Document identifier: S32K3XXRM
Rev. 12, 2025-11-11

S32K3xx Reference Manual

Supports S32K310, S32K311, S32K312, S32K314, S32K322, S32K324, S32K328, S32K338, S32K341, S32K342, S32K344, S32K348, S32K356, S32K358, S32K388, and S32K389

Contents

Chapter 1 About This Manual..... 7

Chapter 2 Introduction..... 15

Chapter 3 Memory Map..... 34

Chapter 4 Signal Multiplexing..... 38

Chapter 5 Cortex-M7 Overview..... 50

Chapter 6 Miscellaneous Control Module (MCM)..... 55

Chapter 7 Miscellaneous System Control Module (MSCM).....76

Chapter 8 Virtualization Wrapper (VIRT_WRAPPER) for S32K388 and S32K389.....148

Chapter 9 Virtualization Wrapper (VIRT_WRAPPER) for all chips except S32K388 and S32K389.....211

Chapter 10 System Integration Unit Lite2 (SIUL2).....302

Chapter 11 Touch Sensing Pin Coupling (TSPC).....492

Chapter 12 Crossbar Switch (AXBS)..... 500

Chapter 13 Peripheral Bridge (AIPS_Lite)..... 517

Chapter 14 Direct Memory Access Multiplexer (DMAMUX)..... 519

Chapter 15 Enhanced Direct Memory Access (eDMA).....529

Chapter 16 Interrupt Monitor (INTM).....633

Chapter 17 Semaphores2 (SEMA42)..... 642

Chapter 18 Crossbar Integrity Checker (XBIC).....651

Chapter 19 Extended Resource Domain Controller (XRDC)..... 675

Chapter 20 Memory and Memory Interfaces..... 767

Chapter 21 Embedded Flash Memory (c40asf)..... 781

Chapter 22 Flash Memory Controller (PFLASH)..... 845

Chapter 23 RAM Controller (PRAMC)..... 896

Chapter 24 Clocking..... 901

Chapter 25 Clock Generation Module (MC_CGM)..... 1007

Chapter 26 Fast Internal RC Oscillator (FIRC)..... 1138

Chapter 27 Slow Internal RC Oscillator (SIRC)..... 1141

Chapter 28 Fast Crystal Oscillator Digital Controller (FXOSC).....1144

Chapter 29 Slow Crystal Oscillator Digital Controller (SXOSC).....1151

Chapter 30 PLL Digital Interface (PLLDIG).....1157

Chapter 31 Reset Overview.....1175

Chapter 32 Boot Overview.....1199

Chapter 33 Reset Generation Module (MC_RGM)..... 1220

Chapter 34 Power-on Reset Watchdog (POR_WDG)..... 1248

Chapter 35 Security Overview..... 1251

Chapter 36 Hardware Security Engine (HSE).....1253

Chapter 37 Device Configuration Format (DCF) records.....1273

Chapter 38 Device Configuration Module General-Purpose Registers (DCM_GPR)..... 1278

Chapter 39 Device Configuration Module (DCM)..... 1467

Chapter 40 Messaging Unit (MU)..... 1509

Chapter 41 Power Management.....1658

Chapter 42 Power Management Controller (PMC for S32K34x, S32K322, S32K324, and S32K314).....1684

Chapter 43 Power Management Controller (PMC for S32K310, S32K311 and S32K312)..... 1696

Chapter 44 Power Management Controller (PMC for S32K358, S32K356, S32K328, S32K338, and S32K348).....1709

Chapter 45 Power Management Controller (PMC for S32K388 and S32K389)1724

Chapter 46 Mode Entry Module (MC_ME).....1739

Chapter 47 Power Control Unit (MC_PCU).....1882

Chapter 48 Wakeup Unit (WKPU)..... 1885

Chapter 49 Safety Overview..... 1915

Chapter 50 Error Injection Module (EIM)..... 1921

Chapter 51 Error Reporting Module (ERM)..... 2209

Chapter 52 Fault Collection and Control Unit (FCCU)..... 2268

Chapter 53 Self-test General-Purpose Registers (SELFTEST_GPR)..... 2327

Chapter 54 Self-Test Control Unit (STCU2).....2331

Chapter 55 Register Protection (REG_PROT).....2392

Chapter 56 Clock Monitoring Unit – Frequency Check (CMU_FC).....2405

Chapter 57 Clock Monitoring Unit – Frequency Meter (CMU_FM)..... 2421

Chapter 58 Cyclic Redundancy Check (CRC).....2429

Chapter 59 Power Conversion and Motor Control (PCMC).....2440

Chapter 60 Analog-to-Digital Converter (ADC).....2457

Chapter 61 Low Power Comparator (LPCMP).....2620

Chapter 62 Logic Control Unit (LCU).....2661

Chapter 63 Enhanced Modular IO Subsystem (eMIOS).....2737

Chapter 64 Body Cross-triggering Unit (BCTU).....2824

Chapter 65 Trigger MUX (TRGMUX).....2864

Chapter 66 Software Watchdog Timer (SWT)..... 2927

Chapter 67 System Timer Module (STM)..... 2945

Chapter 68 Periodic Interrupt Timer (PIT).....2954

Chapter 69 Real Time Clock (RTC).....2979

Chapter 70 Low Power Serial Peripheral Interface (LPSPI)..... 2992

Chapter 71 Low Power Inter-Integrated Circuit (LPI2C)..... 3046

Chapter 72 Flexible I/O (FlexIO).....3108

Chapter 73 CAN (FlexCAN).....3187

Chapter 74 Synchronous Audio Interface (SAI).....3363

Chapter 75 Ethernet Media Access Controller (EMAC).....3413

Chapter 76 Gigabit Ethernet Media Access Controller (GMAC)..... 4003

Chapter 77 Low Power Universal Asynchronous Receiver/Transmitter (LPUART).....4731

Chapter 78 Quad Serial Peripheral Interface (QuadSPI) for S32K322, S32K342, S32K341, S32K314, S32K324, and S32K344..... 4817

Chapter 79 Quad Serial Peripheral Interface (QuadSPI) for S32K358, S32K356, S32K348, S32K338, and S32K328..... 4905

Chapter 80 Quad Serial Peripheral Interface (QuadSPI) for S32K388 and S32K389.....5055

Chapter 81 Ultra Secured Digital Host Controller (uSDHC).....5185

Chapter 82 Debug Subsystem.....5298

Chapter 83 JTAG Controller (JTAGC)..... 5349

Chapter 84 JTAG Data Communication (JDC)..... 5358

Chapter 85 Temperature Sensor (TempSense)..... 5367

Appendix A General Changes..... 5374

Appendix B Release notes.....5375

Legal information..... 5392

Chapter 1
About This Manual

1.1 Audience

This reference manual (RM) is intended for system software, hardware developers, and applications programmers who need to develop products using this chip. It assumes that its users understand operating systems, microprocessor system design, and basic principles of software and hardware.

1.2 Organization

This manual has two main sets of chapters.

- Chapters in the first set contain information that applies to all components on the chip.
- Chapters in the second set are organized into functional groupings that detail particular areas of functionality.
  - Examples of these groupings are clocking, timers, and communication interfaces.
  - Each grouping includes chapters that provide a technical description of individual modules.

1.2.1 Attachments

This manual includes key information in the files attached to it. For example, memory map and I/O details. Use the content in these attachments in conjunction with this manual's content.

NOTE

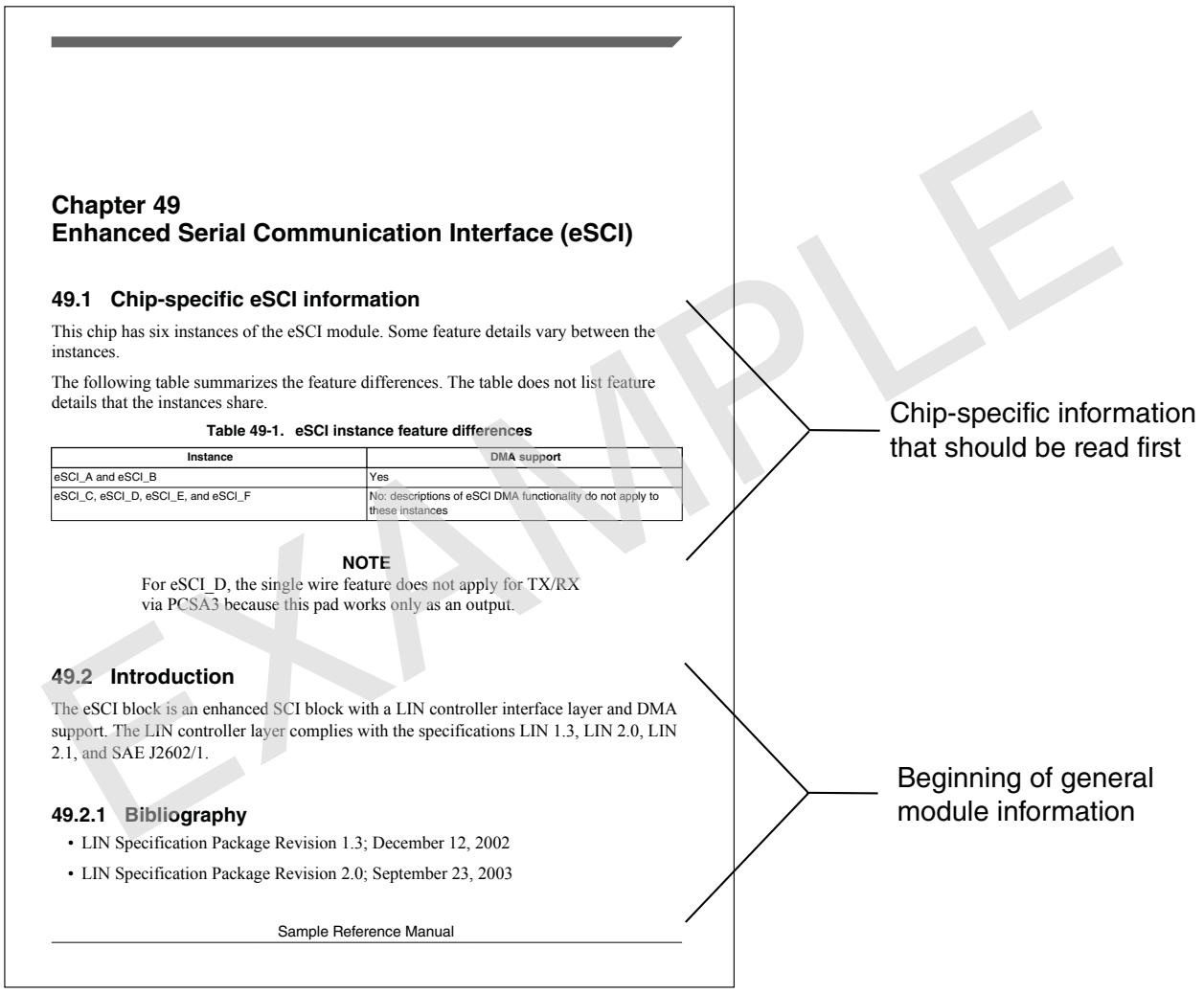
Select the paperclip icon on the left side of the PDF window to see the list of attachments.

1.3 Module descriptions

Each module chapter has two main parts:

- The first section, *chip-specific [module name]* information, provides details such as the number of module instances on the chip and connections between that module and the other ones. Read this section *first* because its content is crucial for understanding the information in the other sections of the chapter.
- The subsequent sections provide general information about the module, including its signals, registers, and functional description.

The following figure shows you an example of this demarcation.



Chapter 49
Enhanced Serial Communication Interface (eSCI)

49.1 Chip-specific eSCI information

This chip has six instances of the eSCI module. Some feature details vary between the instances.

The following table summarizes the feature differences. The table does not list feature details that the instances share.

Table 49-1. eSCI instance feature differences

Instance	DMA support
eSCI_A and eSCI_B	Yes
eSCI_C, eSCI_D, eSCI_E, and eSCI_F	No: descriptions of eSCI DMA functionality do not apply to these instances

NOTE

For eSCI_D, the single wire feature does not apply for TX/RX via PCSA3 because this pad works only as an output.

49.2 Introduction

The eSCI block is an enhanced SCI block with a LIN controller interface layer and DMA support. The LIN controller layer complies with the specifications LIN 1.3, LIN 2.0, LIN 2.1, and SAE J2602/1.

49.2.1 Bibliography

- LIN Specification Package Revision 1.3; December 12, 2002
- LIN Specification Package Revision 2.0; September 23, 2003

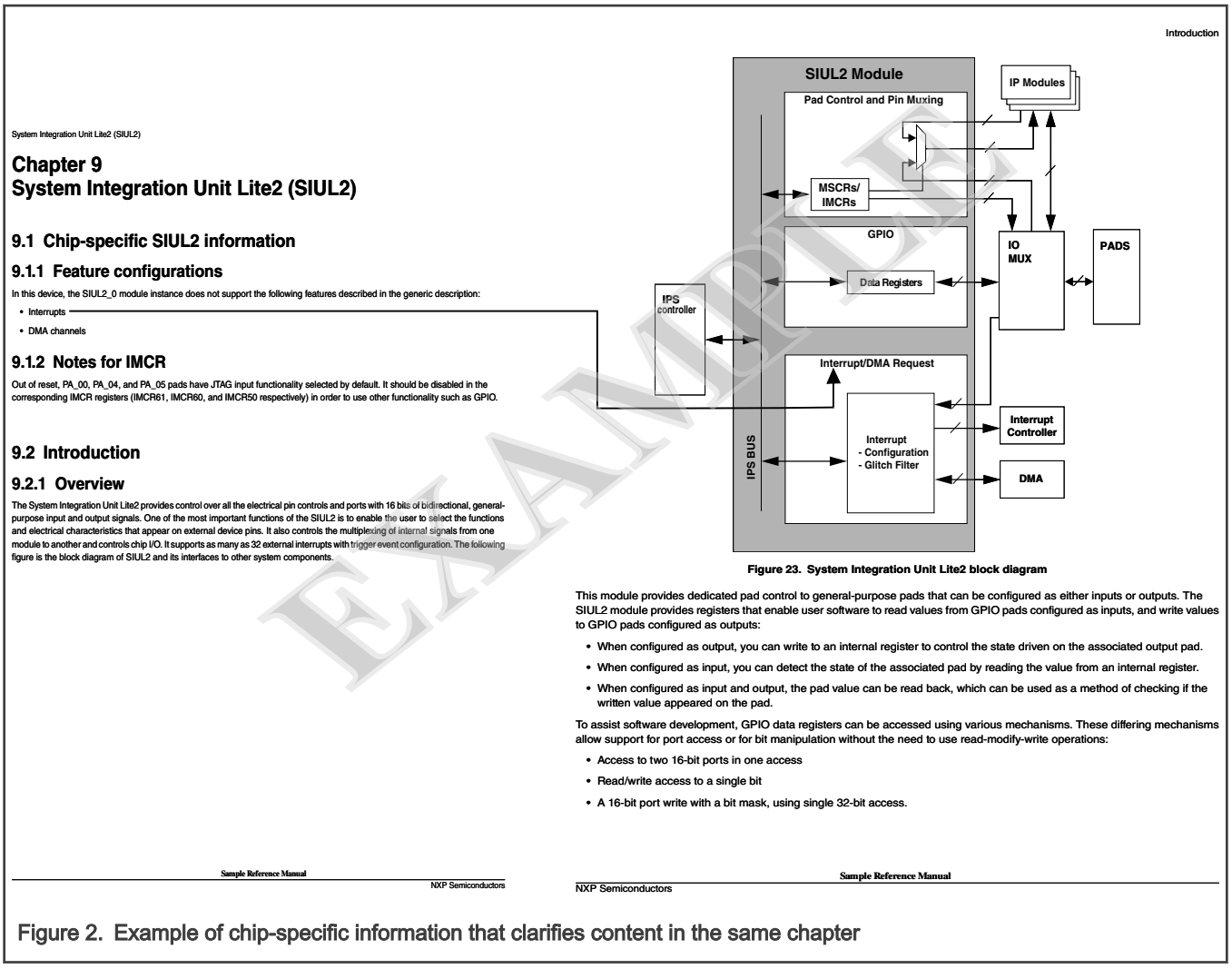
Sample Reference Manual

EXAMPLE

Figure 1. Example of chapter chip-specific information and general module information

1.3.1 Chip-specific information that clarifies content in the same chapter

The following figure shows an example of chip-specific information that clarifies general module information presented later in the chapter. In this case, the chip-specific register reset values supersede the reset values that appear in the register diagram.



Introduction

System Integration Unit Lite2 (SIUL2)

Chapter 9
System Integration Unit Lite2 (SIUL2)

9.1 Chip-specific SIUL2 information

9.1.1 Feature configurations

In this device, the SIUL2_0 module instance does not support the following features described in the generic description:

- Interrupts
- DMA channels

9.1.2 Notes for IMCR

Out of reset, PA_00, PA_04, and PA_05 pads have JTAG input functionality selected by default. It should be disabled in the corresponding IMCR registers (IMCR61, IMCR60, and IMCR50 respectively) in order to use other functionality such as GPIO.

9.2 Introduction

9.2.1 Overview

The System Integration Unit Lite2 provides control over all the electrical pin controls and ports with 16 bits of bidirectional, general-purpose input and output signals. One of the most important functions of the SIUL2 is to enable the user to select the functions and electrical characteristics that appear on external device pins. It also controls the multiplexing of internal signals from one module to another and controls chip I/O. It supports as many as 32 external interrupts with trigger event configuration. The following figure is the block diagram of SIUL2 and its interfaces to other system components.

Figure 23. System Integration Unit Lite2 block diagram

This module provides dedicated pad control to general-purpose pads that can be configured as either inputs or outputs. The SIUL2 module provides registers that enable user software to read values from GPIO pads configured as inputs, and write values to GPIO pads configured as outputs:

- When configured as output, you can write to an internal register to control the state driven on the associated output pad.
- When configured as input, you can detect the state of the associated pad by reading the value from an internal register.
- When configured as input and output, the pad value can be read back, which can be used as a method of checking if the written value appeared on the pad.

To assist software development, GPIO data registers can be accessed using various mechanisms. These differing mechanisms allow support for port access or for bit manipulation without the need to use read-modify-write operations:

- Access to two 16-bit ports in one access
- Read/write access to a single bit
- A 16-bit port write with a bit mask, using single 32-bit access.

Sample Reference Manual
NXP Semiconductors

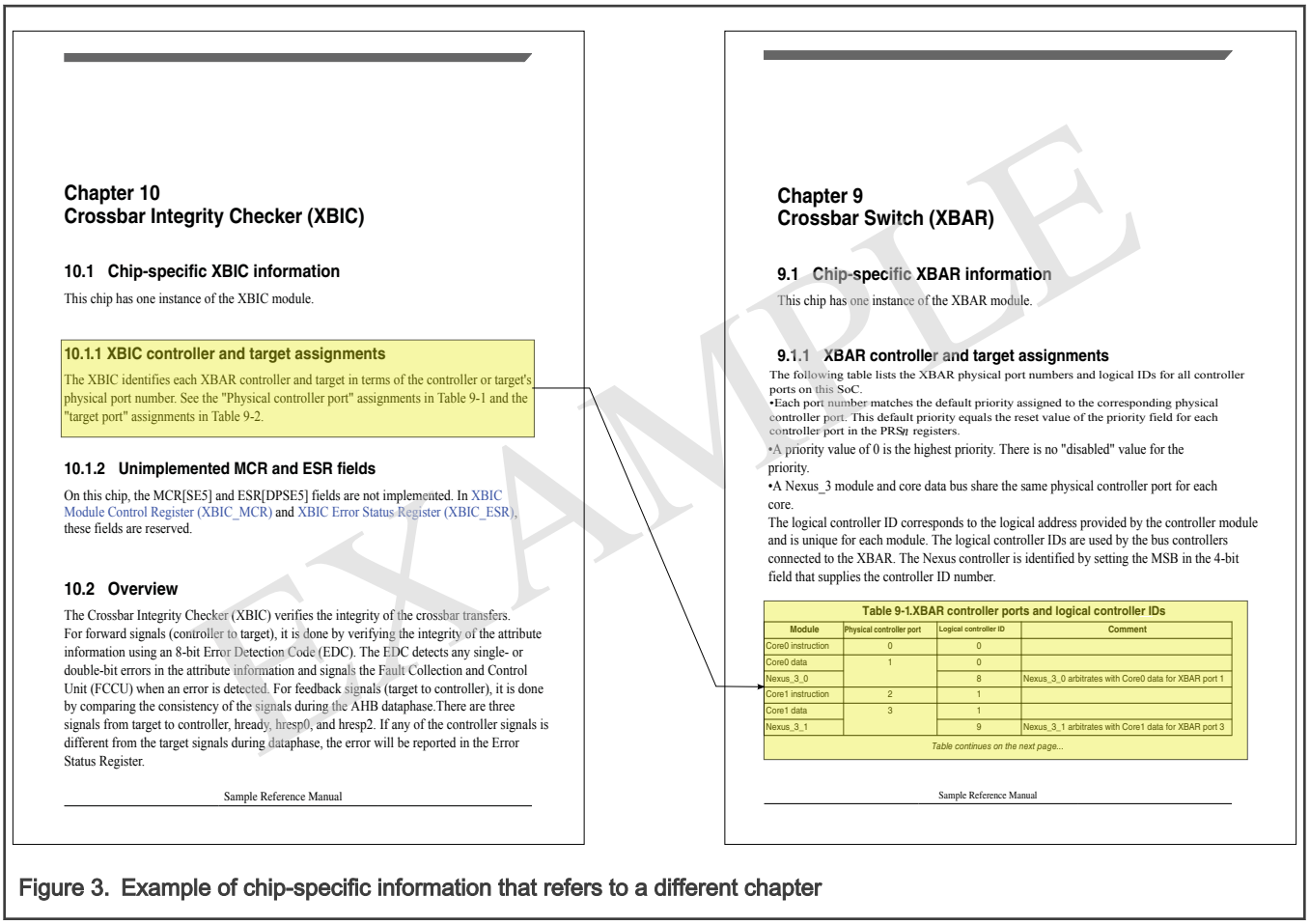
Sample Reference Manual
NXP Semiconductors

EXAMPLE

Figure 2. Example of chip-specific information that clarifies content in the same chapter

1.3.2 Chip-specific information that refers to a different chapter

Related chip-specific information may be provided in different chapters of the manual. The following figure shows an example of two such connected pieces of information. In this case, read both before you proceed.



Chapter 10
Crossbar Integrity Checker (XBIC)

10.1 Chip-specific XBIC information

This chip has one instance of the XBIC module.

10.1.1 XBIC controller and target assignments

The XBIC identifies each XBAR controller and target in terms of the controller or target's physical port number. See the "Physical controller port" assignments in Table 9-1 and the "target port" assignments in Table 9-2.

10.1.2 Unimplemented MCR and ESR fields

On this chip, the MCR[SE5] and ESR[DPSE5] fields are not implemented. In XBIC Module Control Register (XBIC_MCR) and XBIC Error Status Register (XBIC_ESR), these fields are reserved.

10.2 Overview

The Crossbar Integrity Checker (XBIC) verifies the integrity of the crossbar transfers. For forward signals (controller to target), it is done by verifying the integrity of the attribute information using an 8-bit Error Detection Code (EDC). The EDC detects any single- or double-bit errors in the attribute information and signals the Fault Collection and Control Unit (FCCU) when an error is detected. For feedback signals (target to controller), it is done by comparing the consistency of the signals during the AHB dataphase.There are three signals from target to controller, hready, hresp0, and hresp2. If any of the controller signals is different from the target signals during dataphase, the error will be reported in the Error Status Register.

Sample Reference Manual

Chapter 9
Crossbar Switch (XBAR)

9.1 Chip-specific XBAR information

This chip has one instance of the XBAR module.

9.1.1 XBAR controller and target assignments

The following table lists the XBAR physical port numbers and logical IDs for all controller ports on this SoC.

- Each port number matches the default priority assigned to the corresponding physical controller port. This default priority equals the reset value of the priority field for each controller port in the PRSn registers.
- A priority value of 0 is the highest priority. There is no "disabled" value for the priority.
- A Nexus_3 module and core data bus share the same physical controller port for each core.

The logical controller ID corresponds to the logical address provided by the controller module and is unique for each module. The logical controller IDs are used by the bus controllers connected to the XBAR. The Nexus controller is identified by setting the MSB in the 4-bit field that supplies the controller ID number.

Table 9-1.XBAR controller ports and logical controller IDs

Module	Physical controller port	Logical controller ID	Comment
Core0 instruction	0	0	
Core0 data	1	0	
Nexus_3_0		8	Nexus_3_0 arbitrates with Core0 data for XBAR port 1
Core1 instruction	2	1	
Core1 data	3	1	
Nexus_3_1		9	Nexus_3_1 arbitrates with Core1 data for XBAR port 3

Table continues on the next page...

Sample Reference Manual

EXAMPLE

Figure 3. Example of chip-specific information that refers to a different chapter

1.4 Register descriptions

Module chapters present register information in the following:

- Memory maps, which contain:
  - An offset from the module's base address
  - The mnemonic and name of each register
  - The width of each register (in bits)
  - The reset value of each register
- Register figures
- Field-description tables
- Associated text

The following figure shows register figure conventions used throughout the manual.

Access type	Access description	DITA output	Effect of Write on Value	Readback Value
RW	Read/write	R / W: Mnemonic	Changes to Written Value	Current Value
RO	Read-only	R: Mnemonic / W:	No Effect	Current Value
RU	Reserved, unimplemented	R / W: Reserved	No Effect	Value Undefined
ROZ	Reserved, Read-only zero	R: 0 / W:	No Effect	Returns All Zero
ROO	Reserved, Read-only one	R: 1 / W:	No Effect	Returns All Ones
WO	Write-only	R: / W: Mnemonic	Changed to Written Value	Value Undefined
WOZ	Reserved, Write-only zero	R: / W: 0	Write Value Must Be All 0s.	Value Undefined
WOO	Reserved, Write-only one	R: / W: 1	Write Value Must Be All 1s.	Value Undefined
W0C	Write zero to clear	R: Mnemonic / W: w0c	If a Bit in the Written Value is a 0, the Corresponding Bit in the Field is Set to 0. Otherwise, the Field Bit is Not Affected.	Current Value
W1C	Write one to clear	R: Mnemonic / W: w1c	If a Bit in the Written Value is a 1, the Corresponding Bit in the Field is set to 0. Otherwise, the Field Bit is Not Affected.	Current Value
R2C	Read to clear		Value is cleared following the read operation	Current Value
ROWZ	Read-only, writes zero	R: Mnemonic / W: 0	Write Value Must Be All 0s	Current Value
ROWO	Read-only writes one	R: Mnemonic / W: 1	Write Value Must Be All 1s	Current Value
ROWU	Ready-only writes undefined	R: Mnemonic / W: —	Writes Operation Undefine	Current Value
WORZ	Write-only reads zero	R: 0 / W: Mnemonic	Changes to Written Value	Returns All Zero
WORO	Write-only reads one	R: 1 / W: Mnemonic	Changes to Written Value	Returns All Ones
WORU	Write-only reads undefined	R: — / W: Mnemonic	Changes to Written Value	Value Undefined

Figure 4. Register figure conventions

NOTE

Reset values of reserved locations documented in this manual are subject to change and must not be used for diagnostic purposes.

1.5 Conventions

1.5.1 Notes and cautions

Specific information is provided as part of notes and cautions throughout this manual.

NOTE

Emphasizes information that deserves extra attention.

CAUTION

Informs you of situations that could lead to highly undesirable outcomes—such as damage to the chip or irreversible malfunction.

1.5.2 Numbering systems

The following suffixes identify different numbering systems:

Table 1. Numbering systems

This suffix	Identifies a
b	Binary number. For example, the binary equivalent of the number 5 is mentioned as 101b. In some cases, *0b* is prefixed to binary numbers.
d	Decimal number. Decimal numbers are followed by this suffix only when there is a possibility of confusion. In general, decimal numbers are used without a suffix.
h	Hexadecimal number. For example, the hexadecimal equivalent of the number 60 is mentioned as 3Ch. In some cases, *0x* is prefixed to hexadecimal numbers.

1.5.3 Typographic notation

The following typographic notations are used throughout this document:

Table 2. Typographic notation

Example	Description
x and other italicized text	The italicized, lowercase *x* is used as a placeholder for replaceable numbers. In general, italicized text is used for titles of publications and for emphasis. Additionally, italics could be used for metasymbols in syntax descriptions. Plain lowercase letters are used as placeholders for single letters and numbers.
`code font`	Fixed-width font (such as Courier) used for code. It is used for a letter, word, or phrase that you want the user to type. For example, "Type `Read` and press Enter." This type of font is also used for instruction mnemonics, directives, symbols, subcommands, parameters, operators, computer-language elements, code listings, commands that appear in running text, and for sample code. Instruction mnemonics and directives in text and tables are mentioned in all caps; for example, BSR.
SR[SCM]	A mnemonic in square brackets represents the name of a register field. This example refers to the Scaling Mode (SCM) field in the Status Register (SR).
REVNO[6:4], XAD[7:0]	Numbers in brackets that are separated by a colon represent either: • A subset of a register's named field

Table continues on the next page...

Table 2. Typographic notation (continued)

Example	Description
	For example, REVNO[6:4] refers to bits 6-4 that are part of the COREREV field occupying bits 6-0 of the REVNO register. • A continuous range of individual signals of a bus For example, XAD[7:0] refers to signals 7-0 of the XAD bus.
MOD.REG	A period separates the elements of a hierarchy: subsystem.module.register. For example: • SWT.TO means that the TO register is located in the SWT module. • SMU.XRDC.CR means that the CR register is located in the XRDC module within the SMU subsystem.

1.5.4 Special terms

The following terms have special meanings.

Table 3. Special terms

Term	Meaning
Asserted	Refers to the state of a signal as follows: • An active-high signal is asserted when high (1). • An active-low signal is asserted when low (0).
Deasserted	Refers to the state of a signal as follows: • An active-high signal is deasserted when low (0). • An active-low signal is deasserted when high (1). In some cases, deasserted signals are described as *negated*.
Reserved	Refers to memory space, register, field, or programming setting. Writes to a reserved location can result in unpredictable functionality or behavior. You must: • Before writing to a location which contain reserved bits user must make sure the write operation will write the reserved bit with value specified as the reset value in NXP reference manual. • Consider undefined locations in memory to be reserved as reset value 0 shall be assumed. You might get a BERR(transfer error) response on access to undefined locations in memory. • If user reads data from memory area containing reserved bit, the value of reserved bits should be ignored and not used for any functional purposes. **NOTE** BootROM could modify the reserved bit values after reset. Please refer to the BootROM settings attachment.
Write 1 to clear (w1c)	Refers to the access type of a register field that is used to clear the field by writing the value 1 to it.
Undefined (u)	Refers to undefined reset values

1.6 Editorial changes

Each new release of this document includes editorial improvements such as:

- Spelling
- Grammar
- Punctuation
- Voice
- Tense
- Capitalization
- Formatting
- Presentation
- Navigation

Chapter 2
Introduction

2.1 Overview

The S32K3xx product series further extends the highly-scalable portfolio of Arm ® Cortex ® - M0+/M4F S32K1xx chips in the automotive industry with the Arm Cortex-M7 core at higher frequency, more memory, ASIL-B and D rating and advanced security module. With a focus on automotive environment robustness, the S32K3xx product series devices are well suited to a wide range of applications in electrical harsh environments, and are optimized for cost-sensitive applications offering new, space saving package options. The S32K3xx series offers a broad range of memory, peripherals and performance options. Devices in this series share common peripherals and pin-out, allowing developers to migrate easily within a chip series or among other chip series to take advantage of more memory or feature integration.

2.2 S32K3xx product series

The S32K3xx series comes to market together with easy-to-use enablement software, application specific software, and various development tools, supported by broad third parties in different development phases.

The portfolio scalability, future-proof feature like advanced security, as well as software/tool/third-party development support allows developers to standardize on the S32K series for their end product platforms, maximizing hardware and software reuse, and reducing time-to-market.

Following are the general features of the S32K3xx series chips:

- 32-bit Arm Cortex-M7 core with IEEE-754 compliant SPFPU, executing up to 320 MHz
- Scalable memory footprints up to 12 MB flash memory and up to 2.25 MB SRAM
- Precision mixed-signal capability with low power comparators (LPCMP) and multiple 12-bit ADCs
- Powerful timers for a broad range of applications including motor control, lighting control and body applications
- Serial communication interfaces such as LPUART, LPSPI, LPI2C, FlexCAN with ISOCAN-FD support, Ethernet and QuadSPI. FXIO configuration allows other communication options including SENT.
- EVITA full and light functionality compliant HSE_B
- Power supply (3.0 – 5.5 V) with fully functional flash memory program/erase/read operations
- Functional safety compliance with ISO26262 levels B or D (features depend on device specification)
  - Lockstep cores
  - Multiple internal watchdogs
  - Voltage monitors
  - Clock monitors
  - Memory protection
  - Data transport checks
  - ECC on memories
  - Cyclic redundancy checking
- Ambient operation temperature range: –40°C to 125°C
- Junction temperature range: –40°C to 150°C
- Tools solutions:
  - S32 Design Studio IDE
  - NXP GCC compilers for Cortex-M7 and IDE plugins for partner compilers support

  - Low cost debugger with multi-core debug, tracing and profiling support
  - S32 Configuration Tools:
    - Pin wizard with detailed pin configuration report
    - Graphical clock tree for clock configuration
    - Peripheral configuration
  - Model-Based Design Toolbox:
    - Integrated Simulink®-embedded target for direct rapid prototyping and PIL development workflows
    - Peripheral device interface blocks and drivers
    - Target-optimized math and motor control algorithm blocks for efficient execution on the target chip
    - Bit-accurate simulation results in the Simulink® simulation environment
  - FreeMaster
- Software solutions:
  - RTD (Real Time Drivers)
    - AUTOSAR compliant MCAL plus complex device drivers for external ICs (e.g. PMIC) and miscellaneous SW components (e.g. inter core communication)
    - Low level drivers covering all device periphery
    - Example projects for each driver
    - Configuration components for Tresos Studio and S32ConfigTools framework
    - Integrated with S32 Design Studio
  - Stacks/libraries
    - Communication stacks:
      - LIN, TCP/IP, gPTP, AVB, WiFi, AWS IOT
    - Security:
      - mBedTLS
    - Safety:
      - Safety Software Framework
    - Real time control:
      - AMMCLib
      - ISELED
    - RTOS:
      - FreeRTOS
    - Firmware:
      - Security firmware for on-chip crypto engine
    - Core to core communication:
      - IPCF
    - Edge to cloud connectivity:
      - AWS IOT
- Designed to work in conjunction with NXP SBC UJA124x and SBC FS2600

2.3 Feature summary

The S32K3xx product family includes the Arm Cortex-M7 core features described in the following table.

Table 4. S32K3xx chip's feature summary

Feature	Inclusions
Core and architecture	• Arm Cortex-M7 core running up to 320 MHz • Arm core based on the Armv7 architecture and Thumb-2 ISA • Upto 16 KB data and 16 KB instruction cache for optimizing wait state execution from memories • 96 KB Tightly Coupled Memory associated with each core • On-core MPU for dynamic task protection (16 regions) • SPFPU, IEEE 754 compliant • Harvard bus architecture implementing dedicated instruction and data path • 5-stage pipeline with branch speculation • XRDC integrated with a crossbar switch to provide memory and peripheral protection • DSP • I/O protection (VIRT_WRAPPER) • ETM supporting instruction trace • Arm third-party ecosystem support: software and tools to help minimize development time and cost
DMA	• Up to 2x64-channel DMAMUX • eDMA with up to 32 channels • Complex data transfers performed with minimal intervention from a host processor • Programmable support for scatter-gather DMA processing
System and power management	• Support for simplified power modes (Run and Standby) • Support for clock gating of unused modules; specific peripherals continue to work in low-power modes • Support for external ballast transistor to generate core supply • Fully independent CPU and peripheral clocking scheme • Rapid start-up from a 48 MHz FIRC • Various low-power oscillators such as the 32 kHz SIRC and support for an external 32 kHz crystal (SXOSC) • PMC with LVD and selectable trip points • NMI

Table continues on the next page...

Table 4. S32K3xx chip's feature summary (continued)

Feature	Inclusions
Memory and memory interfaces	• Up to 12 MB program flash memory, up to 256 KB data flash memory, and up to 2304 KB SRAM, all with an ECC • 4-bit/8-bit QuadSPI
Clocks	• External 8 MHz–40 MHz crystal oscillator or resonator • External 32 kHz crystal oscillator • Internal clock references — 48 MHz FIRC ±5% — 32 kHz SIRC ±10% • Up to 1280 MHz PLL for divided system clock operation • Auxiliary PLL at up to 1 GHz for additional clocking options (on selected devices)
Security and integrity	• Hardware security engine (HSE_B) — Upgradable Firmware delivered by NXP, to be programmed by the user • Security ciphers: — Symmetric: AES-128/192/256 — AES_ACCEL: Additional AES Accelerator for low latency AES 128/256 signature/verify [1] — Cipher modes: ECB, CBC, CMAC, GMAC, CTR, OFB, CCM, and GCM — Asymmetric: RSA (up to 4096 bits) and ECC (up to 521 bits) — Hash: Miyaguchi-Preneel, SHA-2/SHA-3 (up to 512 bits) — Number of keys is configurable and controlled by HSE FW — Random number generator • Security use case supported: — Secure boot — Secure communication — Component protection — Secure storage — Key exchange
Safety ISO26262	• Classification up to ASIL-D • ERM and EIM support • WDOG with an independent clock source • Voltage monitors

Table continues on the next page...

Table 4. S32K3xx chip's feature summary (continued)

Feature	Inclusions
	• Bandgap voltage available as ADC input • External clock source monitoring using an independent reference • PLL lock and loss-of-lock protection • XRDC • ECC on code flash memory, data flash memory, and system RAM • ADC self-test feature • Internal analog monitoring of all supplies available • CRC generation module • FCCU failure output
Analog	• 12-bit ADC — Up to 72 external analog inputs — 1 µs conversion time — Internal bandgap voltage reference channel, supporting automatic compare and an optional hardware trigger — Up to five internal reference inputs — Automatic compare with interrupt — Self-test and self-calibration scheme • LPCMP with an internal 8-bit DAC as a reference — Low power analog comparator with both positive and negative inputs, separately selectable interrupts on rising and falling comparator output — Ability to cross trigger the timers from both the ADC and LPCMP outputs • Temperature Sensor (TempSense) with output measurable by ADC.
Timers	• 16-bit or 24-bit eMIOS timers, offering up to 72 standard channels — Input capture, output compare, and PWM modes — Fault input support with global fault control — Multiple features such as deadtime insertion, configurable polarity, quadrature decoding, and so on • 32-bit PITs, with four channels, for raising interrupts and triggering DMA channels • 32-bit RTC • Motor control and power conversion using combination of eMIOS, LCU , BCTU, and ADC • Up to four STMs, with four channels each

Table continues on the next page...

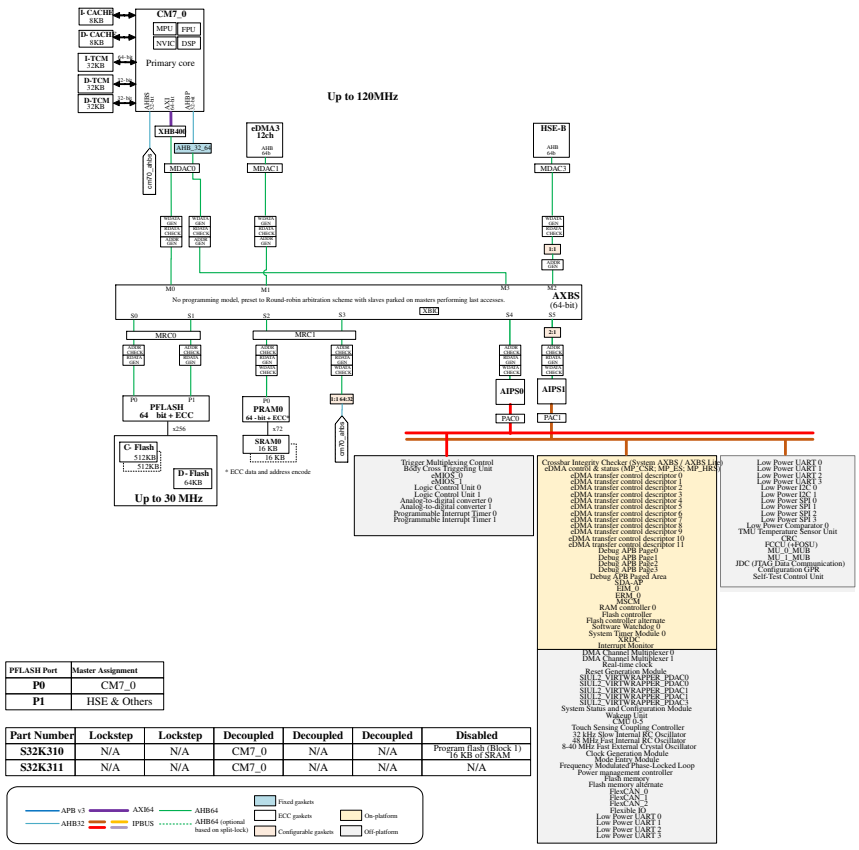
Table 4. S32K3xx chip's feature summary (continued)

Feature	Inclusions
Communications	• LPSPI supporting DMA with full-duplex or single-wire bidirectional communication in Master or Slave mode • FlexIO, with an option to configure it as different communication peripherals, offering support for SENT • LPI2C modules with DMA support, low-power availability, master or slave support, and system management bus • LPUART with DMA support, having an optional 13-bit break, full-duplex non-return- to-zero (NRZ), low-power availability and supports LIN protocol versions 1.3, 2.0, 2.1, 2.2A, and SAE J2602 with using SW LIN driver • FlexCAN modules with ISOCAN-FD and DMA support • SAI capable of supporting stereo audio channels • uSDHC interface to interface with SD/SDIO/MMC cards. • EMAC complex (10/100 Ethernet) that supports 1588 timers, MII/RMII interface, AVB, and TSN support • GMAC (Gigabit Ethernet) with support for AVB (3.3 V only for RGMII) and Time Sensitive Networking (TSN) capability
Debug	• DWT, with four configurable comparators as hardware watchpoints • SWO-synchronous trace data support • ITM with software and hardware trace, plus time stamping • FPB with an ability to patch code and data from code space to system space • Trace of all execution units and bus masters made available through an Arm TPIU over GPIO pins; a very low bandwidth trace option also available via the SWO • Embedded trace FIFO (ETF)—a dedicated trace buffer available for each of the core masters, allowing data to be captured internally before being optionally routed to external trace pins • SWV—trace capability providing displays of reads, writes, exceptions, PC samples, and print
I/O and package	• Up to 320 GPIO pins • Up to 144 GPIO pins with interrupt functionality • Up to 60 GPIO pins with wakeup capability • Pseudo open drain support on LPUART, FlexIO, LPI2C • Package options of 289 MAPBGA, 257 MAPBGA, 172 HDQFP, 100 HDQFP, 100 LQFP, 172 HDQFP-EP, 48 LQFP, and 437 MAPBGA

1. Available only in S32K388 and S32K389

2.4 Block diagram

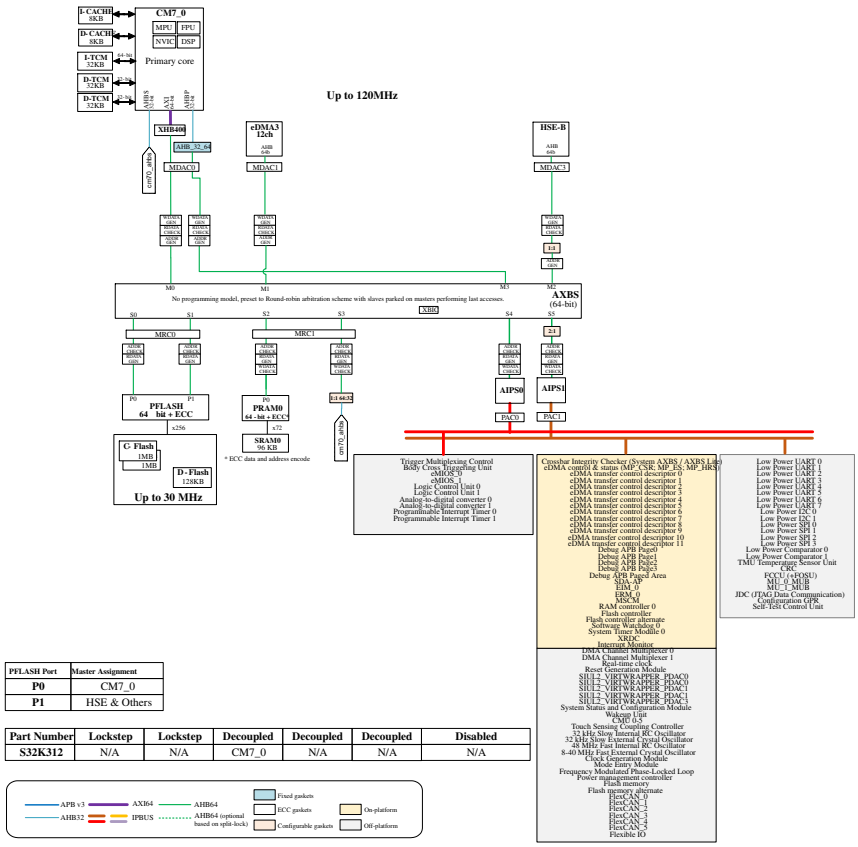
The following figures depict the block diagrams of S32K3xx family devices.



PFLASH Port	Master Assignment
P0	CM7_0
P1	HSE & Others

Part Number	Lockstep	Lockstep	Decoupled	Decoupled	Decoupled	Disabled
S32K310	N/A	N/A	CM7_0	N/A	N/A	Program flash (Block 1) 16 KB of SRAM
S32K311	N/A	N/A	CM7_0	N/A	N/A	N/A

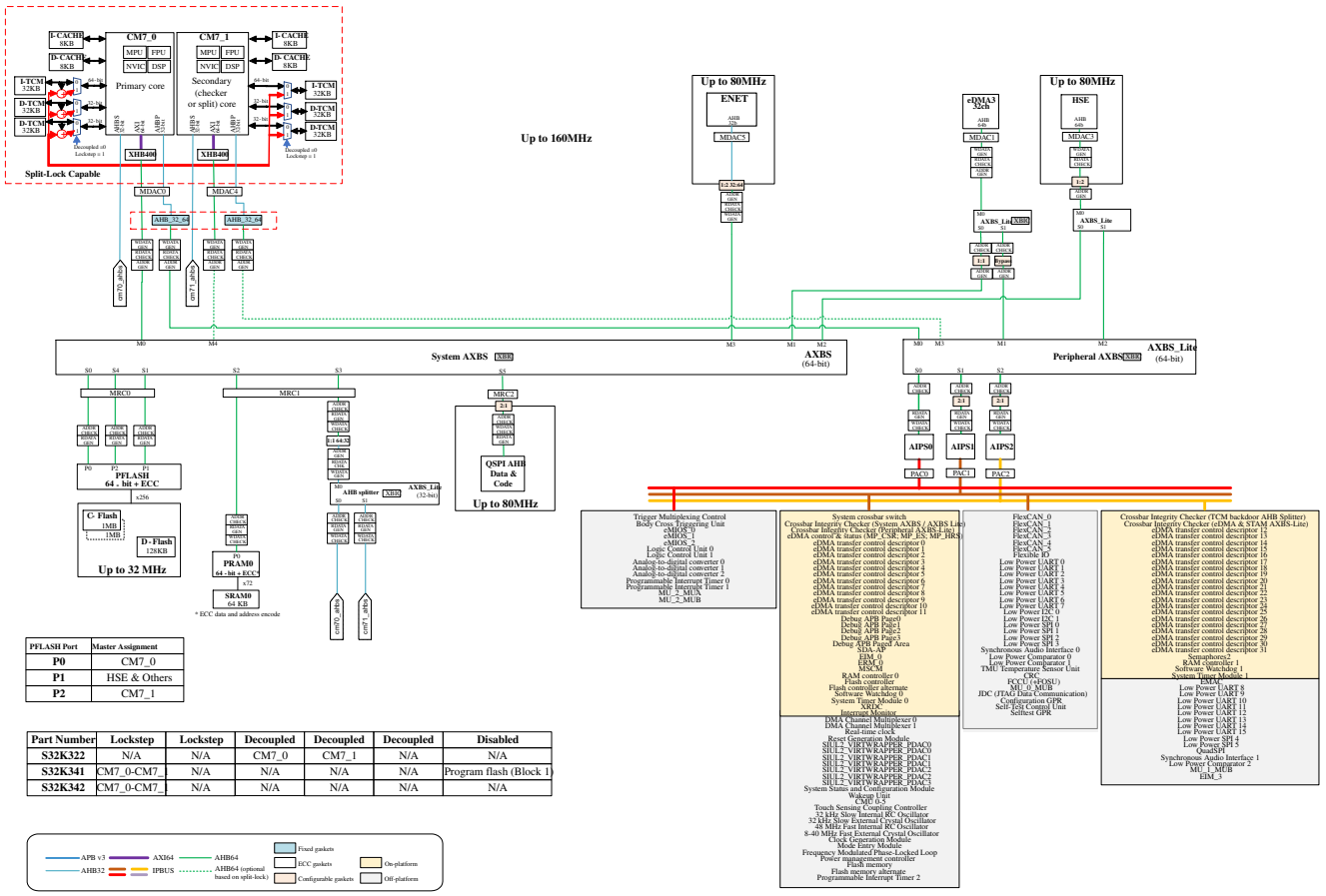
Figure 5. Block diagram – S32K311 and S32K310



PFLASH Port	Master Assignment
P0	CM7_0
P1	HSE & Others

Part Number	Lockstep	Lockstep	Decoupled	Decoupled	Decoupled	Disabled
S32K312	N/A	N/A	CM7_0	N/A	N/A	N/A

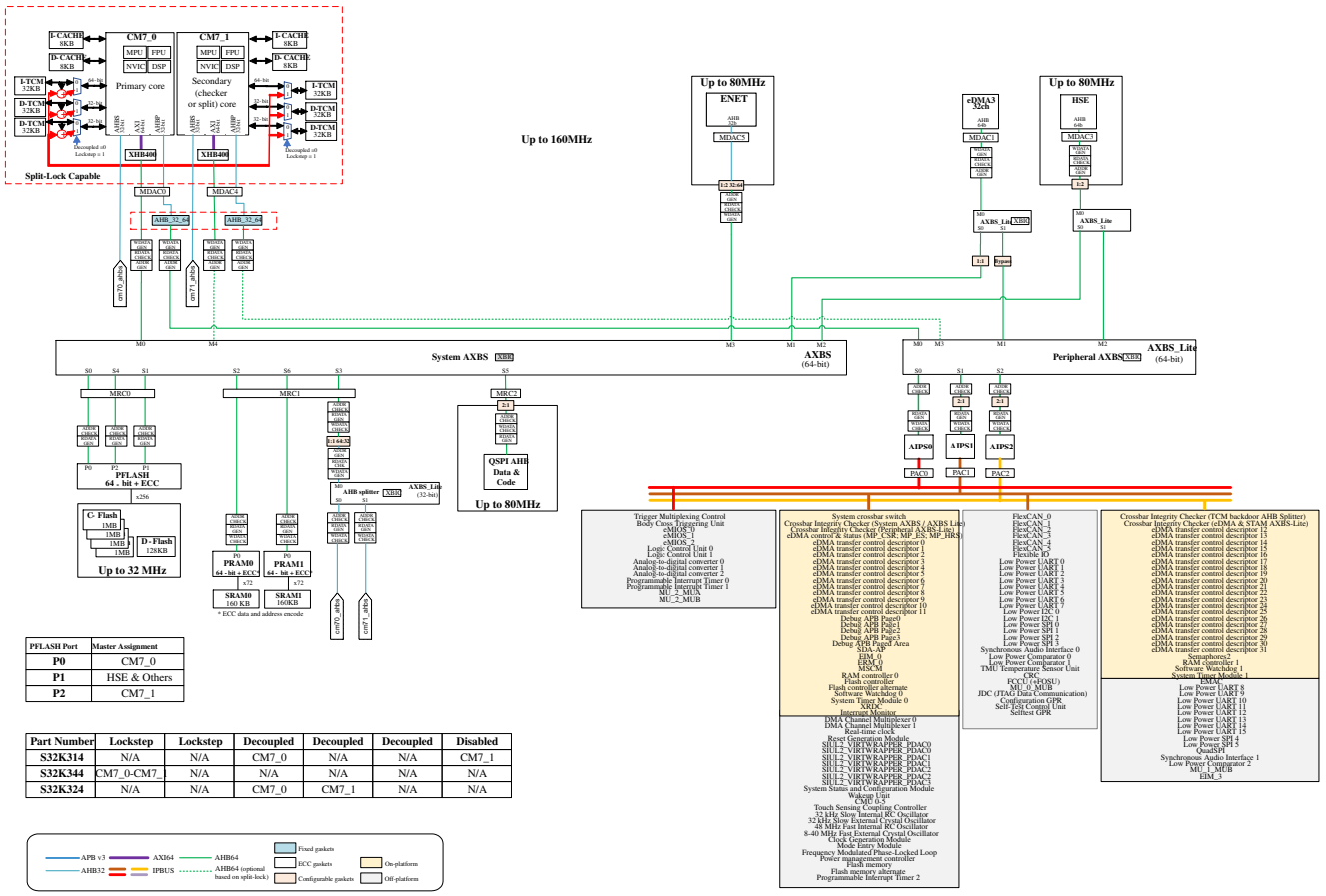
Figure 6. Block diagram – S32K312



PFLASH Port	Master Assignment
P0	CM7_0
P1	HSE & Others
P2	CM7_1

Part Number	Lockstep	Lockstep	Decoupled	Decoupled	Decoupled	Disabled
S32K322	N/A	N/A	CM7_0	CM7_1	N/A	N/A
S32K341	CM7_0-CM7_	N/A	N/A	N/A	N/A	Program flash (Block 1)
S32K342	CM7_0-CM7_	N/A	N/A	N/A	N/A	N/A

Figure 7. Block diagram – S32K322, S32K342 and S32K341



PFLASH Port	Master Assignment
P0	CM7_0
P1	HSE & Others
P2	CM7_1

Part Number	Lockstep	Lockstep	Decoupled	Decoupled	Decoupled	Disabled
S32K314	N/A	N/A	CM7_0	N/A	N/A	CM7_1
S32K344	CM7_0-CM7_	N/A	N/A	N/A	N/A	N/A
S32K324	N/A	N/A	CM7_0	CM7_1	N/A	N/A

Figure 8. Block diagram – S32K324, S32K344 and S32K314

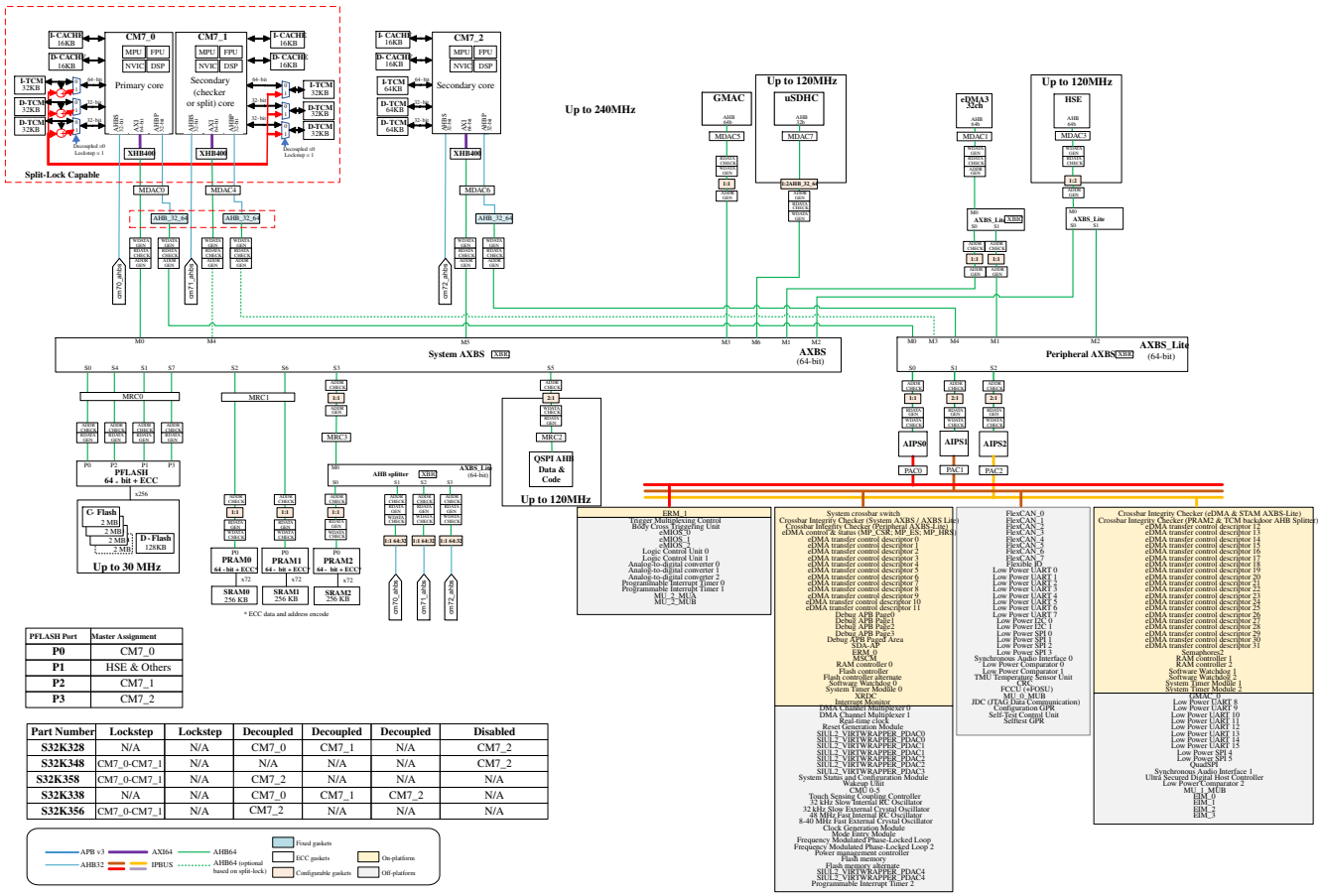
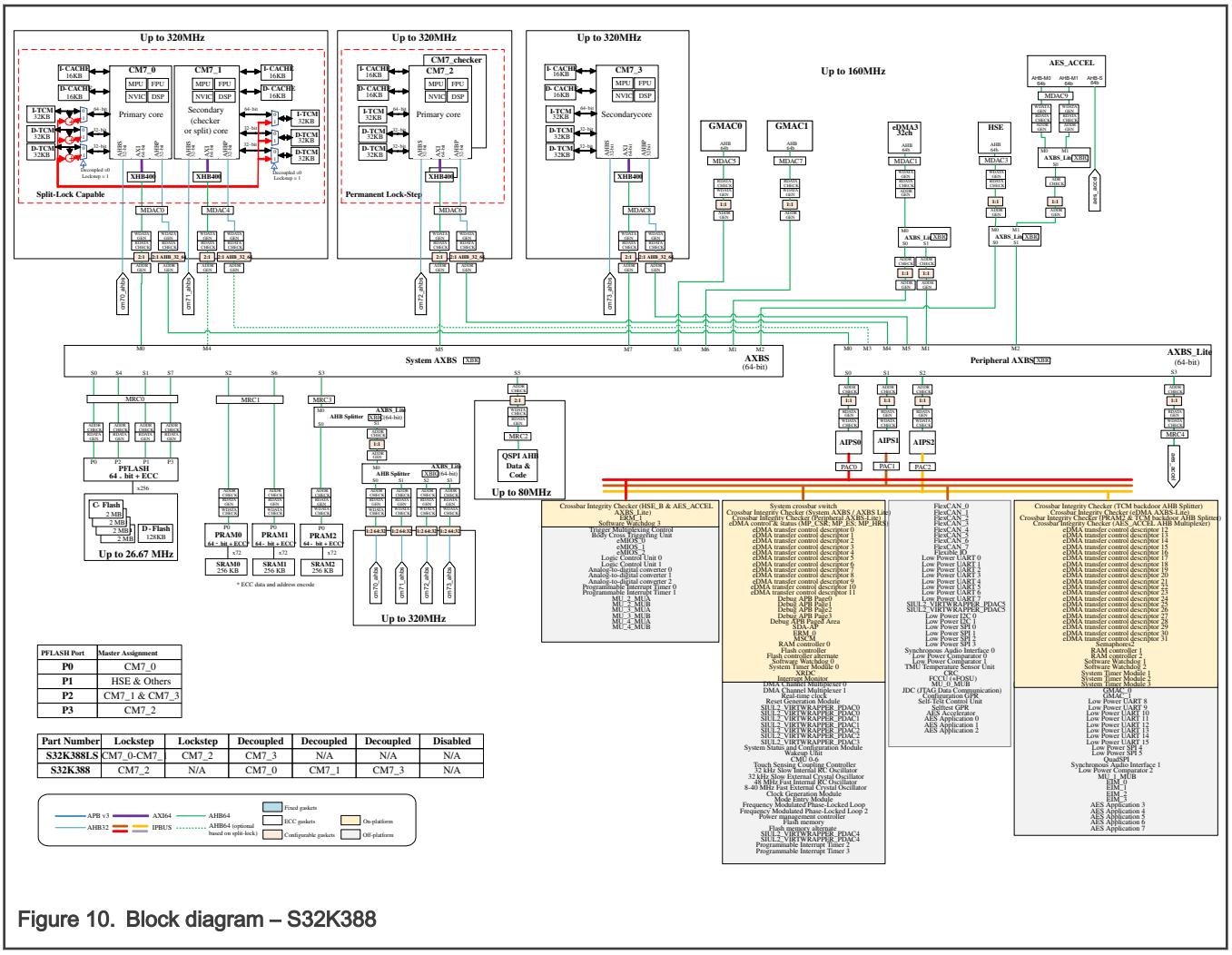


Figure 9. Block diagram – S32K338, S32K358, S32K356, S32K348 and S32K328



PFLASH Port	Master Assignment
P0	CM7_0
P1	HSE & Others
P2	CM7_1 & CM7_3
P3	CM7_2

Part Number	Lockstep	Lockstep	Decoupled	Decoupled	Decoupled	Disabled
S32K388LS	CM7_0-CM7_	CM7_2	CM7_3	N/A	N/A	N/A
S32K388	CM7_2	N/A	CM7_0	CM7_1	CM7_3	N/A

Figure 10. Block diagram – S32K388

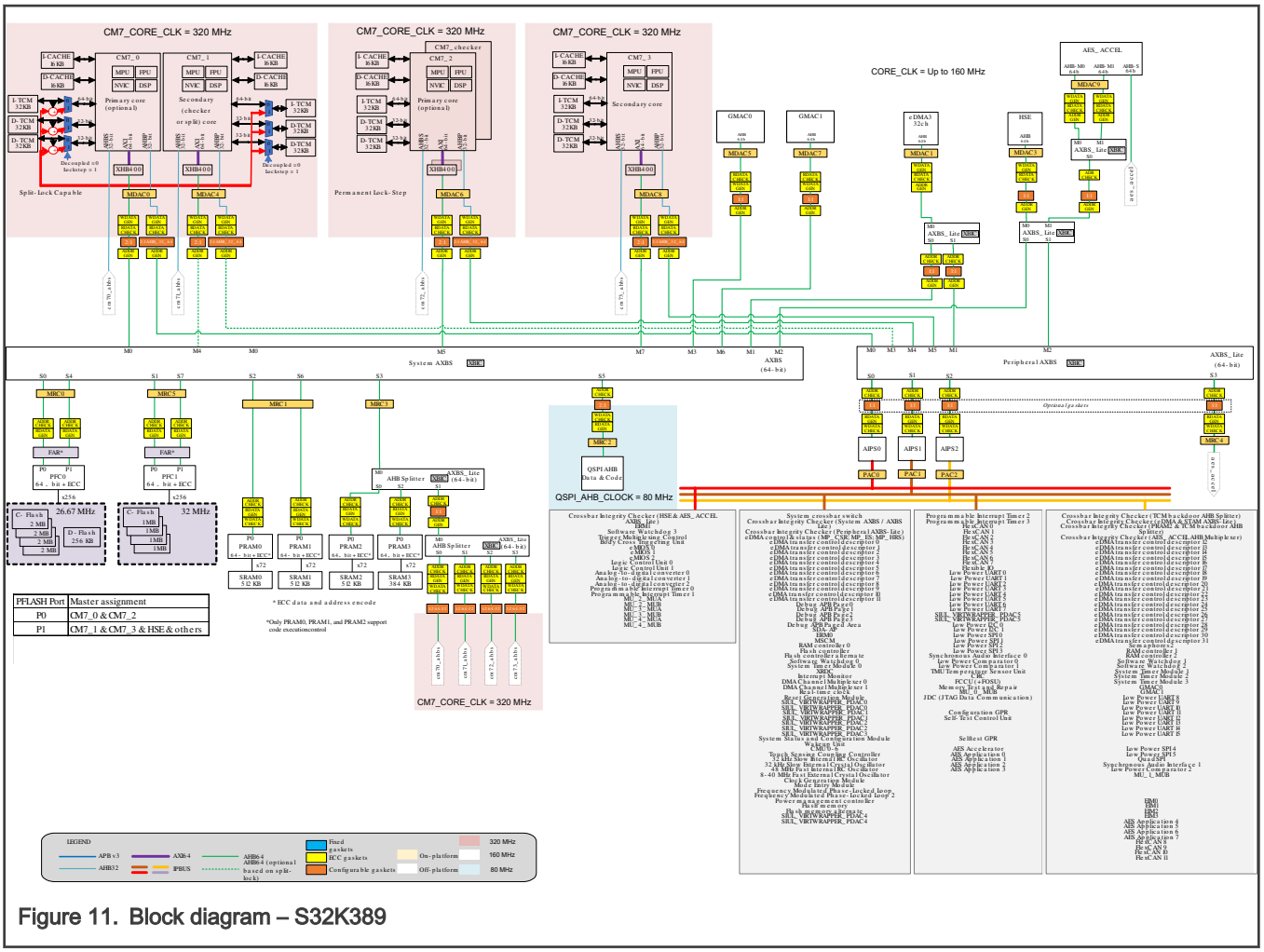


Figure 11. Block diagram – S32K389

2.5 Feature comparison

The following table compares some of the prominent features related to memory and package options of these chips from the S32K3xx family/product series:

- S32K310
- S32K311
- S32K312
- S32K322
- S32K341
- S32K342
- S32K314
- S32K324
- S32K344
- S32K328
- S32K338
- S32K348

- S32K356
- S32K358
- S32K388
- S32K389

Table 5. S32K3xx chip's feature comparison

Feature	Chip															
	S32K310	S32K311	S32K312	S32K322	S32K341	S32K342	S32K314	S32K324	S32K344	S32K328	S32K338	S32K348	S32K356	S32K358	S32K388	S32K389
Safety/ ASIL	B				D		B		D	B			D			
Program flash memory	512 KB	1 MB	2 MB		1 MB	2 MB	4 MB			8 MB			6 MB	8 MB		12 MB
Data flash memory (KB)	64		128							128						256
Total RAM (KB)	112KB (incl. 96KB TCM)	128KB (incl. 96KB TCM)	192KB (incl. 96KB TCM)	256KB (incl. 192KB TCM)			512KB (including 96KB TCM)	512KB (incl. 192KB TCM)		1152 KB (incl. 192KB TCM)	1152 KB (incl. 384KB TCM)	1152 KB (incl. 192KB TCM)	1152KB (incl. 384KB TCM)			2304 KB (incl. 384KB TCM)
Standby RAM	16 KB	32 KB								64 KB						
Security	HSE_B														HSE B + AES_ACCEL	
Core quantity	1 x M7			2 x M7	1 x M7 LS		1 x M7	2 x M7	1 x M7 LS	2 x M7	3 x M7	1 x M7 LS	1xM7 LS + 1xM7		1xM7 LS+3 xM7 or 2xM7 LS+1 xM7	1xM 7 LS + 3xM 7 or 2xM 7 LS +1x M7
Frequency (MHz)	120			160						240					320	
DMA channels	12			32												

Table continues on the next page...

Table 5. S32K3xx chip's feature comparison (continued)

Feature	Chip															
	S32K310	S32K311	S32K312	S32K322	S32K341	S32K342	S32K314	S32K324	S32K344	S32K328	S32K338	S32K348	S32K356	S32K358	S32K388	S32K389
ASIL-B DMIPS [1] [2]	277-387-813			738-1032-2168	—		369-516-1084	738-1032-2168	—	1108-1550-3254	1662-2325-4881	—	554-775-1627		739-1033-2169 [3] 2217-3099-6507 [4]	
ASIL-D DMIPS [1] [2]	—				369-516-1084		—		369-516-1084	—			554-775-1627		1478-2066-4338 [3] 739-1033-2169 [4]	
ASIL-B CoreMark score [1] [5]	634			1692	—		846	1692	—	2538	3807	—	1269		1692 [3] 5078 [4]	
ASIL-D CoreMark score [1] [5]	—				846		—		846	—			1269		3384 [3] 1692 [4]	
FlexCAN instances	3		6	4			6			8						12
EMAC instances	—			1						—						
GMAC instances	—									1					2	
SAI instances	—			2												
LPUART instances	4		8	4			16									
LPSPI instances	4						6									

Table continues on the next page...

Table 5. S32K3xx chip's feature comparison (continued)

Feature	Chip															
	S32K310	S32K311	S32K312	S32K322	S32K341	S32K342	S32K314	S32K324	S32K344	S32K328	S32K338	S32K348	S32K356	S32K358	S32K388	S32K389
I^2C instances	2															
FlexIO (incl. SENT support) channels	16		32													
QuadSPI instances	—			1[6]						1[7]					1[6]	
uSDHC instances	—									1					—	
ADC instances	2						3									
LPCMP instances	1		2				3									
PIT instances	2			3											4	
SWT instances	1		2	1			2	1	2	3	1	2			4	
STM instances	1			2						3					4	
LCU instances	2															
BCTU instances	1															
TRGMUX instances	1															

Table continues on the next page...

Table 5. S32K3xx chip's feature comparison (continued)

Feature	Chip														
	S32K310	S32K311	S32K312	S32K322	S32K341	S32K342	S32K314	S32K324	S32K344	S32K328	S32K338	S32K348	S32K356	S32K358	S32K388	S32K389
eMIOS instances	2						3									
RTC instances	1															
437-ball MAPBGA package	No															Yes
289-ball MAPBGA package	No									Yes						No
257-ball MAPBGA package	No						Yes			No						
172-HDQFP package	No		Yes							No						
172-HDQFP - EP package	No									Yes					No	
100-HDQFP package	Yes						No									
100-LQFP package	No		Yes		No		Yes			No						
48-pin LQFP package	Yes		No													

1. ASIL-B and ASIL-D performance is available simultaneously. ASIL-D performance can also be used for ASIL-B performance.
2. The first result abides by all of the "ground rules" out in Dhrystone documentation, the second permits inlining of functions, not just permitted C strings libraries, while the third additionally permits simultaneous ("multi-file") compilation. All are with the original (K and R) v2.1 of Dhrystone. Arm Compiler 6.17. See https://developer.arm.com/Processors/Cortex-M7 for details.
3. Core configuration is 2xLS + 1 independent core
4. Core configuration is 1xLS + 3 independent cores
5. Results depends on specific compiler version, contact NXP sales representative for more details.

6. 4-bit data width, SDR mode only
7. 8-bit data width, SDR and DDR mode

2.6 Glossary

AES	Advanced encryption standard
ASIL	Automotive safety integrity level. This is a risk classification scheme as defined by ISO 26262 for automotive standard.
AVB	Audio video bridging
CBC	Cipher block chaining
CCM	Counter with CBC MAC (Cipher block chaining message authentication code)
CMAC	Cipher-based message authentication code
CTR	Counter-based block cipher mode
DSP	Digital signal processor
DWT	Debug watchpoint and trace
ECB	Electronic code book
ECC	Elliptic curve cryptography/ Error code correction
ETM	Embedded trace macrocell
ETF	Embedded trace FIFO
EVITA	E-Safety vehicle intrusion protected applications
FPB	Flash patch and breakpoint unit
GCM	Galois/Counter mode, an encryption algorithm
GMAC	Galois message authentication code
GPIO	General purpose input/output
ITM	Instrumentation trace macrocell
ISOCAN-FD	ISO 11898-1 compliant CAN with FD (Flexible datarate)
LVD	Low voltage detection
NMI	Non-maskable interrupt
OFB	Output feedback based block cipher mode
PIL	Processor-in-the-loop
PLL	Phase locked loop oscillator
PWM	Pulse width modulation
RTD	Real-Time Drivers
SDK	Software development kit
SENT	Single edge nibble transmission
SWV	Serial wire viewer
SPFPU	Single precision floating point unit
SWO	Serial wire output

TPIU	Trace port interface unit
TSN	Time sensitive networking
uSDHC	Ultra Secure Digital Host Controller
WDOG	Windowed watchdog

Chapter 3
Memory Map

3.1 Introduction

This chip contains various memories and memory-mapped peripherals that are placed in a 32-bit contiguous memory space, and this chapter describes the memory and peripheral locations within that memory space.

For high-level chip memory map details, see the memory map file attached to this document.

3.2 SRAM memory map

The memory map file attached to this document provides a complete architectural address space definition for various sections that the RAM is partitioned into and across the S32K3xx product series. Based on the physical sizes of the memories and peripherals, the actual address regions used may be smaller. For details see chapter 'Memory and Memory Interfaces'.

3.3 Access-related details of the memory types used in this chip

The Cortex-M7 core can access these memories sequentially:

- ITCM
- DTCM
- I-cache
- D-cache

ITCM and DTCM can be accessed via 32-bit AHBS interface by any master, e.g., different Cortex-M7 cores, eDMA, etc to bootload instructions in ITCM. EMAC is another master that can access DTCM. See 'Block diagram' in the 'Introduction' chapter for details on the transaction path.

Access to SRAM beyond the RAM available on the chip terminates the bus cycle with an error followed by an appropriate response in the requesting bus master.

3.4 TCM as system memory

On multi–core device, all enabled core and non–core masters can use TCMs of the disabled core. In order to allow use of ITCM and DTCM of the disabled core as system memories the following steps must be executed by enabled core:

1. Write 1 to MC_ME's PRTN2_COFB1_CLKEN[REQ62] field for Cortex-M7_0, PRTN2_COFB1_CLKEN[REQ63] field for Cortex-M7_1, PRTN2_COFB2_CLKEN[REQ64] field for Cortex-M7_2, and PRTN2_COFB2_CLKEN[REQ65] field for Cortex-M7_3. This enables the Cortex-M7 core's TCM controller clock.
2. . Write 1 to DCM_GPR's DCMRWF4[CM7_0_CPUWAIT] field for Cortex-M7_0, DCMRWF4[CM7_1_CPUWAIT] field for Cortex-M7_1, DCMRWF4[CM7_2_CPUWAIT] field for Cortex-M7_2, and DCMRWF4[CM7_3_CPUWAIT] field for Cortex-M7_3. This configures the core operation in Wait mode.
3. Write 1 to MC_ME's PRTN0_CORE0_PCONF[CCE] field for Cortex-M7_0, PRTN0_CORE1_PCONF[CCE] field for Cortex-M7 _1, PRTN0_CORE4_PCONF[CCE] field for Cortex-M7_2, and PRTN0_CORE3_PCONF[CCE] field for Cortex-M7_3. This enables the Cortex-M7 core's clock.

Table 6. TCM modes of operation

Description	Control bit (Internal signal)	Cortex-M7 and TCM mode

Table continues on the next page...

Table 6. TCM modes of operation (continued)

	PRTN2_COFB*i*_CLKEN[REQ62+n] [1]	DCMRWF4[CM7_n_CPUWAIT]	PRTN0_COREn_PCONF[CCE]	CM7_n mode	CM7_n_TCM backdoor enabled
Application configurations	—	0	1	RUN	Yes
	0	1	1	WAIT	Yes
	1	1	1	WAIT	Yes
	—	—	0	Disabled	No

1. where *i* represent 1 for Cortex-M7_0/1 and 2 for Cortex-M7_2/3

3.5 Considerations related to TCM's implementation

You must first initialize TCM (ITCM and DTCM) and system RAMs by 64-bit writes before performing read accesses. The system RAM can be initialized using eDMA and core. The ITCM initialization can be performed only by core using either direct or back-door accesses. ITCM initialization via back-door can be done by using STM (Store Multiple) with even number of registers. STRD (Store Dual) instruction would not work.

The DTCM can be initialized also by 32-bit writes performed either using core's direct and back-door accesses using eDMA. These writes are required to set up the initial ECC code words after chip power-on reset.

Each Cortex-M7 core is equipped with a 32 KB ITCM and 64 KB DTCM with a zero wait-state access. In the lockstep operation, the checker core's TCM is added to the primary core.

See table 'Memory ECC initialization summary' in chapter 'Memory and Memory Interfaces' for details on memory ECC initialization.

3.6 Flash memory map

For details, see the memory map file attached to this document.

3.7 AIPS-Lite memory map

You can access the peripheral memory map via a crossbar slave port. The next table shows the three regions associated with peripheral space.

Table 7. Regions associated with peripheral space

Address of region	Region description
4000_0000h–401F_FFFFh	This 2048 KB region (AIPS_Lite_0) is partitioned into 128 spaces, each 16 KB in size, having 32 on-platform and 96 off-platform spaces. AIPS_Lite generates unique module enables for all the 32 on-platform spaces.
4020_0000h–403F_FFFFh	This 2048 KB region (AIPS_Lite_1) is partitioned into 128 spaces, each 16 KB in size, having 32 on-platform and 96 off-platform spaces. AIPS_Lite generates unique module enables for all the 32 on-platform spaces.
4040_0000h–405F_FFFFh	This 2048 KB region (AIPS_Lite_2) is partitioned into 128 spaces, each 16 KB in size, having 32 on-platform and 96 off-platform spaces. AIPS_Lite generates unique module enables for all the 32 on-platform spaces.

Modules that are disabled via their clock gate control fields in the MC_CGM registers disable the associated AIPS_Lite slots. Access to any address within an unimplemented or disabled peripheral bridge slot results in a transfer error termination.

Multiple instances of same peripherals are connected to different bridges on the interconnect. For details, see the memory map file attached to this document.

3.8 Serialization of memory operations

In particular cases, you must complete the process of writing to a peripheral before the subsequent action occurs. Examples of such situations include:

- Exiting an interrupt service routine
- Changing a mode
- Configuring a function

In these situations, you must perform a read-after-write sequence to achieve the required serialization of memory operations. The following table provides this sequence.

Table 8. Read-after-write sequence for serialization of memory operations

Step	Action
1	Write to the associated peripheral register.
2	Read the register to verify the write process.
3	Continue with the subsequent operations.

3.9 PPB memory map

PPB is a part of the defined Arm bus architecture and provides access to specific processor-local modules. You can access these modules only through the core, and not through other system masters.

Table 9. PPB memory map

Starting hex address	Ending hex address	Size (KB)	Module
E000_0000	E000_0FFF	4	ITM
E000_1000	E000_1FFF		DWT
E000_2000	E000_2FFF		FPB
E000_3000	E000_DFFF	44	—
E000_E000	E000_EFFF	4	SCS
E000_F000	E003_FFFF	196	Reserved
E004_0000	E004_0FFF	4	TPIU
E004_1000	E004_1FFF		ETM
E004_2000	E004_2FFF		CTI
E004_3000	E004_3FFF		—
E004_4000	E004_4FFF		
E004_5000	E004_5FFF		
E004_6000	E007_FFFF	232	
E008_0000	E008_0FFF	4	MCM
E008_1000	E008_1FFF		—

Table continues on the next page...

Table 9. PPB memory map (continued)

Starting hex address	Ending hex address	Size (KB)	Module
E008_2000	E008_2FFF		
E008_3000	E00F_EFFF	496	
E00F_F000	E00F_FFFF	4	Cortex-M7 PPB ROM table

3.10 Glossary

CTI	Cross trigger interface
DTCM	Data tightly coupled memory
D-cache	Data cache
DWT	Debug watchpoint and trace
ETM	Embedded trace macrocell
FPB	Flash patch and breakpoints
ITCM	Instruction tightly coupled memory
I-cache	Instruction cache
ITM	Instrumentation trace macrocells
PPB	Private peripheral bus
SCS	System control space
SRAM	Static random access memory
TPIU	Trace port interface unit

Chapter 4
Signal Multiplexing

4.1 Introduction

The signal multiplexing enables the sharing of single pad for multiple functions.

The signal multiplexing unit comprises control signals from SIUL2 and pad interface logic. The signal multiplexing unit consists of several individual sub-units, each handling the signal multiplexing of one pad.

The "SIUL2 Multiplexed Signal Configuration Register (MSCR)" controls the module specific pad settings (pull-up etc.) and the signal present on the external pin. See SIUL2_MSCR for the description of control signals. The pad attributes may vary depending on the pad type.

For the pad attributes of each pad type and their reset values per port, see the IOMUX file attached to this document. The pads specific to the packages and their multiplexing is also documented in the IOMUX file attached to this document.

NOTE

The input functions for the protocols which are not to be used should be appropriated configured as 'disabled low'/'disabled high' with appropriate SIUL2.IMCR configurations corresponding for that function.

4.2 Pad description

Following figure shows the basic representation of a GPIO pad.

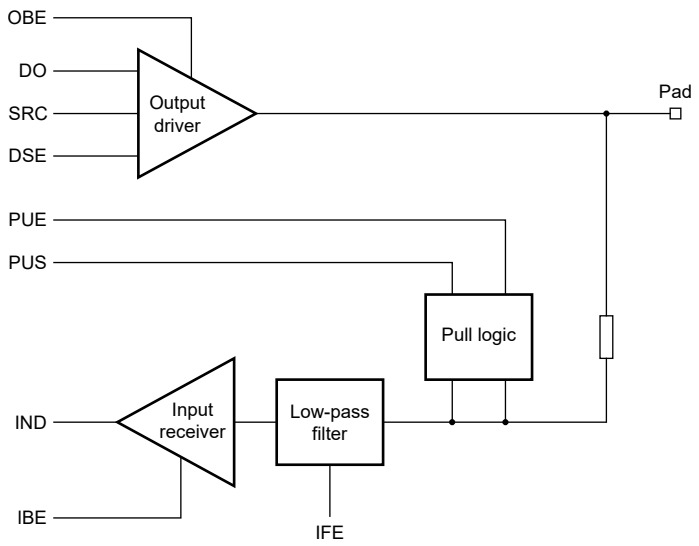


Figure 12. GPIO pad representation

Table 10. Pad Signal description

Signal name	Direction	Description
Pad	I/O	I/O to external world
DO	I	Data coming from the core into the pad
OBE	I	Enable output driver

Table continues on the next page...

Table 10. Pad Signal description (continued)

Signal name	Direction	Description
PUE	I	0: Disable internal pullup or pulldown resistor 1: Enable internal pullup or pulldown resistor
PUS	I	0: Enable internal pulldown resistor if pue is set 1: Enable internal pullup resistor if pue is set
IBE	I	Enable input receiver
IND	O	Data coming out of the pad into the core
SRC	I	Slew Rate Control
PKE	I	Pad keeping enable
IFE	I	Input filter enable
DSE	I	Drive Strength enable

Table 11. Input buffer enable

IBE	Pad	IND	Description
0	X	0	Input buffer disabled, ind gets low
1	0/1	0/1	Input buffer enabled, ind = pad

Table 12. Output buffer enable

OBE	DO	Pad	Description
0	X	Z	Output buffer disabled, pad hi-Z (If not configured as input)
1	0/1	0/1	Output buffer enable, pad = do

Table 13. Input filter enable

IFE	Description
1	Input filter enabled
0	Input filter disabled

Table 14. Pull up/Pull down

PUE	PUS	Pad	Description
0	X	-	Weak pull disabled. Pad retains previous state
1	0	0	Weak pull down enabled
1	1	1	Weak pull up enabled

Table 15. Slew rate control

SRC	Description
0	Slew rate enabled
1	Slew rate disabled

Table 16. Drive strength enable

DSE	Description
1	Drive strength supported
0	Drive strength not supported

Table 17. Pad keeping enable

PKE	Description
0	Pad keeping disabled
1	Pad keeping enabled

NOTE

The default state of GPIO pins on a reset event is high-Z. The high-Z state might settle to active high or active low at chip depending on the supply, temperature and other factors. Hence, it is recommended to use external pulls to ensure safe inactive state in event of a reset.

4.3 Functional description

The signal multiplexing architectural implementation is as shown in the following figure.

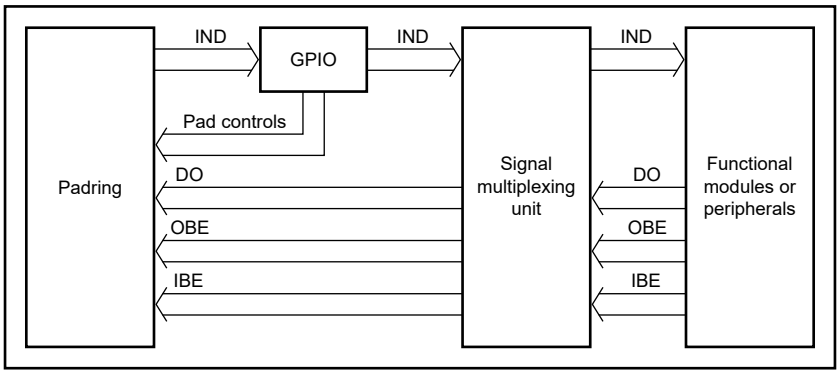


Figure 13. Signal Multiplexing

4.4 Signal Multiplexing sheet

IO Signal Description Input Multiplexing sheet(s) attached to the Reference Manual contains information on pins/balls of this device.

The 'IO Signal Table' and 'Input Muxing' tabs in the sheet correspond to the signal multiplexing information. The 'IO Signal Table' consists of all the pin muxing details and the 'Input Muxing' specifies the priority for the input muxing where an input path is driven by more than one pad.

4.4.1 IO Signal Table

Following is an example snippet of IO Signal Table. For selecting any functionality, the pad MSCR register (refer to SIUL2_MSCRn) needs to be configured accordingly.

Po	CR	SSS	Function	Module	Description	Direction	Pad Type	32K34I_257bga	32K34I_172bga	I/O Power Dom	Pad State During Destructive Rese	Pad State After Destructive Reset	Pad State During Functional Reset	Pad State After Functional Reset	Pad State After Selft	Pad State After FCCU Error	I/O Power Segme	MSCR									
																		[OBD:31]	[21:19]	[18:17]	[16:15]	[14:13]	[12:11]	[10:8]	[7:6]	[5:3]	[SSS:0]
PTA0	SIUL_MSCR0	0000_0000	GPIO[0]	SIUL		I/O	GPIO-STANDARD	A13	137	VDD_HV_A	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z		0	0	0	0	0	0	0	0	0	0000
PTA0		0001_0001	LPSPI4_PCS2	LPSPI4	Peripheral Chip Select 2	O																					
PTA0		0010_0010	eMIOS_0_CH[17]_Y	eMIOS_0	eMIOS Channel	O																					
PTA0		0011_0011	LCU0_OUT4	LCU0	LCU Output	O																					
PTA0		0100_0100	FXIO_D2	FXIO	FlexIO Bi-directional Shift/timer I/O	O																					
PTA0		0101_0101	eMIOS_1_CH[0]_X	eMIOS_1	eMIOS Channel	O																					
PTA0		0110_0110	LPSPI0_PCS7	LPSPI0	Peripheral Chip Select 7	O																					
PTA0		0111_0111	TRGMUX_OUT3	TRGMUX	Trigger Mux Output	O																					
PTA0	-	-	ADC0_S8	ADC0	ADC Standard Input	I																					
PTA0			CMP1_IN0	CMP1	Comparator Input Signal	I																					
PTA0	SIUL_IMCR528	0000_0001	EIRQ[0]	SIUL	External Interrupt	I																					
PTA0	SIUL_IMCR577	0000_0010	eMIOS_0_CH[17]_Y	eMIOS_0	eMIOS Channel	I																					
PTA0	SIUL_IMCR595	0000_0011	eMIOS_1_CH[0]_X	eMIOS_1	eMIOS Channel	I																					
PTA0	SIUL_IMCR649	0000_0100	FXIO_D2	FXIO	FlexIO Bi-directional Shift/timer I/O	I																					
PTA0	SIUL_IMCR747	0000_0101	LPSPI0_PCS7	LPSPI0	Peripheral Chip Select 7	I																					
PTA0	SIUL_IMCR765	0000_0110	LPSPI4_PCS2	LPSPI4	Peripheral Chip Select 2	I																					
PTA0	SIUL_IMCR35	0000_0011	LPUART0_CTS	LPUART0	Clear To Send (bar)	I																					

Figure 14. IO signal table snippet

The columns of the above figure are described below:

- Port: This field in IO Signal Table specifies the PAD names of the device.
- CR(Control Register): This field specifies the name of MSCR corresponding to the Port field. 'On this device, there are up to twelve port groups (PTA, PTB, PTC, PTD, PTE, PTF, PTG, PTH,PTI,PTJ and PTK) that are controlled by SIUL2_MSCRn registers. The below table shows the mapping of ports with respect to SIUL2_MSCRn registers.

Table 18. Port/MSCR mapping

Port	SIUL2_MSCRn index
PortA[0-31]	MSCR0 - MSCR31
PortB[0-31]	MSCR32 - MSCR63
PortC[0-31]	MSCR64 - MSCR95
PortD[0-31]	MSCR96 - MSCR127
PortE[0-31]	MSCR128 - MSCR159
PortF[0-31]	MSCR160 - MSCR191
PortG[0-31]	MSCR192 - MSCR223
PortH[0-31]	MSCR224 - MSCR255
PortI[0-31]	MSCR256 - MSCR287
PortJ[0-31]	MSCR288 - MSCR319
PortK[0-3]	MSCR320 - MSCR323

See SIUL2_MSCR/IMCR (See following Input Muxing section for details on IMCR) for description of MSCR/IMCR fields.

- SSS: This field specifies the ALT mode of operation as per MSCR[Mux_mode]. Not all pins support all pin muxing slots. Unimplemented pin muxing slots are reserved. The corresponding pin is configured in the following pin muxing slot as follows:
  - 0000: Alternative 0 (GPIO)
  - 0001: Alternative 1 (chip-specific)
  - 0010: Alternative 2 (chip-specific)
  - 0011: Alternative 3 (chip-specific)
  - 0100: Alternative 4 (chip-specific)
  - 0101: Alternative 5 (chip-specific)
  - 0110: Alternative 6 (chip-specific)
  - 0111: Alternative 7 (chip-specific)

- 1000: Alternative 8 (chip-specific)
- 1001: Alternative 9 (chip-specific)
- 1010: Alternative 10 (chip-specific)
- 1011: Alternative 11 (chip-specific)
- 1100: Alternative 12 (chip-specific)
- 1101: Alternative 13 (chip-specific)
- 1110: Alternative 14 (chip-specific)
- 1111: Alternative 15 (chip-specific)

NOTE

The analog functionalities are specified with '-' in this field.

- Function: This field specifies the functionality of the pad as per the corresponding ALT mode specified by SSS field.
- Module: The Module field contains the module which is governing the pad for the ALT mode.
- Description: This field mentions a short description of pad functionality.
- Direction: This field specifies the direction (Input, Output or Input/Output) of the pad for the concerned functionality.
- Pad Type: This field mentions the pad type of the corresponding pad.
  - GPIO-Standard:
    - Switching up to 10 MHz
    - High drive-strength not supported.
    - Slew-rate control not supported.
  - GPIO-Standard plus:
    - Switching up to 25 MHz
    - Supports high drive-strength.
    - Slew-rate control not supported.
  - GPIO-Medium:
    - Switching up to 50 MHz
    - Supports high drive-strength.
    - Supports slew-rate control.
  - GPIO-Fast:
    - Switching up to 120 MHz

      NOTE

      For S32K328, S32K338, S32K348, S32K358, S32K356, S32K388, and S32K389 switching frequency up to 125 MHz.

    - Supports high drive-strength.
    - Supports slew-rate control.
- The next columns specify the pin number in the supported packages for the device.
- I/O Power Domain: This field refers to the power domain of associated pad (VDD_HV_A/VDD_HV_B).
- Pad State during/after Reset: These fields represent the pad states during and post different device resets.

- MSCR: This field specifies the default MSCR value for corresponding pad. Refer SIUL2_MSCRn for description of MSCR fields.
- The next two columns specify the reset value and the configurable bit fields of MSCR corresponding to pad.

4.4.2 Input muxing table

As the same function can be multiplexed to several pads by configuring their respective IMCRs, there is priority input muxing. In case of same input being driven from multiple pads, the one with highest priority (1 being the highest) will drive the input. Following is a snippet of Input Muxing Table.

Destination Instance	Destination Function	CR Instance	Input CR#	Input SSS	Source Instance	Source Signal	S32K344_257bga	S32K344_172hdqfp_D
CAN0	CAN0_RX	SIUL	SIUL_IMCR512	0000_0000	-	disable low		
				0000_0001	IO_PAD	PTC2	T4	50
				0000_0010	IO_PAD	PTA6	M15	102
				0000_0011	IO_PAD	PTB0	P16	95
				0000_0100	IO_PAD	PTA28	N2	30
				0000_0101	IO_PAD	PTF21	K14	

Figure 15. Input muxing table snippet

The columns of the figure are briefly described below:

- Destination Instance: This field contains the instance name of the input path to where the signal will propagate from padring.
- Destination Function: This field mentions the function name of the input path.
- Input SSS: This field specifies the IMCR[Mux_mode] value corresponding to the pad specified in source signal column.
- Source Instance: This field specifies the source pad type. A blank is mentioned for the default source when no pad is driving the input path.
- Source Signal: This field mentions the pad name. A ‘disable low’/’disable high’ specifies the signal behavior when none of the pads are driving the input path.
- The next columns specify the pin number in the supported packages for the device.

4.4.3 MSCR/IMCR description/explanation

MSCR assignments

The Implemented SIUL2 Multiplexed Single Configuration Register details is provided in the I/O Signal Description Table attached as excel.

Example:

If user wants to configure PTB0 as CAN0RX and PTB1 as CAN0TX, then the following configuration can be used in SIUL2 registers:

```
//.CAN0.RX.(PTB0):
SIUL2.IMCR0.B.SSS.=.0b011;.//.Select.CAN0-RX
SIUL2.MSCR32.B.IBE.=.1;....//.Enable.the.input.buffer

//.CAN0.TX.(PTB1):
SIUL2.MSCR33.B.SSS.=.0b101;.//.Select.CAN0-TX
SIUL2.MSCR33.B.OBE.=.1;.//.Enable.the.output.buffer
```

MSCR bit fields correspond to pin/pad basis, these are independent of muxing implemented on that specific pin/pad. As an example, pad PTA31 has SSS, SMC, DSE, PUS, PKE, INV, IBE and OBE are implemented with the reset value 0 and SRC is implemented with reset value 1.

In the 'IO Signal Table' tab of the IOMUX file attached, the MSCR register bits shows ‘0’, ‘1’, and ‘-’ for state of all bits associated with different ports. ‘0’ or ’1’ represents the reset state and ‘-’ represents the bit is not supported on the respective port.

NOTE

The GPIOs 38 and 39 are used for direct connections and FXOSC.

4.4.4 Pinout diagrams

See IO Signal Description Input Multiplexing sheet(s) attached to the Reference Manual for pinout diagrams corresponding to available packages.

4.5 Pin States

The tables in the upcoming sections mention the state of pins of the device under various conditions/event like Functional Reset, Destructive Reset, Power Up condition, during Selftest Phase etc. The details of the Reset events can be found in the Reset overview Chapter. The Details of Selftest can be found in the STCU chapter.

NOTE

For S32K358, S32K356, S32K348, S32K338, and S32K328: Pad states of PTE13 are defined when this is not configured as VRC_CTRL pin.

4.5.1 Pin numbers

See IOMUX sheet for the pin numbers of the functions mentioned in tables given in following sections.

4.5.2 Power up

This table mentions the pin behavior on power up condition of the device.

Table 19. Power up

Pin function	POWER UP until SW comes up[1]
RESET_B	LOW
JTAG_TMS,JTAG_TCK,JTAG_TDI,JTAG_TDO as JTAG	PULLED Values: (TMS = HIGH, TCK = LOW, TDI =HIGH) TDO: HIGH Z
ETM_TRACE	HIGH Z
FCCU_ERR	HIGH Z
CLKOUT_STANDBY	HIGH Z
GPIOs	HIGH Z
EXTAL	HIGH Z till extal is connected
XTAL	HIGH Z till xtal is connected

1. The IO pad states are undefined until Supply rises sufficiently to enable the POR circuits.

4.5.3 Destructive Reset

This table mentions the Pin behavior when any Destructive Reset event is triggered.

Table 20. Destructive Reset

Pins	ON Destructive Reset until SW comes up
RESET_B	LOW
TMS,TCK,TDI,TDO as JTAG	PULLED Values: (TMS = HIGH TCK = LOW, TDI =HIGH) TDO: HIGH Z
ETM_TRACE (Only on 17trace Pins)	HIGH Z
FCCU_ERR	HIGH Z
CLKOUT_STANDBY	Clkout expose if CLKOUT_STANDBY is configured to be Enabled Else HIGH Z
GPIOs	HIGH Z

4.5.4 Functional Reset including Functional entry sequence

This table mentions the Pin behavior when any functional Reset event is triggered. Also, when the FCCU reaction is configured as Functional Reset , then also the same behavior is achieved.

Table 21. Functional Reset including Functional entry sequence

Pins	Functional Reset Entry	Reset until SW comes up
RESET_B	LOW	LOW
TMS,TCK,TDI,TDO as JTAG	TMS,TCK,TDI: PULLED Values if configured as JTAG (TMS = HIGH TCK = LOW, TDI =HIGH) HIGH Z if configured as GPIO TDO: HIGH Z	PULLED Values (TMS = HIGH TCK = LOW, TDI =HIGH) TDO: HIGH Z
ETM_TRACE	Trace if MDMAPCTL[DBGRSTSLOWPAD] or MDMAPCTL[DBGRSTFASTPAD] is configured to be Enabled Else HIGH Z	Trace if MDMAPCTL[DBGRSTSLOWPAD] or MDMAPCTL[DBGRSTFASTPAD] is configured to be Enabled Else HIGH Z
FCCU_ERR	ERR from FCCU if UTEST_MISC[FCCU_EOUT_DEDICATED] is configured to be Enabled Else HIGH Z	ERR from FCCU if UTEST_MISC[FCCU_EOUT_DEDICATED] is configured to be Enabled. Else HIGH Z
CLKOUT_STANDBY	Clkout expose if DCMRWP1[CLKOUT_STANDBY] is configured to be Enabled Else HIGH Z	Clkout expose if DCMRWP1[CLKOUT_STANDBY] is configured to be Enabled Else HIGH Z

Table continues on the next page...

Table 21. Functional Reset including Functional entry sequence (continued)

Pins	Functional Reset Entry	Reset until SW comes up
GPIOs	HIGH Z	HIGH Z

4.5.5 SELFTEST(MC_RGM.ERCTRL[ERASSERT] configured as 1)

Table 22. SELFTEST(MC_RGM.ERCTRL[ERASSERT] configured as 1)

Pins	Selftest	Reset until SW comes up
RESET_B	LOW if configured as Reset Pin High Z if configured as GPIO	LOW if configured as Reset Pin High Z if configured as GPIO
TMS,TCK,TDI,TDO as JTAG	PULLED Values if configured as JTAG (TMS = HIGH, TCK = LOW, TDI = HIGH) TDO: HIGH Z	PULLED Values (TMS = HIGH, TCK = LOW, TDI = HIGH) TDO: HIGH Z
ETM_TRACE	Trace if MDMAPCTL[DBGRSTSLOWPAD] or MDMAPCTL[DBGRSTFASTPAD] is configured to be Enabled. Else HIGH Z	Trace if MDMAPCTL[DBGRSTSLOWPAD] or MDMAPCTL[DBGRSTFASTPAD] is configured to be Enabled. Else HIGH Z
FCCU_ERR	FCCU Error State if UTEST_MISC[FCCU_EOUT_DEDICATED] is Enabled and DCMRWD2[EOUT_STAT_DUR_STEST] is Enabled Else HIGH Z	FCCU Error State if UTEST_MISC[FCCU_EOUT_DEDICATED] is Enabled and DCMRWD2[EOUT_STAT_DUR_STEST] is Enabled Else HIGH Z
CLKOUT_STANDBY	Clkout expose if DCMRWP1[CLKOUT_STANDBY] is configured to be Enabled Else HIGH Z	Clkout expose if DCMRWP1[CLKOUT_STANDBY] is configured to be Enabled Else HIGH Z
GPIOs	HIGH Z	HIGH Z

4.5.6 SELFTEST (RGM_ERCTRL[ERASSERT]=0 and RGM_FRBE[ST_DONE] =0)

Table 23. SELFTEST (RGM_ERCTRL[ERASSERT]=0 and RGM_FRBE[ST_DONE] =0)

Pins	Selftest	Reset after Selftest until SW comes up
RESET_B	Same state as before selftest	Same state as before selftest
TMS,TCK,TDI,TDO as JTAG	Same state as before selftest	PULLED Values (TMS = HIGH, TCK = LOW, TDI = HIGH) TDO: No change
ETM_TRACE	Same state as before selftest	Same state as before selftest

Table continues on the next page...

Table 23. SELFTEST (RGM_ERCTRL[ERASSERT]=0 and RGM_FRBE[ST_DONE] =0) (continued)

Pins	Selftest	Reset after Selftest until SW comes up
FCCU_ERR	FCCU Error State if UTEST_MISC[FCCU_EOUT_DEDICATED] is Enabled and DCMRWD2[EOUT_STAT_DUR_STEST] is Enabled Else No change	FCCU Error State if UTEST_MISC[FCCU_EOUT_DEDICATED] is Enabled and DCMRWD2[EOUT_STAT_DUR_STEST] is Enabled Else No change
CLKOUT_STANDBY	Same state as before selftest	Same state as before selftest
GPIOs	Same state as before selftest	Same state as before selftest

4.5.7 FCCU fault in RUN mode when the fault reaction is not Reset

NOTE

For any GPIO PAD If SMC=1 then the pad would retain it's state during the fault mode else if SMC=0 then the pad would become high Z during the fault mode.

Table 24. FCCU fault in RUN mode, fault reaction not Reset

Pins	ON FCCU Fault
RESET_B	If RESET_B is assigned as a dedicated PAD then no change in the state of the PAD else if RESET_B is configured as GPIO then it would behave as per the SMC configuration. • Case 1: If dcf_client_reset_pad_dedicated[reset pad dedicated] is configured as 1'b1, the pad functions as per the reset state machine. • Case 2: If dcf_client_reset_pad_dedicated[reset pad dedicated] is configured as 1'b0 and SIUL2.MSCR5[SMC] is configured as 1'b1, the pad functions as per the configured protocol status, the protocol being configured via SIUL2.MSCR5[SSS]. • Case 3: If dcf_client_reset_pad_dedicated[reset pad dedicated] is configured as 1'b0 and SIUL2.MSCR5[SMC] is configured as 1'b0, the pin gets tristated (high-Z).
TMS, TCK, TDI, TDO as JTAG	If the PAD is configured as a JTAG PAD then no change and the pad would continue performing the operation else if it is a GPIO PAD then it would behave as per the SMC configuration. • Case 1: If the pad is configured for JTAG mode and SMC bit of SIUL2's MSCR register of the corresponding pin is configured as 1'b1, the pad functions as per the JTAGC state machine or protocol. • Case 2: If the pad is configured for non-JTAG mode and SMC bit of SIUL2's MSCR register of the corresponding pin is configured as 1'b0, the pins get tristated (high-Z).
ETM_TRACE	If the PAD is configured as a dedicated PAD then no change and the pad would continue performing the operation else if it is a GPIO PAD then it would behave as per the SMC configuration. • Case 1: If the pad is configured for ETM_TRACE functionality and SMC bit of SIUL2's MSCR register of the corresponding pin is configured as 1'b1, the pad functions as per the ETM trace control logic. • Case 2: If the pad is configured for non-ETM_TRACE mode and SMC bit of SIUL2's MSCR register of the corresponding pin is configured as 1'b0, the pins get tristated (high-Z).

Table continues on the next page...

Table 24. FCCU fault in RUN mode, fault reaction not Reset (continued)

Pins	ON FCCU Fault
FCCU_ERR	If the PAD is configured as a dedicated PAD by programming the UTEST_MISC[FCCU_EOUT_DEDICATED] as well as Eout indication is enabled then it would be indicating error state else if it is a GPIO PAD then it would behave as per the SMC configuration. • Case 1: If dcf_client_utest_misc[FCCU_EOUT_DEDICATED] is configured as 1'b1, the FCCU_ERR pins indicate the error state. • Case 2: If dcf_client_utest_misc[FCCU_EOUT_DEDICATED] is configured as 1'b0 and SMC bit of the SIUL2's corresponding MSCR register is configured as 1'b1, the pin retains its state. • Case 3: If dcf_client_utest_misc[FCCU_EOUT_DEDICATED] is configured as 1'b0 and SIUL2.MSCR5[SMC] is configured as 1'b0, the pin gets tristated (high-Z).
CLKOUT_STANDBY	If the PAD is configured as a dedicated PAD then no change and the pad would continue performing the operation else if it is a GPIO PAD then it would behave as per the SMC configuration. • Case 1: If the pad is configured for CLKOUT_STANDBY functionality and SMC bit of SIUL2's MSCR register of the corresponding pin is configured as 1'b1, the pin continues functioning as the CLKOUT_STANDBY pin. • Case 2: If the SMC bit of SIUL2's MSCR register of the corresponding pin is configured as 1'b0, the pin gets tristated (high-Z).
GPIOs	If the PAD is configured as a dedicated PAD then no change and the pad would continue performing the operation else if it is a GPIO PAD then it would behave as per the SMC configuration. • Case 1: If the SMC bit of SIUL2's MSCR register of the corresponding pin is configured as 1'b1, the pin continues its operation. • Case 2: If the SMC bit of SIUL2's MSCR register of the corresponding pin is configured as 1'b0, the pins get tristated (high-Z).

4.5.8 STANDBY (DCMRWF1[STANDBY_IO_CONFIG] must be configured as 1)

- Please refer to Padkeeping Section for steps to configure the various pad states during standby

Table 25. STANDBY

Pins	Standby until SW comes up
RESET_B	Alive / HighZ /LOW/HIGH (Depends on Software configuration before entering STANDBY)
TMS,TCK,TDI,TDO as JTAG	HighZ /LOW/HIGH (Depends on Software configuration before entering STANDBY)
ETM_TRACE	HighZ /LOW/HIGH (Depends on Software configuration before entering STANDBY)
FCUU_ERR	HighZ /LOW/HIGH (Depends on Software configuration before entering STANDBY)
CLKOUT_STANDBY	Alive / HighZ /LOW/HIGH (Depends on Software configuration before entering STANDBY)
GPIOs	HighZ /LOW/HIGH (Depends on Software configuration before entering STANDBY)

4.6 Glossary

DSE	Drive strength enable
EXTAL	External crystal input
GPIO	General purpose input/output
IBE	Input buffer enable
INV	Invert enable
JTAG_TMS	JTAG test mode select
JTAG_TCK	JTAG test clock input
JTAG_TDI	JTAG test data input
JTAG_TDO	JTAG test data output
OBE	Output buffer enable
PUS	Pullup and pulldown select
PKE	Pad keeping enable
SMC	Safe mode control
SRC	Slew rate control
SSS	Source signal select
XTAL	External crystal output

Chapter 5
Cortex-M7 Overview

5.1 Introduction

Cortex-M7 is a high-performance embedded processor intended for deeply embedded applications that require fast interrupt response features. The configuration of the processor is based on little-endian format, and you must compile the execution testbench tests in this format too.

Table 26. Cortex-M7 instances

Instances	S32K388/ S32K389	S32K338	S32K358/ S32K356	S32K328/ S32K324/S32K322	S32K348/S32K344/S32K342/ S32K341/S32K314/S32K312/ S32K311/S32K310
CM7_0	Yes	Yes	Yes	Yes	Yes
CM7_1	Yes	Yes	No	Yes	No
CM7_2	Yes	Yes	Yes	No	No
CM7_3	Yes	No	No	No	No

5.1.1 Features

Cortex-M7 provides:

- Low interrupt latency
- Low-cost debug
- Backwards compatibility with existing Cortex-M profile processors
- In-order superscalar pipeline
- Dual-issue support for load/load and load/store instruction pairs to multiple memory interfaces
- An MPU that you could configure to protect regions of memory
- An NVIC
- A debug and trace unit (CoreSight components)
- Floating-point arithmetic functionality, with support for single-precision arithmetic
- The ability to perform speculative load from any normal type memory space through its AXIM bus, if D-cache is enabled
- Several memory interfaces that include:
  - Harvard architecture-based instruction and data caches, and an AXIM interface
  - A dedicated low-latency AHBP interface
  - A 64-bit AXI AMBA4 memory interface with a 16 KB instruction cache and a 16 KB data cache for efficient access to external resources. The instruction and data caches are ECC protected.
  - A 32-bit AHBS for interfacing with slaves such as DMA
  - 64-bit and 32-bit memory interfaces for the connection to local Tightly Coupled Memories called ITCM and DTCM. For details see Core configuration

5.1.2 Related information

For detailed information on:

- Cortex-M7 processor, see Arm Cortex-M7 Processor Technical Reference Manual.
- Cortex-M7 peripherals and control, see Arm Cortex-M7 Devices Generic User Guide.
- System memory map, see the memory map file attached to this document.

5.1.3 Buses, interconnects, and interfaces

This table discusses Cortex-M7 buses and their associated interconnects and interfaces.

Table 27. Buses and associated information

Bus name	Description
AXIM	Using the XHB400 module, this bus is first translated to the AHB-Lite bus that interfaces with the AXBS crossbar switch providing high-bandwidth access to on-chip memories and peripherals.
AHBP	This bus connects to the AXBS crossbar switch providing high-bandwidth access to on-chip peripherals.
PPB	This bus provides access to these modules: • Arm modules such as NVIC, ETM, ITM, DWT, and ROM tables • Miscellaneous Control Module (MCM)

NOTE

S32K3xx AHBP bus is enabled after reset. Therefore, accesses of all cores to on-chip peripherals are performed exclusively through this bus.

5.1.4 Core configuration

This table describes Cortex-M7 parameter settings.

Table 28. Core configuration

Parameter	Configuration[1]
FPU	Single precision
DSP extension instructions	• Single cycle 16/32-bit MAC • Single cycle dual 16-bit MAC • 8/16-bit SIMD arithmetic • Hardware divide (2-12 cycles)
Armv8-M security extensions	Not implemented
I-cache	Implemented
D-cache	Implemented
Caches ECC	Implemented
On core MPU region	16
Number of IRQs	240
IRQ priority width configuration	4 (16 interrupt priority levels)
Debug (breakpoint/watchpoint)	Full comparator set: 4 DWT and 8 FPB comparators

Table continues on the next page...

Table 28. Core configuration (continued)

Parameter	Configuration[1]
Internal trace support	ITM and DWT trace functionality implemented
ETM support	Instruction and data ETM interface implemented
CTI	Implemented
WIC support	Not implemented
Dual-redundant core (lock-step) CPU functionality	Not implemented[2]
RAR	All asynchronously reset
I-cache size	Implemented [3]
D-cache size	Implemented [3]
ITCM	• All chips except S32K388/S32K389: 32 KB for CM7_0/1, 64 KB for CM7_2 • S32K388/S32K389: 32 KB for all cores
DTCM0	• All chips except S32K388/S32K389: 32 KB for CM7_0/1, 64 KB for CM7_2 • S32K388/S32K389: 32 KB for all cores
DTCM1	• All chips except S32K388/S32K389: 32 KB for CM7_0/1, 64 KB for CM7_2 • S32K388/S32K389: 32 KB for all cores

1. Armv7-M (Harvard architecture), six-stage superscalar plus branch prediction
2. Two different Cortex-M7 cores are used physically in the S32K342, S32K344, S32K348, S32K358, S32K356, S32K388, and S32K389 chips (dual Cortex-M7 lock-step parts)
3. See chapter 'Memory and Memory Interfaces' for details

5.2 Speculative accesses

The Arm Cortex M7 processor can issue speculative read accesses that may access any location within the complete memory address range. This behavior can be controlled, but not disabled. Corresponding effects must be considered for a proper operation of your system.

Speculative accesses do not cause any processor faults. The processor is aware whether an access is speculative, and ignores any error response signaled by the system due to the speculative access. However, the system that is integrating the processor cannot distinguish speculative accesses from non-speculative accesses.

Addresses used by speculative accesses are not validated against the memory map of the device, and may attempt to also access non-existing memory regions or hardware elements having side effects. For details about corresponding behavior of the Arm Cortex M7 processor see Related information. Important processing aspects are listed in section “Memory Model” within the Generic User Guide.

Speculative accesses can result in improved performance when the related memory regions are properly characterized. It is imperative to properly setup the attributes within the Memory Protection Unit (MPU) to avoid any unwanted impact of a speculative access. Examples for possible, corresponding issues are usually the result of side effects unknown to the processor:

- Speculative access to an uninitialized RAM memory location that causes a double bit error {the related fault processing by the FCCU is performed independently from the processor}.
- Speculative access to a peripheral that causes an unwanted operation; for example, a FIFO read {the corresponding data may be removed from the FIFO without being processed}.

- Speculative access to a peripheral that is not clocked, powered down, or cannot respond {the access may not be terminated, resulting an access that is stalled, system blockage}.
- Speculative access to an address range that causes an unexpected error being reported; for example, a read-while-write error of the embedded flash (indicated by setting MCRS[RWE]) during a flash erase or program operation {which may hide real errors}.

Speculative accesses can be controlled by a proper assignment of memory regions within the MPU:

- Speculative instruction fetches are never made to memory addresses in an Execute Never region.
- Speculative data reads are never made to memory addresses marked as non-accessible in the MPU.
- Speculative cache line-fills are never made to non-cacheable memory addresses.
- Speculative data reads and speculative cache line-fills are never made to memory addresses in a region having a Device or Strongly-ordered attribute.
- Speculative reads are never made on the AHBP interface.
- Speculative writes are never made.

Memory regions mapped to a TCM are always treated as Normal Memory (equivalent to the MPU attribute) and are therefore always subject to speculation. Related issues can be avoided by properly initializing any TCM memory before a corresponding access may occur and by ensuring that corresponding faults are not being processed before the appropriate management is in place.

When no speculative accesses should be initiated to a memory region, it is recommended to set all of the following attributes within the MPU for this region: Device or Strongly-ordered, and Execute Never. These attributes are often also used for address ranges associated with peripherals.

Unwanted processing of side effects caused by a speculative access can also be inhibited by disabling the related events while a speculative access may occur. As an example, the FCCU can be enabled after the ECC of the RAM memories has been initialized. As a second example, read accesses to a Flash block can be inhibited by configuring an MPU region while it is being erased.

5.3 Debug facilities

This chip has extensive debug capabilities such as run control and tracing. It includes the standard Arm debug port that supports the JTAG and SWD interfaces.

5.4 Vector fetch behavior on Cortex-M7

In Cortex-M7, the vector fetches are looked up into the I-Cache. if the vector table is located in a region of memory that is cacheable, any load or store to the vector must be treated as self-modifying code and cache maintenance instructions should be used to synchronize the updates to the data and instruction caches. The Cortex-M7 Device Generic User Guide chapter 'Cache maintenance design hints and tips' specifies a recommendation for synchronization of the D-Cache and I-Cache.

If cache maintenance is to be avoided each time when the vector table gets updated, then the vector table must be allocated in the ITCM or DTCM, as those are non-cacheable regions. Alternatively, the I-Cache must be enabled after the vector table has been initialized.

5.5 TCM retry

TCM retry is disabled and software should disable the TCM retry bit at startup by programming the relevant core's CM7_ITCMCR[RETEN] and CM7_DTCMR[RETEN] fields to disable state.

5.6 Glossary

AHBP AHB-lite peripheral

AHBS AHB-slave port

AXIM	Advanced extensible interface master
DWT	Data watchpoint and trace unit
ETM	Embedded trace macrocell
FPU	Floating point unit
FPB	Flash patch and breakpoint
ITM	Instrumentation trace macrocell
MAC	Multiplier accumulator (refers to a multiplier accumulator unit as well as multiplier accumulator operation)
NVIC	Nested vectored interrupt controller
RCCU	Redundancy control checking unit
RAR	Reset-all-registers
SIMD	Single instruction multiple data

Chapter 6
Miscellaneous Control Module (MCM)

6.1 Chip-specific MCM information

6.1.1 MCM instances and configuration

This chip supports up to four instances of MCM:

- MCM_0
- MCM_1
- MCM_2
- MCM_3

NOTE

For S32K358, S32K356, S32K338, S32K388, and S32K389 the reset value for CM7_2 is:

- LMEM_DESC_0: 8706_0000h
- LMEM_DESC_1-2 : 8704_2000h

Table 29. MCM instances

Instances	S32K388/ S32K389	S32K358/ S32K356/ S32K348/ S32K338/ S32K328	S32K322/S32K324/S32K344/S32K342	S32K312/S32K311/ S32K310/S32K314
MCM_0	Yes	Yes	Yes	Yes
MCM_1	Yes	Yes	Yes	No
MCM_2	Yes	Yes	No	No
MCM_3	Yes	No	No	No

Table 30. Memories for all chips except S32K358/S32K356/S32K348/S32K338/S32K328/S32K388/S32K389

Memory	S32K311/S32K310/S32K312/S32K314 (Single core)	S32K322/S32K324 (Dual core)	S32K342/S32K344 (Lockstep mode)
Icache	8 KB	8 KB (per core)	8 KB
Dcache	8 KB	8 KB (per core)	8 KB
ITCM	32 KB	32 KB (per core)	64 KB
DTCM	64 KB	64 KB (per core)	128 KB

Table 31. Memories for S32K358/S32K356/S32K348/S32K338/S32K328/S32K388/S32K389

Memory	Dual core CM7_0 and CM7_1	Single core CM7_2 [1]	CM7_0 and CM7_1 (Lockstep mode	CM7_2 (Lockstep mode) [2]	CM7_3[2]
Icache	16 KB (per core)	16 KB	16 KB	16 KB	16 KB

Table continues on the next page...

Table 31. Memories for S32K358/S32K356/S32K348/S32K338/S32K328/S32K388/S32K389 (continued)

Memory	Dual core CM7_0 and CM7_1	Single core CM7_2 [1]	CM7_0 and CM7_1 (Lockstep mode	CM7_2 (Lockstep mode) [2]	CM7_3[2]
Dcache	16 KB (per core)	16 KB	16 KB	16 KB	16 KB
ITCM	32 KB (per core)	64 KB	64 KB	32 KB	32 KB
DTCM	64 KB (per core)	128 KB	128 KB	64 KB	64 KB

1. Available only in S32K358/S32K356
2. Available only in S32K388/S32K389

6.2 Overview

MCM provides miscellaneous control functions and contains local memory descriptors for the Cortex-M7 core. For more information about core-related registers, see the Cortex-M7 core overview chapter.

NOTE

The terminology in this chapter has been updated to align with Arm's AMBA AHB Protocol Specification, as shown in the table below.

Table 32. Updated terms

Updated term	Deprecated term
Manager	Master
Subordinate	Slave

6.2.1 Features

The MCM includes the following features:

- Program-visible information on the platform configuration and revision
- Floating Point Exception monitor and interrupt control
- Local memory descriptors:
  - ITCM
  - D0TCM
  - D1TCM
  - I-cache
  - D-cache

6.3 Functional description

6.3.1 Interrupts

MCM generates an interrupt if any of the following are true:

- FPU input denormal interrupt is enabled (FIDCE) and an input is denormalized (FIDC).
- FPU inexact interrupt is enabled (FIXCE) and a number is inexact (FIXC).
- FPU underflow interrupt is enabled (FUFCE) and an underflow occurs (FUFC).
- FPU overflow interrupt is enabled (FOFCE) and an overflow occurs (FOFC).

- FPU divide-by-zero interrupt is enabled (FDZCE) and a divide-by-zero occurs (FDZC).
- FPU invalid operation interrupt is enabled (FIOCE) and an invalid operation occurs (FIOC).
- Write abort interrupt is enabled (WABE) and a write abort occurs (CORTEX-M7 WABORTS INDICATOR).

Determining interrupt source

To determine the exact source of the interrupt for Cortex-M7 core, qualify the interrupt status flags with the corresponding interrupt enable fields.

- MCM_ISCR[31:16] && MCM_ISCR[15:0]
- Search the result for asserted flags, which indicate the exact interrupt sources.

6.4 Memory map and register descriptions

The memory map and register descriptions below describe the registers using byte addresses.

NOTE

The following actions result in bus errors:

- Writing to read-only registers at 0x0.
- Reading from or writing to an address from offset 480h and above.
- Accessing any MCM register while in User mode. These registers are only accessible while in Supervisor mode.

6.4.1 MCM register descriptions

6.4.1.1 MCM memory map

MCM_0_CM7 base address: E008_0000h

MCM_1_CM7 base address: E008_0000h

MCM_2_CM7 base address: E008_0000h

MCM_3_CM7 base address: E008_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Chip-Defined Platform Revision (PLREV)	16	R	0000h
2h	Processor Core Type (PCT)	16	R	AC70h
Ch	Core Platform Control (CPCR)	32	RW	0000_0200h
10h	Interrupt Status and Control (ISCR)	32	RW	0000_0000h
30h	Process Identifier (PID)	8	RW	00h
400h	Local Memory Descriptor 0 (LMEM_DESC_0)	32	R	8606_0000h
404h - 408h	Local Memory Descriptor a (LMEM_DESC_1 - LMEM_DESC_2)	32	R	8604_2000h
40Ch	Local Memory Descriptor 3 (LMEM_DESC_3)	32	R	8526_4000h
410h	Local Memory Descriptor 4 (LMEM_DESC_4)	32	R	8544_6000h

6.4.1.2 Chip-Defined Platform Revision (PLREV)

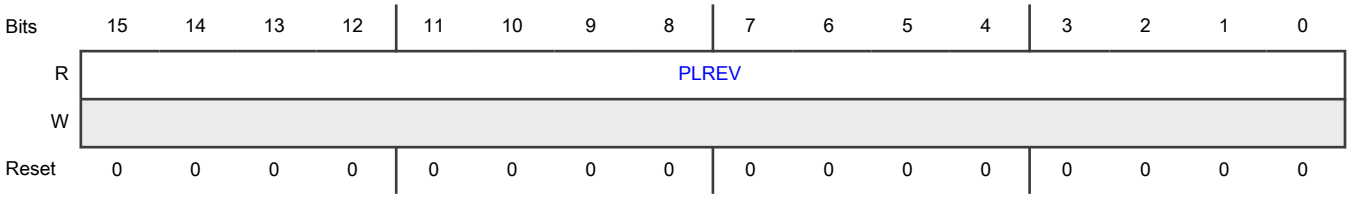
Offset

Register	Offset
PLREV	0h

Function

Specifies a chip-defined platform revision number. A platform input signal defines the state of this register; it can only be read from the IPS programming model. Any attempted write is ignored.

Diagram



Fields

Field	Function
15-0 PLREV	Defines the software-visible revision number, specified by a platform input signal.

6.4.1.3 Processor Core Type (PCT)

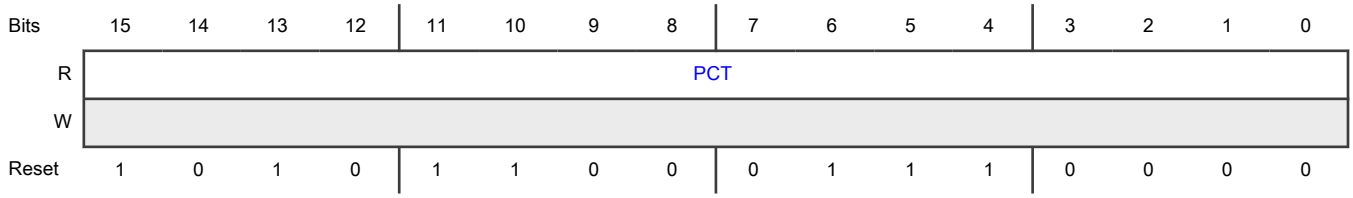
Offset

Register	Offset
PCT	2h

Function

Specifies the architecture of the processor core within the platform on the chip. A module input signal defines the state of this register, which can only be read from the IPS programming model. Any attempted write is ignored.

Diagram



Fields

Field	Function
15-0 PCT	Core Complex Identifier Identifies the core complex. This MCM design supports the Arm Cortex M7 core. 1010_1100_0111_0000b - Arm Cortex-M7

6.4.1.4 Core Platform Control (CPCR)

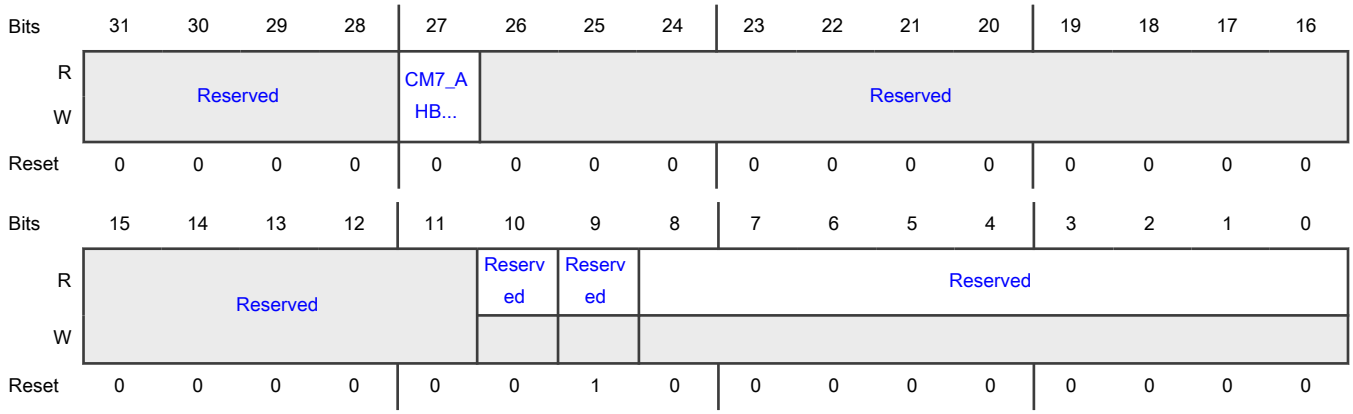
Offset

Register	Offset
CPCR	Ch

Function

Defines the arbitration and protection schemes for the two system RAM arrays.

Diagram



Fields

Field	Function
31-28	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
27 CM7_AHBSPRI	AHB Subordinate Priority Indicates the access priority on the AHBS port of the Cortex-M7 core. **NOTE** This setting has no effect unless enabled by AHBSCR[CTL][1] of the Cortex-M7 core. 0b - Uses a round-robin arbitration scheme 1b - AHB-subordinate access has priority over a core access
26-11 —	Reserved
10 —	Reserved
9 —	Reserved
8-0 —	Reserved

1. For more information see Cortex-M7 documentation: Arm Cortex-M7 Processor Technical Reference Manual at www.arm.com.

6.4.1.5 Interrupt Status and Control (ISCR)

Offset

Register	Offset
ISCR	10h

Function

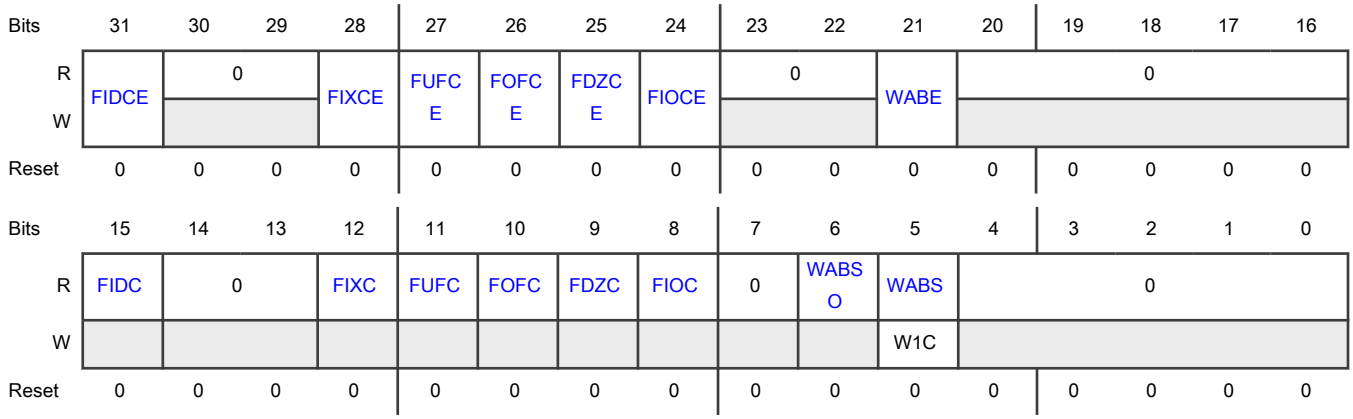
Defines the configuration and reports the status for a number of core-related interrupt exception conditions. It includes:

- Enable and status fields associated with the core's floating-point exceptions
- Bus errors associated with the core's cache write buffer

The individual event indicators are first qualified with their exception enables, and then logically summed to form an interrupt request sent to the core's NVIC.

Bits 15-8 are read-only indicator flags based on the processor's FPSCR register. Attempted writes to these fields are ignored. When these flags are 1, they retain this value until software clears the corresponding FPSCR field. For more information see Cortex-M7 documentation: Arm Cortex-M7 Processor Technical Reference Manual at www.arm.com.

Diagram



Fields

Field	Function
31 FIDCE	FPU Input Denormal Interrupt Enable 0b - Disable 1b - Enable
30-29 —	Reserved
28 FIXCE	FPU Inexact Interrupt Enable 0b - Disable 1b - Enable
27 FUFCE	FPU Underflow Interrupt Enable 0b - Disable 1b - Enable
26 FOFCE	FPU Overflow Interrupt Enable 0b - Disable 1b - Enable
25 FDZCE	FPU Divide-by-Zero Interrupt Enable 0b - Disable 1b - Enable
24 FIOCE	FPU Invalid Operation Interrupt Enable 0b - Disable 1b - Enable
23-22	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
21 WABE	TCM Write Abort Interrupt Enable 0b - Disable 1b - Enable
20-16 —	Reserved
15 FIDC	FPU Input Denormal Interrupt Status Indicates that an input denormalized number has been detected in the processor's FPU. This field is a copy of the core's FPSCR[IDC] field. When this field is 1, it retains this value until software clears the FPSCR[IDC] field. 0b - No interrupt 1b - Interrupt occurred
14-13 —	Reserved
12 FIXC	FPU Inexact Interrupt Status Indicates that an inexact number has been detected in the processor's FPU. This field is a copy of the core's FPSCR[IXC] field. When this field is 1, it retains this value until software clears the FPSCR[IXC] field. 0b - No interrupt 1b - Interrupt occurred
11 FUFC	FPU Underflow Interrupt Status Indicates that an underflow has been detected in the processor's FPU. This field is a copy of the core's FPSCR[UFC] field. When this field is 1, it retains this value until software clears the FPSCR[UFC] field. 0b - No interrupt 1b - Interrupt occurred
10 FOFC	FPU Overflow Interrupt Status Indicates that an overflow has been detected in the processor's FPU. This field is a copy of the core's FPSCR[OFC] field. When this field is 1, it retains this value until software clears the FPSCR[OFC] field. 0b - No interrupt 1b - Interrupt occurred
9 FDZC	FPU Divide-by-Zero Interrupt Status Indicates that a divide-by-zero operation has been detected in the processor's FPU. This field is a copy of the core's FPSCR[DZC] field. When this field is 1, it retains this value until software clears the FPSCR[DZC] field.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No interrupt 1b - Interrupt occurred
8 FIOC	FPU Invalid Operation Interrupt Status Indicates that an illegal operation has been detected in the processor's FPU. This field is a copy of the core's FPSCR[IOC] field. When this field is 1, it retains this value until software clears the FPSCR[IOC] field. 0b - No interrupt 1b - Interrupt occurred
7 —	Reserved
6 WABSO	Write Abort on Subordinate Overrun Indicates whether an abort overrun has occurred. The overrun conditions are reported only if WABE=1. 0b - No write abort overrun 1b - Write abort overrun occurred
5 WABS	Write Abort on Subordinate Indicates when a write abort has occurred on the AHBS interface. 0b - No write abort occurred on AHBS interface 1b - Write abort occurred on AHBS interface
4-0 —	Reserved

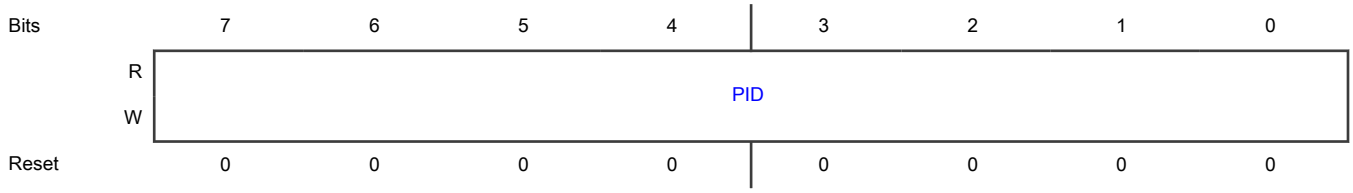
6.4.1.6 Process Identifier (PID)

Offset

Register	Offset
PID	30h

Function
Contains the CPU process ID.

Diagram



Fields

Field	Function
7-0 PID	Process Identifier Identifies the CPU process.

6.4.1.7 Local Memory Descriptor 0 (LMEM_DESC_0)

Offset

Register	Offset
LMEM_DESC_0	400h

Function

NOTE

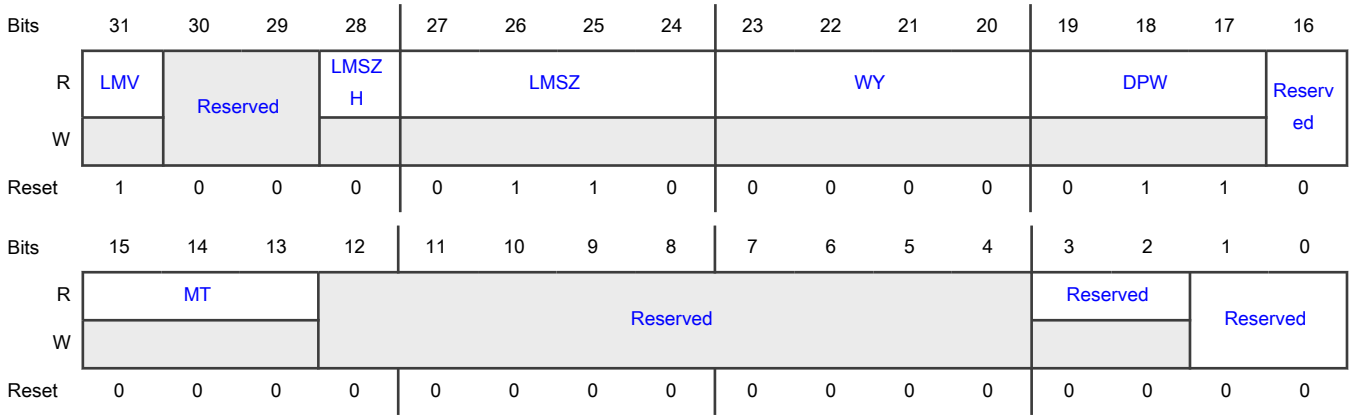
The DESC_a registers map to the LMEMs in this way:

- DESC_0: ITCM
- DESC_1: D0TCM
- DESC_2: D1TCM
- DESC_3: I-cache
- DESC_4: D-cache

NOTE

You can read and write to the reserved fields (instead of read as zero and write ignored). Writing to any of these fields has no functional impact.

Diagram



Fields

Field	Function
31 LMV	Local Memory Valid Defines the validity (presence) of the local memory. 0b - LMEM*n* not present 1b - LMEM*n* present
30-29 —	Reserved
28 LMSZH	LMEM Size Hole Used for local memories that are not fully populated (that is, local memories that include a memory "hole" in the upper 25 % of the address range). 0b - LMEM*n* is a power-of-2 capacity 1b - LMEM*n* is not a power-of-2, with capacity of 0.75 × LMSZ
27-24 LMSZ	Local Memory Size Indicates the local memory size. 0000b - 0 KB 0001b - 1 KB 0010b - 2 KB 0011b - 4 KB 0100b - 8 KB 0101b - 16 KB 0110b - 32 KB 0111b - 64 KB 1000b - 128 KB

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1001b - 256 KB 1010b - 512 KB 1011b - 1024 KB 1100b - 2048 KB 1101b - 4096 KB 1110b - 8192 KB 1111b - 16384 KB
23-20 WY	Level 1 Cache Ways Defines the level 1 cache ways. 0000b - No cache 0010b - 2-way set associative 0100b - 4-way set associative
19-17 DPW	Data Path Width Defines the LMEM*n* data path width, which is the width of the local memory. 000b-001b - Reserved 010b - 32 bits 011b - 64 bits 100b-111b - Reserved
16 —	Reserved
15-13 MT	Memory Type Defines the memory type. 000b - ITCM 001b - DTCM 010b - I-cache 011b - D-cache
12-4 —	Reserved
3-2 —	Reserved
1-0	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	

6.4.1.8 Local Memory Descriptor a (LMEM_DESC_1 - LMEM_DESC_2)

Offset

Register	Offset
LMEM_DESC_1	404h
LMEM_DESC_2	408h

Function

NOTE

The DESC_a registers map to the LMEMs in this way:

- DESC_0: ITCM
- DESC_1: D0TCM
- DESC_2: D1TCM
- DESC_3: I-cache
- DESC_4: D-cache

NOTE

You can read and write to the reserved fields (instead of read as zero and write ignored). Writing to any of these fields has no functional impact.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LMV	Reserved		LMSZH	LMSZ				WY				DPW			Reserved
W																
Reset	1	0	0	0	0	1	1	0	0	0	0	0	0	1	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MT			Reserved									Reserved		Reserved	
W																
Reset	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 LMV	Local Memory Valid Defines the validity (presence) of the local memory. 0b - LMEM*n* not present 1b - LMEM*n* present
30-29 —	Reserved
28 LMSZH	LMEM Size Hole Used for local memories that are not fully populated (that is, local memories that include a memory "hole" in the upper 25 % of the address range). 0b - LMEM*n* is a power-of-2 capacity 1b - LMEM*n* is not a power-of-2, with capacity of 0.75 × LMSZ
27-24 LMSZ	Local Memory Size Indicates the local memory size. 0000b - 0 KB 0001b - 1 KB 0010b - 2 KB 0011b - 4 KB 0100b - 8 KB 0101b - 16 KB 0110b - 32 KB 0111b - 64 KB 1000b - 128 KB 1001b - 256 KB 1010b - 512 KB 1011b - 1024 KB 1100b - 2048 KB 1101b - 4096 KB 1110b - 8192 KB 1111b - 16384 KB
23-20 WY	Level 1 Cache Ways Defines the level 1 cache ways. 0000b - No cache

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0010b - 2-way set associative 0100b - 4-way set associative
19-17 DPW	Data Path Width Defines the LMEM*n* data path width, which is the width of the local memory. 000b-001b - Reserved 010b - 32 bits 011b - 64 bits 100b-111b - Reserved
16 —	Reserved
15-13 MT	Memory Type Defines the memory type. 000b - ITCM 001b - DTCM 010b - I-cache 011b - D-cache
12-4 —	Reserved
3-2 —	Reserved
1-0 —	Reserved

6.4.1.9 Local Memory Descriptor 3 (LMEM_DESC_3)

Offset

Register	Offset
LMEM_DESC_3	40Ch

Function

NOTE

The DESC_a registers map to the LMEMs in this way:

- DESC_0: ITCM
- DESC_1: D0TCM
- DESC_2: D1TCM
- DESC_3: I-cache
- DESC_4: D-cache

NOTE

You can read and write to the reserved fields (instead of read as zero and write ignored). Writing to any of these fields has no functional impact.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LMV	Reserved		LMSZH	LMSZ				WY				DPW			Reserved
W																
Reset	1	0	0	0	0	1	0	1	0	0	1	0	0	1	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MT			Reserved									Reserved		Reserved	
W																
Reset	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 LMV	Local Memory Valid Defines the validity (presence) of the local memory. 0b - LMEM*n* not present 1b - LMEM*n* present
30-29 —	Reserved
28 LMSZH	LMEM Size Hole Used for local memories that are not fully populated (that is, local memories that include a memory "hole" in the upper 25 % of the address range). 0b - LMEM*n* is a power-of-2 capacity 1b - LMEM*n* is not a power-of-2, with capacity of 0.75 × LMSZ

Table continues on the next page...

Table continued from the previous page...

Field	Function
27-24 LMSZ	Local Memory Size Indicates the local memory size. 0000b - 0 KB 0001b - 1 KB 0010b - 2 KB 0011b - 4 KB 0100b - 8 KB 0101b - 16 KB 0110b - 32 KB 0111b - 64 KB 1000b - 128 KB 1001b - 256 KB 1010b - 512 KB 1011b - 1024 KB 1100b - 2048 KB 1101b - 4096 KB 1110b - 8192 KB 1111b - 16384 KB
23-20 WY	Level 1 Cache Ways Defines the level 1 cache ways. 0000b - No cache 0010b - 2-way set associative 0100b - 4-way set associative
19-17 DPW	Data Path Width Defines the LMEM*n* data path width, which is the width of the local memory. 000b-001b - Reserved 010b - 32 bits 011b - 64 bits 100b-111b - Reserved
16 —	Reserved
15-13	Memory Type

Table continues on the next page...

Table continued from the previous page...

Field	Function
MT	Defines the memory type. 000b - ITCM 001b - DTCM 010b - I-cache 011b - D-cache
12-4 —	Reserved
3-2 —	Reserved
1-0 —	Reserved

6.4.1.10 Local Memory Descriptor 4 (LMEM_DESC_4)

Offset

Register	Offset
LMEM_DESC_4	410h

Function

NOTE

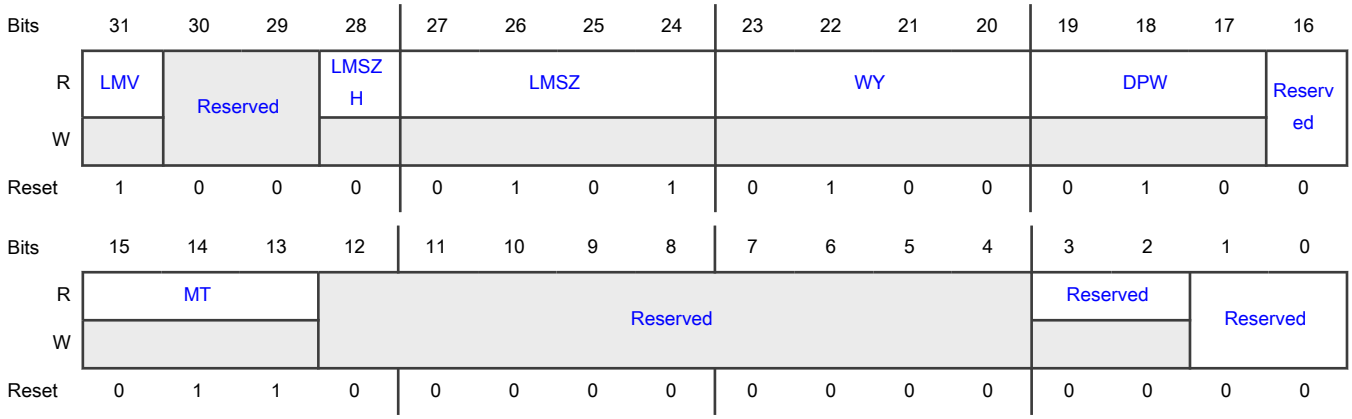
The DESC_a registers map to the LMEMs in this way:

- DESC_0: ITCM
- DESC_1: D0TCM
- DESC_2: D1TCM
- DESC_3: I-cache
- DESC_4: D-cache

NOTE

You can read and write to the reserved fields (instead of read as zero and write ignored). Writing to any of these fields has no functional impact.

Diagram



Fields

Field	Function
31 LMV	Local Memory Valid Defines the validity (presence) of the local memory. 0b - LMEM*n* not present 1b - LMEM*n* present
30-29 —	Reserved
28 LMSZH	LMEM Size Hole Used for local memories that are not fully populated (that is, local memories that include a memory "hole" in the upper 25 % of the address range). 0b - LMEM*n* is a power-of-2 capacity 1b - LMEM*n* is not a power-of-2, with capacity of 0.75 × LMSZ
27-24 LMSZ	Local Memory Size Indicates the local memory size. 0000b - 0 KB 0001b - 1 KB 0010b - 2 KB 0011b - 4 KB 0100b - 8 KB 0101b - 16 KB 0110b - 32 KB 0111b - 64 KB 1000b - 128 KB

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1001b - 256 KB 1010b - 512 KB 1011b - 1024 KB 1100b - 2048 KB 1101b - 4096 KB 1110b - 8192 KB 1111b - 16384 KB
23-20 WY	Level 1 Cache Ways Defines the level 1 cache ways. 0000b - No cache 0010b - 2-way set associative 0100b - 4-way set associative
19-17 DPW	Data Path Width Defines the LMEM*n* data path width, which is the width of the local memory. 000b-001b - Reserved 010b - 32 bits 011b - 64 bits 100b-111b - Reserved
16 —	Reserved
15-13 MT	Memory Type Defines the memory type. 000b - ITCM 001b - DTCM 010b - I-cache 011b - D-cache
12-4 —	Reserved
3-2 —	Reserved
1-0	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	

6.5 Glossary

ITCM	Instruction Tightly-Coupled Memory
DTCM	Data Tightly Coupled Memory
I-cache	Instruction Cache Memory
D-cache	Data Cache Memory

Chapter 7
Miscellaneous System Control Module (MSCM)

7.1 Chip-specific MSCM information

7.1.1 MSCM instance

This chip has one instance of MSCM.

NOTE

The XN_CTRL register is used to restrict execution from SRAM, including TCMs and their backdoors, which will be permanent until next device reset, while still allowing data R/W.

XN_CTRL register is reserved for S32K344/S32K324/S32K314.

7.1.2 Reporting of core-to-core interrupts

For all the variants the core-to-core interrupts are reported to both INTM and MSCM for CM7_0 and CM7_1. But, for CM7_2 the core-to-core interrupts are reported to MSCM, instead of INTM. See the interrupt map file attached to this document for details.

7.1.3 ENEDC register implementation

In S32K310, S32K311, S32K312, S32K314, S32K322, S32K324, S32K341, S32K342, and S32K344 there are additional bit field as compared to what is mentioned in section 'MSCM memory map'. See following table for details.

Table 33. Bitfield details

Bitfield name	Bitfield position
ENEDC[EN_WR_TCM][1]	16
ENEDC[EN_ADD_TCM] [1]	17

1. See section EN_WR_TCM and EN_ADD_TCM definition for details

7.1.3.1 EN_WR_TCM and EN_ADD_TCM definition

Register ENEDC has the following additional bitfields:

Table 34. Bitfield definition

Bitfield position	Bitfield name	Bitfield description
16	EN_WR_TCM (Enable Write Data Check TCM)	Enables or disables the write data check for TCM 64-bit path. • 0b-Disabled • 1b-Enabled
17	EN_ADD_TCM (Enable Address Check TCM)	Enables or disables the address check for TCM 64-bit path. • 0b-Disabled • 1b-Enabled

7.1.4 ENEDC and ENEDC1 register implementation for S32K358, S32K356, S32K348, S32K338, and S32K328

In S32K358, S32K356, S32K348, S32K338, and S32K328 there are some differences in register[bit field] as compared to what is mentioned in section 'MSCM memory map'. See following table for details.

Table 35. Bitfield details

Register[Bitfield name]	Bitfield position
ENEDC[ADD_TCM_BACKDOOR]	17
ENEDC1[CM7_3_WDATA_CHK][1]	12
ENEDC1[CM7_3_ADDR_CHK][1]	13
ENEDC1[USDHC]	16
ENEDC1[CM7_3_AHBM][1]	17
ENEDC1[CM7_3_AHBP][1]	18
ENEDC1[MSTR_CHK_ACE_RESULT_CHK][1]	19
ENEDC1[MSTR_CHK_ACE_FEED_CHK][1]	20
ENEDC1[SLV_CHK_ACE_ADDR_CHK][1]	21
ENEDC1[SLV_CHK_ACE_ACCEL_RESULT_M1_GSKT_WDATA_CHK][1]	22
ENEDC1[SLV_CHK_ACE_ACCEL_RESULT_M1_GSKT_ADDR_CHK][1]	23
ENEDC1[TCM_GSKT_ADDR_CHK][1]	24

1. This field is Reserved for S32K358, S32K356, S32K348, S32K338, and S32K328.

7.1.4.1 ENEDC[ADD_TCM_BACKDOOR] and ENEDC1 [USDHC] definition

In S32K358, S32K356, S32K348, S32K338, and S32K328 ENEDC register has the following additional bitfields:

Table 36. Bitfield definition

Bitfield position	Bitfield name	Bitfield description
17	ADD_TCM_BACKDOOR (Write Data Check For TCM Backdoor)	Enables or disables the address check for the TCM backdoor path. • 0b-Disabled • 1b-Enabled

In S32K358, S32K356, S32K348, S32K338, and S32K328 ENEDC1 register has the following additional bitfields:

Table 37. Bitfield definition

Bitfield position	Bitfield name	Bitfield description
16	USDHC (Enable Read Data Check uSDHC)	Enables or disables the read data check for the uSDHC path. • 0b-Disabled • 1b-Enabled

7.2 Overview

MSCM contains registers for:

- CPU configuration
- On-chip memory control
- Interrupt router control

- Message-based interrupt configuration
- Virtual management

7.2.1 Features

- Software-accessible processor core configuration information
- Support for interrupt router control
- Support for message-based interrupt configuration

7.3 Functional description

MSCM provides information of the system cores and can identify the core that is running currently.

7.3.1 MSI routing

MSIs are interrupts that are indirectly broadcast to a target core by writing configuration bits in MSCM. These MSIs can be initiated by one core targeting another core in the system (known as core-to-core interrupts). These MSIs are initiated via writes to Interrupt Router CPn Interrupt Generation (IRCP0IGR0 - IRCP3IGR3) and managed through Interrupt Router CPn Interrupt Status (IRCP0ISR0 - IRCP3ISR3). The Cortex-M7 cores can support up to four outstanding core-to-core interrupts.

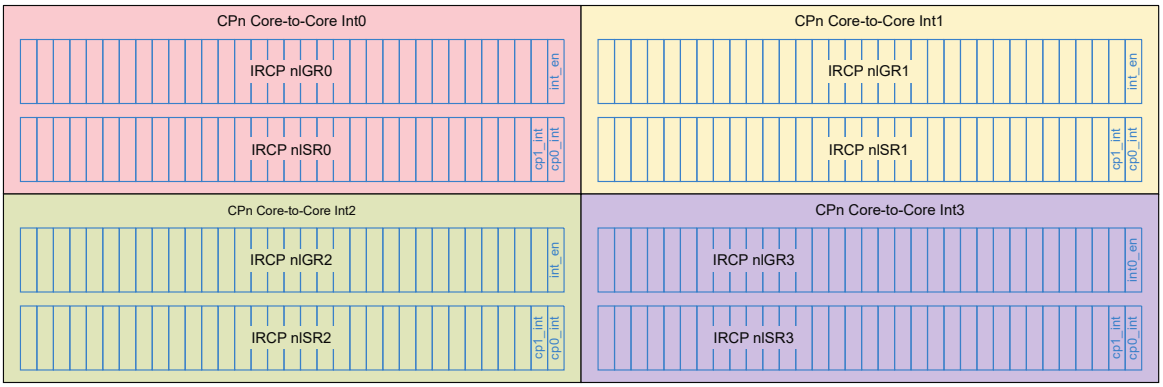


Figure 16. IRCP*n*IGR*m*/IRCP*n*ISR*m* pairs for one core

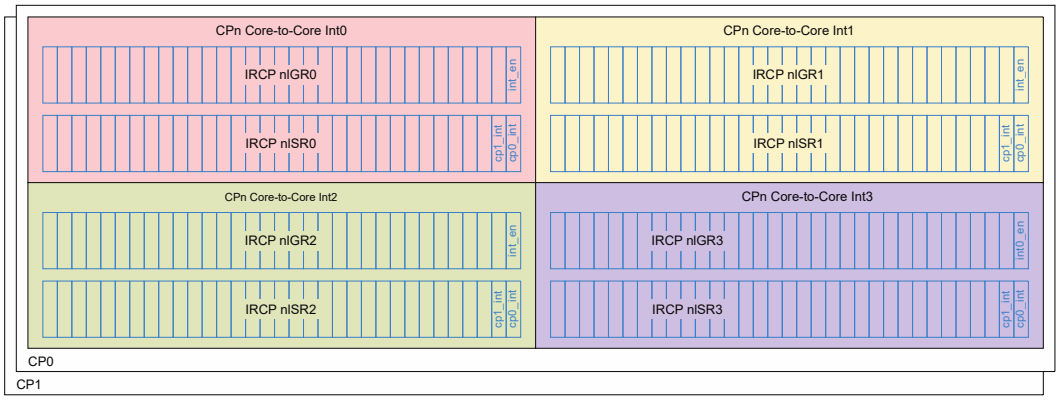


Figure 17. IRCP*n*IGR*m*/IRCP*n*ISR*m* pairs per core

7.3.1.1 Core-to-core MSIs

The next figure depicts the sequence for initiating a core-to-core MSI, in which *m* represents the initiating core, *n* represents the target core, and *x* indicates the MSI number. CP*m* writes to IRCP*n*IGR*x* to initiate an MSI. The outstanding MSI that CP*m* initiates, targeting CP*n*, is reflected in the corresponding bit-mapped field in IRCP*n*ISR*x*.

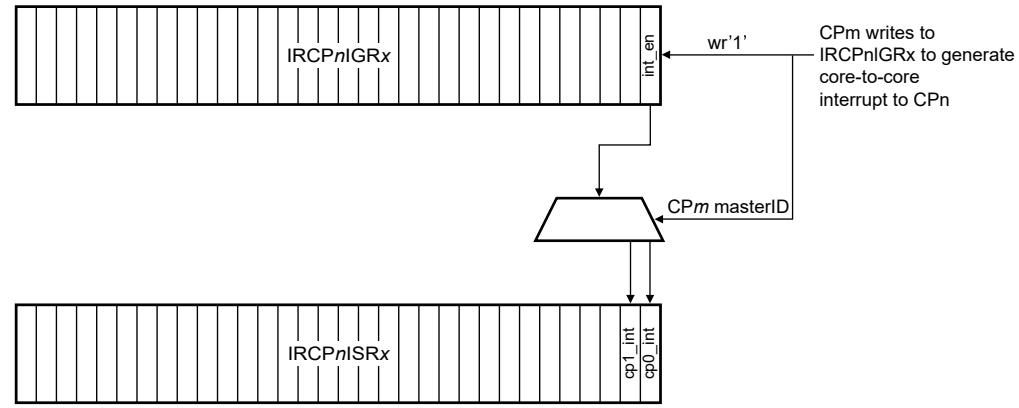


Figure 18. Initiating a core-to-core MSI via IRCP*n*IGR*x*/IRCP*n*ISR*x*

7.3.2 Interrupt steering and semaphores

7.3.2.1 Interrupt handling overview

The interrupt handling mechanisms of the Cortex-M7 cores are very similar. These cores have an NVIC tightly coupled to the processor core. The real-time performance of the cores means the NVIC directly provides an appropriate interrupt vector, in the form of the starting instruction address for the interrupt service routine, to the core. These core architectural features directly translate into a faster ISR entry and exit capabilities, coupled with an improved runtime performance. See the Arm modules and Arm core technical reference manuals for details.

In this architecture, a total of 240 IRQs are supported, where this parameter is defined by the realistic limits of the NVIC implementation, both in terms of silicon size and supported frequency of operation. These 240 IRQs are split into a total of four directed requests and 236 shared peripheral requests. Unless noted otherwise, let the directed requests be defined as IRQ[3:0] and the shared peripheral requests as IRQ[239:4]. See the interrupt map file attached to this document for details.

7.3.2.2 MSCM interrupt router functional description

As described in MSCM register descriptions, the interrupt routing registers enable the steering of requests to the processor cores.

7.3.3 Clocking

This module has no clocking considerations.

7.4 External signals

This module has no external signals.

7.5 Initialization

This module does not require initialization.

7.6 Memory map and register definition

7.6.1 Core configuration registers

These read-only registers contain data that defines the core setup for this chip. You can access the registers using 32-bit read references; other access sizes terminate with an error. Attempted write accesses to the read-only core configuration registers also terminate with an error.

The core configuration portion of the MSCM programming model map is organized based on the logical core number, and not on any type of physical port number. The following table shows how the configuration is partitioned.

Table 38. MSCM core configuration partitioning

Offset address range	Purpose
0h–018h	Defines the generic core x configuration information. Only the MSCM associated controllers can access this region in either User or Privileged mode; reads by noncore bus controllers (including the debugger) are treated as read as zero (RAZ) accesses. Write attempts are not permitted and terminate with a system bus error.
020h–038h	Defines the configuration information for core 0 (CP0). Any bus controller can access this region in either User or Privileged mode. Write attempts are not permitted and terminate with a system bus error.
040h–058h	Defines the configuration information for core 1 (CP1). A bus controller can access this region in either User or Privileged mode. Write attempts are not permitted and terminate with a system bus error.
060h–078h	Defines the configuration information for core 2 (CP2). Any bus controller can access this region in either User or Privileged mode. Write attempts are not permitted and terminate with a system bus error.
080h–098h	Defines the configuration information for core 3 (CP3). Any bus controller can access this region in either User or Privileged mode. Write attempts are not permitted and terminate with a system bus error.

NOTE

Attempted accesses to reserved locations are not permitted and terminate with a system bus error.

7.6.2 Shared peripheral interrupt (SPI) routing

The SPI router portion of MSCM provides a set of memory-mapped registers defining the interrupt routing for all the SPIs on this chip.

7.6.3 MSCM register descriptions

7.6.3.1 MSCM memory map

MSCM base address: 4026_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Processor X Type (CPXTYPE)	32	R	See section
4h	Processor X Number (CPXNUM)	32	R	See section
8h	Processor X Revision (CPXREV)	32	R	See section
Ch	Processor X Configuration 0 (CPXCFG0)	32	R	0602_0604h
10h	Processor X Configuration 1 (CPXCFG1)	32	R	See section

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
14h	Processor X Configuration 2 (CPXCFG2)	32	R	See section
18h	Processor X Configuration 3 (CPXCFG3)	32	R	0000_000Bh
20h	Processor 0 Type (CP0TYPE)	32	R	434D_3730h
24h	Processor 0 Number (CP0NUM)	32	R	See section
28h	Processor 0 Count (CP0REV)	32	R	See section
2Ch	Processor 0 Configuration 0 (CP0CFG0)	32	R	0502_0504h
30h	Processor 0 Configuration 1 (CP0CFG1)	32	R	0000_0000h
34h	Processor 0 Configuration 2 (CP0CFG2)	32	R	See section
38h	Processor 0 Configuration 3 (CP0CFG3)	32	R	0000_000Bh
40h	Processor 1 Type (CP1TYPE)	32	R	434D_3731h
44h	Processor 1 Number (CP1NUM)	32	R	0000_0001h
48h	Processor 1 Count (CP1REV)	32	R	See section
4Ch	Processor 1 Configuration 0 (CP1CFG0)	32	R	0502_0504h
50h	Processor 1 Configuration 1 (CP1CFG1)	32	R	See section
54h	Processor 1 Configuration 2 (CP1CFG2)	32	R	See section
58h	Processor 1 Configuration 3 (CP1CFG3)	32	R	0000_000Bh
60h	Processor 2 Type (CP2TYPE)	32	R	434D_3732h
64h	Processor 2 Number (CP2NUM)	32	R	0000_0002h
68h	Processor 2 Count (CP2REV)	32	R	See section
6Ch	Processor 2 Configuration 0 (CP2CFG0)	32	R	0602_0604h
70h	Processor 2 Configuration 1 (CP2CFG1)	32	R	See section
74h	Processor 2 Configuration 2 (CP2CFG2)	32	R	See section
78h	Processor 2 Configuration 3 (CP2CFG3)	32	R	0000_000Bh
80h	Processor 3 Type (CP3TYPE)	32	R	434D_3733h
84h	Processor 3 Number (CP3NUM)	32	R	0000_0003h
88h	Processor 3 Count (CP3REV)	32	R	See section
8Ch	Processor 3 Configuration 0 (CP3CFG0)	32	R	0602_0604h
90h	Processor 3 Configuration 1 (CP3CFG1)	32	R	See section
94h	Processor 3 Configuration 2 (CP3CFG2)	32	R	See section
98h	Processor 3 Configuration 3 (CP3CFG3)	32	R	0000_000Bh
200h	Interrupt Router CP0 Interrupt Status (IRCP0ISR0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
204h	Interrupt Router CP0 Interrupt Generation (IRCP0IGR0)	32	RW	0000_0000h
208h	Interrupt Router CP0 Interrupt Status (IRCP0ISR1)	32	RW	0000_0000h
20Ch	Interrupt Router CP0 Interrupt Generation (IRCP0IGR1)	32	RW	0000_0000h
210h	Interrupt Router CP0 Interrupt Status (IRCP0ISR2)	32	RW	0000_0000h
214h	Interrupt Router CP0 Interrupt Generation (IRCP0IGR2)	32	RW	0000_0000h
218h	Interrupt Router CP0 Interrupt Status (IRCP0ISR3)	32	RW	0000_0000h
21Ch	Interrupt Router CP0 Interrupt Generation (IRCP0IGR3)	32	RW	0000_0000h
220h	Interrupt Router CP1 Interrupt Status (IRCP1ISR0)	32	RW	0000_0000h
224h	Interrupt Router CP1 Interrupt Generation (IRCP1IGR0)	32	RW	0000_0000h
228h	Interrupt Router CP1 Interrupt Status (IRCP1ISR1)	32	RW	0000_0000h
22Ch	Interrupt Router CP1 Interrupt Generation (IRCP1IGR1)	32	RW	0000_0000h
230h	Interrupt Router CP1 Interrupt Status (IRCP1ISR2)	32	RW	0000_0000h
234h	Interrupt Router CP1 Interrupt Generation (IRCP1IGR2)	32	RW	0000_0000h
238h	Interrupt Router CP1 Interrupt Status (IRCP1ISR3)	32	RW	0000_0000h
23Ch	Interrupt Router CP1 Interrupt Generation (IRCP1IGR3)	32	RW	0000_0000h
240h	Interrupt Router CP2 Interrupt Status (IRCP2ISR0)	32	RW	0000_0000h
244h	Interrupt Router CP2 Interrupt Generation (IRCP2IGR0)	32	RW	0000_0000h
248h	Interrupt Router CP2 Interrupt Status (IRCP2ISR1)	32	RW	0000_0000h
24Ch	Interrupt Router CP2 Interrupt Generation (IRCP2IGR1)	32	RW	0000_0000h
250h	Interrupt Router CP2 Interrupt Status (IRCP2ISR2)	32	RW	0000_0000h
254h	Interrupt Router CP2 Interrupt Generation (IRCP2IGR2)	32	RW	0000_0000h
258h	Interrupt Router CP2 Interrupt Status (IRCP2ISR3)	32	RW	0000_0000h
25Ch	Interrupt Router CP2 Interrupt Generation (IRCP2IGR3)	32	RW	0000_0000h
260h	Interrupt Router CP3 Interrupt Status (IRCP3ISR0)	32	RW	0000_0000h
264h	Interrupt Router CP3 Interrupt Generation (IRCP3IGR0)	32	RW	0000_0000h
268h	Interrupt Router CP3 Interrupt Status (IRCP3ISR1)	32	RW	0000_0000h
26Ch	Interrupt Router CP3 Interrupt Generation (IRCP3IGR1)	32	RW	0000_0000h
270h	Interrupt Router CP3 Interrupt Status (IRCP3ISR2)	32	RW	0000_0000h
274h	Interrupt Router CP3 Interrupt Generation (IRCP3IGR2)	32	RW	0000_0000h
278h	Interrupt Router CP3 Interrupt Status (IRCP3ISR3)	32	RW	0000_0000h
27Ch	Interrupt Router CP3 Interrupt Generation (IRCP3IGR3)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
400h	Interrupt Router Configuration (IRCPCFG)	32	RW	0000_0000h
500h	Memory Execution Controls (XN_CTRL)	32	RW	4000_0000h
600h	Enable Interconnect Error Detection (ENEDC)	32	RW	0000_0000h
604h	Enable Interconnect Error Detection (ENEDC1)	32	RW	0000_0000h
700h	AHB Gasket Configuration (IAHBCFGREG)	32	RW	0000_0000h
880h - A5Eh	Interrupt Router Shared Peripheral Routing Control (IRSPRC0 - IRSPRC239)	16	RW	000Fh

7.6.3.2 Processor X Type (CPXTYPE)

Offset

Register	Offset
CPXTYPE	0h

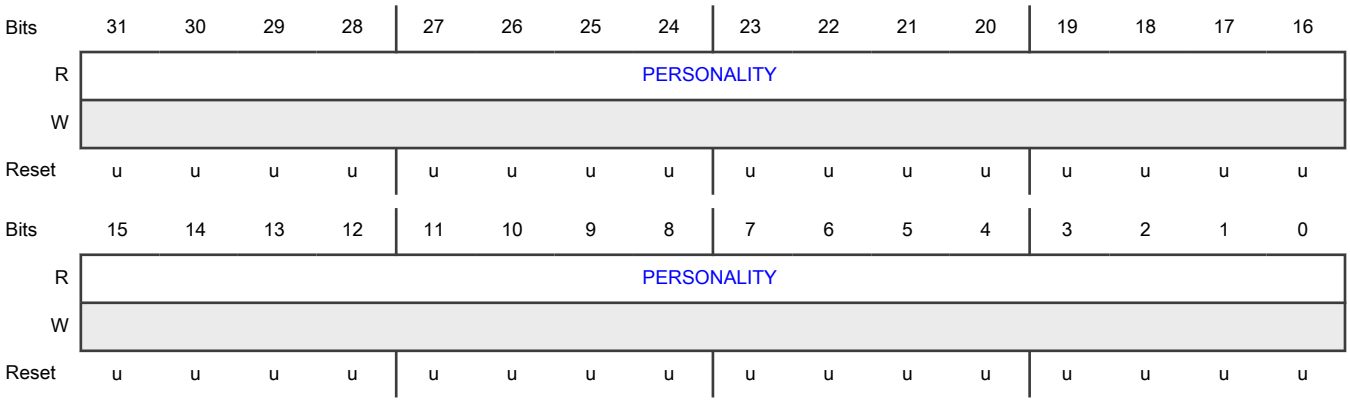
Function

Provides a CPU-specific response indicating the personality of the core making the access. The 32-bit response includes four ASCII characters defining the CPU type (Cortex-M7 cores) along with a byte defining the logical revision number and a byte defining the instance number of the core.

A read from Cortex-M7 returns the appropriate processor information. Reads from any other bus controller return all 0s and attempted write accesses terminate with an error.

Access: User or privileged read-only

Diagram



Fields

Field	Function	
31-0 PERSONALITY	Personality of CPx Defines the processor personality for CP*x*.	
	Processor	**Personality**
	CP*x* = Cortex-M7_0	43_4D_37_30h
	CP*x* = Cortex-M7_1	43_4D_37_31h
	CP*x* = Cortex-M7_2	43_4D_37_32h
	CP*x* = Cortex-M7_3	43_4D_37_33h

7.6.3.3 Processor X Number (CPXNUM)

Offset

Register	Offset
CPXNUM	4h

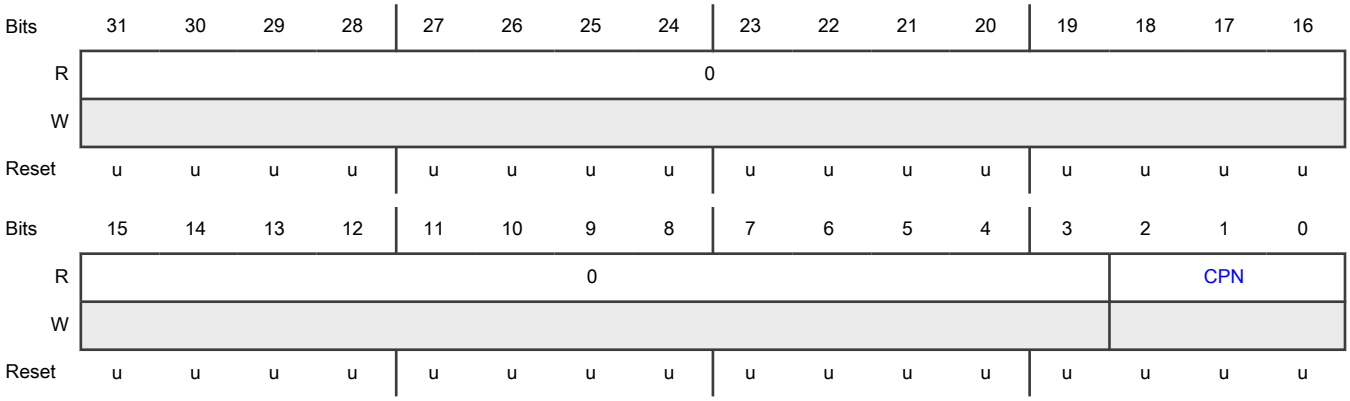
Function

Provides a CPU-specific response indicating the logical processor number of the core making the access.

A read from the Cortex-M7 cores returns the appropriate processor information. Reads from any other bus controller return all 0s and attempted write accesses terminate with an error.

Access: User or privileged read-only

Diagram



Fields

Field	Function
31-3 —	Reserved
2-0 CPN	Processor Number Defines the logical processor number for CP*x*. **NOTE** CPN in MSCM indicates only the on-platform cores and not the HSE_B core. CPN = 0 represents Cortex-M7_0 if it is a lockstep or dual core. In Lockstep mode, CPN = 1 does not read from Processor X Number (CPXNUM). 000b - Cortex-M7 core 0 001b - Cortex-M7 core 1 010b - Cortex-M7 core 2 011b - Cortex-M7 core 3

7.6.3.4 Processor X Revision (CPXREV)

Offset

Register	Offset
CPXREV	8h

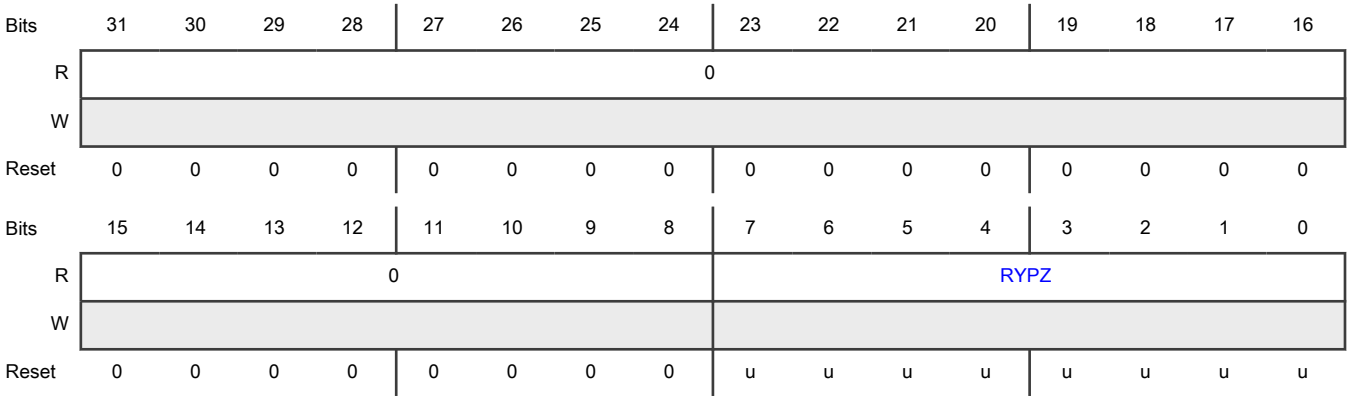
Function

Provides a CPU-specific response indicating the logical revision number of the core.

A read from the Cortex-M7 cores returns the appropriate processor information. Reads from any other bus controller return all 0s and attempted write accesses terminate with an error.

Access: User or privileged read-only

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 RYPZ	Processor Revision Defines the processor revision for CP*x*. For the Cortex-M7 cores in this chip, RYPZ = 12h corresponding to the r1p2 core release.

7.6.3.5 Processor X Configuration 0 (CPXCFG0)

Offset

Register	Offset
CPXCFG0	Ch

Function

Provides a CPU-specific response detailing configuration information. In this case, it is information on Level 1 (L1) cache, if present.

A read from Cortex-M7 returns the appropriate processor information. Reads from any other bus controller return all 0s and attempted write accesses terminate with an error.

Access: User or privileged read-only

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ICSZ								ICWY							
W																
Reset	0	0	0	0	0	1	1	0	0	0	0	0	0	0	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DCSZ								DCWY							
W																
Reset	0	0	0	0	0	1	1	0	0	0	0	0	0	1	0	0

Fields

Field	Function
31-24 ICSZ	Level 1 Instruction Cache Size

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Provides an encoded value of the instruction cache size. The capacity of the memory is derived using the formula $2^{(8+SZ)}$ and expressed as bytes. Here, ICSZ is a nonzero value and ICSZ = 0 indicates that the memory is not present. For information about cache sizes, see the "Miscellaneous Control Module (MCM)" chapter.
23-16 ICWY	L1 Instruction Cache Ways Provides the number of cache ways for the instruction cache. For the Cortex-M7 cores in this chip, ICWY = 2h (2-way set-associative).
15-8 DCSZ	L1 Data Cache Size Provides an encoded value of the data cache size. The capacity of the memory is derived using the formula $2^{(8+SZ)}$ and expressed as bytes. Here, DCSZ is a nonzero value and DCSZ = 0 indicates that the memory is not present. For information about cache sizes, see the "Miscellaneous Control Module (MCM)" chapter.
7-0 DCWY	L1 Data Cache Ways Provides the number of cache ways for the data cache. For the Cortex-M7 cores in this chip, DCWY = 4h (4-way set-associative).

7.6.3.6 Processor X Configuration 1 (CPXCFG1)

Offset

Register	Offset
CPXCFG1	10h

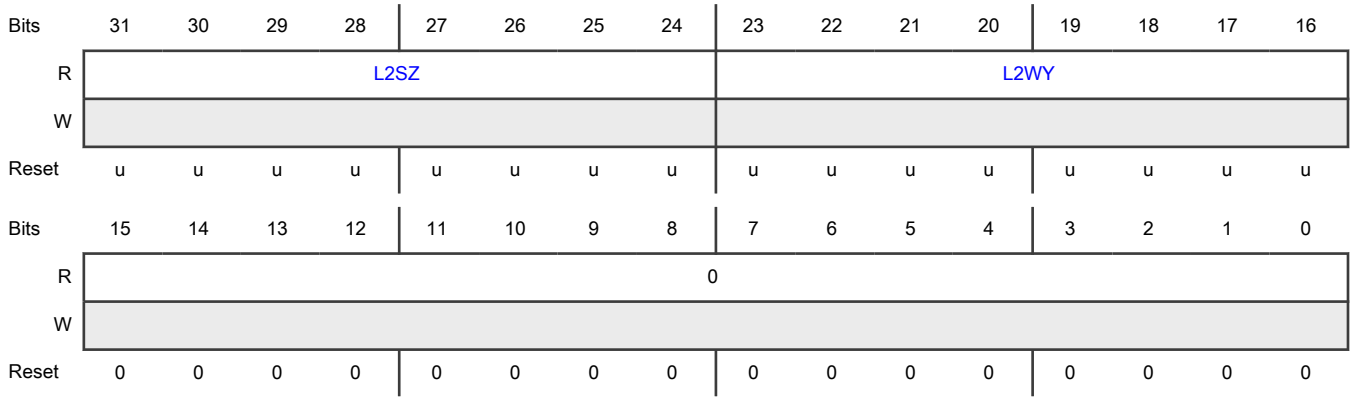
Function

Provides a CPU-specific response detailing configuration information. In this case, it is information on Level 2 (L2) cache, if present.

A read from the Cortex-M7 cores returns the appropriate processor information. Reads from any other bus controller return all 0s and attempted write accesses terminate with an error.

Access: User or privileged read-only

Diagram



Fields

Field	Function
31-24 L2SZ	L2 Cache Size Provides an encoded value of the L2 cache size. The capacity of the memory is derived using the formula $2^{(8+SZ)}$ and expressed as bytes. Here, L2SZ is a nonzero value, and L2SZ = 0 indicates that the memory is not present. For the Cortex-M7 cores in this chip, L2SZ = 0h (not present).
23-16 L2WY	L2 Cache Ways Provides the number of cache ways for the L2 cache. For the Cortex-M7 cores in this chip, L2WY = 0h (not present).
15-0 —	Reserved

7.6.3.7 Processor X Configuration 2 (CPXCFG2)

Offset

Register	Offset
CPXCFG2	14h

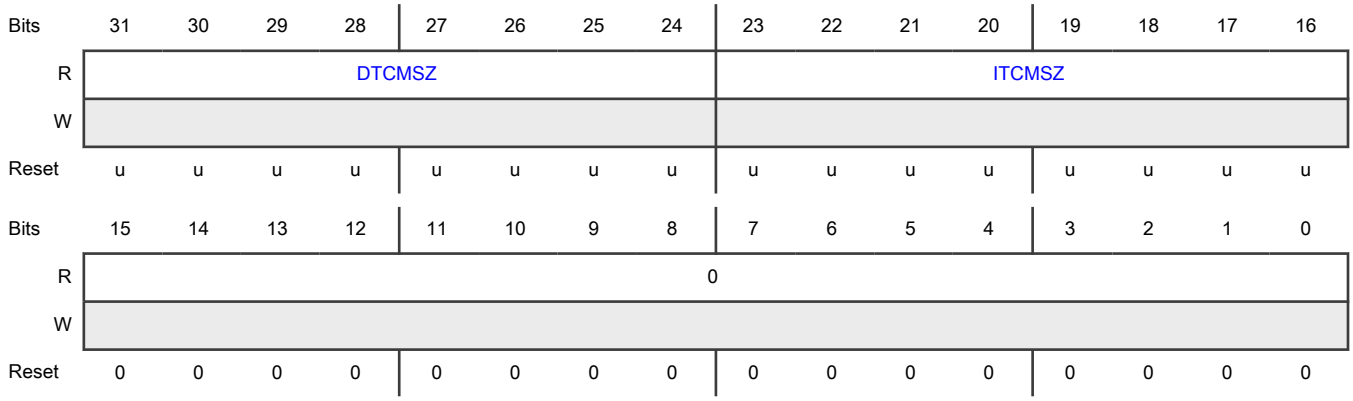
Function

Provides a CPU-specific response detailing configuration information. In this case, it is information on tightly coupled local memories, if present.

A read from the Cortex-M7 cores returns the appropriate processor information. Reads from any other bus controller return all 0s and attempted write accesses terminate with an error.

Access: User or privileged read-only

Diagram



Fields

Field	Function
31-24 DTCMSZ	Tightly Coupled Data Memory Size Provides an encoded value of the tightly coupled local data memory size. The capacity of the memory is derived using the formula $2^{(8+SZ)}$ and expressed as bytes. Here, DTCMSZ is a nonzero value and DTCMSZ = 0 indicates that the memory is not present. For the Cortex-M7 cores in this chip: • DTCMSZ = 08 in Decoupled mode (64 KB) • DTCMSZ = 09 for Cortex-M7_0 in Lockstep mode (128 KB)
23-16 ITCMSZ	Instruction Tightly Coupled Memory Size Provides an encoded value of the tightly coupled local instruction memory size. The capacity of the memory is derived using the formula $2^{(8+SZ)}$ and expressed as bytes. Here, ITCMSZ is a nonzero value, and ITCMSZ = 0 indicates that the memory is not present. For the Cortex-M7 cores in this chip: • ITCMSZ = 07 in Decoupled mode (32 KB) • ITCMSZ = 08 for Cortex-M7_0 in Lockstep mode (64 KB)
15-0 —	Reserved

7.6.3.8 Processor X Configuration 3 (CPXCFG3)

Offset

Register	Offset
CPXCFG3	18h

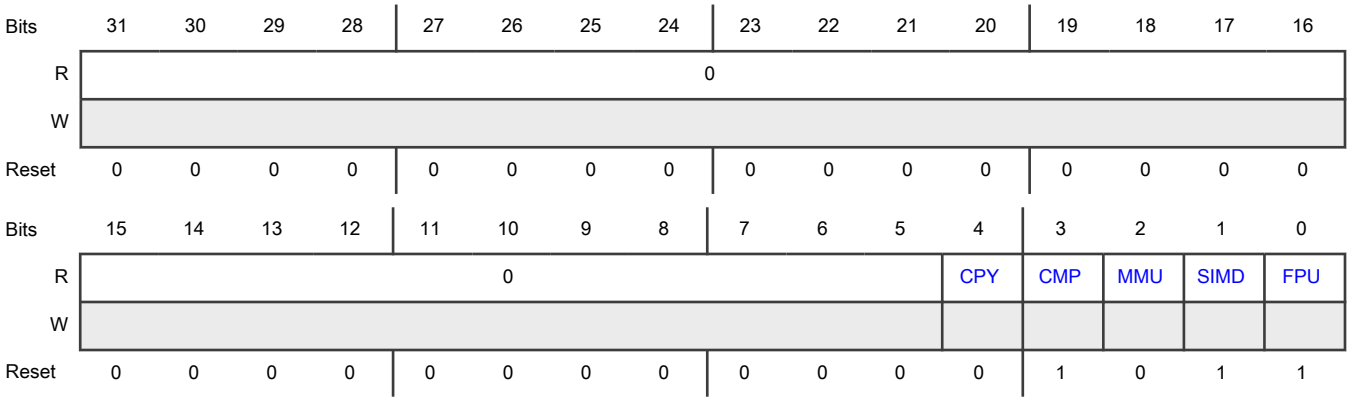
Function

Provides a CPU-specific response detailing configuration information. In this case, it is information on processor options.

A privileged read from Cortex-M7 returns the appropriate processor information. Reads from any other bus controller return all 0s and attempted write accesses terminate with an error.

Access: User or privileged read-only

Diagram



Fields

Field	Function
31-5 —	Reserved
4 CPY	Cryptography Indicates whether the cryptography extensions are supported in the core. For the Cortex-M7 cores in this chip, CPY = 0h. 0b - Not supported 1b - Supported
3 CMP	Core Memory Protection Unit Indicates whether the core memory protection hardware is included in this core. For the Cortex-M7 cores in this chip, CMP = 1h. 0b - Not included 1b - Included
2 MMU	Memory Management Unit Indicates whether virtual management capabilities are supported in this core. For the Cortex-M7 cores in this chip, MMU = 0h. 0b - Not supported 1b - Supported
1 SIMD	SIMD/NEON Instruction Support

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Indicates whether the instruction set extensions supporting SIMD and/or NEON capabilities are included in the processor. For the Cortex-M7 cores in this chip, SIMD = 1h. 0b - Not included 1b - Included
0 FPU	Floating Point Unit Indicates whether hardware support for floating point capabilities is provided in the processor. For the Cortex-M7 cores in this chip, FPU = 1h. 0b - Not provided 1b - Provided

7.6.3.9 Processor 0 Type (CP0TYPE)

Offset

Register	Offset
CP0TYPE	20h

Function

Defines the configuration information for processor 0 (CP0). It has the same field definitions and functionality as provided in Processor X Type (CPXTYPE).

A read from any bus controller returns the appropriate processor information and attempted write accesses terminate with an error.

Access: User or privileged read-only

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PERSONALITY															
W																
Reset	0	1	0	0	0	0	1	1	0	1	0	0	1	1	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PERSONALITY															
W																
Reset	0	0	1	1	0	1	1	1	0	0	1	1	0	0	0	0

Fields

Field	Function
31-0 PERSONALITY	Processor Personality Defines the processor personality for CP0. For Cortex-M7 core 0, personality = 43_4D_37_30h.

7.6.3.10 Processor 0 Number (CP0NUM)

Offset

Register	Offset
CP0NUM	24h

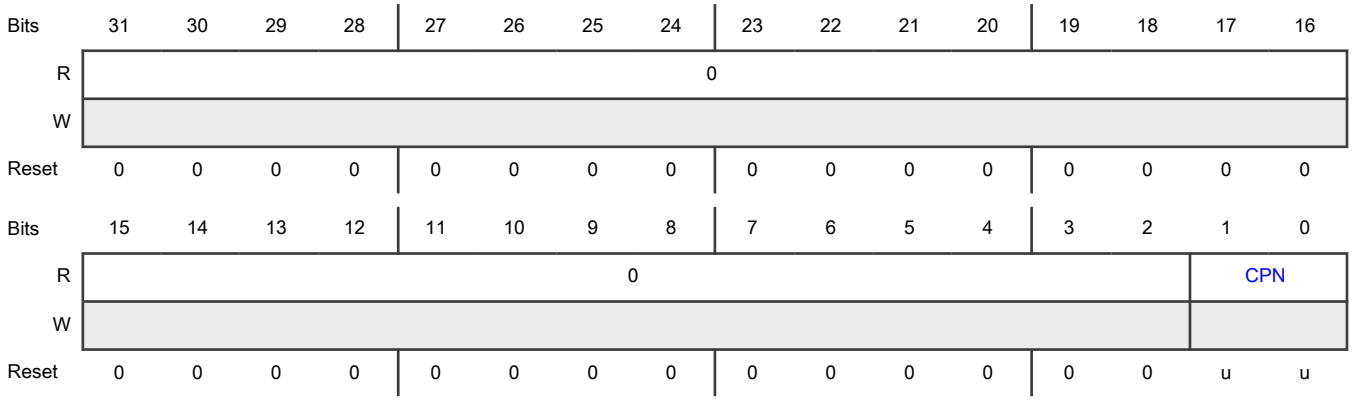
Function

Defines the configuration information for processor 0 (CP0). It has the same field definitions and functionality as provided in Processor X Number (CPXNUM).

A privileged read from any bus controllers returns the appropriate processor information and attempted write accesses terminate with an error.

Access: User or privileged read-only

Diagram



Fields

Field	Function
31-2 —	Reserved
1-0 CPN	Processor Number Defines the logical processor number for CP0. For processor Cortex-M7 core 0, processor number = 0.

7.6.3.11 Processor 0 Count (CP0REV)

Offset

Register	Offset
CP0REV	28h

Function

Defines the configuration information for processor 0 (CP0). It has the same field definitions and functionality as provided in Processor X Revision (CPXREV).

A read from any bus controller returns the appropriate processor information and attempted write accesses terminate with an error.

Access: User or privileged read-only

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								RYPZ							
W																
Reset	0	0	0	0	0	0	0	0	u	u	u	u	u	u	u	u

Fields

Field	Function
31-8 —	Reserved
7-0 RYPZ	Processor Revision Defines the processor revision for CP0. For the Cortex-M7 processor, RYPZ = 12h corresponding to the r1p2 core release.

7.6.3.12 Processor 0 Configuration 0 (CP0CFG0)

Offset

Register	Offset
CP0CFG0	2Ch

Function

Defines the configuration information for processor 0 (CP0). It has the same field definitions and functionality as provided in Processor X Configuration 0 (CPXCFG0).

A read from any bus controller returns the appropriate processor information and attempted write accesses terminate with an error.

Access: User or privileged read-only

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ICSZ								ICWY							
W																
Reset	0	0	0	0	0	1	0	1	0	0	0	0	0	0	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DCSZ								DCWY							
W																
Reset	0	0	0	0	0	1	0	1	0	0	0	0	0	1	0	0

Fields

Field	Function
31-24 ICSZ	Level 1 Instruction Cache Size Provides an encoded value of the instruction cache size. The capacity of the memory is derived using the formula $2^{(8+SZ)}$ and expressed as bytes. Here, ICSZ is a nonzero value, and ICSZ = 0 indicates that the memory is not present. For information about cache sizes, see the "Miscellaneous Control Module (MCM)" chapter.
23-16 ICWY	L1 Instruction Cache Ways Provides the number of cache ways for the instruction cache. For the Cortex-M7 cores in this chip, ICWY = 2h (2-way set-associative).
15-8 DCSZ	L1 Data Cache Size Provides an encoded value of the data cache size. The capacity of the memory is derived using the formula $2^{(8+SZ)}$ and expressed as bytes. Here, DCSZ is a nonzero value, and DCSZ = 0 indicates that the memory is not present. For information about cache sizes, see the "Miscellaneous Control Module (MCM)" chapter.
7-0 DCWY	L1 Data Cache Ways Provides the number of cache ways for the data cache. For the Cortex-M7 cores in this chip, DCWY = 4h (4-way set-associative).

7.6.3.13 Processor 0 Configuration 1 (CP0CFG1)

Offset

Register	Offset
CP0CFG1	30h

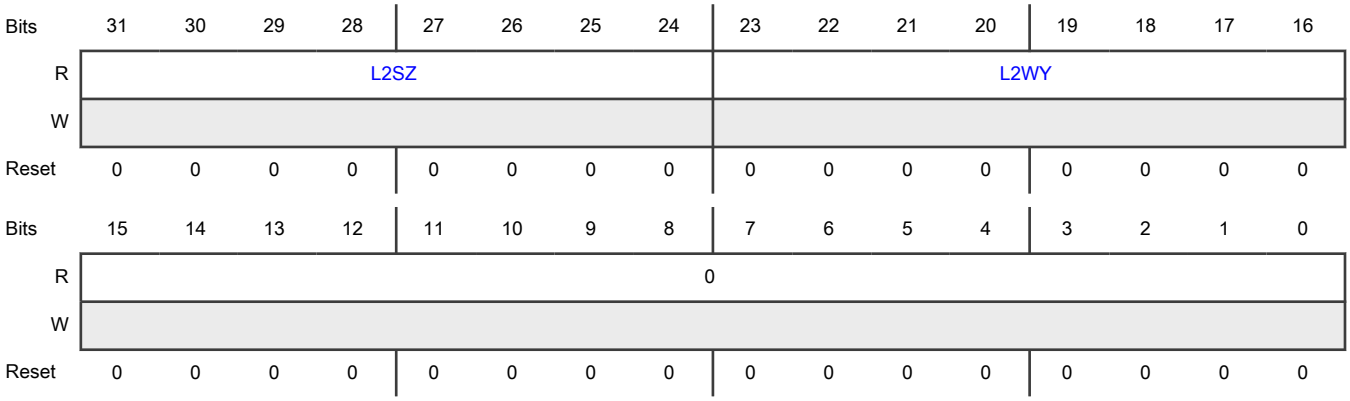
Function

Defines the configuration information for processor 0 (CP0). It has the same field definitions and functionality as provided in Processor X Configuration 1 (CPXCFG1).

A privileged read from any bus controller returns the appropriate processor information and attempted write accesses terminate with an error.

Access: User or privileged read-only

Diagram



Fields

Field	Function
31-24 L2SZ	L2 Cache Size Provides an encoded value of the L2 cache size. The capacity of the memory is derived using the formula $2^{(8+SZ)}$ and expressed as bytes. Here, L2SZ is a nonzero value, and L2SZ = 0 indicates that the memory is not present. For the Cortex-M7 cores in this chip, L2SZ = 0h (not present).
23-16 L2WY	L2 Cache Ways Provides the number of cache ways for the L2 cache. For the Cortex-M7 cores in this chip, L2WY = 0h (not present).
15-0 —	Reserved

7.6.3.14 Processor 0 Configuration 2 (CP0CFG2)

Offset

Register	Offset
CP0CFG2	34h

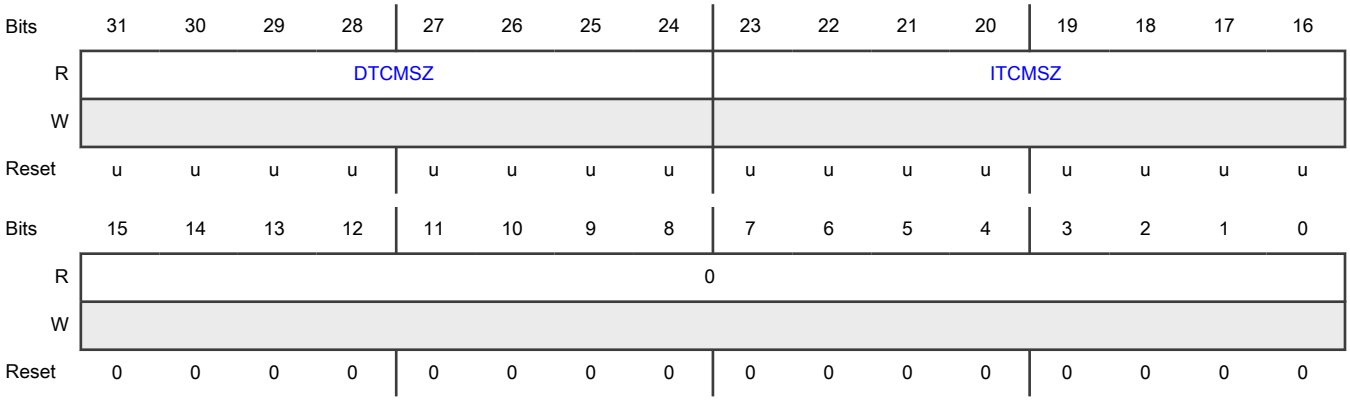
Function

Defines the configuration information for processor 0 (CP0). It has the same field definitions and functionality as provided in Processor X Configuration 2 (CPXCFG2).

A privileged read from any bus controller returns the appropriate processor information and attempted write accesses terminate with an error.

Access: User or privileged read-only

Diagram



Fields

Field	Function
31-24 DTCMSZ	Tightly Coupled Data Memory Size Provides an encoded value of the tightly coupled local data memory size. The capacity of the memory is derived using the formula $2^{(8+SZ)}$ and expressed as bytes. Here, TMLSZ is a nonzero value, and TMLSZ = 0 indicates that the memory is not present. For the Cortex-M7 cores in this chip: • DTCMSZ = 8h in Decoupled mode (64 KB) • DTCMSZ = 9h in Lockstep mode (128 KB)
23-16 ITCMSZ	Instruction Tightly Coupled Memory Size Provides an encoded value of the tightly coupled local instruction memory size. The capacity of the memory is derived using the formula $2^{(8+SZ)}$ and expressed as bytes. Here, TMUSZ is a nonzero value, and TMUSZ = 0 indicates that the memory is not present.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	For the Cortex-M7 cores in this chip: • ITCMSZ = 7h in Decoupled mode (32 KB) • ITCMSZ = 8h in Lockstep mode (64 KB)
15-0 —	Reserved

7.6.3.15 Processor 0 Configuration 3 (CP0CFG3)

Offset

Register	Offset
CP0CFG3	38h

Function

Defines the configuration information for processor 0 (CP0). It has the same field definitions and functionality as provided in the CPXCFG3 register.

A privileged read from any bus controller returns the appropriate processor information and attempted write accesses terminate with an error.

Access: User or privileged read-only

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0											CPY	CMP	MMU	SIMD	FPU
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1

Fields

Field	Function
31-5 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
4 CPY	Cryptography Indicates whether the cryptography extensions are supported in the core. For the Cortex-M7 cores in this chip, CPY = 0h. 0b - Not supported 1b - Supported
3 CMP	Core Memory Protection Unit Indicates whether the core memory protection hardware is included in this core. For the Cortex-M7 cores in this chip, CMP = 1h. 0b - Not included 1b - Included
2 MMU	Memory Management Unit Indicates whether virtual management capabilities are supported in this core. For the Cortex-M7 cores in this chip, MMU = 0h. 0b - Not supported 1b - Supported
1 SIMD	SIMD/NEON Instruction Support Indicates whether the instruction set extensions supporting SIMD and/or NEON capabilities are included in the processor. For the Cortex-M7 cores in this chip, SIMD = 1h. 0b - Not included 1b - Included
0 FPU	Floating Point Unit Indicates whether hardware support for floating point capabilities is provided in the processor. For the Cortex-M7 cores in this chip, FPU = 1h. 0b - Not provided 1b - Provided

7.6.3.16 Processor 1 Type (CP1TYPE)

Offset

Register	Offset
CP1TYPE	40h

Function

Defines the configuration information for processor 1 (CP1). It has the same field definitions and functionality as provided in Processor X Type (CPXTYPE).

A read from any bus controller returns the appropriate processor information and attempted write accesses terminate with an error.

Access: User or privileged read-only

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PERSONALITY															
W																
Reset	0	1	0	0	0	0	1	1	0	1	0	0	1	1	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PERSONALITY															
W																
Reset	0	0	1	1	0	1	1	1	0	0	1	1	0	0	0	1

Fields

Field	Function
31-0 PERSONALITY	Personality Processor Defines the processor personality for CP1. CP1 = Cortex-M7 core 1 and processor personality = 43_4D_37_31h.

7.6.3.17 Processor 1 Number (CP1NUM)

Offset

Register	Offset
CP1NUM	44h

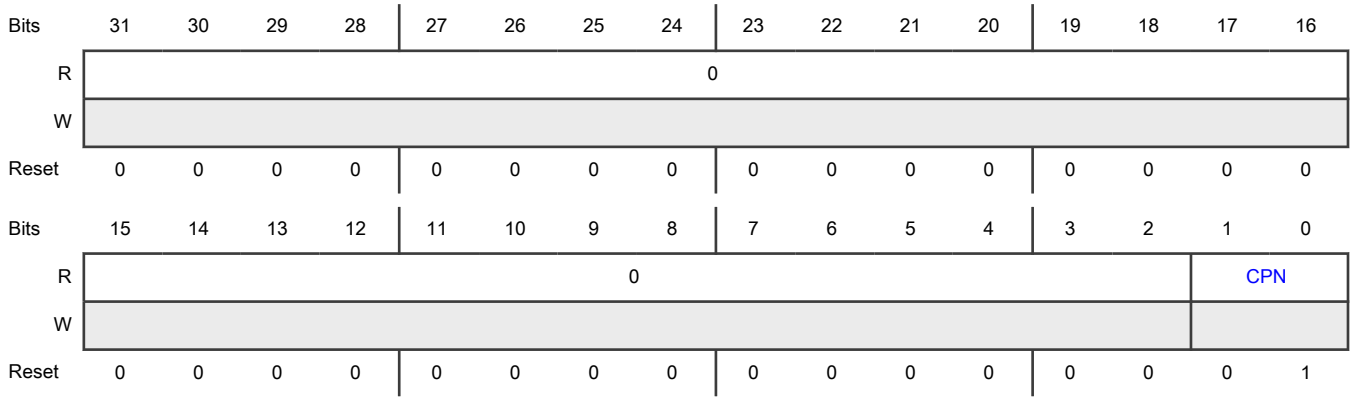
Function

Defines the configuration information for processor 1 (CP1). It has the same field definitions and functionality as provided in Processor X Number (CPXNUM).

A read from any bus controller returns the appropriate processor information and attempted write accesses terminate with an error.

Access: User or privileged read-only

Diagram



Fields

Field	Function
31-2 —	Reserved
1-0 CPN	Processor Number Defines the logical processor number for CP1. For the Cortex-M7 core 1 processor, the processor number = 1. The processor personality is 0000_0001h.

7.6.3.18 Processor 1 Count (CP1REV)

Offset

Register	Offset
CP1REV	48h

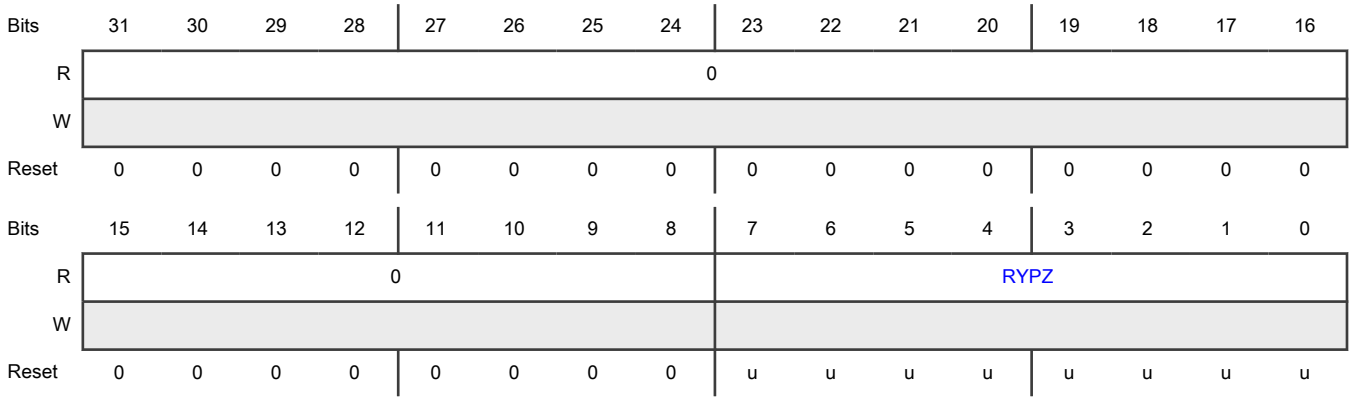
Function

Defines the configuration information for processor 1 (CP1). It has the same field definitions and functionality as provided in Processor X Revision (CPXREV).

A read from any bus controller returns the appropriate processor information and attempted write accesses terminate with an error.

Access: User or privileged read-only

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 RYPZ	Processor Revision Defines the processor revision for CP1. For the Cortex-M7 processor, RYPZ = 12h corresponding to the r1p2 core release.

7.6.3.19 Processor 1 Configuration 0 (CP1CFG0)

Offset

Register	Offset
CP1CFG0	4Ch

Function

Defines the configuration information for processor 1 (CP1). It has the same field definitions and functionality as provided in Processor X Configuration 0 (CPXCFG0).

A privileged read from any bus controller returns the appropriate processor information and attempted write accesses terminate with an error.

Access: User or privileged read-only

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ICSZ								ICWY							
W																
Reset	0	0	0	0	0	1	0	1	0	0	0	0	0	0	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DCSZ								DCWY							
W																
Reset	0	0	0	0	0	1	0	1	0	0	0	0	0	1	0	0

Fields

Field	Function
31-24 ICSZ	Level 1 Instruction Cache Size Provides an encoded value of the instruction cache size. The capacity of the memory is derived using the formula $2^{(8+SZ)}$ and expressed as bytes. Here, ICSZ is a nonzero value, and ICSZ = 0 indicates that the memory is not present. For information about cache sizes, see the "Miscellaneous Control Module (MCM)" chapter.
23-16 ICWY	Level 1 Instruction Cache Ways Provides the number of cache ways for the instruction cache. For the Cortex-M7 cores in this chip, ICWY = 2h (2-way set-associative).
15-8 DCSZ	L1 Data Cache Size Provides an encoded value of the data cache size. The capacity of the memory is derived using the formula $2^{(8+SZ)}$ and expressed as bytes. Here, DCSZ is a nonzero value, and DCSZ = 0 indicates that the memory is not present. For information about cache sizes, see the "Miscellaneous Control Module (MCM)" chapter.
7-0 DCWY	L1 Data Cache Ways Provides the number of cache ways for the data cache. For the Cortex-M7 cores in this chip, DCWY = 4h (4-way set-associative).

7.6.3.20 Processor 1 Configuration 1 (CP1CFG1)

Offset

Register	Offset
CP1CFG1	50h

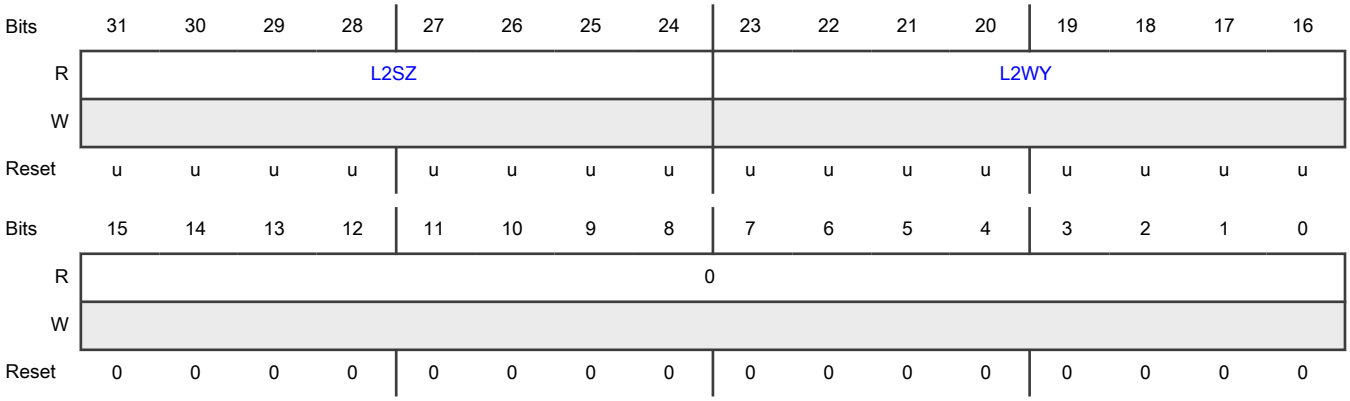
Function

Defines the configuration information for processor 1 (CP1). It has the same field definitions and functionality as provided in Processor X Configuration 1 (CPXCFG1).

A privileged read from any bus controller returns the appropriate processor information and attempted write accesses terminate with an error.

Access: User or privileged read-only

Diagram



Fields

Field	Function
31-24 L2SZ	L2 Cache Size Provides an encoded value of the L2 cache size. The capacity of the memory is derived using the formula $2^{(8+SZ)}$ and expressed as bytes. Here, L2SZ is a nonzero value, and L2SZ = 0 indicates that the memory is not present. For the Cortex-M7 cores in this chip, L2SZ = 0h (not present).
23-16 L2WY	L2 Cache Ways Provides the number of cache ways for the L2 cache. For the Cortex-M7 cores in this chip, L2WY = 0h (not present).
15-0 —	Reserved

7.6.3.21 Processor 1 Configuration 2 (CP1CFG2)

Offset

Register	Offset
CP1CFG2	54h

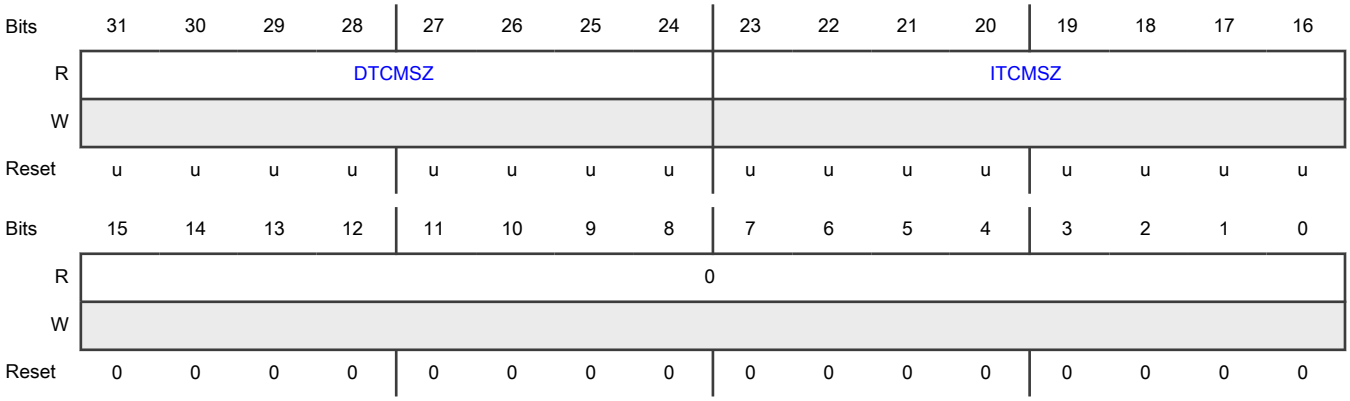
Function

Defines the configuration information for processor 1 (CP1). It has the same field definitions and functionality as provided in Processor X Configuration 2 (CPXCFG2).

A privileged read from any bus controller returns the appropriate processor information and attempted write accesses terminate with an error.

Access: User or privileged read-only

Diagram



Fields

Field	Function
31-24 DTCMSZ	Tightly Coupled Data Memory Size Provides an encoded value of the tightly coupled local data memory size. The capacity of the memory is derived using the formula $2^{(8+SZ)}$ and expressed as bytes. Here, TMLSZ is a nonzero value, and TMLSZ = 0 indicates that the memory is not present. For the Cortex-M7 cores in this chip, DTCMSZ = 8h in Decoupled mode (64 KB), and this value is not applicable in Lockstep mode.
23-16 ITCMSZ	Instruction Tightly Coupled Memory Size Provides an encoded value of the tightly coupled local instruction memory size. The capacity of the memory is derived using the formula $2^{(8+SZ)}$ and expressed as bytes. Here, TMUSZ is a nonzero value, and TMUSZ = 0 indicates that the memory is not present. For the Cortex-M7 cores in this chip, ITCMSZ = 7h in Decoupled mode (32 KB); this value is not applicable in Lockstep mode.
15-0 —	Reserved

7.6.3.22 Processor 1 Configuration 3 (CP1CFG3)

Offset

Register	Offset
CP1CFG3	58h

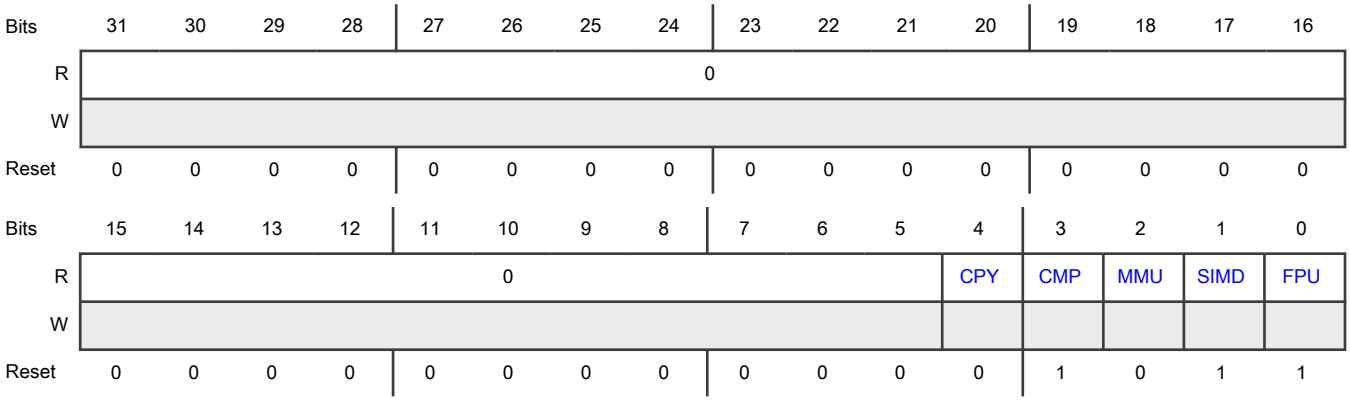
Function

Defines the configuration information for processor 1 (CP1). It has the same field definitions and functionality as provided in the CPXCFG3 register.

A privileged read from any bus controller returns the appropriate processor information and attempted write accesses terminate with an error.

Access: User or privileged read-only

Diagram



Fields

Field	Function
31-5 —	Reserved
4 CPY	Cryptography Indicates whether cryptography extensions are supported in the core. For the Cortex-M7 cores in this chip, CPY = 0h. 0b - Not supported 1b - Supported
3 CMP	Core Memory Protection Unit Indicates whether the core memory protection hardware is included in this core. For the Cortex-M7 cores in this chip, CMP = 1h.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Not included 1b - Included
2 MMU	Memory Management Unit Indicates whether virtual management capabilities are supported in this core. For the Cortex-M7 cores in this chip, MMU = 0h. 0b - Not supported 1b - Supported
1 SIMD	SIMD/NEON Instruction Support Indicates whether the instruction set extensions supporting SIMD and/or NEON capabilities are included in the processor. For the Cortex-M7 cores in this chip, SIMD = 1h. 0b - Not included 1b - Included
0 FPU	Floating Point Unit Indicates whether the processor includes hardware support for floating point capabilities. For the Cortex-M7 cores in this chip, FPU = 1h. 0b - Not included 1b - Included

7.6.3.23 Processor 2 Type (CP2TYPE)

Offset

Register	Offset
CP2TYPE	60h

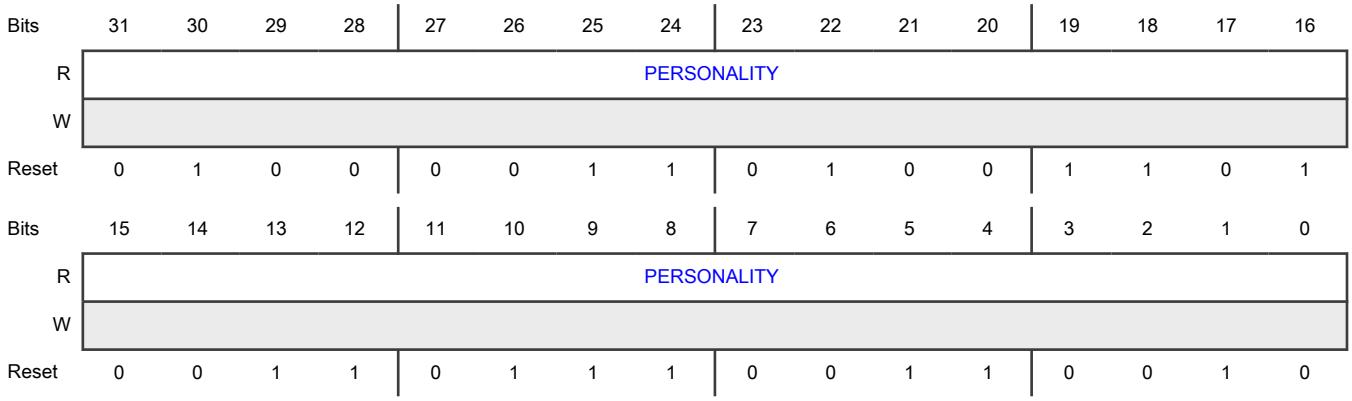
Function

Defines the configuration information for processor 2 (CP2). It has the same field definitions and functionality as provided in Processor X Type (CPXTYPE).

A read from any bus controller returns the appropriate processor information and attempted write accesses terminate with an error.

Access: User or privileged read-only

Diagram



Fields

Field	Function
31-0 PERSONALITY	Processor Personality Defines the processor personality for CP2. CP2 = Cortex-M7 core 2 and processor personality = 43_4D_37_32h.

7.6.3.24 Processor 2 Number (CP2NUM)

Offset

Register	Offset
CP2NUM	64h

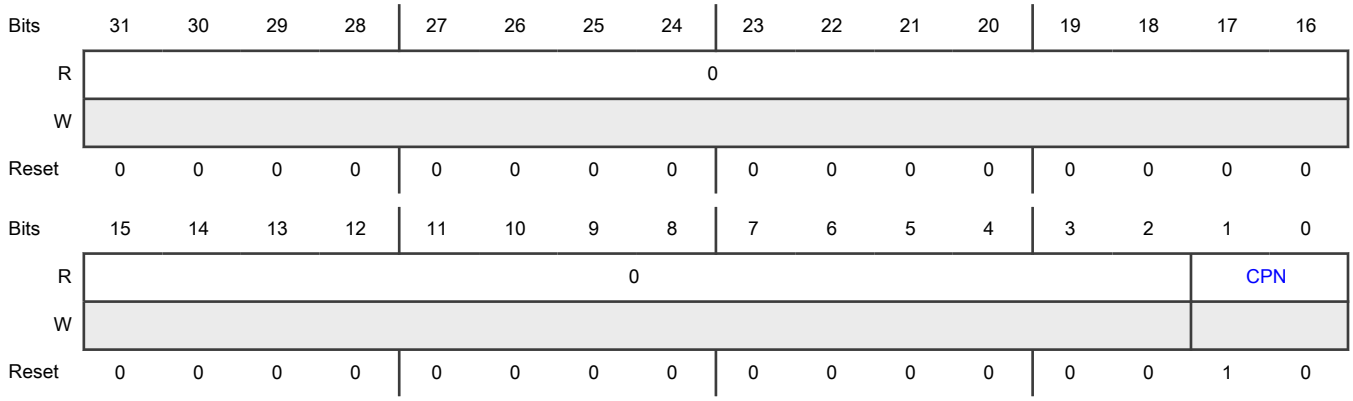
Function

Defines the configuration information for processor 2 (CP2). It has the same field definitions and functionality as provided in Processor X Number (CPXNUM).

A read from any bus controller returns the appropriate processor information and attempted write accesses terminate with an error.

Access: User or privileged read-only

Diagram



Fields

Field	Function
31-2 —	Reserved
1-0 CPN	Processor Number Defines the logical processor number for CP2. For Cortex-M7 core 2, processor number = 2. The processor personality is 0000_0002h.

7.6.3.25 Processor 2 Count (CP2REV)

Offset

Register	Offset
CP2REV	68h

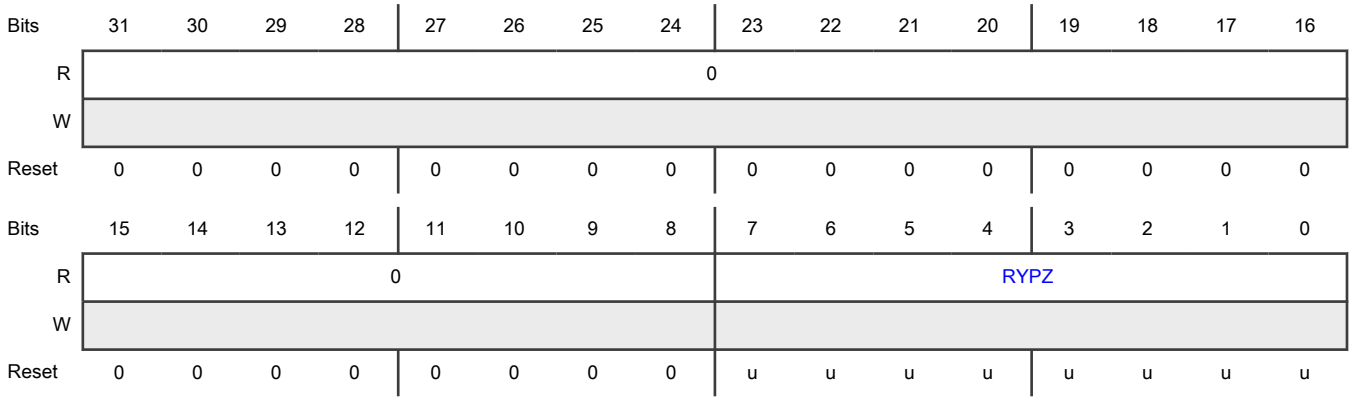
Function

Defines the configuration information for processor 2 (CP2). It has the same field definitions and functionality as provided in Processor X Revision (CPXREV).

A read from any bus controller returns the appropriate processor information and attempted write accesses terminate with an error.

Access: User or privileged read-only

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 RYPZ	Processor Revision Defines the processor revision for CP2. For the Cortex-M7 processor, RYPZ = 12h corresponding to the r1p2 core release.

7.6.3.26 Processor 2 Configuration 0 (CP2CFG0)

Offset

Register	Offset
CP2CFG0	6Ch

Function

Defines the configuration information for processor 2 (CP2). It has the same field definitions and functionality as provided in Processor X Configuration 0 (CPXCFG0).

A privileged read from any bus controller returns the appropriate processor information and attempted write accesses terminate with an error.

Access: User or privileged read-only

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ICSZ								ICWY							
W																
Reset	0	0	0	0	0	1	1	0	0	0	0	0	0	0	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DCSZ								DCWY							
W																
Reset	0	0	0	0	0	1	1	0	0	0	0	0	0	1	0	0

Fields

Field	Function
31-24 ICSZ	Level 1 Instruction Cache Size Provides an encoded value of the instruction cache size. The capacity of the memory is derived using the formula $2^{(8+SZ)}$ and expressed as bytes. Here, ICSZ is a nonzero value, and ICSZ = 0 indicates that the memory is not present. For information about cache sizes, see the "Miscellaneous Control Module (MCM)" chapter.
23-16 ICWY	Level 1 Instruction Cache Ways Provides the number of cache ways for the instruction cache. For the Cortex-M7 cores in this chip, ICWY = 2h (2-way set-associative).
15-8 DCSZ	L1 Data Cache Size Provides an encoded value of the data cache size. The capacity of the memory is derived using the formula $2^{(8+SZ)}$ and expressed as bytes. Here, DCSZ is a nonzero value, and DCSZ = 0 indicates that the memory is not present. For information about cache sizes, see the "Miscellaneous Control Module (MCM)" chapter.
7-0 DCWY	L1 Data Cache Ways Provides the number of cache ways for the data cache. For the Cortex-M7 cores in this chip, DCWY = 4h (4-way set-associative).

7.6.3.27 Processor 2 Configuration 1 (CP2CFG1)

Offset

Register	Offset
CP2CFG1	70h

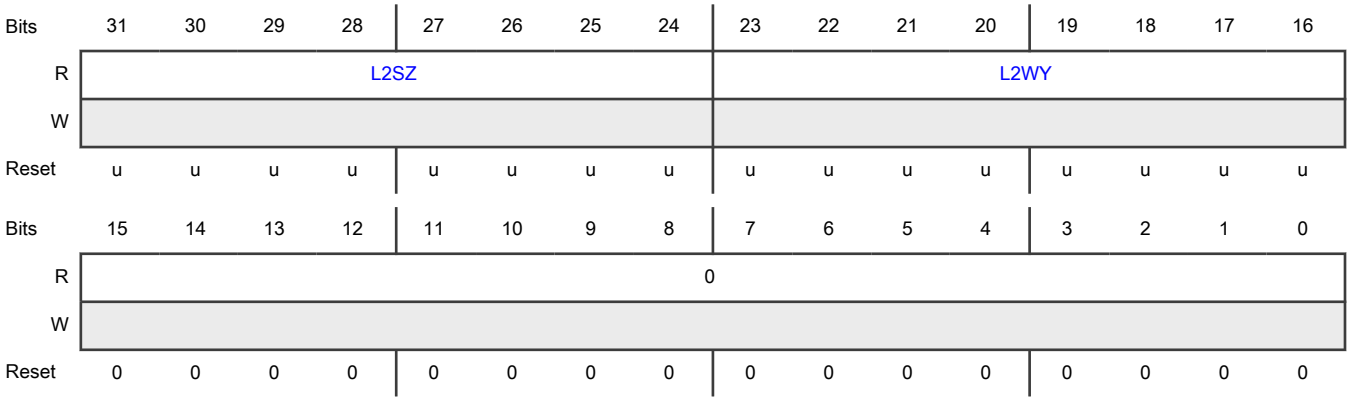
Function

Defines the configuration information for processor 2 (CP2). It has the same field definitions and functionality as provided in Processor X Configuration 1 (CPXCFG1).

A privileged read from any bus controller returns the appropriate processor information and attempted write accesses terminate with an error.

Access: User or privileged read-only

Diagram



Fields

Field	Function
31-24 L2SZ	L2 Cache Size Provides an encoded value of the L2 cache size. The capacity of the memory is derived using the formula $2^{(8+SZ)}$ and expressed as bytes. Here, L2SZ is a nonzero value, and L2SZ = 0 indicates that the memory is not present. For the Cortex-M7 cores in this chip, L2SZ = 0h (not present).
23-16 L2WY	L2 Cache Ways Provides the number of cache ways for the L2 cache. For the Cortex-M7 cores in this chip, L2WY = 0h (not present).
15-0 —	Reserved

7.6.3.28 Processor 2 Configuration 2 (CP2CFG2)

Offset

Register	Offset
CP2CFG2	74h

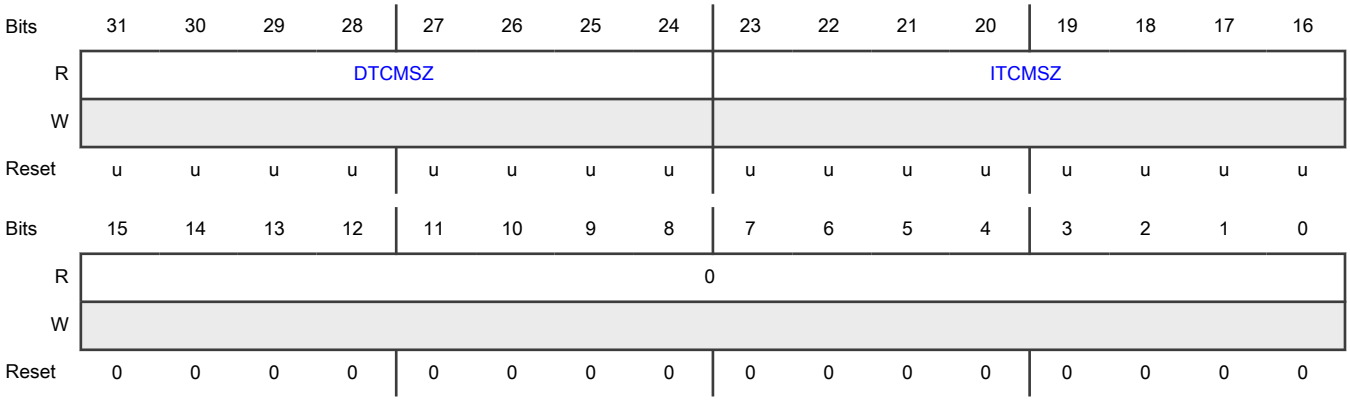
Function

Defines the configuration information for processor 2 (CP2). It has the same field definitions and functionality as provided in Processor X Configuration 2 (CPXCFG2).

A privileged read from any bus controller returns the appropriate processor information and attempted write accesses terminate with an error.

Access: User or privileged read-only

Diagram



Fields

Field	Function
31-24 DTCMSZ	Tightly Coupled Data Memory Size Provides an encoded value of the tightly coupled local data memory size. The capacity of the memory is derived using the formula $2^{(8+SZ)}$ and expressed as bytes. Here, TMLSZ is a nonzero value, and TMLSZ = 0 indicates that the memory is not present. For the Cortex-M7 cores in this chip, DTCMSZ = 8h in Decoupled mode (64 KB), and this value is not applicable in Lockstep mode.
23-16 ITCMSZ	Instruction Tightly Coupled Memory Size Provides an encoded value of the tightly coupled local instruction memory size. The capacity of the memory is derived using the formula $2^{(8+SZ)}$ and expressed as bytes. Here, TMUSZ is a nonzero value, and TMUSZ = 0 indicates that the memory is not present. For the Cortex-M7 cores in this chip, ITCMSZ = 7h in Decoupled mode (32 KB); this value is not applicable in Lockstep mode.
15-0 —	Reserved

7.6.3.29 Processor 2 Configuration 3 (CP2CFG3)

Offset

Register	Offset
CP2CFG3	78h

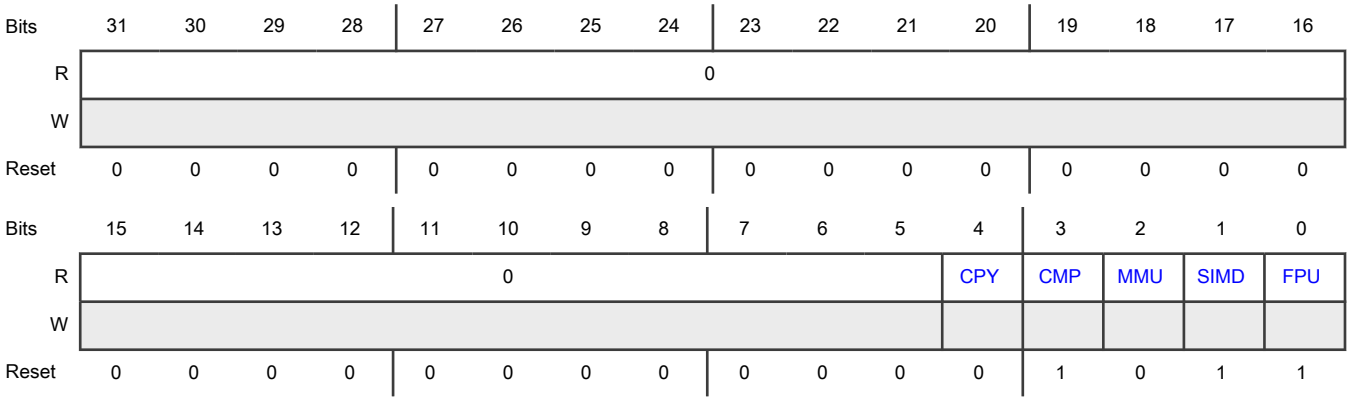
Function

Defines the configuration information for processor 2 (CP2). It has the same field definitions and functionality as provided in the CPXCFG3 register.

A privileged read from any bus controller returns the appropriate processor information and attempted write accesses terminate with an error.

Access: User or privileged read-only

Diagram



Fields

Field	Function
31-5 —	Reserved
4 CPY	Cryptography Indicates whether cryptography extensions are supported in the core. For the Cortex-M7 cores in this chip, CPY = 0h. 0b - Not supported 1b - Supported
3 CMP	Core Memory Protection Unit Indicates whether the core memory protection hardware is included in this core. For the Cortex-M7 cores in this chip, CMP = 1h.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Not included 1b - Included
2 MMU	Memory Management Unit Indicates whether virtual management capabilities are supported in this core. For the Cortex-M7 cores in this chip, MMU = 0h. 0b - Not supported 1b - Supported
1 SIMD	SIMD/NEON Instruction Support Indicates whether the instruction set extensions supporting SIMD and/or NEON capabilities are included in the processor. For the Cortex-M7 cores in this chip, SIMD = 1h. 0b - Not included 1b - Included
0 FPU	Floating Point Unit Indicates whether the processor includes hardware support for floating point capabilities. For the Cortex-M7 cores in this chip, FPU = 1h. 0b - Not included 1b - Included

7.6.3.30 Processor 3 Type (CP3TYPE)

Offset

Register	Offset
CP3TYPE	80h

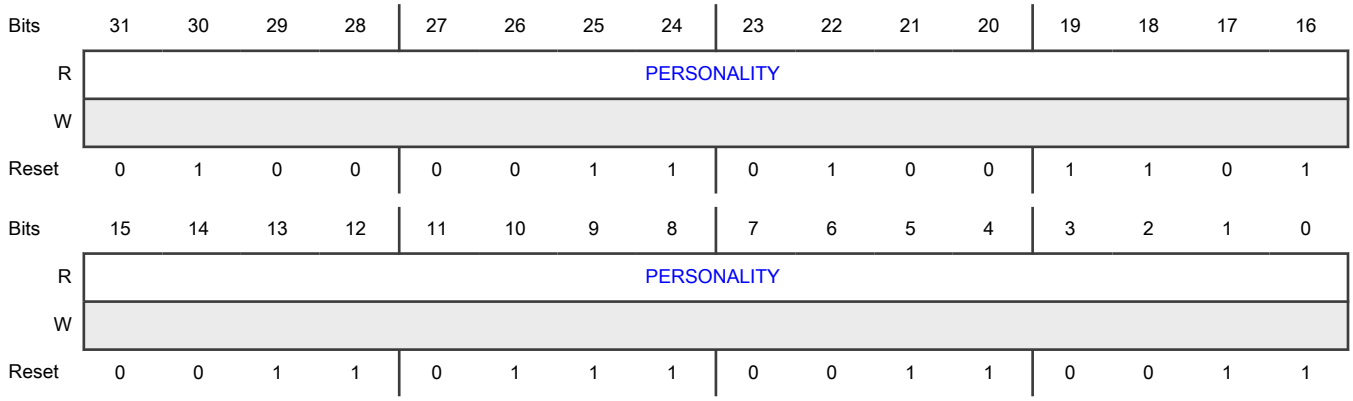
Function

Defines the configuration information for processor 3 (CP3). It has the same field definitions and functionality as provided in Processor X Type (CPXTYPE).

A read from any bus controller returns the appropriate processor information and attempted write accesses terminate with an error.

Access: User or privileged read-only

Diagram



Fields

Field	Function
31-0 PERSONALITY	Processor Personality Defines the processor personality for CP3. CP3 = Cortex-M7 core 3 and processor personality = 43_4D_37_33h.

7.6.3.31 Processor 3 Number (CP3NUM)

Offset

Register	Offset
CP3NUM	84h

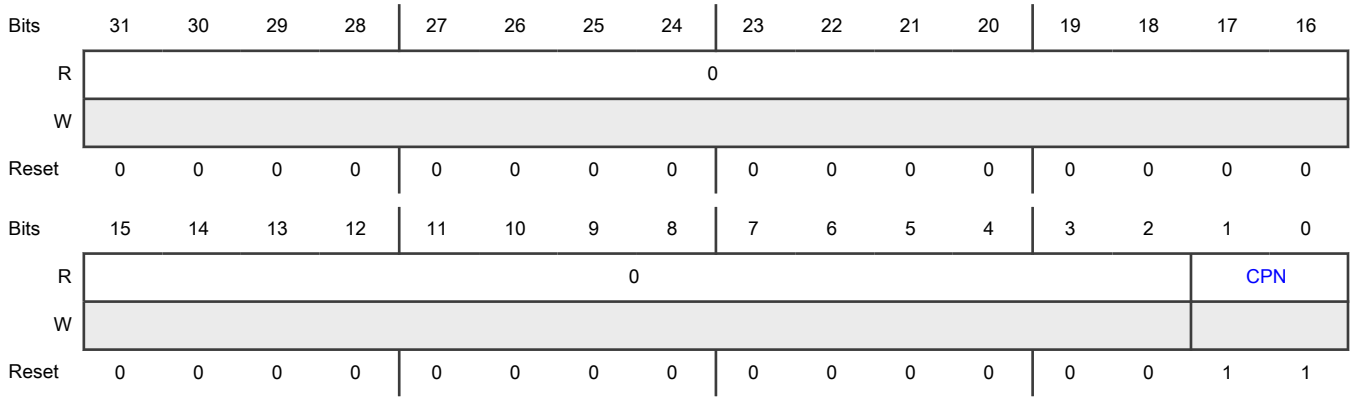
Function

Defines the configuration information for processor 3 (CP3). It has the same field definitions and functionality as provided in Processor X Number (CPXNUM).

A read from any bus controller returns the appropriate processor information and attempted write accesses terminate with an error.

Access: User or privileged read-only

Diagram



Fields

Field	Function
31-2 —	Reserved
1-0 CPN	Processor Number Defines the logical processor number for CP3. For Cortex-M7 core 3, processor number = 3. The processor personality is 0000_0003h.

7.6.3.32 Processor 3 Count (CP3REV)

Offset

Register	Offset
CP3REV	88h

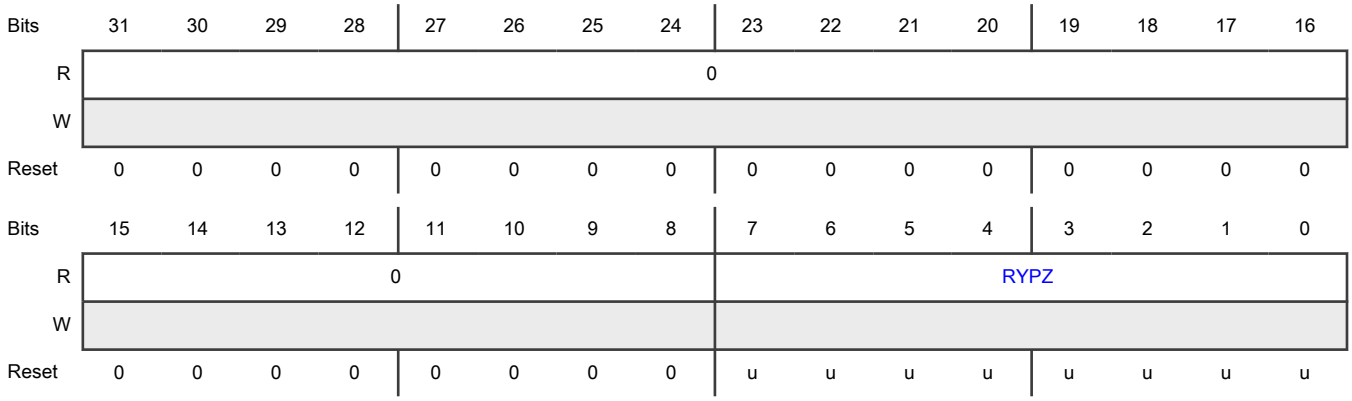
Function

Defines the configuration information for processor 3 (CP3). It has the same field definitions and functionality as provided in Processor X Revision (CPXREV).

A read from any bus controller returns the appropriate processor information and attempted write accesses terminate with an error.

Access: User or privileged read-only

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 RYPZ	Processor Revision Defines the processor revision for CP3. For the Cortex-M7 processor, RYPZ = 12h corresponding to the r1p2 core release.

7.6.3.33 Processor 3 Configuration 0 (CP3CFG0)

Offset

Register	Offset
CP3CFG0	8Ch

Function

Defines the configuration information for processor 3 (CP3). It has the same field definitions and functionality as provided in Processor X Configuration 0 (CPXCFG0).

A privileged read from any bus controller returns the appropriate processor information and attempted write accesses terminate with an error.

Access: User or privileged read-only

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ICSZ								ICWY							
W																
Reset	0	0	0	0	0	1	1	0	0	0	0	0	0	0	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DCSZ								DCWY							
W																
Reset	0	0	0	0	0	1	1	0	0	0	0	0	0	1	0	0

Fields

Field	Function
31-24 ICSZ	Level 1 Instruction Cache Size Provides an encoded value of the instruction cache size. The capacity of the memory is derived using the formula $2^{(8+SZ)}$ and expressed as bytes. Here, ICSZ is a nonzero value, and ICSZ = 0 indicates that the memory is not present. For information about cache sizes, see the "Miscellaneous Control Module (MCM)" chapter.
23-16 ICWY	Level 1 Instruction Cache Ways Provides the number of cache ways for the instruction cache. For the Cortex-M7 cores in this chip, ICWY = 2h (2-way set-associative).
15-8 DCSZ	L1 Data Cache Size Provides an encoded value of the data cache size. The capacity of the memory is derived using the formula $2^{(8+SZ)}$ and expressed as bytes. Here, DCSZ is a nonzero value, and DCSZ = 0 indicates that the memory is not present. For information about cache sizes, see the "Miscellaneous Control Module (MCM)" chapter.
7-0 DCWY	L1 Data Cache Ways Provides the number of cache ways for the data cache. For the Cortex-M7 cores in this chip, DCWY = 4h (4-way set-associative).

7.6.3.34 Processor 3 Configuration 1 (CP3CFG1)

Offset

Register	Offset
CP3CFG1	90h

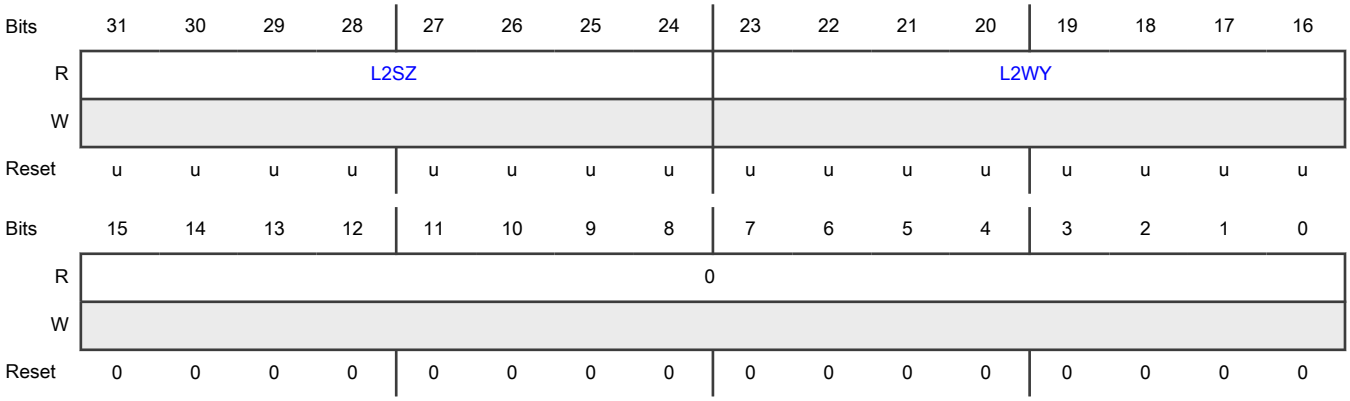
Function

Defines the configuration information for processor 3 (CP3). It has the same field definitions and functionality as provided in Processor X Configuration 1 (CPXCFG1).

A privileged read from any bus controller returns the appropriate processor information and attempted write accesses terminate with an error.

Access: User or privileged read-only

Diagram



Fields

Field	Function
31-24 L2SZ	L2 Cache Size Provides an encoded value of the L2 cache size. The capacity of the memory is derived using the formula $2^{(8+SZ)}$ and expressed as bytes. Here, L2SZ is a nonzero value, and L2SZ = 0 indicates that the memory is not present. For the Cortex-M7 cores in this chip, L2SZ = 0h (not present).
23-16 L2WY	L2 Cache Ways Provides the number of cache ways for the L2 cache. For the Cortex-M7 cores in this chip, L2WY = 0h (not present).
15-0 —	Reserved

7.6.3.35 Processor 3 Configuration 2 (CP3CFG2)

Offset

Register	Offset
CP3CFG2	94h

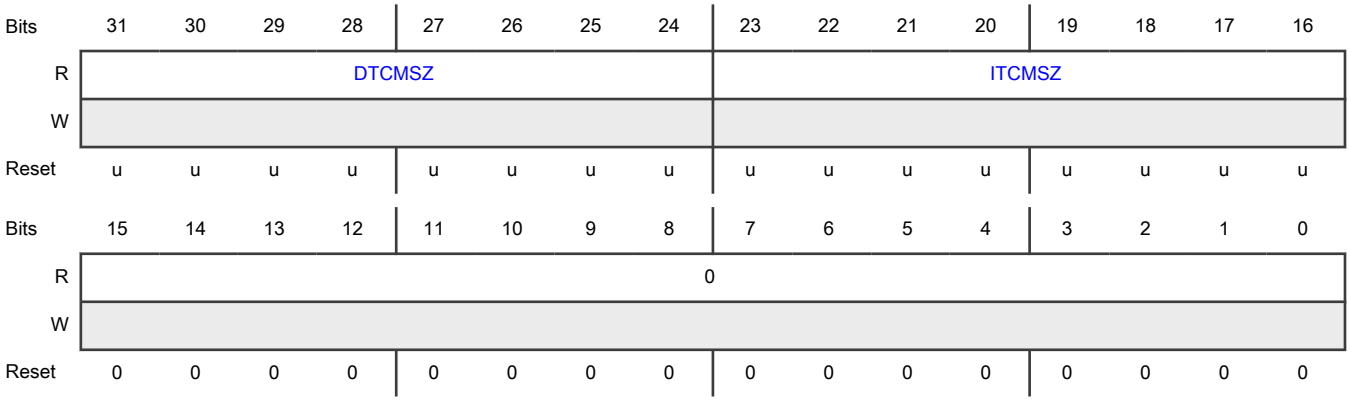
Function

Defines the configuration information for processor 3 (CP3). It has the same field definitions and functionality as provided in Processor X Configuration 2 (CPXCFG2).

A privileged read from any bus controller returns the appropriate processor information and attempted write accesses terminate with an error.

Access: User or privileged read-only

Diagram



Fields

Field	Function
31-24 DTCMSZ	Tightly Coupled Data Memory Size Provides an encoded value of the tightly coupled local data memory size. The capacity of the memory is derived using the formula $2^{(8+SZ)}$ and expressed as bytes. Here, TMLSZ is a nonzero value, and TMLSZ = 0 indicates that the memory is not present. For the Cortex-M7 cores in this chip, DTCMSZ = 8h in Decoupled mode (64 KB), and this value is not applicable in Lockstep mode.
23-16 ITCMSZ	Instruction Tightly Coupled Memory Size Provides an encoded value of the tightly coupled local instruction memory size. The capacity of the memory is derived using the formula $2^{(8+SZ)}$ and expressed as bytes. Here, TMUSZ is a nonzero value, and TMUSZ = 0 indicates that the memory is not present. For the Cortex-M7 cores in this chip, ITCMSZ = 7h in Decoupled mode (32 KB); this value is not applicable in Lockstep mode.
15-0 —	Reserved

7.6.3.36 Processor 3 Configuration 3 (CP3CFG3)

Offset

Register	Offset
CP3CFG3	98h

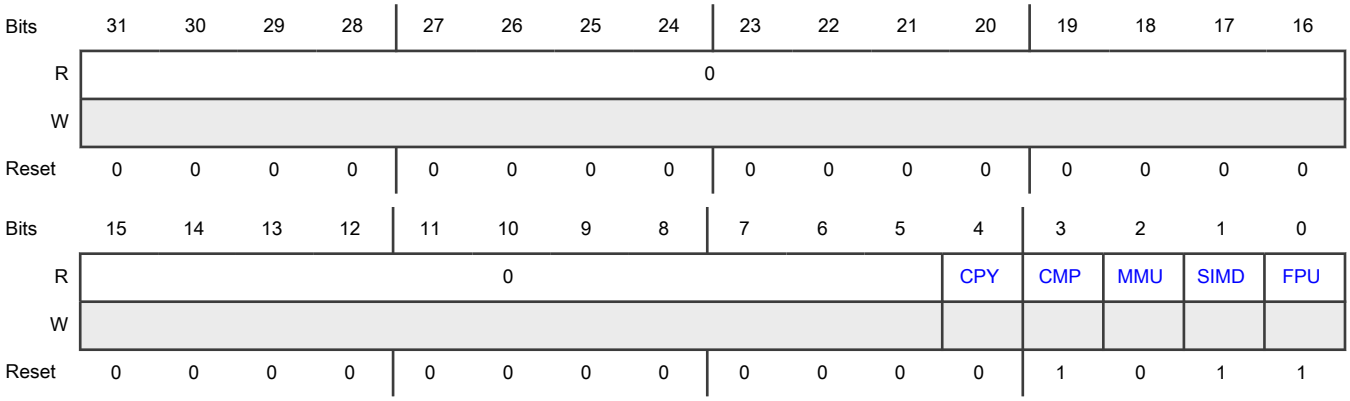
Function

Defines the configuration information for processor 3 (CP3). It has the same field definitions and functionality as provided in the CPXCFG3 register.

A privileged read from any bus controller returns the appropriate processor information and attempted write accesses terminate with an error.

Access: User or privileged read-only

Diagram



Fields

Field	Function
31-5 —	Reserved
4 CPY	Cryptography Indicates whether cryptography extensions are supported in the core. For the Cortex-M7 cores in this chip, CPY = 0h. 0b - Not supported 1b - Supported
3 CMP	Core Memory Protection Unit Indicates whether the core memory protection hardware is included in this core. For the Cortex-M7 cores in this chip, CMP = 1h.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Not included 1b - Included
2 MMU	Memory Management Unit Indicates whether virtual management capabilities are supported in this core. For the Cortex-M7 cores in this chip, MMU = 0h. 0b - Not supported 1b - Supported
1 SIMD	SIMD/NEON Instruction Support Indicates whether the instruction set extensions supporting SIMD and/or NEON capabilities are included in the processor. For the Cortex-M7 cores in this chip, SIMD = 1h. 0b - Not included 1b - Included
0 FPU	Floating Point Unit Indicates whether the processor includes hardware support for floating point capabilities. For the Cortex-M7 cores in this chip, FPU = 1h. 0b - Not included 1b - Included

7.6.3.37 Interrupt Router CPn Interrupt Status (IRCP0ISR0 - IRCP3ISR3)

Offset

For n = 0 to 3; m = 0 to 3:

Register	Offset
IRCPnISRm	200h + (n × 20h) + (m × 8h)

Function

Provides an interrupt bit map, where each bit defines the state of a unique MSI based on the initiating core. An MSI interrupt clears in an interrupt service routine by writing 1 to the appropriate field in IRCP*n*ISR*m*.

In this discussion, CP*m* represents the initiating core and CP*n* represents the target core for a core-to-core interrupt. For more information on interrupt source mapping, see the interrupt map file attached to this document.

For read access:

- Reads to IRCP*n*ISR*m* are only accessible in Privileged mode using 32-bit (word) accesses.
- Privileged 32-bit read accesses from noncore (and nondebugger) bus controllers are treated as RAZ.

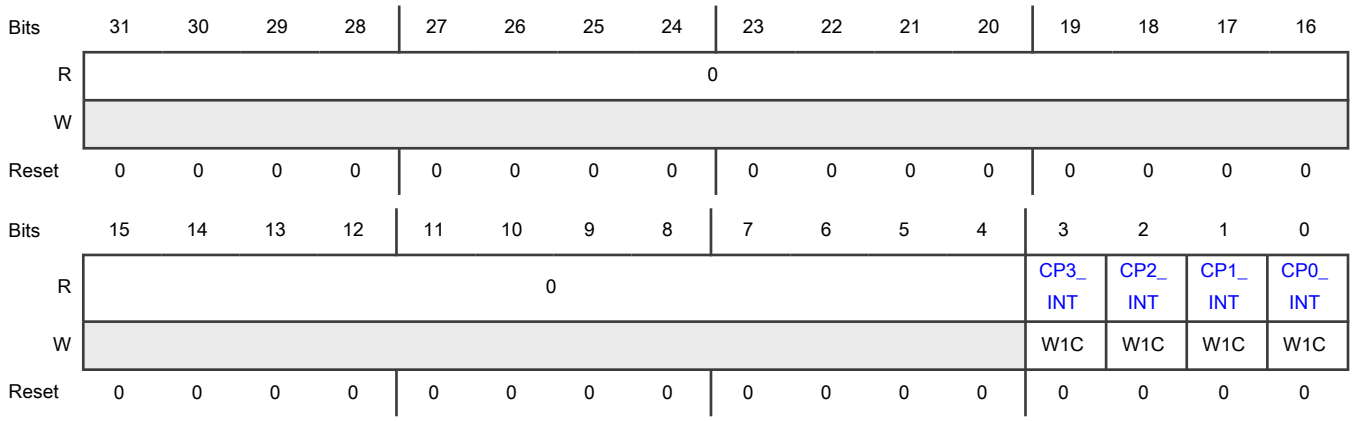
- Attempted accesses in User mode or the ones using a size other than 32 bits are not permitted. They terminate with an error.
- When CP*n* requests to read IRCP*n*ISR*m*, MSCM returns the entire content of IRCP*n*ISR*m*.
- When a trusted core, as the IRCPCFG register indicates, requests to read IRCP*n*ISR*m*, MSCM returns the entire content of IRCP*n*ISR*m*.
- When the debugger requests to read IRCP*n*ISR*m*, MSCM returns the entire content of IRCP*n*ISR*m*.
- When CP*m* requests to read IRCP*n*ISR*m*, MSCM returns the value of the corresponding status, CP*m*_INT, while not exposing all the other pending interrupts that the cores initiated.
- When CP*m* requests to read IRCP*n*ISR*m*, MSCM returns the value of the corresponding status, CP*m*_INT, in bit position 0, reflecting how CP*m* set the MSI when it wrote to IRCP*n*IGR*m*. All the other fields on the returned read value are zero-filled.

For write access:

- Writes to IRCP*n*ISR*m* are only accessible in Privileged mode using 32-bit (word) accesses.
- Attempted accesses in User mode or the ones using a size other than 32 bits are not permitted. They terminate with an error.
- Writes to IRCP*n*ISR*m* follow the Write 1 to Clear (W1C) protocol, whereby writing 1 causes the corresponding field to become 0, and writing 0 is ignored.
- The target core, CP*n*, has full access to write to all the fields of IRCP*n*ISR*m*.
- A trusted core, as the IRCPCFG register indicates, has full access to write to all the fields of IRCP*n*ISR*m*.
- When CP*m* is different from CP*n*, the W1C action by CP*m* only clears IRCP*n*ISR*m*[CP*m*_INT].
- The CP*m* field must present W1C in bit position 0 to clear its corresponding interrupt. Write data bits 1-31 that CP*m* presents are ignored.
- Privileged write accesses from the noncore (and nondebugger) bus controllers are treated as Writes Ignored (WI).
- Privileged write accesses from the debugger are treated as WI.

Access: Privileged mode only

Diagram



Fields

Field	Function
31-4	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
3 CP3_INT	CP3-to-CPn Interrupt Generates a directed interrupt initiated by core 3 targeting core *n*, if the appropriate interrupt routing bit is enabled. The interrupt is negated when the target core, a trusted core, or core 3 writes 1 to clear the field. 0b - No interrupt is asserted to CPn 1b - Interrupt to CPn is asserted
2 CP2_INT	CP2-to-CPn Interrupt Generates a directed interrupt initiated by core 2 targeting core *n*, if the appropriate interrupt routing bit is enabled. The interrupt is negated when the target core, a trusted core, or core 2 writes 1 to clear the field. 0b - No interrupt is asserted to CPn 1b - Interrupt to CPn is asserted
1 CP1_INT	CP1-to-CPn Interrupt Generates a directed interrupt initiated by core 1 targeting core *n*, if the appropriate interrupt routing bit is enabled. The interrupt is negated when the target core, a trusted core, or core 1 writes 1 to clear the field. 0b - No interrupt is asserted to CPn 1b - Interrupt to CPn is asserted
0 CP0_INT	CP0-to-CPn Interrupt Generates a directed interrupt initiated by core 0 targeting core *n*, if the appropriate interrupt routing bit is enabled. The interrupt is negated when the target core, a trusted core, or core 0 writes 1 to clear the field. 0b - No interrupt asserted to CPn 1b - Interrupt to CPn asserted

7.6.3.38 Interrupt Router CPn Interrupt Generation (IRCP0IGR0 - IRCP3IGR3)

Offset

For n = 0 to 3; m = 0 to 3:

Register	Offset
IRCPnIGRm	204h + (n × 20h) + (m × 8h)

Function

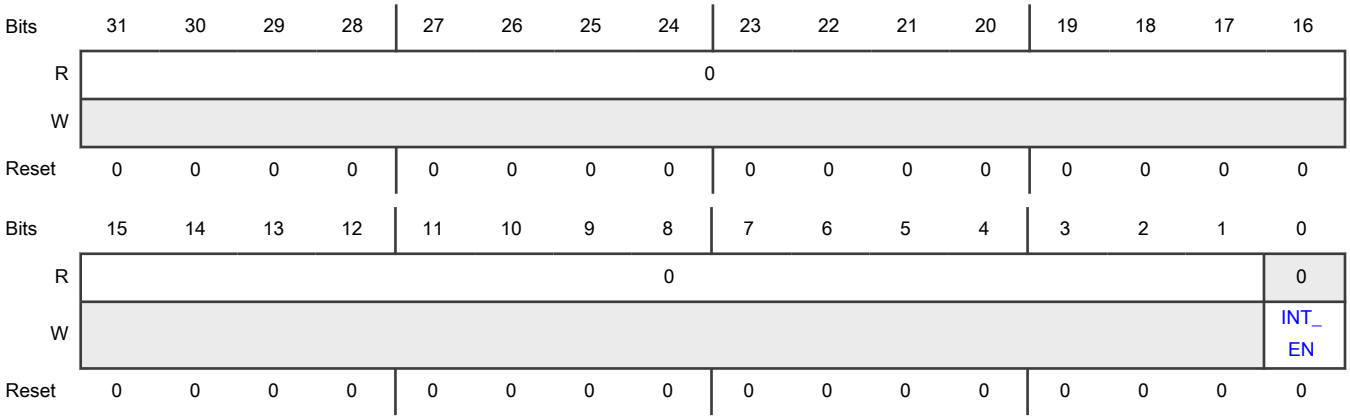
Provides a mechanism for cores to initiate an MSI to another core in the system.

Privileged, 32-bit accesses from the:

- Cortex-M7 cores are treated as RAZ/W.
- Debugger are treated as RAZ/WI.
- Noncore (and nondebugger) bus controllers are treated as RAZ/WI.

Access: Privileged mode only. Attempted accesses in User mode or the ones using a size other than 32 bits are not permitted and terminate with an error.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 INT_EN	Interrupt Enable Initiates a core-to-core interrupt targeting CP*n*, if CP*m* writes to this field (where *n* indicates the logical core number (0-1) and *m* represents the interrupt number (0-3)).

7.6.3.39 Interrupt Router Configuration (IRCPCFG)

Offset

Register	Offset
IRCPCFG	400h

Function

Provides a mechanism to designate specific cores in the system as trusted. These trusted cores are allowed to access and manage outstanding MSIs.

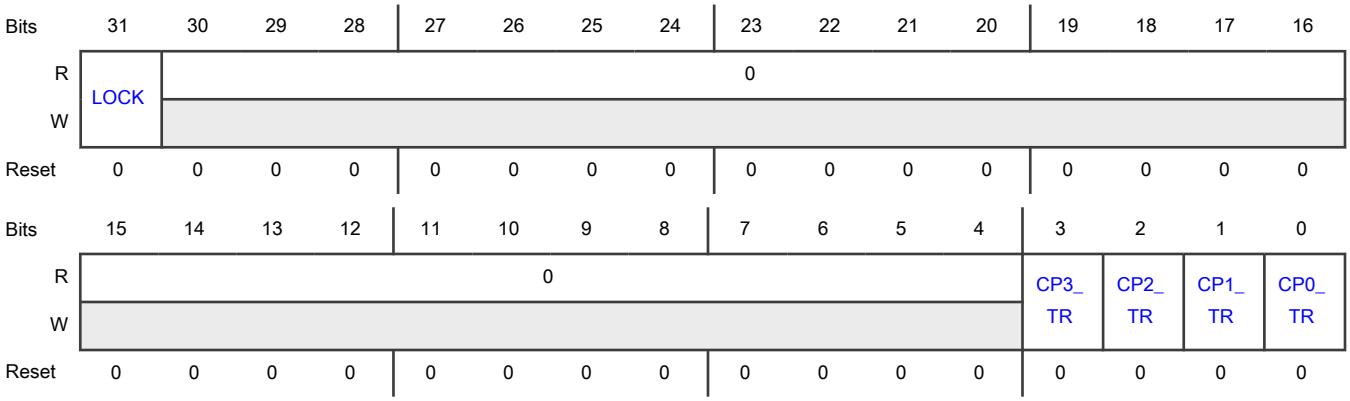
Privileged, 32-bit accesses from the:

- Cortex-M7 cores are treated as R/W.
- Debugger are treated as R/WI.
- Noncore (and nondebugger) bus controllers are treated as RAZ/WI.

Attempted accesses in User mode or the ones using a size other than 32 bits are not permitted. They terminate with an error.

Access: Privileged mode only

Diagram



Fields

Field	Function
31 LOCK	Lock Provides a locking mechanism that can be used to limit the ability to write to the register. After you write 1 to this field, it remains 1 until the next reset. 0b - Register can be written by any privileged write 1b - Register is locked (read-only) until the next reset
30-4 —	Reserved
3 CP3_TR	CP3 as Trusted Core Indicates whether CP3 is a trusted core with access to read the full contents of IRCP*n*ISR*m*. 0b - Not trusted 1b - Trusted
2 CP2_TR	CP2 as Trusted Core Indicates whether CP2 is a trusted core with access to read the full contents of IRCP*n*ISR*m*. 0b - Not trusted 1b - Trusted
1 CP1_TR	CP1 as Trusted Core Indicates whether CP1 is a trusted core with access to read the full contents of IRCP*n*ISR*m*. 0b - Not trusted 1b - Trusted

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 CP0_TR	CP0 as Trusted Core Indicates whether CP0 is a trusted core with access to read the full contents of IRCP*n*ISR*m*. 0b - Not trusted 1b - Trusted

7.6.3.40 Memory Execution Controls (XN_CTRL)

Offset

Register	Offset
XN_CTRL	500h

Function

Controls whether an instruction fetch, also known as a code fetch or executable fetch, is allowed for SRAM and TCM.

NOTE

- This register does not inhibit HSE_B instruction accesses.
- Only PRAM0, PRAM1, and PRAM2 support code execution control.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	HLK	SLK	0						CM7_3_D...	CM7_2_D...	CM7_1_D...	CM7_0_D...	CM7_3_D...	CM7_2_D...	CM7_1_D...	CM7_0_D...
W																
Reset	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CM7_3_I...	CM7_2_I...	CM7_1_I...	CM7_0_I...	CM7_3_D...	CM7_2_D...	CM7_1_D...	CM7_0_D...	0				PRAM_3	PRAM_2	PRAM_1	PRAM_0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 HLK	Hard Lock Enables hard lock. This field locks the register to disable writes until the next hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disable 1b - Enable
30 SLK	Soft Lock Enables soft lock. This field locks the register to disable writes until this field becomes 0. 0b - Disable 1b - Enable
29-24 —	Reserved
23 CM7_3_DTCM	Transaction Control For Cortex-M7_3 DTCM Does not select Cortex-M7_3 as target. All code fetch or executable fetch transactions to Cortex-M7_3 DTCM are not generated because Cortex-M7_3 DTCM is not selected (backdoor access). 0b - Transaction enabled 1b - Transaction disabled
22 CM7_2_DTCM	Transaction Control For Cortex-M7_2 DTCM Does not select Cortex-M7_2 as target. All code fetch or executable fetch transactions to Cortex-M7_2 DTCM are not generated because Cortex-M7_2 DTCM is not selected (backdoor access). 0b - Transaction enabled 1b - Transaction disabled
21 CM7_1_DTCM	Transaction Control For Cortex-M7_1 DTCM Does not select Cortex-M7_1 as target. All code fetch or executable fetch transactions to Cortex-M7_1 DTCM are not generated because Cortex-M7_1 DTCM is not selected (backdoor access). 0b - Transaction enabled 1b - Transaction disabled
20 CM7_0_DTCM	Transaction Control For Cortex-M7_0 DTCM Does not select Cortex-M7_0 as target. All code fetch or executable fetch transactions to Cortex-M7_0 DTCM are not generated because Cortex-M7_0 DTCM is not selected (backdoor access). 0b - Transaction enabled 1b - Transaction disabled
19 CM7_3_DIS_D0_D1TCM_EXEC	Disable D0 and D1 TCM Execution For Cortex-M7_3 Disables D0 and D1 TCM execution for Cortex-M7_3. It is provided to the core. If execution is disabled, it is taken as illegal access by the core.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Enable 1b - Disable
18 CM7_2_DIS_D0_D1TCM_EXEC	Disable D0 and D1 TCM Execution For Cortex-M7_2 Disables D0 and D1 TCM execution for Cortex-M7_2. It is provided to the core. If execution is disabled, it is taken as illegal access by the core. 0b - Enable 1b - Disable
17 CM7_1_DIS_D0_D1TCM_EXEC	D0 and D1 TCM Execution For Cortex-M7_1 Disables D0 and D1 TCM execution for Cortex-M7_1. It is provided to the core. If execution is disabled, it is taken as illegal access by the core. 0b - Enable 1b - Disable
16 CM7_0_DIS_D0_D1TCM_EXEC	D0 And D1 TCM Execution For Cortex-M7_0 Disables D0 and D1 TCM execution for Cortex-M7_0. It is provided to the core. If execution is disabled, it is taken as illegal access by the core. 0b - Enable 1b - Disable
15 CM7_3_ITCM	Transaction Control For Cortex-M7_3 ITCM Does not select Cortex-M7_3 ITCM as target. All code fetch or executable fetch transactions to Cortex-M7_3 ITCM are not generated because Cortex-M7_3 ITCM is not selected (backdoor access). 0b - Execution enabled 1b - Execution disabled
14 CM7_2_ITCM	Transaction Control For Cortex-M7_2 ITCM Does not select Cortex-M7_2 ITCM as target. All code fetch or executable fetch transactions to Cortex-M7_2 ITCM are not generated because Cortex-M7_2 ITCM is not selected (backdoor access). 0b - Execution enabled 1b - Execution disabled
13 CM7_1_ITCM	Transaction Control For Cortex-M7_1 ITCM Does not select Cortex-M7_1 ITCM as target. All code fetch or executable fetch transactions to Cortex-M7_1 ITCM are not generated because Cortex-M7_1 ITCM is not selected (backdoor access). 0b - Execution enabled 1b - Execution disabled
12	Transaction Control For Cortex-M7_0 ITCM

Table continues on the next page...

Table continued from the previous page...

Field	Function
CM7_0_ITCM	Specifies the execution status. This field does not select Cortex-M7_0 ITCM as target. All code fetch or executable fetch transactions to Cortex-M7_0 ITCM are not generated because Cortex-M7_0 ITCM is not selected (backdoor access). 0b - Execution enabled 1b - Execution disabled
11 CM7_3_DIS_ITCM_EXEC	ITCM Execution for Cortex-M7_3 Disables ITCM execution for Cortex-M7_3. It is provided to the core. If execution is disabled, it is considered an illegal access by the core. 0b - Enable 1b - Disable
10 CM7_2_DIS_ITCM_EXEC	ITCM Execution for Cortex-M7_2 Disables ITCM execution for Cortex-M7_2. It is provided to the core. If execution is disabled, it is considered an illegal access by the core. 0b - Enable 1b - Disable
9 CM7_1_DIS_ITCM_EXEC	ITCM Execution For Cortex-M7_1 Disables ITCM execution for Cortex-M7_1. It is provided to the core. If execution is disabled, it is considered an illegal access by the core. 0b - Enable 1b - Disable
8 CM7_0_DIS_ITCM_EXEC	ITCM Execution For Cortex-M7_0 Disables ITCM execution for Cortex-M7_0. It is provided to the core. If execution is disabled, it is considered an illegal access by the core. 0b - Enable 1b - Disable
7-4 —	Reserved
3 PRAM_3	Transaction Control For PRAM 3 Does not select PRAM3 as target. All code fetch or executable fetch transactions to PRAM3 are not generated because PRAM3 is not selected. 0b - Transaction enabled 1b - Transaction disabled
2	Transaction Control For PRAM 2

Table continues on the next page...

Table continued from the previous page...

Field	Function
PRAM_2	Does not select PRAM2 as target. All code fetch or executable fetch transactions to PRAM2 are not generated because PRAM2 is not selected. 0b - Transaction enabled 1b - Transaction disabled
1 PRAM_1	Transaction Control For PRAM 1 Does not select PRAM1 as target. All code fetch or executable fetch transactions to PRAM1 are not generated because PRAM1 is not selected. 0b - Transaction enabled 1b - Transaction disabled
0 PRAM0	Transaction Control For PRAM 0 Does not select PRAM0 as target. All code fetch or executable fetch transactions to PRAM0 are not generated because PRAM0 is not selected. 0b - Transaction enabled 1b - Transaction disabled

7.6.3.41 Enable Interconnect Error Detection (ENEDC)

Offset

Register	Offset
ENEDC	600h

Function

Enables interconnect error detection.

For more information, see the FCCU file attached to this document.

Access: Privileged mode only

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		ADD_CM7...	CM7_1_T...	ADD_CM7...	CM7_0_T...	ADD_AIP...	AIPS2	ADD_AIP...	AIPS1	ADD_AIP...	AIPS0	ADD_QSPI	QSPI	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ADD_PRA...	PRAM1	ADD_PRA...	PRAM0	PF1_P0_...	PF0_P1_...	PF0_P0_...	0	CM7_1_A...	CM7_1_A...	ENET	HSE	0	EDMA	CM7_0_A...	CM7_0_A...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29 ADD_CM7_1_TCM	Address Check For Cortex-M7_1_TCM Enables address check for the Cortex-M7_1_TCM backdoor path. 0b - Disable 1b - Enable
28 CM7_1_TCM	Write Data Check For Cortex-M7_1_TCM Enables write data check for the Cortex-M7_1_TCM backdoor path. 0b - Disable 1b - Enable
27 ADD_CM7_0_TCM	Address Check For Cortex-M7_0_TCM Enables address check for the Cortex-M7_0_TCM backdoor path. 0b - Disable 1b - Enable
26 CM7_0_TCM	Write Data Check For Cortex-M7_0_TCM Enables write data check for the Cortex-M7_0_TCM backdoor path. 0b - Disable 1b - Enable
25 ADD_AIPS2	Address Check For AIPS2 Enables address check for the AIPS2 path. 0b - Disable 1b - Enable
24 AIPS2	Write Data Check For AIPS2 Enables write data check for the AIPS2 path. 0b - Disable 1b - Enable
23 ADD_AIPS1	Address Check For AIPS1 Enables address check for the AIPS1 path. 0b - Disable 1b - Enable
22	Write Data Check For AIPS1

Table continues on the next page...

Table continued from the previous page...

Field	Function
AIPS1	Enables write data check for the AIPS1 path. 0b - Disable 1b - Enable
21 ADD_AIPS0	Address Check For AIPS0 Enables address check for the AIPS0 path. 0b - Disable 1b - Enable
20 AIPS0	Write Data Check For AIPS0 Enables write data check for the AIPS0 path. 0b - Disable 1b - Enable
19 ADD_QSPI	Address Check For QuadSPI Enables address check for the QuadSPI path. 0b - Disable 1b - Enable
18 QSPI	Write Data Check For QuadSPI Enables write data check for the QuadSPI path. 0b - Disable 1b - Enable
17 —	Reserved
16 —	Reserved
15 ADD_PRAM1	Address Check For PRAM1 Enables address check for the PRAM1 path. 0b - Disable 1b - Enable
14 PRAM1	Write Data Check For PRAM1 Enables write data check for the PRAM1 path. 0b - Disable 1b - Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
13 ADD_PRAM0	Address Check For PRAM0 Enables address check for the PRAM0 path. 0b - Disable 1b - Enable
12 PRAM0	Write Data Check For PRAM0 Enables write data check for the PRAM0 path. 0b - Disable 1b - Enable
11 PF1_P0_ACHK	Address Check for PF1 P0 Enables address check for flash memory controller 1 P0. 0b - Disable 1b - Enable
10 PF0_P1_ACHK	Address Check for PF0 P1 Enables address check for flash memory controller 0 P1. 0b - Disable 1b - Enable
9 PF0_P0_ACHK	Address Check for PF0 P0 Enables address check for flash controller 0 P0. 0b - Disable 1b - Enable
8 —	Reserved
7 CM7_1_AHBP	Read Data Check For Cortex-M7_1_AHBP Enables read data check for the Cortex-M7_1_AHBP path. 0b - Disable 1b - Enable
6 CM7_1_AHBM	Read Data Check For Cortex-M7_1_AHBM Enables read data check for the Cortex-M7_1_AHBM path. 0b - Disable 1b - Enable
5	Read Data Check For ENET

Table continues on the next page...

Table continued from the previous page...

Field	Function
ENET	Enables read data check for the ENET path. 0b - Disable 1b - Enable
4 HSE	Read Data Check For HSE_B Enables read data check for the HSE_B path. 0b - Disable 1b - Enable
3 —	Reserved
2 EDMA	Read Data Check For eDMA Enables read data check for the eDMA path. 0b - Disable 1b - Enable
1 CM7_0_AHBP	Read Data Check For Cortex-M7_0_AHBP Enables read data check for the Cortex-M7_0_AHBP path. 0b - Disable 1b - Enable
0 CM7_0_AHBM	Read Data Check For Cortex-M7_0_AHBM Enables read data check for the Cortex-M7_0_AHBM path. 0b - Disable 1b - Enable

7.6.3.42 Enable Interconnect Error Detection (ENEDC1)

Offset

Register	Offset
ENEDC1	604h

Function

Enables interconnect error detection.

For more information, see the FCCU file attached to this document.

Access: Privileged mode only

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0			PRAM3_A...	PRAM3_W...	0		TCM_GSK...	SLV_CHK...	SLV_CHK...	SLV_CHK...	MSTR_CH...	MSTR_CH...	CM7_3_A...	CM7_3_A...	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ADD_CM7...	CM7_2_T...	CM7_3_A...	CM7_3_W...	ADD_EN_...	EN_PRAM2	0		MSTR_CH...	EDMA_S1	EDMA_S0	PF1_P1_...	CM7_2_A...	CM7_2_A...	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	Reserved
28 PRAM3_ACHK	Address Check for PRAM3 Enables address check for PRAM3. 0b - Disable 1b - Enable
27 PRAM3_WCHK	Write Data Check for PRAM3 Enables write data check for PRAM3. 0b - Disable 1b - Enable
26-25 —	Reserved
24 TCM_GSKT_ADDR_CHK	TCM Gasket Address Check Enables address check for TCM gasket. 0b - Disable 1b - Enable
23 SLV_CHK_ACE_ACCEL_RESULT_M1_GSKT_ADDR_CHK	Target Check Accelerator Result M1 Gasket Address Check Enables gasket address check for target accelerator result. 0b - Disable 1b - Enable
22	Target Check Accelerator Result M1 Gasket Write Data Check

Table continues on the next page...

Table continued from the previous page...

Field	Function
SLV_CHK_ACE_ACCEL_RESULT_M1_GSKT_WDATA_CHK	Enables gasket write data check for target accelerator result. 0b - Disable 1b - Enable
21 SLV_CHK_ACE_ADDR_CHK	Target Check Accelerator Address Enables target check for accelerator address. 0b - Disable 1b - Enable
20 MSTR_CHK_ACE_FEED_CHK	Controller Check Accelerator Feed Enables controller check for accelerator feed. 0b - Disable 1b - Enable
19 MSTR_CHK_ACE_RESULT_CHK	Controller Check Accelerator Result Enables controller check for accelerator result. 0b - Disable 1b - Enable
18 CM7_3_AHBP	Enable Read Data Check Cortex-M7_3_AHBP Enables the read data check for the Cortex-M7_3_AHBP path. 0b - Disable 1b - Enable
17 CM7_3_AHBM	Enable Read Data Check Cortex-M7_3_AHBM Enables the read data check for the Cortex-M7_3_AHBM path. 0b - Disable 1b - Enable
16 —	Reserved
15 ADD_CM7_2_TCM	Enable Address Check Cortex-M7_2_TCM Enables the address data check for the Cortex-M7_2_TCM backdoor path. 0b - Disable 1b - Enable
14 CM7_2_TCM	Enable Write Data Check Cortex-M7_2_TCM Enables write data check for the Cortex-M7_2_TCM backdoor path.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disable 1b - Enable
13 CM7_3_ADDR_CHK	Enable Address Check Cortex-M7_3_TCM Enables the address check for the Cortex-M7_3_TCM backdoor path. 0b - Disable 1b - Enable
12 CM7_3_WDATA_CHK	Enable Write Data Check Cortex-M7_3_TCM Enables write data check for the Cortex-M7_3_TCM backdoor path. 0b - Disable 1b - Enable
11 ADD_EN_PRAM2	Enable Address Check PRAM 2 Enables the address check for the PRAM 2 path. 0b - Disable 1b - Enable
10 EN_PRAM2	Enable Write Data Check PRAM 2 Enables the write data check for PRAM2. 0b - Disable 1b - Enable
9-8 —	Reserved
7 MSTR_CHECK_ENET1	Controller Check ENET1 Enables the controller check for ENET1. 0b - Disable 1b - Enable
6 EDMA_S1	Enable Address Check eDMA S1 Enables the address check for the eDMA S1 path. 0b - Disable 1b - Enable
5 EDMA_S0	Enable Address Check eDMA S0 Enables address check for the eDMA S0 path.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disable 1b - Enable
4 PF1_P1_ACHK	Enable Address Check PF1 P1 Enables address check for flash controller 1 P1. 0b - Disable 1b - Enable
3 CM7_2_AHBP	Enable Read Data Check Cortex-M7_2_AHBP Enables read data check for the Cortex-M7_2_AHBP path. 0b - Disable 1b - Enable
2 CM7_2_AHBM	Enable Read Data Check Cortex-M7_2_AHBM Enables read data check for the Cortex-M7_2_AHBM path. 0b - Disable 1b - Enable
1-0 —	Reserved

7.6.3.43 AHB Gasket Configuration (IAHBCFGREG)

Offset

Register	Offset
IAHBCFGREG	700h

Function

Controls the functional configuration of the AHB gaskets located on the platform.

Access: Privileged mode only

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved			HSE_CMX...	ACE_GSK...	ACE_ACC...	CM7_3_A...	CM7_3_A...	CM7_2_A...	CM7_1_A...	CM7_0_A...	CM7_3_A...	CM7_2_A...	CM7_1_A...	CM7_0_A...	PRAM2_D...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TCM_PRA...	PRAM1_D...	GMAC1_D...	USDHC_D...	AIPS0_D...	AIPS2_D...	AIPS1_D...	CM7_2_A...	CM7_1_A...	CM7_0_A...	QSPI_DI...	TCM_DIS...	HSE_DIS...	DMA_AXB...	DMA_AXB...	EMAC_DI...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	Reserved
28 HSE_CMX_GSKT_DAP_DISABLE_OPT_WR	HSE CMX Gasket Disable Write Optimization Determines whether write burst optimizations in the HSE_B CMX gasket are enabled or disabled. Enabling optimization allows performance improvements during burst writes. Disabling optimization is required only if you expect an early write burst termination from a controller. 0b - Enable 1b - Disable
27 ACE_GSKT_DISABLE_OPT_WR	ACE Gasket Disable Write Optimization Determines whether write burst optimizations in the ACE gasket are enabled or disabled. Enabling optimization allows performance improvements during burst writes. Disabling optimization is required only if you expect an early write burst termination from a controller. 0b - Enable 1b - Disable
26 ACE_ACCEL_RESULT_M1_GSKT_DISABLE_OPT_WR	Ace Accelerator Disable Write Optimization Determines whether write burst optimizations in the ace accelerator result M1 gasket are enabled or disabled. Enabling optimization allows performance improvements during burst writes. Disabling optimization is required only if you expect an early write burst termination from a controller. 0b - Enable 1b - Disable
25 CM7_3_AHBS_DIS_WR_OPT	Cortex-M7_3 AHBS Disable Write Optimization Determines whether write burst optimizations in the Cortex-M7_3_AHBS gasket are enabled or disabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Enabling optimization allows performance improvements during burst writes. Disabling optimization is required only if you expect an early write burst termination from a controller. 0b - Enable 1b - Disable
24 CM7_3_AHBP_DIS_WR_OPT	Cortex-M7_3 AHBP Disable Write Optimization Determines whether write burst optimizations in the Cortex-M7_3_AHBP gasket are enabled or disabled. Enabling optimization allows performance improvements during burst writes. Disabling optimization is required only if you expect an early write burst termination from a controller. 0b - Enable 1b - Disable
23 CM7_2_AHBP_DIS_WR_OPT	Cortex-M7_2 AHBP Disable Write Optimization Determines whether write burst optimizations in the Cortex-M7_2_AHBP gasket are enabled or disabled. Enabling optimization allows performance improvements during burst writes. Disabling optimization is required only if you expect an early write burst termination from a controller. 0b - Enable 1b - Disable
22 CM7_1_AHBP_DIS_WR_OPT	Cortex-M7_1 AHBP Disable Write Optimization Determines whether write burst optimizations in the Cortex-M7_1_AHBP gasket are enabled or disabled. Enabling optimization allows performance improvements during burst writes. Disabling optimization is required only if you expect an early write burst termination from a controller. 0b - Enable 1b - Disable
21 CM7_0_AHBP_DIS_WR_OPT	Cortex-M7_0 AHBP Disable Write Optimization Determines whether write burst optimizations in the Cortex-M7_0_AHBP gasket are enabled or disabled. Enabling optimization allows performance improvements during burst writes. Disabling optimization is required only if you expect an early write burst termination from a controller. 0b - Enable 1b - Disable
20 CM7_3_AHBM_DIS_WR_OPT	Cortex-M7_3 AHBM Disable Write Optimization Determines whether write burst optimizations in the Cortex-M7_3_AHBM gasket are enabled or disabled. Enabling optimization allows performance improvements during burst writes. Disabling optimization is required only if you expect an early write burst termination from a controller. 0b - Enable 1b - Disable

Table continues on the next page...

Table continued from the previous page...

Field	Function
19 CM7_2_AHBM_DIS_WR_OPT	Cortex-M7_2 AHBM Disable Write Optimization Determines whether write burst optimizations in the Cortex-M7_2_AHBM gasket are enabled or disabled. Enabling optimization allows performance improvements during burst writes. Disabling optimization is required only if you expect an early write burst termination from a controller. 0b - Enable 1b - Disable
18 CM7_1_AHBM_DIS_WR_OPT	Cortex-M7_1 AHBM Disable Write Optimization Determines whether write burst optimizations in the Cortex-M7_1_AHBM gasket are enabled or disabled. Enabling optimization allows performance improvements during burst writes. Disabling optimization is required only if you expect an early write burst termination from a controller. 0b - Enable 1b - Disable
17 CM7_0_AHBM_DIS_WR_OPT	Cortex-M7_0 AHBM Disable Write Optimization Determines whether write burst optimizations in the Cortex-M7_0_AHBM gasket are enabled or disabled. Enabling optimization allows performance improvements during burst writes. Disabling optimization is required only if you expect an early write burst termination from a controller. 0b - Enable 1b - Disable
16 PRAM2_DIS_WR_OPT	PRAM2 Disable Write Optimization Determines whether write burst optimizations in the PRAM2 gasket are enabled or disabled. Enabling optimization allows performance improvements during burst writes. Disabling optimization is required only if you expect an early write burst termination from a controller. 0b - Enable 1b - Disable
15 TCM_PRAM_DIS_WR_OPT	TCM PRAM Disable Write Optimization Determines whether write burst optimizations in the TCM_PRAM gasket are enabled or disabled. Enabling optimization allows performance improvements during burst writes. Disabling optimization is required only if you expect an early write burst termination from a controller. 0b - Enable 1b - Disable
14 PRAM1_DIS_WR_OPT	PRAM1 Disable Write Optimization Determines whether write burst optimizations in the PRAM1 gasket are enabled or disabled. Enabling optimization allows performance improvements during burst writes. Disabling optimization is required only if you expect an early write burst termination from a controller.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Enable 1b - Disable
13 GMAC1_DIS_WR_OPT	GMAC1 Disable Write Optimization Determines whether write burst optimizations in the GMAC1 gasket are enabled or disabled. Enabling optimization allows performance improvements during burst writes. Disabling optimization is required only if you expect an early write burst termination from a controller. 0b - Enable 1b - Disable
12 USDHC_DIS_WR_OPT	uSDHC Disable Write Optimization Determines whether write burst optimizations in the uSDHC gasket are enabled. Enabling optimization allows performance improvements during burst writes. Disabling optimization is required only if you expect an early write burst termination from a controller. 0b - Enable 1b - Disable
11 AIPS0_DIS_WR_OPT	AIPS0 Disable Write Optimization Determines whether write burst optimizations in the AIPS2 AHB gasket are enabled. Enabling optimization allows performance improvements during burst writes. Disabling optimization is required only if you expect an early write burst termination from a controller. 0b - Enable 1b - Disable
10 AIPS2_DIS_WR_OPT	AIPS2 Disable Write Optimization Determines whether write burst optimizations in the AIPS2 AHB gasket are enabled. Enabling optimization allows performance improvements during burst writes. Disabling optimization is required only if you expect an early write burst termination from a controller. 0b - Enable 1b - Disable
9 AIPS1_DIS_WR_OPT	AIPS1 Disable Write Optimization Determines whether write burst optimizations in the AIPS1 AHB gasket are enabled. Enabling optimization allows performance improvements during burst writes. Disabling optimization is required only if you expect an early write burst termination from a controller. 0b - Enable 1b - Disable
8	Cortex-M7_2 AHBS Disable Write Optimization

Table continues on the next page...

Table continued from the previous page...

Field	Function
CM7_2_AHBS_DIS_WR_OPT	Determines whether write burst optimizations in the Cortex-M7_2_AHBS gasket are enabled or disabled. Enabling optimization allows performance improvements during burst writes. Disabling optimization is required only if you expect an early write burst termination from a controller. 0b - Enable 1b - Disable
7 CM7_1_AHBS_DIS_WR_OPT	Cortex-M7_1 AHBS Disable Write Optimization Determines whether write burst optimizations in the Cortex-M7_1_AHBS gasket are enabled or disabled. Enabling optimization allows performance improvements during burst writes. Disabling optimization is required only if you expect an early write burst termination from a controller. 0b - Enable 1b - Disable
6 CM7_0_AHBS_DIS_WR_OPT	Cortex-M7_0 AHBS Disable Write Optimization Determines whether write burst optimizations in the Cortex-M7_0_AHBS gasket are enabled or disabled. Enabling optimization allows performance improvements during burst writes. Disabling optimization is required only if you expect an early write burst termination from a controller. 0b - Enable 1b - Disable
5 QSPI_DIS_WR_OPT	QSPI Disable Write Optimization Determines whether write burst optimizations in the QuadSPI AHB gasket are enabled. Enabling optimization allows performance improvements during burst writes. Disabling optimization is required only if you expect an early write burst termination from a controller. 0b - Enable 1b - Disable
4 TCM_DIS_WR_OPT	TCM Disable Write Optimization Determines whether write burst optimizations in the TCM AHB gasket are enabled. Enabling optimization allows performance improvements during burst writes. Disabling optimization is required only if you expect an early write burst termination from a controller. 0b - Enable 1b - Disable
3 HSE_DIS_WR_OPT	HSE Disable Write Optimization Determines whether write burst optimizations in the HSE_B AHB gasket are enabled. Enabling optimization allows performance improvements during burst writes. Disabling optimization is required only if you expect an early write burst termination from a controller.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Enable 1b - Disable
2 DMA_AXBS_S1_DIS_WR_OPT	DMA AXBS S1 Disable Write Optimization Determines whether write burst optimizations in the DMA AXBS S1 AHB gasket are enabled. Enabling optimization allows performance improvements during burst writes. Disabling optimization is required only if you expect an early write burst termination from a controller. 0b - Enable 1b - Disable
1 DMA_AXBS_S0_DIS_WR_OPT	DMA AXBS S0 Disable Write Optimization Determines whether write burst optimizations in the DMA AXBS S0 AHB gasket are enabled. Enabling optimization allows performance improvements during burst writes. Disabling optimization is required only if you expect an early write burst termination from a controller. 0b - Enable 1b - Disable
0 EMAC_DIS_WR_OPT	EMAC Disable Write Optimization Determines whether write burst optimizations in the EMAC AHB gasket are enabled. Enabling optimization allows performance improvements during burst writes. Disabling optimization is required only if you expect an early write burst termination from a controller. 0b - Enable 1b - Disable

7.6.3.44 Interrupt Router Shared Peripheral Routing Control (IRSPRC0 - IRSPRC239)

Offset

For n = 0 to 239:

Register	Offset
IRSPRCn	880h + (n × 2h)

Function

Provides an array of 16-bit registers, where each register defines the routing control for the corresponding interrupt request, starting from IRQ = 0 (first on-platform interrupt vector). See the interrupt map file attached to this document for details.

For this chip, each interrupt request can be either routed to a subset or to all the cores using the bit-mapped fields in IRSPRC*n*. If all the CP*x*E*n* fields are cleared, the interrupt request is disabled. Each routing control halfword can be locked by writing 1 to the LOCK field.

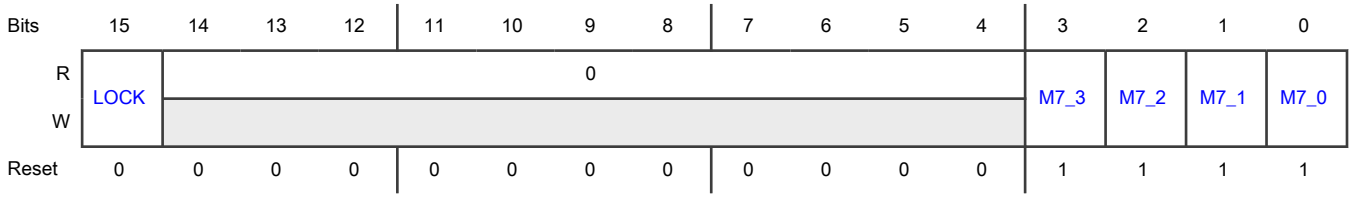
Privileged accesses from noncore (and nondebug) bus controllers are treated as RAZ/WI, and any attempted User mode reference terminates with an error. Attempted accesses using a size other than a 16-bit halfword also terminate with an error.

If you write 1 to all the CP*x*En bits, all the cores service the interrupt. You must ensure that no conflicts arise from this setup either via the interrupt handler or through programming core level interrupt routing (NVIC/GIC).

Reads and writes to this register beyond IRSPRC207 lead to unpredictable results.

Access: Privileged mode only

Diagram



Fields

Field	Function
15 LOCK	Lock Provides a mechanism to lock the routing of the corresponding interrupt request. After you write 1 to this field, attempted writes to IRSPRC*n* are ignored until the next reset writes 0 to the field. 0b - Writes to IRSPRCn allowed 1b - Writes to IRSPRCn ignored
14-4 —	Reserved
3 M7_3	Enable Cortex-M7_3 Interrupt Steering Enables the corresponding interrupt request to route to Cortex-M7_3. 0b - Routing disabled 1b - Routing enabled
2 M7_2	Enable Cortex-M7_2 Interrupt Steering Enables the corresponding interrupt request to route to Cortex-M7_2. 0b - Routing disabled 1b - Routing enabled
1 M7_1	Enable Cortex-M7_1 Interrupt Steering Enables the corresponding interrupt request to route to Cortex-M7_1. 0b - Routing disabled 1b - Routing enabled
0 M7_0	Enable Cortex-M7_0 Interrupt Steering Enables the corresponding interrupt request to route to Cortex-M7_0. 0b - Routing disabled 1b - Routing enabled

7.7 Glossary

GIC	Generic interrupt controller
IRQs	Interrupt requests
ISR	Interrupt service routine
MSI	Message signal interface
NVIC	Nested vector interrupt controller

Chapter 8
Virtualization Wrapper (VIRT_WRAPPER) for S32K388 and S32K389

8.1 Chip-specific VIRT_WRAPPER information

8.1.1 Virtual Wrapper instances

This chip has one instance of Virtual Wrapper.

8.1.2 System overview of PDAC scheme

This figure shows the interaction of XRDC (containing PDACs), related to different cores (domain IDs), with VIRT_WRAPPER.

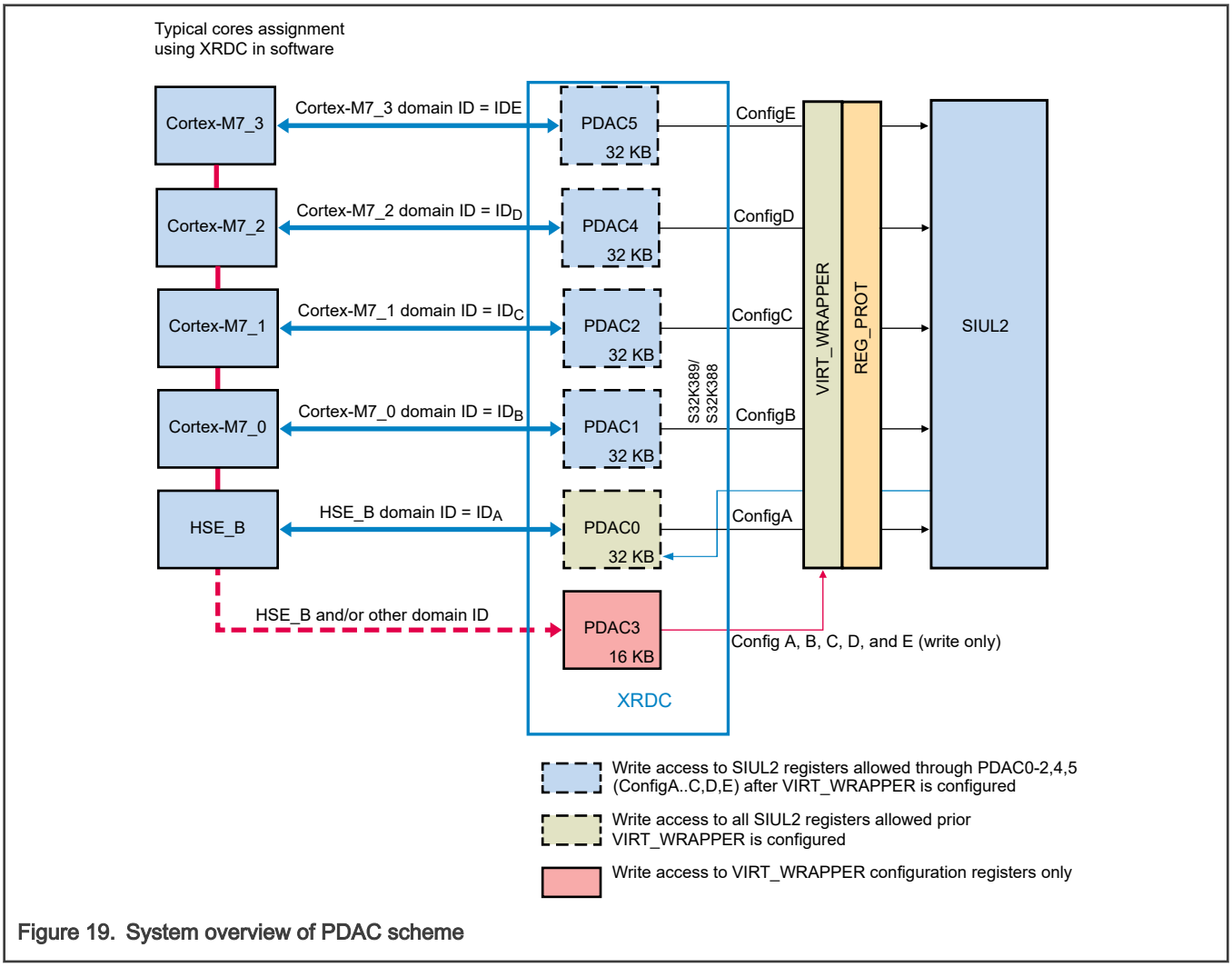


Figure 19. System overview of PDAC scheme

8.1.3 Initial VIRT_WRAPPER operation

Initially, VIRT_WRAPPER:

- Protects the "ConfigB-E" paths to SIUL2. You must configure VIRT_WRAPPER to define the SIUL2 R/W control registers that you can access through the "ConfigB-E" VIRT_WRAPPER access paths.

- Does not protect the direct path to SIUL2. It allows write accesses to all the SIUL2 R/W control registers. If PDAC0 is set to allow write access only to initiator core, for example HSE-B core (variant supported by the chip), then other initiators cannot program the SIUL2 register. However, if these non-HSE_B initiators program SIUL2 registers using other PDAC slots, a transfer error occurs. Any core can act as initiator core.
- Protects the "ConfigB-E" paths to SIUL2. You must configure VIRT_WRAPPER to define the SIUL2 R/W control registers that you can access through the "ConfigB-E" VIRT_WRAPPER access paths.
- The Virtualization Wrapper shall be configured using "IPS register interface" . The Virtualization Wrapper access paths are accessible using IPS register interface.

8.1.4 Additional VIRT_WRAPPER details

See the "SIUL2 memory map overview and protection" table, later in this chapter, for an overview of the SIUL2 registers and their protection attributes.

By default, any core can access SIUL2 registers through PDAC0. This process is called ANY_MASTER. However, after XRDC configuration, HSE_B:

- Assigns PDACs to the respective cores or domain IDs
- Locks the XRDC configuration
- Programs VIRT_WRAPPER registers for pad assignments

HSE_B accesses PDAC0 solely for the pads having 111b as the value in their corresponding registers. Also, HSE_B retains the same value for the pads that it stores. In case HSE_B needs to take control of some pins, it can still configure PDAC*n* to be accessible not only from a specific core initiator, but also from HSE_B, before it allows other cores to execute.

PDAC3 protects the IPS register portal of VIRT_WRAPPER configuration registers that any initiator can access but only via the memory slot assigned to PDAC3. The configuration via IPS register portal can be locked to prevent any further changes until the next functional reset. PDAC3 implements this locking.

The protection information of the parallel port output registers is inherited from the protection information of the included pads, according to these rules:

- When all the pads assigned to a GPIO port share the same protection information, the corresponding parallel port output register for this port inherits the same protection information.
- When at least one of the pads assigned to a GPIO port has different protection information than the other pads assigned to this GPIO port, access to the corresponding parallel port output register for this port is disabled for all the masters. Because of the aforementioned protection group mapping, the data bits of a register encode the protection control information related to one full GPIO port, or a chunk of 16 pads.
- You can assign the following SIUL2 registers to individual PDAC control:
  - DISR0
  - DIRER0
  - DIRSR0
  - IREER0
  - IFEER0
  - IFER0
  - IFMCR0–31
  - IFCPR

  Then, you can access all these registers only through the assigned PDAC because the VIRT_WRAPPER_REG_C1039_1024 register controls the SIUL2 interrupt registers.
- You cannot access the SIUL2 R/W control registers configured through PDAC3 to be accessible through the ConfigA VIRT_WRAPPER access path, using the ConfigB VIRT_WRAPPER access path.

- You cannot access any of the SIUL2 R/W control registers through ConfigB-C VIRT_WRAPPER access paths prior to VIRT_WRAPPER configuration. By default, any core can access SIUL2 registers through PDAC0 after reset.
- The same PDAC that accesses the main SIUL2 registers is used to access the SIUL2-mirrored registers and soft-lock-bit registers. This is because these registers are a part of the same 16 KB space.
- Access to mirrored SIUL2 registers sets the bit in Soft lock bit as an inherent property of register protection. See the "Register Protection" chapter for details on the soft lock bit behavior, when accessing the mirrored region.
- You cannot access any of the SIUL2 R/W control registers through ConfigB-E VIRT_WRAPPER access paths prior to VIRT_WRAPPER configuration. By default, any core can access SIUL2 registers through PDAC0 after reset. Once Virtual wrapper is configured, each attempt to write to SIUL R/W register through any of unassigned paths shall result in a transfer error.

8.2 Introduction

Virtualization refers to the various techniques, methods, or approaches of creating a virtual (rather than actual) version of something, such as a virtual hardware platform, operating system (OS), storage device, or network resources. The term "hardware virtualization" refers to the creation of a virtual machine that acts like a real computer with an operating system. The underlying hardware resources separate the software executed on these virtual machines (named "guest program"). In many cases, the specifically modified guest programs are required to run in such a virtual environment.

There are different types of hardware virtualization. One of them is para-virtualization. The para-virtualization is a non-simulated hardware environment. However, a guest program is executed in its own isolated domain, as if it is running on a separate system. Such a behavior is especially beneficial for the software targeted toward functional safety, because it allows freedom of interference for certain aspects of this software.

The hardware-assisted virtualization is a way of improving the efficiency of hardware virtualization. It involves employing specially designed CPUs and hardware components that help improve the performance of a guest environment. VIRT_WRAPPER, described in this chapter, is such a hardware component.

8.2.1 Overview

VIRT_WRAPPER is an extension of the Register Protection (REG_PROT) wrapper methodology. It works in parallel with an existing REG_PROT wrapper by virtualizing registers within the virtualized module. In this chip, SIUL2 is the virtualized module. Virtualized module is the module instance associated with VIRT_WRAPPER. Virtualization is implemented by protecting (for example, granting or inhibiting) the access to a register within the virtualized module, dependent on the specific criteria. It also virtualizes registers within REG_PROT when associated with registers within the virtualized module.

Registers within virtualized module or the REG_PROT wrapper are virtualized by granting, or inhibiting the access to different PDACs as encoded within virtualization pad assignments in VIRT_WRAPPER configuration registers. For this purpose, the virtualization information is programmed into VIRT_WRAPPER configuration registers that specify the grouping of registers within virtualized module. Upon an access to virtualized module through the peripheral interface, the grouping information is exercised to identify whether the corresponding transaction should be granted, or inhibited. A corresponding signal is forwarded to the REG_PROT wrapper that then grants, or inhibits the transaction. The following figure depicts the usage of VIRT_WRAPPER in combination with the REG_PROT wrapper and virtualized module.

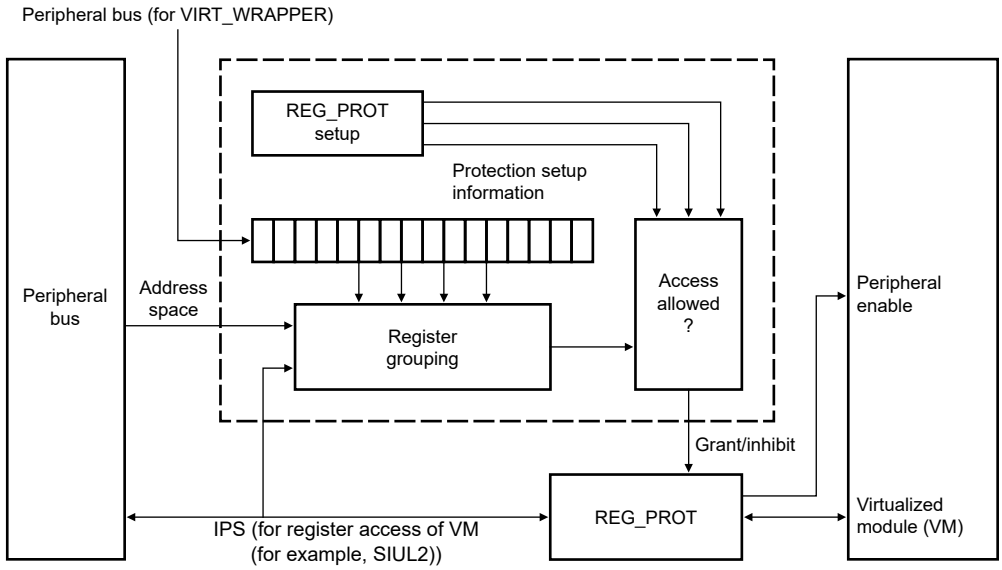


Figure 20. VIRT_WRAPPER block diagram

On a high level, VIRT_WRAPPER is an extension of the REG_PROT wrapper along those lines:

- There is a set of GPRs that receive data over peripheral bus of the VIRT_WRAPPER module. The output of these clients is a quasi-static bus providing the "Protection Control Information" to the remaining logic; this information is kept stable, that is programmed by specific PDAC after boot-up. That should be accessed only by that initiator(s) that is accessing PDAC0. This is taken care by XRDC programming at the SoC.
- The remaining functionality added by VIRT_WRAPPER module consists of two sub-blocks:
  - A first sub-block (the "Grouping PLA") defines the grouping of the registers within the virtualized module. This sub-block is module specific and generated, since this information is specific for the virtualized module, It receives the address information from the peripheral bus and selects the corresponding data from the "Protection Control Information".
  - A second sub-block (the "Protection" sub-block) encodes the previously selected "Protection Control Information" and combines it with the information about the received access request. The current protection methodology allows to inhibit only write accesses for a specific initiator or set of initiators. For this purpose, this sub-block evaluates the received access information; especially the byte enables and the PDAC of the accessing initiator. The output of this sub-block is an internal signal "access granted" that is used to flag an access inhibited by the virtualization information.

8.2.2 Features

VIRT_WRAPPER includes these features:

- A programmable chip-specific virtualization setup, capable to select a subset of the available initiators in combination with up to four available address spaces
- Virtualizing accesses to the registers within a VM by inhibiting accesses to a subset of these registers under control of the specified virtualization information. Only write accesses can be inhibited for a specific initiator or set of initiators.
- Inhibiting accesses to registers within REG_PROT associated with VM; in cases, these registers are associated with a register in the VM to which the access is also inhibited. Additionally, accesses to the global control register (GCR) within REG_PROT can be protected separately.

8.2.3 Modes of operation

VIRT_WRAPPER is operable when the VM is operable. For details about the availability of the VM, see the chapter of the corresponding module. When there is no virtualization information specified for the VIRT_WRAPPER module, if XRDC is not

configured, PDAC slot 0 is open as the default value of configuration register is 111b that is assigned with PDAC slot 0 and any initiator can access through it.

For more details, see "Chip specific VIRT_WRAPPER information".

8.3 Functional description

This section describes the following topics:

- PDAC-based protection scheme
- Register group mapping considering SIUL2
- Access errors

8.3.1 PDAC-based protection scheme

VIRT_WRAPPER uses a protection scheme of the PDACs sub-blocks of XRDC. Each PDAC slot is assigned to one or more cores (by domain ID assignment in XRDC) as shown in Chip-specific VIRT_WRAPPER information section. Only the write accesses are protected by the virtualization feature. Read accesses are not impacted.

For any protection group defined within the register, three bits specify the protection control information according to the following table.

Table 39. Protection control information for a single group

Value	Mnemonic	Protection	Description
000	PDAC1	Restricted access of registers, MSCR, IMCR, and INTINT, and GCR to PDAC slot 1	Protection setup information for PDAC slot 1 is used.
001	PDAC2	Restricted access of registers, MSCR, IMCR, and INTINT, and GCR to PDAC slot 2	Protection setup information for PDAC slot 2 is used.
010	PDAC4	Restricted access of registers, MSCR, IMCR, and INT INT, and GCR to PDAC slot 4	Protection setup information for PDAC slot 4 is used.
011	PDAC5	Restricted access of registers, MSCR, IMCR, and INTINT, and GCR to PDAC slot 5	Protection setup information for PDAC slot 5 is used.
111	PDAC0	By default, any core can access SIUL2 registers through PDAC slot 0.	Protection setup information for PDAC slot 0 is used.

8.3.2 Clocking

This module has no major clocking considerations.

8.3.3 Register group mapping for SIUL2

For the SIUL2 instance, the corresponding virtualization information within the VIRT_WRAPPER module is defined on a per-pad basis. Additionally, the virtualization information protects the control registers (controlling the input multiplexing scheme).

Table 40. Register group mapping details

Register name	Description
Reg A	Virtualization of pad output registers
Reg B	Virtualization of input multiplexing control registers
Reg C	Virtualization of interrupt control registers
Reg D	Virtualization of interrupt control registers

8.3.3.1 Virtualization of pad output registers

Any functional pad (controlled by the SIUL2 instance) is assigned to a separate protection group. This scheme excludes the pads that an MSCR register does not control within an SIUL2 instance (for example, power supply pads). As a result, the protection granularity defined by the virtualization information is a single pad. The control information associated with this protection group affects all the registers related to this pad.

The protection control information for pad *i* is specified within the protection group *i*; thus, it enables a very simple assignment scheme for the related protection control data that is hard-coded. The protection control information for pad *i* affects all the registers related to this pad, in the MSCR*i* and the GPDO*i* registers (in case such registers exist).

SIUL2 can control up to 512 pads that the MSCR, GPDO, and GPDI registers can access individually. The actual number of pads associated with a SIUL2 instance is specific to implementation. GPIO pads are also organized in GPIO ports consisting of a maximum of 16 pads that the parallel port registers (PGPDO, PGPDI, MPGPDO) can access.

Only the pad control and pad output registers are protected. Accesses to the read-only registers GPDI and PGPDI are not affected by virtualization.

As the protection granularity is a single pad, any virtualization information associated with this pad affects the corresponding MCSR and GPDO registers directly. Additionally, any consequence that an illegal access has on the registers implemented within the REG_PROT wrapper is inhibited. For this purpose, any access to the mirrored address space and the corresponding Soft Lock Bit Register (SLBR) is also observed.

The parallel port output registers (PGPDO, MPGPDO) inherit the protection information from the protection information of the included pads according to the following rules:

- When all pads assigned to a GPIO port share the same protection information, the corresponding parallel port output register for this port inherits the same protection information.
- When at least one of the pads assigned to a GPIO port has a different protection information than the other pads assigned to this GPIO port, the write access to the corresponding parallel port output register (PGPDO and MPGPDO) for this port is disabled for all the initiators through all the PDAC slots and a transfer error is generated.

Due to the aforementioned protection group mapping, the data bits of a register encode the protection control information related to one full GPIO port, or a chunk of 16 pads.

8.3.3.2 Virtualization of input multiplexing control registers

Additionally, the SIUL2 instance supports the control of an input multiplexer (INMUX) scheme. This scheme provides the capability to select one of a set of the input pads to be the source of an input function for some of the peripheral modules. Any INMUX controlled by the SIUL2 instance is assigned to a separate protection group. It therefore provides an equivalent protection granularity within the virtualization information.

The protection group 512+*i* specifies the protection control information for the input multiplexer INMUX *i*. This enables a very simple assignment scheme for the related protection control data that is hard-coded. The protection control information for INMUX *i* affects the input multiplexing control register related to this input—namely, IMCR*i*.

SIUL2 can control up to 512 MSCRs and 512 IMCRs input multiplexers that the IMCRs can access individually. GPIO inputs are not affected by the input multiplexer scheme. Therefore, the associated virtualization information does not affect the corresponding pad input registers.

As the protection granularity is a single INMUX, any virtualization information associated with this INMUX affects the corresponding IMCR directly. Additionally, any consequence that an illegal access has on the registers implemented within the REG_PROT wrapper is inhibited. For this purpose, any access to the mirrored address space and the corresponding SLBR is also observed.

8.3.3.3 Virtualization of interrupt control registers

Additionally, the complete set of interrupt control registers within SIUL2 (address offset range 0010h–00C3h) can be protected as a separate group (protection group #1024).

Another additional protection group (protection group #1055) is defined for the global control register (GCR) of the related REG_PROT wrapper.

The PDAC that accesses the main SIUL2 registers is used to access the SIUL2-mirrored registers and soft-lock-bit registers. This is because these registers are a part of the same 16 KB space.

Access to mirrored SIUL2 registers sets the bit in Soft lock bit as an inherent property of register protection. See the "Register Protection" chapter for details on the soft lock bit behavior, when accessing the mirrored region.

The following table shows an overview of the SIUL2 registers and their protection attributes.

Table 41. SIUL2 memory map overview and protection

Offset range	Register	Size (bits)	Protected	Description
0000–000Fh	MIDR1, MIDR2	32	N	Read-only registers, not protected
0010–00C3h	Interrupt registers	32	Y	Protected as a single group—no individual protection of related registers
0240–A3Fh [1]	MSCR	32	Y	Amount of registers defines maximum number of PADs to be controlled and protected
0A40–0123Fh	IMCR	32	Y	Amount of registers defines maximum number of INMUXes to be controlled and protected
1300–14FFh	GPDO GPDO[0] (8-bit) register controls single PAD[0] pad means that if PAD[0] is assigned to PDAC slot 1 then PDAC slot 1 can write to GPDO[0] (8-bit).	8	Y	There are fewer GPDO registers than MSCR registers, as some PADs are not made available as GPIO PADs, but their electrical characteristics can still be programmed
1500–16FFh	GPDI	8	N	PAD input register, not protected
1700–173Fh	PGPDO PGPDO[0] (16-bit) register controls PAD[0-15] pads meaning if only all the PADs[0-15] are assigned to PDAC slot 1, then PDAC slot 1 can write to PGPDO[0] (16-bit).	16	Y	Writable with 8-, 16-, and as a pair with 32-bit accesses
1740–177Fh	PGPDI	16	N	PAD input register, not protected

Table continues on the next page...

Table 41. SIUL2 memory map overview and protection (continued)

Offset range	Register	Size (bits)	Protected	Description
1780–17FFh	MPGPDO MPGPDO[0] (32-bit) register controls PAD[0-15] pads meaning if only all the PADs[0-15] are assigned to PDAC slot 1, then PDAC slot 1 can write to MPGPDO[0] (32-bit).	32	Y	Only writable with 32-bit accesses
2010–20C3h	Interrupt registers (mirror) The DISR0 DIRER0 DIRSR0 IREER0 IFEER0 IFER0 IFMCR0-31 IFCPR0 can be assigned to individual PDAC control, then all these registers are only accessible through assigned PDAC.[2]	32	Y	Mirrored access to interrupt registers in SIUL2 instance
2240–2A3Fh	MSCR (mirror)[3]	32	Y	Mirrored access to MSCRs in SIUL2 instance
2A40–323Fh	IMCR (mirror)[3]	32	Y	Mirrored access to IMCRs in SIUL2 instance
3300–34FFh	GPDO (mirror)[4]	8	Y	Mirrored access to GPDOs in SIUL2 instance
4000–47FFh	SLBR[5]	8	Y	Soft lock bit registers within REG_PROT wrapper for this SIUL2 instance
4800–48FFh	GCR	32	Y	Global configuration register within REG_PROT wrapper for this SIUL2 instance. There is a single register in configuration space that is area 4 of REG_PROT. The size mentioned is to make the whole area a multiple of 4 kB. See the table "Address accesses as a function of PROT_MEM" in the REG_PROT chapter to know the exact area for GCRs.

1. The SIUL2 block guide specifies the support of a maximum of 512 MSCRs.
2. Access to mirrored register address range results in a change of the related SLBR within the REG_PROT wrapper if REG_PROT protects the register. For registers unprotected by REG_PROT, a write to a register mirror has the same impact as a write to the register.
3. Access to mirrored register address range results in a change of the related SLBR within the REG_PROT wrapper if REG_PROT protects the register. For registers unprotected by REG_PROT, write to register mirror has the same impact as write to register.
4. Access to mirrored register address range results in a change of the related SLBR within the REG_PROT wrapper if REG_PROT protects the register. For registers unprotected by REG_PROT, a write to a register mirror has the same impact as a write to the register.

5. Only SLBRs protecting accesses to the interrupt registers, IMCRs, and the MSCR registers are protected. See the register protection sheet attached to the reference manual for a complete list of SIUL2 registers that REG_PROT protects.

8.4 External signals

This module has no external signals.

8.5 Initialization

This module does not require initialization.

8.6 Application Information

This module supports virtualization of accesses to the registers within a SIUL2 by inhibiting or granting accesses to a subset of these registers under control of the specified initiator(core) information.

8.7 VIRT_WRAPPER register descriptions

NOTE

Access to reserved spaces outside the register bank and holes (unimplemented registers) within register bank generates the transfer error. Access to offset C00h does not generate any transfer error.

8.7.1 VIRT_WRAPPER memory map

VIRT_WRAPPER base address: 402A_8000h

Offset	Register	Width (In bits)	Access	Reset value
0h - 20h	Parameter_n Register (REG_A3_0 - REG_A35_32)	32	RW	0707_0707h
24h	Parameter_n Register (REG_A39_36)	32	RW	0707_0707h
28h - 88h	Parameter_n Register (REG_A43_40 - REG_A139_136)	32	RW	0707_0707h
8Ch	Parameter_n Register (REG_A143_140)	32	RW	0707_0707h
90h - 140h	Parameter_n Register (REG_A147_144 - REG_A323_320)	32	RW	0707_0707h
400h	Parameter_n Register (REG_B3_0)	32	RW	0707_0707h
404h	Parameter_n Register (REG_B7_4)	32	RW	0707_0707h
410h	Parameter_n Register (REG_B19_16)	32	RW	0707_0707h
414h	Parameter_n Register (REG_B23_20)	32	RW	0707_0707h
418h	Parameter_n Register (REG_B27_24)	32	RW	0707_0707h
41Ch	Parameter_n Register (REG_B31_28)	32	RW	0707_0707h
420h	Parameter_n Register (REG_B35_32)	32	RW	0707_0707h
424h	Parameter_n Register (REG_B39_36)	32	RW	0707_0707h
428h	Parameter_n Register (REG_B43_40)	32	RW	0707_0707h
42Ch	Parameter_n Register (REG_B47_44)	32	RW	0707_0707h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
430h	Parameter_n Register (REG_B51_48)	32	RW	0707_0707h
434h	Parameter_n Register (REG_B55_52)	32	RW	0707_0707h
438h	Parameter_n Register (REG_B59_56)	32	RW	0707_0707h
43Ch	Parameter_n Register (REG_B63_60)	32	RW	0707_0707h
440h	Parameter_n Register (REG_B67_64)	32	RW	0707_0707h
444h	Parameter_n Register (REG_B71_68)	32	RW	0707_0707h
450h	Parameter_n Register (REG_B83_80)	32	RW	0707_0707h
454h	Parameter_n Register (REG_B87_84)	32	RW	0707_0707h
458h	Parameter_n Register (REG_B91_88)	32	RW	0707_0707h
45Ch	Parameter_n Register (REG_B95_92)	32	RW	0707_0707h
460h	Parameter_n Register (REG_B99_96)	32	RW	0707_0707h
464h	Parameter_n Register (REG_B103_100)	32	RW	0707_0707h
470h	Parameter_n Register (REG_B115_112)	32	RW	0707_0707h
474h	Parameter_n Register (REG_B119_116)	32	RW	0707_0707h
478h	Parameter_n Register (REG_B123_120)	32	RW	0707_0707h
47Ch	Parameter_n Register (REG_B127_124)	32	RW	0707_0707h
480h	Parameter_n Register (REG_B131_128)	32	RW	0707_0707h
484h	Parameter_n Register (REG_B135_132)	32	RW	0707_0707h
490h	Parameter_n Register (REG_B147_144)	32	RW	0707_0707h
494h	Parameter_n Register (REG_B151_148)	32	RW	0707_0707h
498h - 4C4h	Parameter_n Register (REG_B155_152 - REG_B199_196)	32	RW	0707_0707h
4C8h	Parameter_n Register (REG_B203_200)	32	RW	0707_0707h
4D0h	Parameter_n Register (REG_B211_208)	32	RW	0707_0707h
4D4h - 508h	Parameter_n Register (REG_B215_212 - REG_B267_264)	32	RW	0707_0707h
50Ch	Parameter_n Register (REG_B271_268)	32	RW	0707_0707h
520h	Parameter_n Register (REG_B291_288)	32	RW	0707_0707h
524h - 530h	Parameter_n Register (REG_B295_292 - REG_B307_304)	32	RW	0707_0707h
534h	Parameter_n Register (REG_B311_308)	32	RW	0707_0707h
538h	Parameter_n Register (REG_B315_312)	32	RW	0707_0707h
53Ch - 540h	Parameter_n Register (REG_B319_316 - REG_B323_320)	32	RW	0707_0707h
544h	Parameter_n Register (REG_B327_324)	32	RW	0707_0707h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
554h	Parameter_n Register (REG_B343_340)	32	RW	0707_0707h
558h - 56Ch	Parameter_n Register (REG_B347_344 - REG_B367_364)	32	RW	0707_0707h
570h	Parameter_n Register (REG_B371_368)	32	RW	0707_0707h
574h	Parameter_n Register (REG_B375_372)	32	RW	0707_0707h
578h	Parameter_n Register (REG_B379_376)	32	RW	0707_0707h
584h	Parameter_n Register (REG_B391_388)	32	RW	0707_0707h
58Ch	Parameter_n Register (REG_B399_396)	32	RW	0707_0707h
598h	Parameter_n Register (REG_B411_408)	32	RW	0707_0707h
59Ch	Parameter_n Register (REG_B415_412)	32	RW	0707_0707h
5A0h	Parameter_n Register (REG_B419_416)	32	RW	0707_0707h
5B8h	Parameter_n Register (REG_B443_440)	32	RW	0707_0707h
5C0h - 5D4h	Parameter_n Register (REG_B451_448 - REG_B471_468)	32	RW	0707_0707h
5D8h	Parameter_n Register (REG_B475_472)	32	RW	0707_0707h
800h	Parameter_n Register (REG_C)	32	RW	0000_0007h
C00h	Parameter_n Register (REG_D)	32	RW	0000_0007h

8.7.2 Parameter_n Register (REG_A3_0 - REG_A35_32)

Offset

For a = 0 to 8:

Register	Offset
REG_A(4 * a + 3)_(4 * a)	0h + (a × 4h)

Function

This register set is for PAD 0-511. They control MSCR, GPDO, PGPDO, and MPGPDO. Eight bits assigned per PDAC slot have attributes of one of the implemented PDAC slots:

- 000-SIUL2_VIRTWRAPPER_PDAC1
- 001-SIUL2_VIRTWRAPPER_PDAC2
- 010-SIUL2_VIRTWRAPPER_PDAC4
- 011-SIUL2_VIRTWRAPPER_PDAC5
- 111-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any initiator can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any initiator can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0					PAD_3			0					PAD_2		
W																
Reset	0	0	0	0	0	1	1	1	0	0	0	0	0	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0					PAD_1			0					PAD_0		
W																
Reset	0	0	0	0	0	1	1	1	0	0	0	0	0	1	1	1

Fields

Field	Function
31-27 —	Reserved
26-24 PAD_3	PAD_3
23-19 —	Reserved
18-16 PAD_2	PAD_2
15-11 —	Reserved
10-8 PAD_1	PAD_1
7-3 —	Reserved
2-0 PAD_0	PAD_0

8.7.3 Parameter_n Register (REG_A39_36)

Offset

Register	Offset
REG_A39_36	24h

Function

This register set is for PAD 0-511. They control MSCR, GPDO, PGPDO, and MPGPDO. Eight bits assigned per PDAC slot have attributes of one of the implemented PDAC slots:

- 000-SIUL2_VIRTWRAPPER_PDAC1
- 001-SIUL2_VIRTWRAPPER_PDAC2
- 010-SIUL2_VIRTWRAPPER_PDAC4
- 011-SIUL2_VIRTWRAPPER_PDAC5
- 111-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any initiator can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any initiator can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0					Reserved			0					Reserved		
W																
Reset	0	0	0	0	0	1	1	1	0	0	0	0	0	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0					PAD_1			0					PAD_0		
W																
Reset	0	0	0	0	0	1	1	1	0	0	0	0	0	1	1	1

Fields

Field	Function
31-27 —	Reserved
26-24 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
23-19 —	Reserved
18-16 —	Reserved
15-11 —	Reserved
10-8 PAD_1	PAD_1
7-3 —	Reserved
2-0 PAD_0	PAD_0

8.7.4 Parameter_n Register (REG_A43_40 - REG_A139_136)

Offset

For a = 10 to 34:

Register	Offset
REG_A(4 * a + 3)_(4 * a)	0h + (a × 4h)

Function

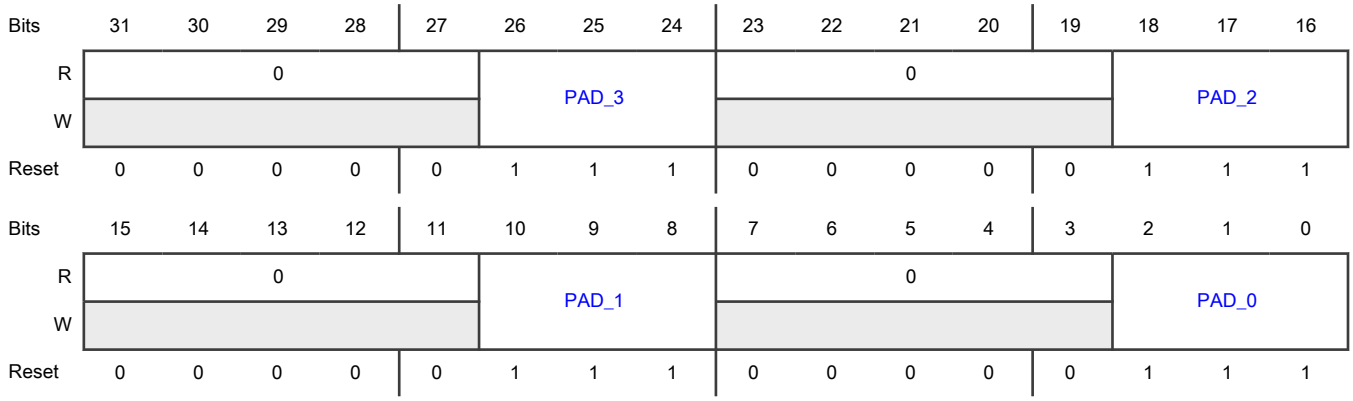
This register set is for PAD 0-511. They control MSCR, GPDO, PGPDO, and MPGPDO. Eight bits assigned per PDAC slot have attributes of one of the implemented PDAC slots:

- 000-SIUL2_VIRTWRAPPER_PDAC1
- 001-SIUL2_VIRTWRAPPER_PDAC2
- 010-SIUL2_VIRTWRAPPER_PDAC4
- 011-SIUL2_VIRTWRAPPER_PDAC5
- 111-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any initiator can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any initiator can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-24 PAD_3	PAD_3
23-19 —	Reserved
18-16 PAD_2	PAD_2
15-11 —	Reserved
10-8 PAD_1	PAD_1
7-3 —	Reserved
2-0 PAD_0	PAD_0

8.7.5 Parameter_n Register (REG_A143_140)

Offset

Register	Offset
REG_A143_140	8Ch

Function

This register set is for PAD 0-511. They control MSCR, GPDO, PGPDO, and MPGPDO. Eight bits assigned per PDAC slot have attributes of one of the implemented PDAC slots:

- 000-SIUL2_VIRTWRAPPER_PDAC1
- 001-SIUL2_VIRTWRAPPER_PDAC2
- 010-SIUL2_VIRTWRAPPER_PDAC4
- 011-SIUL2_VIRTWRAPPER_PDAC5
- 111-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any initiator can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any initiator can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0					PAD_3			0					PAD_2		
W																
Reset	0	0	0	0	0	1	1	1	0	0	0	0	0	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0					Reserved			0					PAD_0		
W																
Reset	0	0	0	0	0	1	1	1	0	0	0	0	0	1	1	1

Fields

Field	Function
31-27 —	Reserved
26-24 PAD_3	PAD_3

Table continues on the next page...

Table continued from the previous page...

Field	Function
23-19 —	Reserved
18-16 PAD_2	PAD_2
15-11 —	Reserved
10-8 —	Reserved
7-3 —	Reserved
2-0 PAD_0	PAD_0

8.7.6 Parameter_n Register (REG_A147_144 - REG_A323_320)

Offset

For a = 36 to 80:

Register	Offset
REG_A(4 * a + 3)_(4 * a)	0h + (a × 4h)

Function

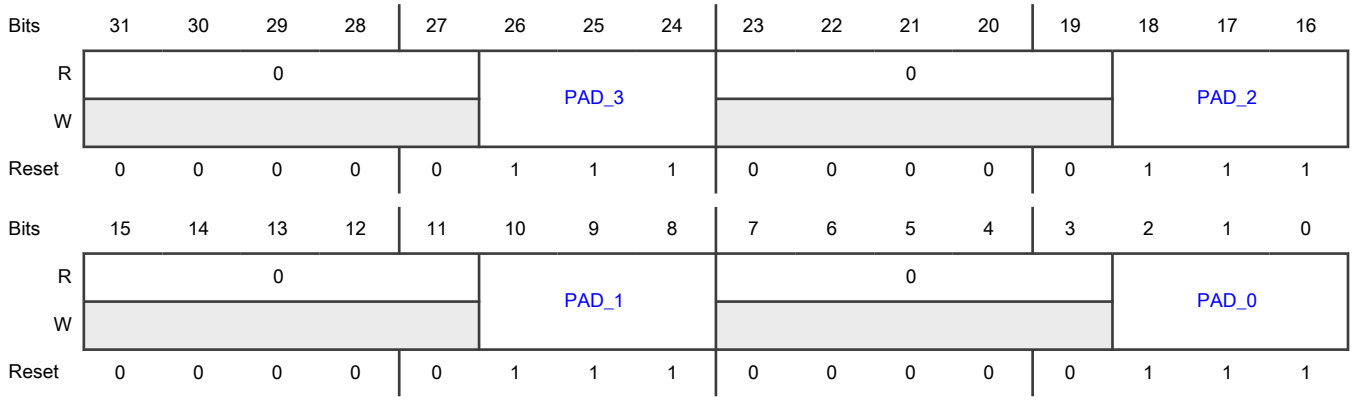
This register set is for PAD 0-511. They control MSCR, GPDO, PGPDO, and MPGPDO. Eight bits assigned per PDAC slot have attributes of one of the implemented PDAC slots:

- 000-SIUL2_VIRTWRAPPER_PDAC1
- 001-SIUL2_VIRTWRAPPER_PDAC2
- 010-SIUL2_VIRTWRAPPER_PDAC4
- 011-SIUL2_VIRTWRAPPER_PDAC5
- 111-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any initiator can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any initiator can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-24 PAD_3	PAD_3
23-19 —	Reserved
18-16 PAD_2	PAD_2
15-11 —	Reserved
10-8 PAD_1	PAD_1
7-3 —	Reserved
2-0 PAD_0	PAD_0

8.7.7 Parameter_n Register (REG_B3_0)

Offset

Register	Offset
REG_B3_0	400h

Function

Controls access to PAD 0-511 specific for IMCR registers. Eight bits assigned per IMCR register have attributes of one of the implemented PDAC slots:

- 000-SIUL2_VIRTWRAPPER_PDAC1
- 001-SIUL2_VIRTWRAPPER_PDAC2
- 010-SIUL2_VIRTWRAPPER_PDAC4
- 011-SIUL2_VIRTWRAPPER_PDAC5
- 111-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any initiator can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any initiator can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0					INMUX_3			0					INMUX_2		
W																
Reset	0	0	0	0	0	1	1	1	0	0	0	0	0	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0					INMUX_1			0					INMUX_0		
W																
Reset	0	0	0	0	0	1	1	1	0	0	0	0	0	1	1	1

Fields

Field	Function
31-27 —	Reserved
26-24 INMUX_3	INMUX_3

Table continues on the next page...

Table continued from the previous page...

Field	Function
23-19 —	Reserved
18-16 INMUX_2	INMUX_2
15-11 —	Reserved
10-8 INMUX_1	INMUX_1
7-3 —	Reserved
2-0 INMUX_0	INMUX_0

8.7.8 Parameter_n Register (REG_B7_4)

Offset

Register	Offset
REG_B7_4	404h

Function

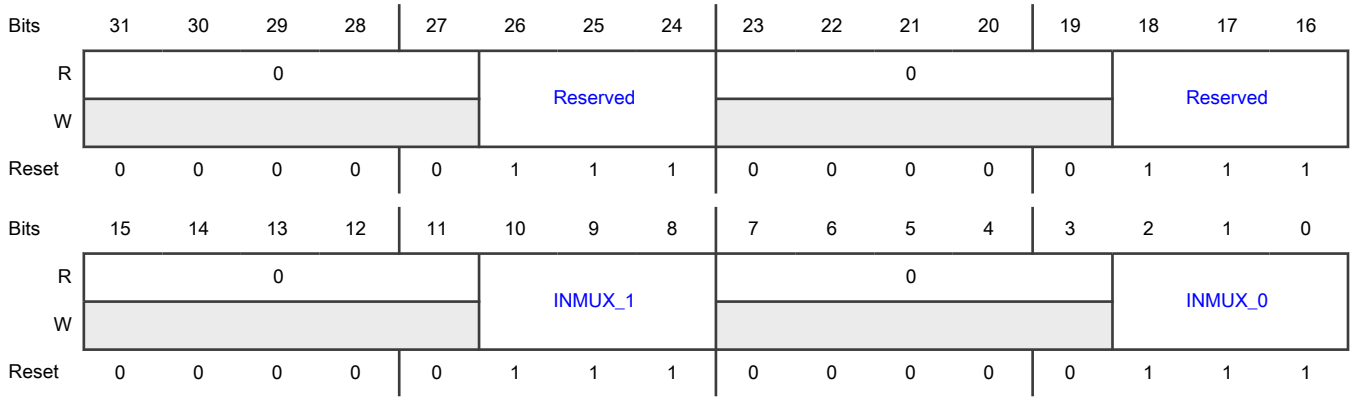
Controls access to PAD 0-511 specific for IMCR registers. Eight bits assigned per IMCR register have attributes of one of the implemented PDAC slots:

- 000-SIUL2_VIRTWRAPPER_PDAC1
- 001-SIUL2_VIRTWRAPPER_PDAC2
- 010-SIUL2_VIRTWRAPPER_PDAC4
- 011-SIUL2_VIRTWRAPPER_PDAC5
- 111-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any initiator can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any initiator can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-24 —	Reserved
23-19 —	Reserved
18-16 —	Reserved
15-11 —	Reserved
10-8 INMUX_1	INMUX_1
7-3 —	Reserved
2-0 INMUX_0	INMUX_0

8.7.9 Parameter_n Register (REG_B19_16 - REG_B147_144)

Offset

Register	Offset
REG_B19_16	410h
REG_B23_20	414h
REG_B27_24	418h
REG_B31_28	41Ch
REG_B35_32	420h
REG_B39_36	424h
REG_B43_40	428h
REG_B47_44	42Ch
REG_B51_48	430h
REG_B55_52	434h
REG_B59_56	438h
REG_B63_60	43Ch
REG_B67_64	440h
REG_B71_68	444h
REG_B83_80	450h
REG_B87_84	454h
REG_B91_88	458h
REG_B95_92	45Ch
REG_B99_96	460h
REG_B103_100	464h
REG_B115_112	470h
REG_B119_116	474h
REG_B123_120	478h
REG_B127_124	47Ch
REG_B131_128	480h
REG_B135_132	484h
REG_B147_144	490h

Function

Controls access to PAD 0-511 specific for IMCR registers. Eight bits assigned per IMCR register have attributes of one of the implemented PDAC slots:

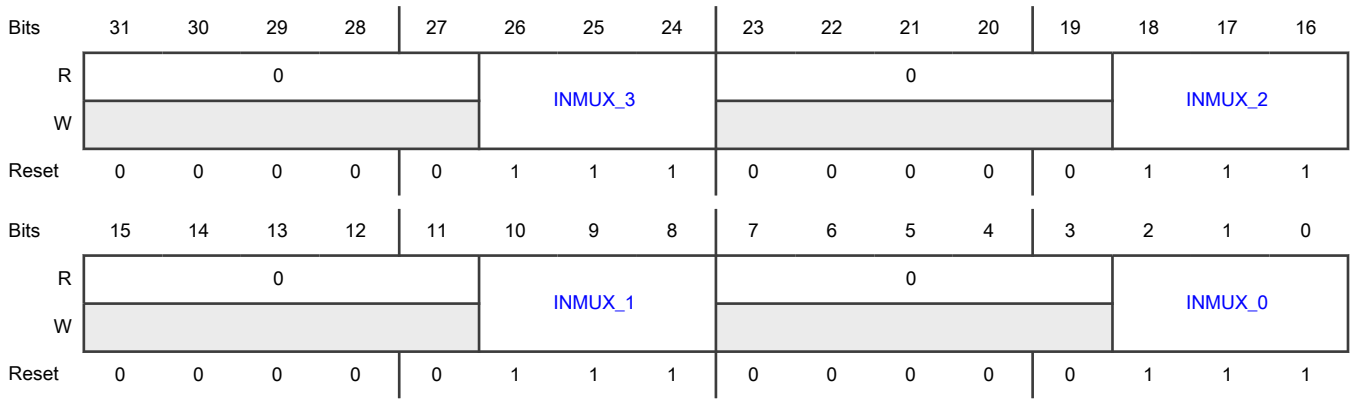
- 000-SIUL2_VIRTWRAPPER_PDAC1

- 001-SIUL2_VIRTWRAPPER_PDAC2
- 010-SIUL2_VIRTWRAPPER_PDAC4
- 011-SIUL2_VIRTWRAPPER_PDAC5
- 111-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any initiator can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any initiator can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-24 INMUX_3	INMUX_3
23-19 —	Reserved
18-16 INMUX_2	INMUX_2
15-11 —	Reserved
10-8 INMUX_1	INMUX_1

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-3 —	Reserved
2-0 INMUX_0	INMUX_0

8.7.10 Parameter_n Register (REG_B151_148)

Offset

Register	Offset
REG_B151_148	494h

Function

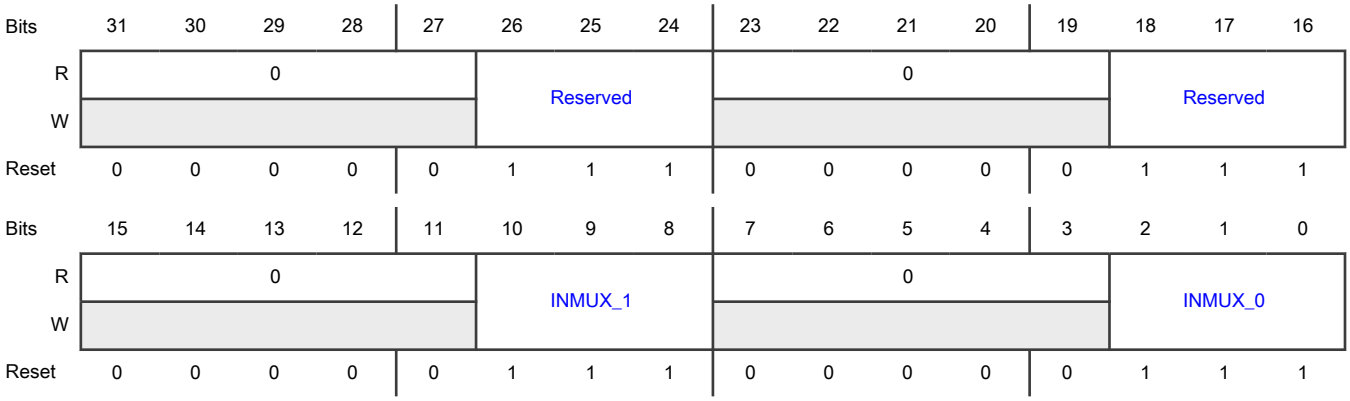
Controls access to PAD 0-511 specific for IMCR registers. Eight bits assigned per IMCR register have attributes of one of the implemented PDAC slots:

- 000-SIUL2_VIRTWRAPPER_PDAC1
- 001-SIUL2_VIRTWRAPPER_PDAC2
- 010-SIUL2_VIRTWRAPPER_PDAC4
- 011-SIUL2_VIRTWRAPPER_PDAC5
- 111-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any initiator can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any initiator can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-24 —	Reserved
23-19 —	Reserved
18-16 —	Reserved
15-11 —	Reserved
10-8 INMUX_1	INMUX_1
7-3 —	Reserved
2-0 INMUX_0	INMUX_0

8.7.11 Parameter_n Register (REG_B155_152 - REG_B199_196)

Offset

For a = 166 to 177:

Register	Offset
REG_B(4 * (a - 128) + 3)_(4 * (a - 128))	200h + (a × 4h)

Function

Controls access to PAD 0-511 specific for IMCR registers. Eight bits assigned per IMCR register have attributes of one of the implemented PDAC slots:

- 000-SIUL2_VIRTWRAPPER_PDAC1
- 001-SIUL2_VIRTWRAPPER_PDAC2
- 010-SIUL2_VIRTWRAPPER_PDAC4
- 011-SIUL2_VIRTWRAPPER_PDAC5
- 111-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any initiator can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any initiator can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0					INMUX_3			0					INMUX_2		
W																
Reset	0	0	0	0	0	1	1	1	0	0	0	0	0	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0					INMUX_1			0					INMUX_0		
W																
Reset	0	0	0	0	0	1	1	1	0	0	0	0	0	1	1	1

Fields

Field	Function
31-27 —	Reserved
26-24 INMUX_3	INMUX_3
23-19 —	Reserved
18-16 INMUX_2	INMUX_2
15-11 —	Reserved
10-8 INMUX_1	INMUX_1
7-3 —	Reserved
2-0 INMUX_0	INMUX_0

8.7.12 Parameter_n Register (REG_B203_200)

Offset

Register	Offset
REG_B203_200	4C8h

Function

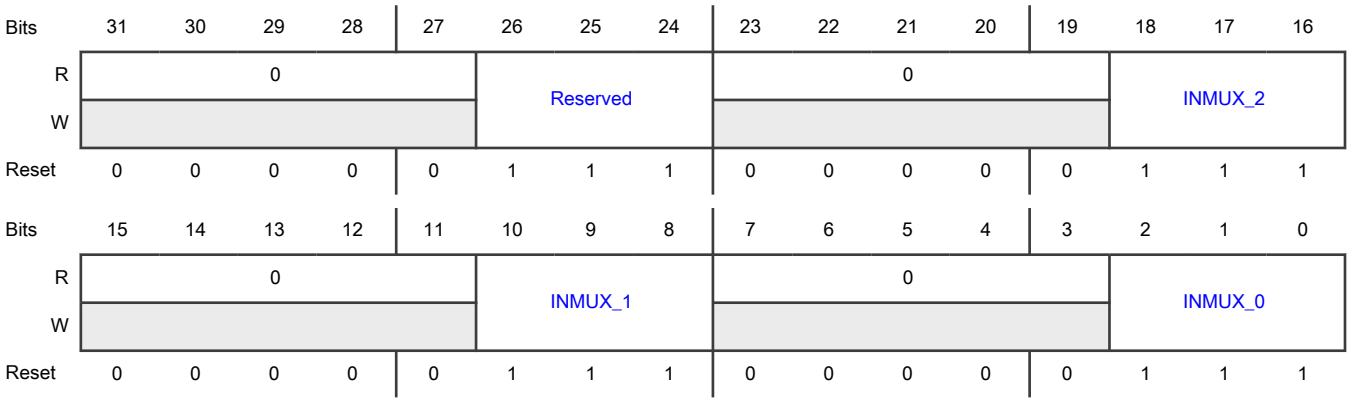
Controls access to PAD 0-511 specific for IMCR registers. Eight bits assigned per IMCR register have attributes of one of the implemented PDAC slots:

- 000-SIUL2_VIRTWRAPPER_PDAC1
- 001-SIUL2_VIRTWRAPPER_PDAC2
- 010-SIUL2_VIRTWRAPPER_PDAC4
- 011-SIUL2_VIRTWRAPPER_PDAC5
- 111-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any initiator can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any initiator can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-24 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
23-19 —	Reserved
18-16 INMUX_2	INMUX_2
15-11 —	Reserved
10-8 INMUX_1	INMUX_1
7-3 —	Reserved
2-0 INMUX_0	INMUX_0

8.7.13 Parameter_n Register (REG_B211_208)

Offset

Register	Offset
REG_B211_208	4D0h

Function

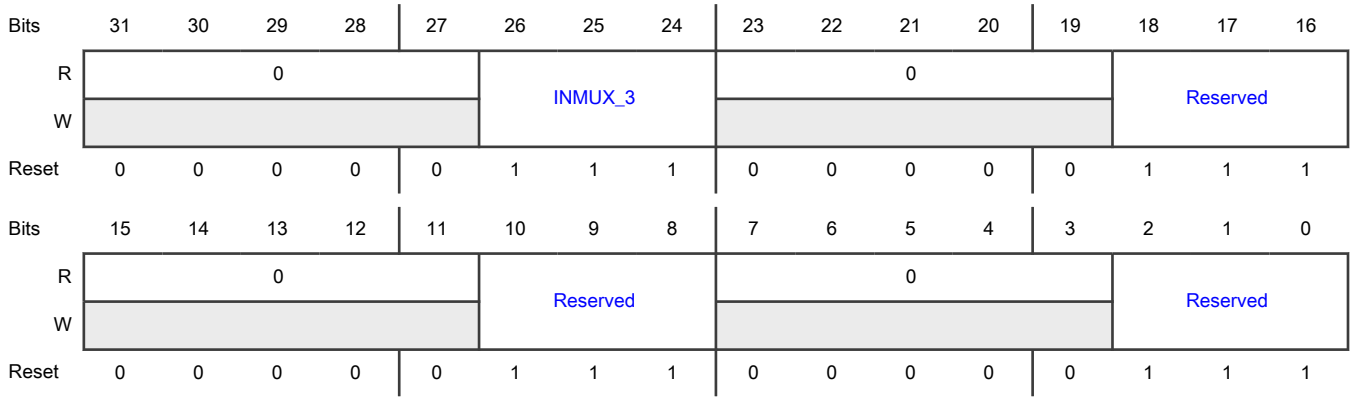
Controls access to PAD 0-511 specific for IMCR registers. Eight bits assigned per IMCR register have attributes of one of the implemented PDAC slots:

- 000-SIUL2_VIRTWRAPPER_PDAC1
- 001-SIUL2_VIRTWRAPPER_PDAC2
- 010-SIUL2_VIRTWRAPPER_PDAC4
- 011-SIUL2_VIRTWRAPPER_PDAC5
- 111-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any initiator can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any initiator can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-24 INMUX_3	INMUX_3
23-19 —	Reserved
18-16 —	Reserved
15-11 —	Reserved
10-8 —	Reserved
7-3 —	Reserved
2-0 —	Reserved

8.7.14 Parameter_n Register (REG_B215_212 - REG_B267_264)

Offset

For a = 181 to 194:

Register	Offset
REG_B(4 * (a - 128) + 3)_(4 * (a - 128))	200h + (a × 4h)

Function

Controls access to PAD 0-511 specific for IMCR registers. Eight bits assigned per IMCR register have attributes of one of the implemented PDAC slots:

- 000-SIUL2_VIRTWRAPPER_PDAC1
- 001-SIUL2_VIRTWRAPPER_PDAC2
- 010-SIUL2_VIRTWRAPPER_PDAC4
- 011-SIUL2_VIRTWRAPPER_PDAC5
- 111-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any initiator can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any initiator can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0					INMUX_3			0					INMUX_2		
W																
Reset	0	0	0	0	0	1	1	1	0	0	0	0	0	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0					INMUX_1			0					INMUX_0		
W																
Reset	0	0	0	0	0	1	1	1	0	0	0	0	0	1	1	1

Fields

Field	Function
31-27 —	Reserved
26-24 INMUX_3	INMUX_3
23-19	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
18-16 INMUX_2	INMUX_2
15-11 —	Reserved
10-8 INMUX_1	INMUX_1
7-3 —	Reserved
2-0 INMUX_0	INMUX_0

8.7.15 Parameter_n Register (REG_B271_268)

Offset

Register	Offset
REG_B271_268	50Ch

Function

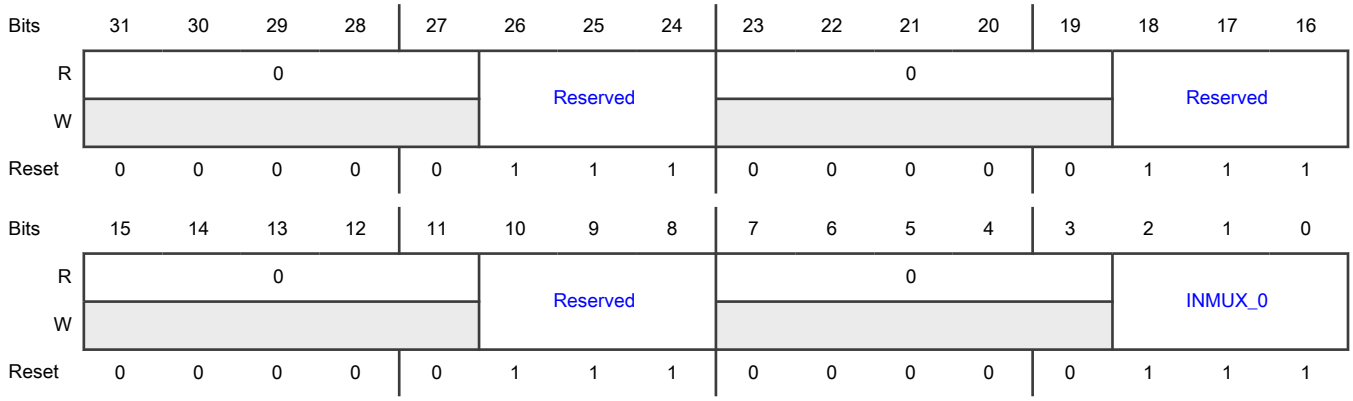
Controls access to PAD 0-511 specific for IMCR registers. Eight bits assigned per IMCR register have attributes of one of the implemented PDAC slots:

- 000-SIUL2_VIRTWRAPPER_PDAC1
- 001-SIUL2_VIRTWRAPPER_PDAC2
- 010-SIUL2_VIRTWRAPPER_PDAC4
- 011-SIUL2_VIRTWRAPPER_PDAC5
- 111-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any initiator can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any initiator can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-24 —	Reserved
23-19 —	Reserved
18-16 —	Reserved
15-11 —	Reserved
10-8 —	Reserved
7-3 —	Reserved
2-0 INMUX_0	INMUX_0

8.7.16 Parameter_n Register (REG_B291_288)

Offset

Register	Offset
REG_B291_288	520h

Function

Controls access to PAD 0-511 specific for IMCR registers. Eight bits assigned per IMCR register have attributes of one of the implemented PDAC slots:

- 000-SIUL2_VIRTWRAPPER_PDAC1
- 001-SIUL2_VIRTWRAPPER_PDAC2
- 010-SIUL2_VIRTWRAPPER_PDAC4
- 011-SIUL2_VIRTWRAPPER_PDAC5
- 111-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any initiator can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any initiator can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0					INMUX_3			0				INMUX_2			
W																
Reset	0	0	0	0	0	1	1	1	0	0	0	0	0	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0					INMUX_1			0				Reserved			
W																
Reset	0	0	0	0	0	1	1	1	0	0	0	0	0	1	1	1

Fields

Field	Function
31-27 —	Reserved
26-24 INMUX_3	INMUX_3

Table continues on the next page...

Table continued from the previous page...

Field	Function
23-19 —	Reserved
18-16 INMUX_2	INMUX_2
15-11 —	Reserved
10-8 INMUX_1	INMUX_1
7-3 —	Reserved
2-0 —	Reserved

8.7.17 Parameter_n Register (REG_B295_292 - REG_B307_304)

Offset

Register	Offset
REG_B295_292	524h
REG_B299_296	528h
REG_B303_300	52Ch
REG_B307_304	530h

Function

Controls access to PAD 0-511 specific for IMCR registers. Eight bits assigned per IMCR register have attributes of one of the implemented PDAC slots:

- 000-SIUL2_VIRTWRAPPER_PDAC1
- 001-SIUL2_VIRTWRAPPER_PDAC2
- 010-SIUL2_VIRTWRAPPER_PDAC4
- 011-SIUL2_VIRTWRAPPER_PDAC5
- 111-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any initiator can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any initiator can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0					INMUX_3			0					INMUX_2		
W																
Reset	0	0	0	0	0	1	1	1	0	0	0	0	0	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0					INMUX_1			0					INMUX_0		
W																
Reset	0	0	0	0	0	1	1	1	0	0	0	0	0	1	1	1

Fields

Field	Function
31-27 —	Reserved
26-24 INMUX_3	INMUX_3
23-19 —	Reserved
18-16 INMUX_2	INMUX_2
15-11 —	Reserved
10-8 INMUX_1	INMUX_1
7-3 —	Reserved
2-0 INMUX_0	INMUX_0

8.7.18 Parameter_n Register (REG_B311_308)

Offset

Register	Offset
REG_B311_308	534h

Function

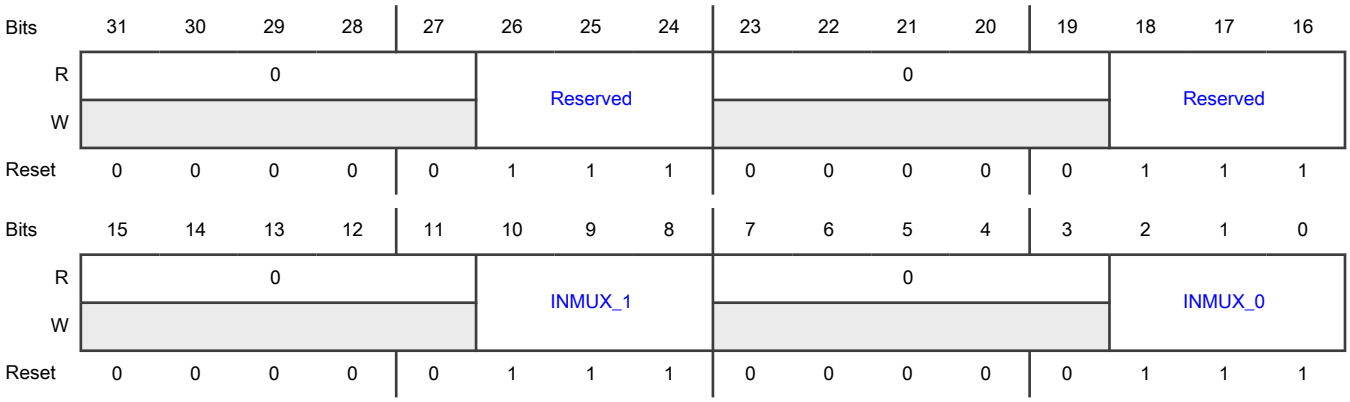
Controls access to PAD 0-511 specific for IMCR registers. Eight bits assigned per IMCR register have attributes of one of the implemented PDAC slots:

- 000-SIUL2_VIRTWRAPPER_PDAC1
- 001-SIUL2_VIRTWRAPPER_PDAC2
- 010-SIUL2_VIRTWRAPPER_PDAC4
- 011-SIUL2_VIRTWRAPPER_PDAC5
- 111-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any initiator can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any initiator can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-24 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
23-19 —	Reserved
18-16 —	Reserved
15-11 —	Reserved
10-8 INMUX_1	INMUX_1
7-3 —	Reserved
2-0 INMUX_0	INMUX_0

8.7.19 Parameter_n Register (REG_B315_312)

Offset

Register	Offset
REG_B315_312	538h

Function

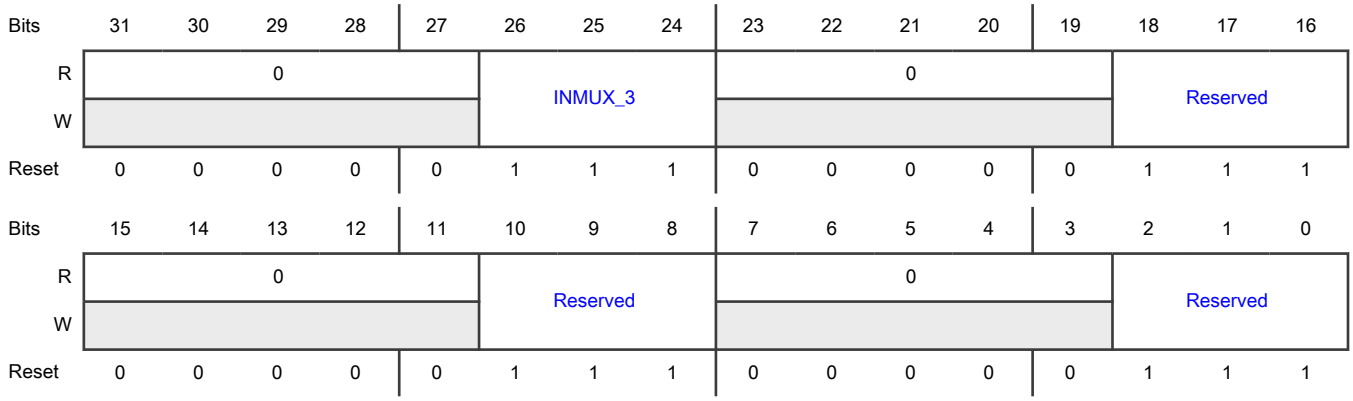
Controls access to PAD 0-511 specific for IMCR registers. Eight bits assigned per IMCR register have attributes of one of the implemented PDAC slots:

- 000-SIUL2_VIRTWRAPPER_PDAC1
- 001-SIUL2_VIRTWRAPPER_PDAC2
- 010-SIUL2_VIRTWRAPPER_PDAC4
- 011-SIUL2_VIRTWRAPPER_PDAC5
- 111-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any initiator can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any initiator can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-24 INMUX_3	INMUX_3
23-19 —	Reserved
18-16 —	Reserved
15-11 —	Reserved
10-8 —	Reserved
7-3 —	Reserved
2-0 —	Reserved

8.7.20 Parameter_n Register (REG_B319_316 - REG_B323_320)

Offset

Register	Offset
REG_B319_316	53Ch
REG_B323_320	540h

Function

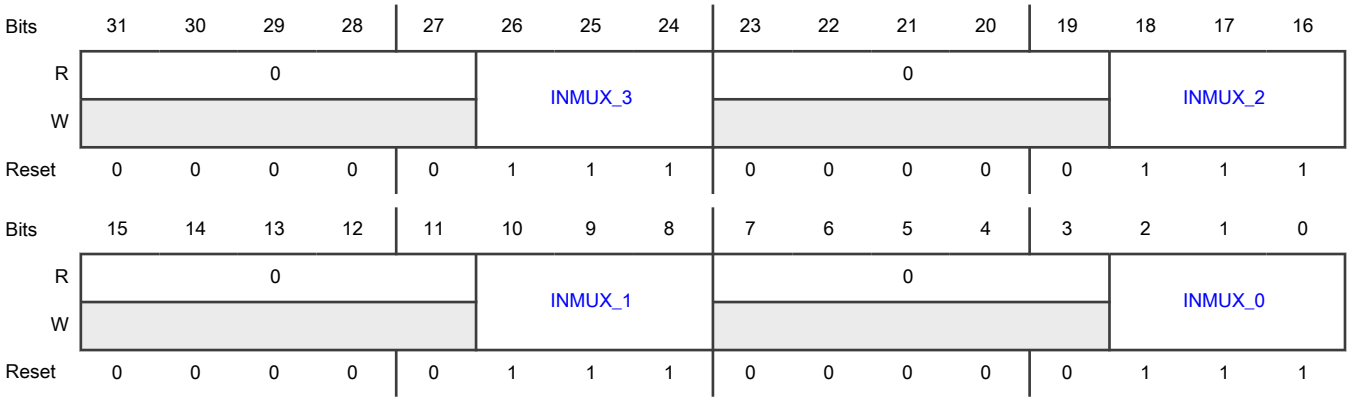
Controls access to PAD 0-511 specific for IMCR registers. Eight bits assigned per IMCR register have attributes of one of the implemented PDAC slots:

- 000-SIUL2_VIRTWRAPPER_PDAC1
- 001-SIUL2_VIRTWRAPPER_PDAC2
- 010-SIUL2_VIRTWRAPPER_PDAC4
- 011-SIUL2_VIRTWRAPPER_PDAC5
- 111-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any initiator can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any initiator can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-24	INMUX_3

Table continues on the next page...

Table continued from the previous page...

Field	Function
INMUX_3	
23-19 —	Reserved
18-16 INMUX_2	INMUX_2
15-11 —	Reserved
10-8 INMUX_1	INMUX_1
7-3 —	Reserved
2-0 INMUX_0	INMUX_0

8.7.21 Parameter_n Register (REG_B327_324)

Offset

Register	Offset
REG_B327_324	544h

Function

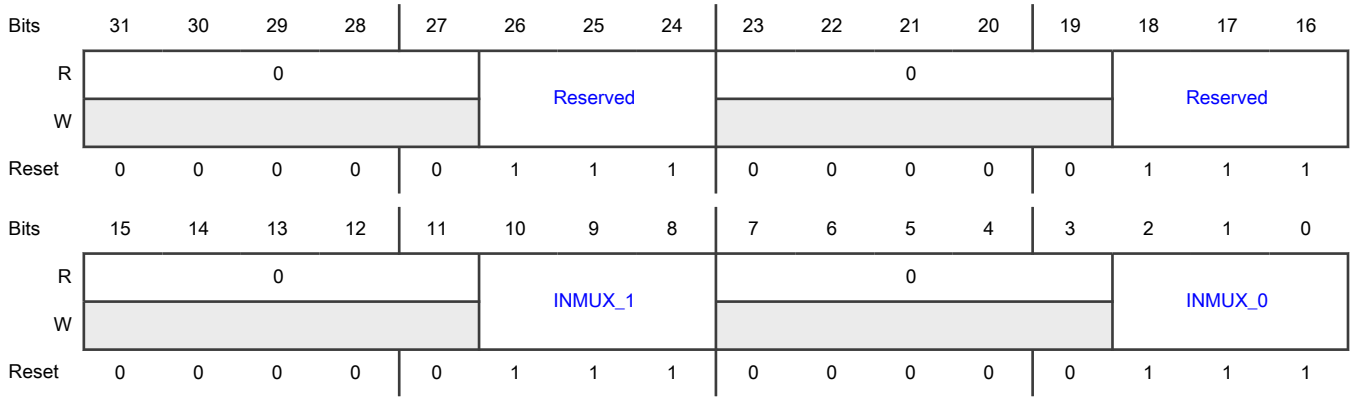
Controls access to PAD 0-511 specific for IMCR registers. Eight bits assigned per IMCR register have attributes of one of the implemented PDAC slots:

- 000-SIUL2_VIRTWRAPPER_PDAC1
- 001-SIUL2_VIRTWRAPPER_PDAC2
- 010-SIUL2_VIRTWRAPPER_PDAC4
- 011-SIUL2_VIRTWRAPPER_PDAC5
- 111-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any initiator can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any initiator can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-24 —	Reserved
23-19 —	Reserved
18-16 —	Reserved
15-11 —	Reserved
10-8 INMUX_1	INMUX_1
7-3 —	Reserved
2-0 INMUX_0	INMUX_0

8.7.22 Parameter_n Register (REG_B343_340)

Offset

Register	Offset
REG_B343_340	554h

Function

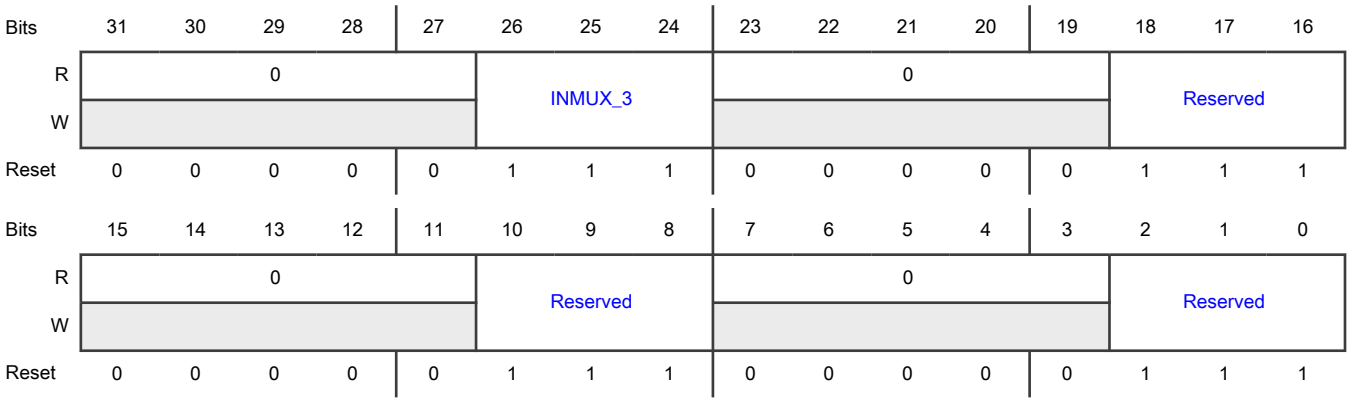
Controls access to PAD 0-511 specific for IMCR registers. Eight bits assigned per IMCR register have attributes of one of the implemented PDAC slots:

- 000-SIUL2_VIRTWRAPPER_PDAC1
- 001-SIUL2_VIRTWRAPPER_PDAC2
- 010-SIUL2_VIRTWRAPPER_PDAC4
- 011-SIUL2_VIRTWRAPPER_PDAC5
- 111-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any initiator can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any initiator can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-24 INMUX_3	INMUX_3

Table continues on the next page...

Table continued from the previous page...

Field	Function
23-19 —	Reserved
18-16 —	Reserved
15-11 —	Reserved
10-8 —	Reserved
7-3 —	Reserved
2-0 —	Reserved

8.7.23 Parameter_n Register (REG_B347_344 - REG_B367_364)

Offset

Register	Offset
REG_B347_344	558h
REG_B351_348	55Ch
REG_B355_352	560h
REG_B359_356	564h
REG_B363_360	568h
REG_B367_364	56Ch

Function

Controls access to PAD 0-511 specific for IMCR registers. Eight bits assigned per IMCR register have attributes of one of the implemented PDAC slots:

- 000-SIUL2_VIRTWRAPPER_PDAC1
- 001-SIUL2_VIRTWRAPPER_PDAC2
- 010-SIUL2_VIRTWRAPPER_PDAC4
- 011-SIUL2_VIRTWRAPPER_PDAC5
- 111-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any initiator can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any initiator can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0					INMUX_3			0					INMUX_2		
W																
Reset	0	0	0	0	0	1	1	1	0	0	0	0	0	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0					INMUX_1			0					INMUX_0		
W																
Reset	0	0	0	0	0	1	1	1	0	0	0	0	0	1	1	1

Fields

Field	Function
31-27 —	Reserved
26-24 INMUX_3	INMUX_3
23-19 —	Reserved
18-16 INMUX_2	INMUX_2
15-11 —	Reserved
10-8 INMUX_1	INMUX_1
7-3 —	Reserved
2-0 INMUX_0	INMUX_0

8.7.24 Parameter_n Register (REG_B371_368)

Offset

Register	Offset
REG_B371_368	570h

Function

Controls access to PAD 0-511 specific for IMCR registers. Eight bits assigned per IMCR register have attributes of one of the implemented PDAC slots:

- 000-SIUL2_VIRTWRAPPER_PDAC1
- 001-SIUL2_VIRTWRAPPER_PDAC2
- 010-SIUL2_VIRTWRAPPER_PDAC4
- 011-SIUL2_VIRTWRAPPER_PDAC5
- 111-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any initiator can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any initiator can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0					Reserved			0					INMUX_2		
W																
Reset	0	0	0	0	0	1	1	1	0	0	0	0	0	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0					INMUX_1			0					INMUX_0		
W																
Reset	0	0	0	0	0	1	1	1	0	0	0	0	0	1	1	1

Fields

Field	Function
31-27 —	Reserved
26-24 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
23-19 —	Reserved
18-16 INMUX_2	INMUX_2
15-11 —	Reserved
10-8 INMUX_1	INMUX_1
7-3 —	Reserved
2-0 INMUX_0	INMUX_0

8.7.25 Parameter_n Register (REG_B375_372)

Offset

Register	Offset
REG_B375_372	574h

Function

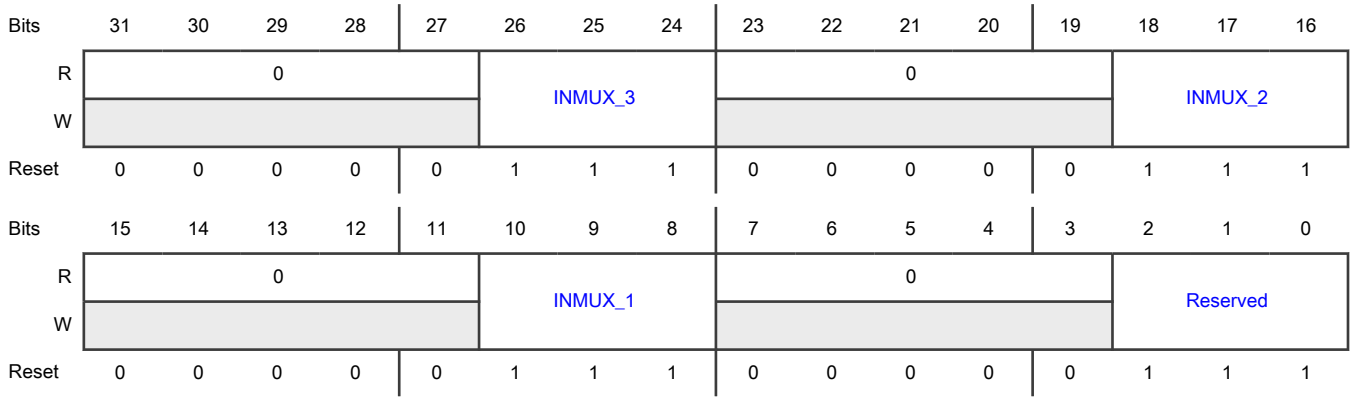
Controls access to PAD 0-511 specific for IMCR registers. Eight bits assigned per IMCR register have attributes of one of the implemented PDAC slots:

- 000-SIUL2_VIRTWRAPPER_PDAC1
- 001-SIUL2_VIRTWRAPPER_PDAC2
- 010-SIUL2_VIRTWRAPPER_PDAC4
- 011-SIUL2_VIRTWRAPPER_PDAC5
- 111-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any initiator can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any initiator can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-24 INMUX_3	INMUX_3
23-19 —	Reserved
18-16 INMUX_2	INMUX_2
15-11 —	Reserved
10-8 INMUX_1	INMUX_1
7-3 —	Reserved
2-0 —	Reserved

8.7.26 Parameter_n Register (REG_B379_376)

Offset

Register	Offset
REG_B379_376	578h

Function

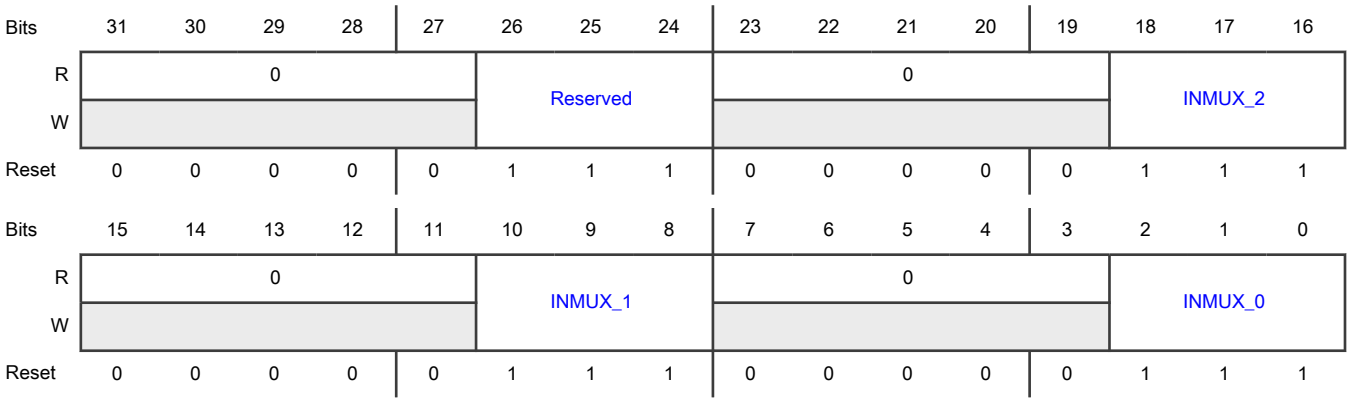
Controls access to PAD 0-511 specific for IMCR registers. Eight bits assigned per IMCR register have attributes of one of the implemented PDAC slots:

- 000-SIUL2_VIRTWRAPPER_PDAC1
- 001-SIUL2_VIRTWRAPPER_PDAC2
- 010-SIUL2_VIRTWRAPPER_PDAC4
- 011-SIUL2_VIRTWRAPPER_PDAC5
- 111-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any initiator can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any initiator can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-24 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
23-19 —	Reserved
18-16 INMUX_2	INMUX_2
15-11 —	Reserved
10-8 INMUX_1	INMUX_1
7-3 —	Reserved
2-0 INMUX_0	INMUX_0

8.7.27 Parameter_n Register (REG_B391_388)

Offset

Register	Offset
REG_B391_388	584h

Function

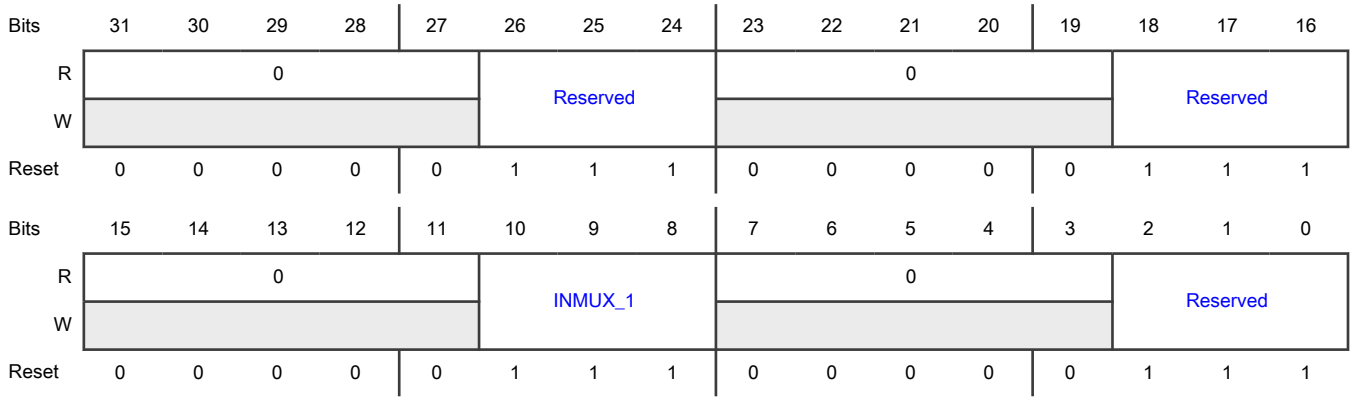
Controls access to PAD 0-511 specific for IMCR registers. Eight bits assigned per IMCR register have attributes of one of the implemented PDAC slots:

- 000-SIUL2_VIRTWRAPPER_PDAC1
- 001-SIUL2_VIRTWRAPPER_PDAC2
- 010-SIUL2_VIRTWRAPPER_PDAC4
- 011-SIUL2_VIRTWRAPPER_PDAC5
- 111-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any initiator can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any initiator can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-24 —	Reserved
23-19 —	Reserved
18-16 —	Reserved
15-11 —	Reserved
10-8 INMUX_1	INMUX_1
7-3 —	Reserved
2-0 —	Reserved

8.7.28 Parameter_n Register (REG_B399_396)

Offset

Register	Offset
REG_B399_396	58Ch

Function

Controls access to PAD 0-511 specific for IMCR registers. Eight bits assigned per IMCR register have attributes of one of the implemented PDAC slots:

- 000-SIUL2_VIRTWRAPPER_PDAC1
- 001-SIUL2_VIRTWRAPPER_PDAC2
- 010-SIUL2_VIRTWRAPPER_PDAC4
- 011-SIUL2_VIRTWRAPPER_PDAC5
- 111-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any initiator can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any initiator can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0					INMUX_3			0				INMUX_2			
W																
Reset	0	0	0	0	0	1	1	1	0	0	0	0	0	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0					Reserved			0				Reserved			
W																
Reset	0	0	0	0	0	1	1	1	0	0	0	0	0	1	1	1

Fields

Field	Function
31-27 —	Reserved
26-24 INMUX_3	INMUX_3

Table continues on the next page...

Table continued from the previous page...

Field	Function
23-19 —	Reserved
18-16 INMUX_2	INMUX_2
15-11 —	Reserved
10-8 —	Reserved
7-3 —	Reserved
2-0 —	Reserved

8.7.29 Parameter_n Register (REG_B411_408)

Offset

Register	Offset
REG_B411_408	598h

Function

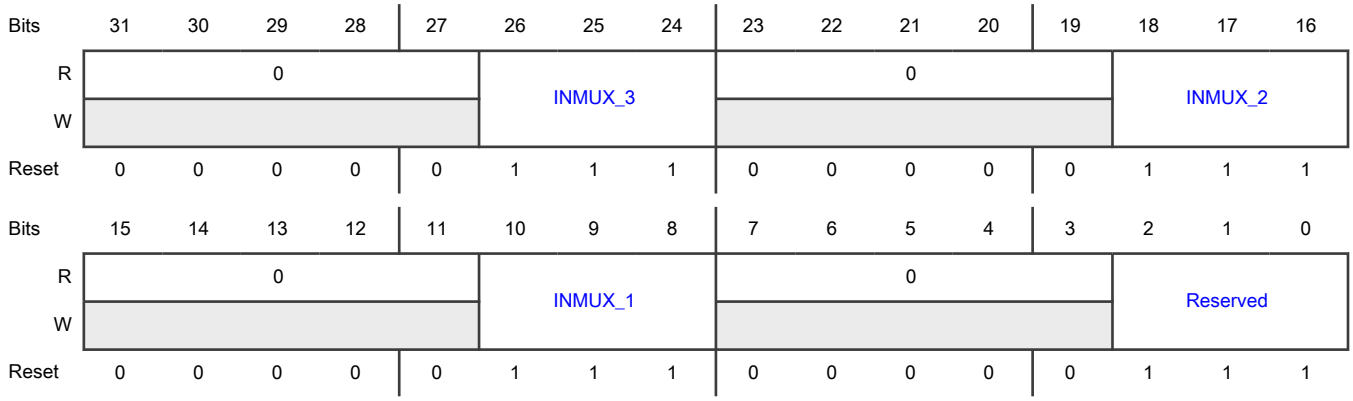
Controls access to PAD 0-511 specific for IMCR registers. Eight bits assigned per IMCR register have attributes of one of the implemented PDAC slots:

- 000-SIUL2_VIRTWRAPPER_PDAC1
- 001-SIUL2_VIRTWRAPPER_PDAC2
- 010-SIUL2_VIRTWRAPPER_PDAC4
- 011-SIUL2_VIRTWRAPPER_PDAC5
- 111-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any initiator can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any initiator can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-24 INMUX_3	INMUX_3
23-19 —	Reserved
18-16 INMUX_2	INMUX_2
15-11 —	Reserved
10-8 INMUX_1	INMUX_1
7-3 —	Reserved
2-0 —	Reserved

8.7.30 Parameter_n Register (REG_B415_412)

Offset

Register	Offset
REG_B415_412	59Ch

Function

Controls access to PAD 0-511 specific for IMCR registers. Eight bits assigned per IMCR register have attributes of one of the implemented PDAC slots:

- 000-SIUL2_VIRTWRAPPER_PDAC1
- 001-SIUL2_VIRTWRAPPER_PDAC2
- 010-SIUL2_VIRTWRAPPER_PDAC4
- 011-SIUL2_VIRTWRAPPER_PDAC5
- 111-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any initiator can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any initiator can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0					INMUX_3			0					INMUX_2		
W																
Reset	0	0	0	0	0	1	1	1	0	0	0	0	0	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0					INMUX_1			0					INMUX_0		
W																
Reset	0	0	0	0	0	1	1	1	0	0	0	0	0	1	1	1

Fields

Field	Function
31-27 —	Reserved
26-24 INMUX_3	INMUX_3

Table continues on the next page...

Table continued from the previous page...

Field	Function
23-19 —	Reserved
18-16 INMUX_2	INMUX_2
15-11 —	Reserved
10-8 INMUX_1	INMUX_1
7-3 —	Reserved
2-0 INMUX_0	INMUX_0

8.7.31 Parameter_n Register (REG_B419_416)

Offset

Register	Offset
REG_B419_416	5A0h

Function

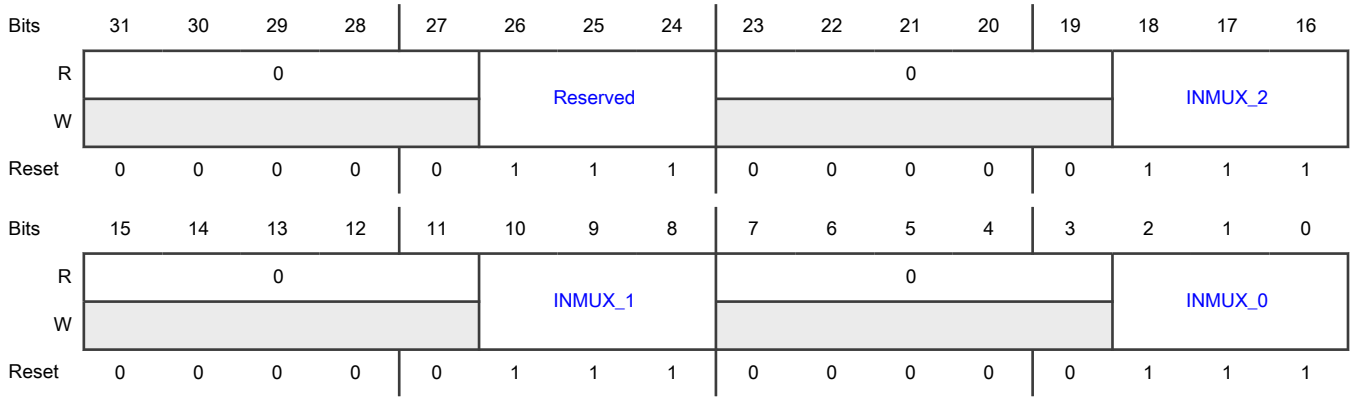
Controls access to PAD 0-511 specific for IMCR registers. Eight bits assigned per IMCR register have attributes of one of the implemented PDAC slots:

- 000-SIUL2_VIRTWRAPPER_PDAC1
- 001-SIUL2_VIRTWRAPPER_PDAC2
- 010-SIUL2_VIRTWRAPPER_PDAC4
- 011-SIUL2_VIRTWRAPPER_PDAC5
- 111-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any initiator can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any initiator can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-24 —	Reserved
23-19 —	Reserved
18-16 INMUX_2	INMUX_2
15-11 —	Reserved
10-8 INMUX_1	INMUX_1
7-3 —	Reserved
2-0 INMUX_0	INMUX_0

8.7.32 Parameter_n Register (REG_B443_440)

Offset

Register	Offset
REG_B443_440	5B8h

Function

Controls access to PAD 0-511 specific for IMCR registers. Eight bits assigned per IMCR register have attributes of one of the implemented PDAC slots:

- 000-SIUL2_VIRTWRAPPER_PDAC1
- 001-SIUL2_VIRTWRAPPER_PDAC2
- 010-SIUL2_VIRTWRAPPER_PDAC4
- 011-SIUL2_VIRTWRAPPER_PDAC5
- 111-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any initiator can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any initiator can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0					Reserved			0					Reserved		
W																
Reset	0	0	0	0	0	1	1	1	0	0	0	0	0	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0					Reserved			0					INMUX_0		
W																
Reset	0	0	0	0	0	1	1	1	0	0	0	0	0	1	1	1

Fields

Field	Function
31-27 —	Reserved
26-24 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
23-19 —	Reserved
18-16 —	Reserved
15-11 —	Reserved
10-8 —	Reserved
7-3 —	Reserved
2-0 INMUX_0	INMUX_0

8.7.33 Parameter_n Register (REG_B451_448 - REG_B471_468)

Offset

Register	Offset
REG_B451_448	5C0h
REG_B455_452	5C4h
REG_B459_456	5C8h
REG_B463_460	5CCh
REG_B467_464	5D0h
REG_B471_468	5D4h

Function

Controls access to PAD 0-511 specific for IMCR registers. Eight bits assigned per IMCR register have attributes of one of the implemented PDAC slots:

- 000-SIUL2_VIRTWRAPPER_PDAC1
- 001-SIUL2_VIRTWRAPPER_PDAC2
- 010-SIUL2_VIRTWRAPPER_PDAC4
- 011-SIUL2_VIRTWRAPPER_PDAC5
- 111-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any initiator can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any initiator can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0					INMUX_3			0					INMUX_2		
W																
Reset	0	0	0	0	0	1	1	1	0	0	0	0	0	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0					INMUX_1			0					INMUX_0		
W																
Reset	0	0	0	0	0	1	1	1	0	0	0	0	0	1	1	1

Fields

Field	Function
31-27 —	Reserved
26-24 INMUX_3	INMUX_3
23-19 —	Reserved
18-16 INMUX_2	INMUX_2
15-11 —	Reserved
10-8 INMUX_1	INMUX_1
7-3 —	Reserved
2-0 INMUX_0	INMUX_0

8.7.34 Parameter_n Register (REG_B475_472)

Offset

Register	Offset
REG_B475_472	5D8h

Function

Controls access to PAD 0-511 specific for IMCR registers. Eight bits assigned per IMCR register have attributes of one of the implemented PDAC slots:

- 000-SIUL2_VIRTWRAPPER_PDAC1
- 001-SIUL2_VIRTWRAPPER_PDAC2
- 010-SIUL2_VIRTWRAPPER_PDAC4
- 011-SIUL2_VIRTWRAPPER_PDAC5
- 111-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any initiator can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any initiator can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0					Reserved			0					Reserved		
W																
Reset	0	0	0	0	0	1	1	1	0	0	0	0	0	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0					INMUX_1			0					INMUX_0		
W																
Reset	0	0	0	0	0	1	1	1	0	0	0	0	0	1	1	1

Fields

Field	Function
31-27 —	Reserved
26-24 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
23-19 —	Reserved
18-16 —	Reserved
15-11 —	Reserved
10-8 INMUX_1	INMUX_1
7-3 —	Reserved
2-0 INMUX_0	INMUX_0

8.7.35 Parameter_n Register (REG_C)

Offset

Register	Offset
REG_C	800h

Function

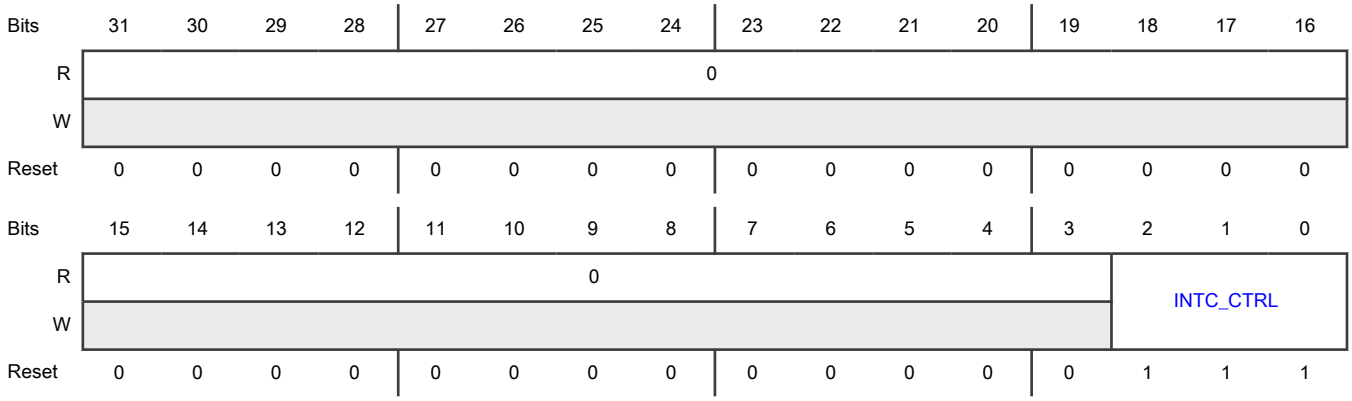
Controls access to DISR0, DIRER0, DIRSR0, IREER0, IFEER0, IFER0, IFMCR, and IFCPR0 interrupt registers. Eight bits assigned per PDAC slot have attributes of one of the implemented PDAC slots:

- 000-SIUL2_VIRTWRAPPER_PDAC1
- 001-SIUL2_VIRTWRAPPER_PDAC2
- 010-SIUL2_VIRTWRAPPER_PDAC4
- 011-SIUL2_VIRTWRAPPER_PDAC5
- 111-SIUL2_VIRTWRAPPER_PDAC0

NOTE

After reset, when XRDC is not configured, any initiator can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram



Fields

Field	Function
31-3 —	Reserved
2-0 INTC_CTRL	Interrupt register control This bit controls the interrupt register.

8.7.36 Parameter_n Register (REG_D)

Offset

Register	Offset
REG_D	C00h

Function

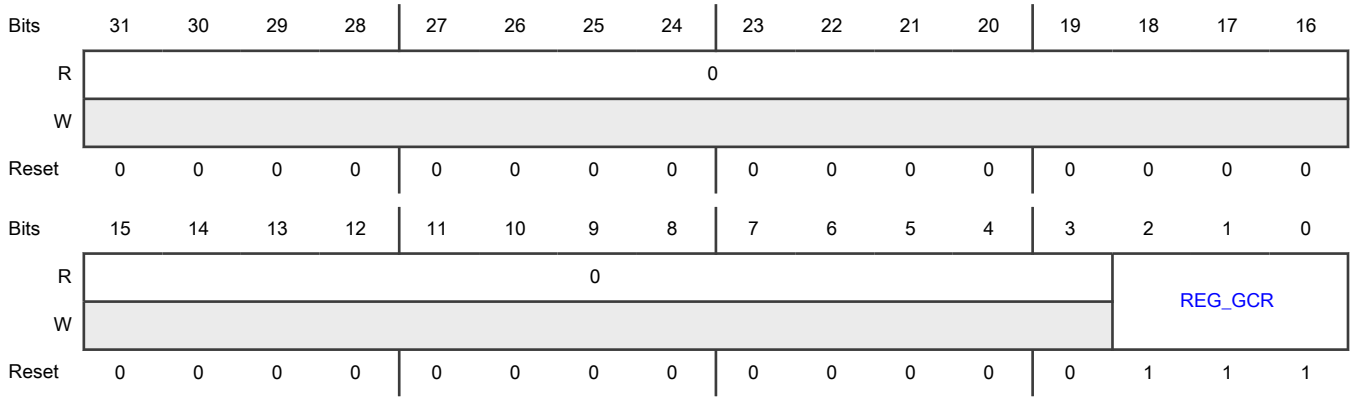
Controls access to GCR protection wrapper registers. Eight bits assigned per PDAC slot have attributes of one of the implemented PDAC slots:

- 000-SIUL2_VIRTWRAPPER_PDAC1
- 001-SIUL2_VIRTWRAPPER_PDAC2
- 010-SIUL2_VIRTWRAPPER_PDAC4
- 011-SIUL2_VIRTWRAPPER_PDAC5
- 111-SIUL2_VIRTWRAPPER_PDAC0

NOTE

After reset, when XRDC is not configured, any initiator can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram



Fields

Field	Function
31-3 —	Reserved
2-0 REG_GCR	GCR Register Of REG_PROT Controls access to GCR protection wrapper registers.

8.8 Glossary

VM Virtualized module, such as SIUL2.

PDAC Peripheral domain access control. PDAC and PDAC slot are interchangeable terms.

Chapter 9
Virtualization Wrapper (VIRT_WRAPPER) for all chips except S32K388 and S32K389

9.1 Chip-specific VIRT_WRAPPER information

9.1.1 Virtual Wrapper instances

This chip has one instance of Virtual Wrapper.

9.1.2 System overview of PDAC scheme

This figure shows the interaction of XRDC (containing PDACs), related to different cores (domain IDs), with VIRT_WRAPPER.

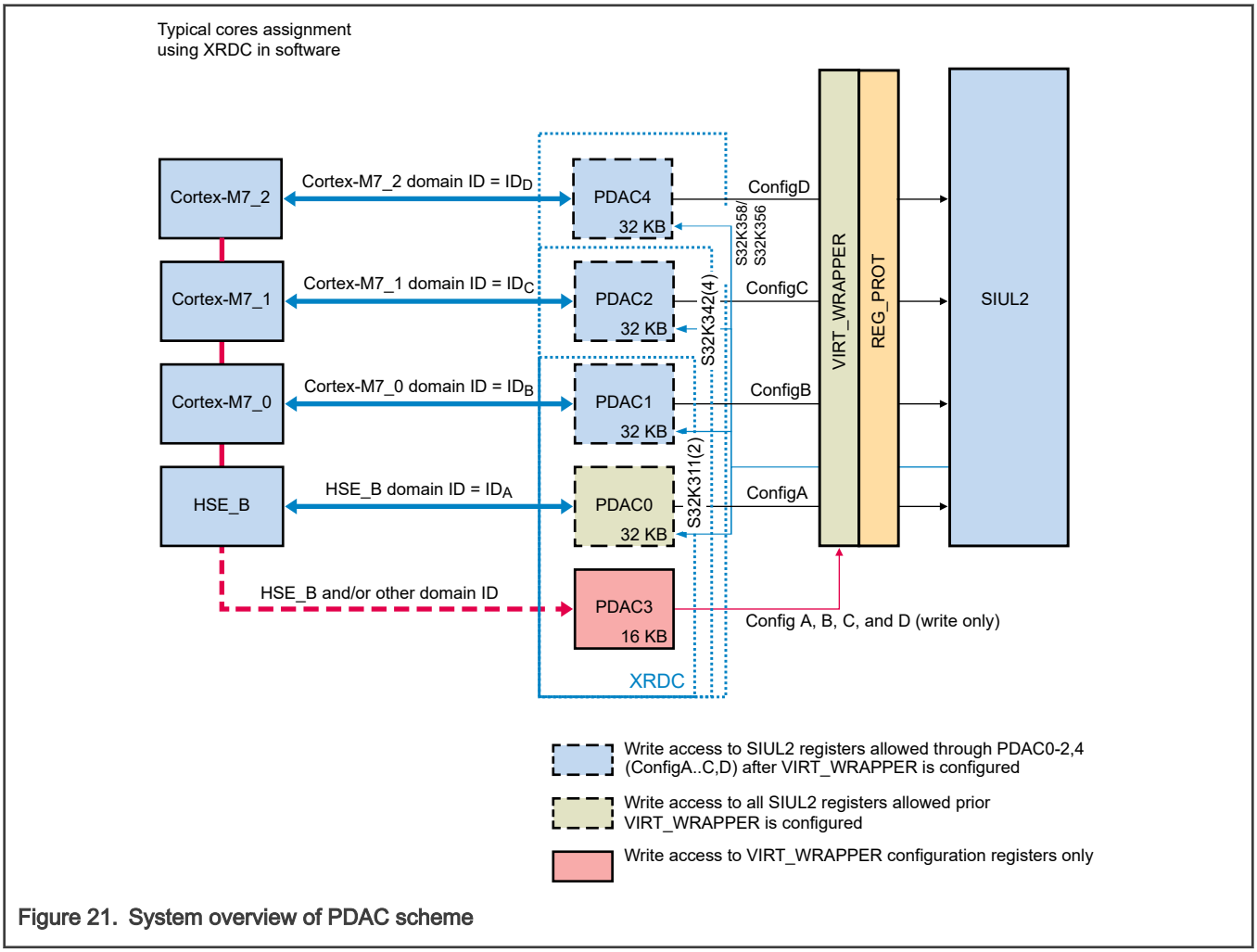


Figure 21. System overview of PDAC scheme

9.1.3 Initial VIRT_WRAPPER operation

Initially, VIRT_WRAPPER:

- Protects the "ConfigB-C" paths to SIUL2. You must configure VIRT_WRAPPER to define the SIUL2 R/W control registers that you can access through the "ConfigB-C" VIRT_WRAPPER access paths.

- Does not protect the direct path to SIUL2. It allows write accesses to all the SIUL2 R/W control registers. If PDAC0 is set to allow write access only to master core, for example HSE-B core (variant supported by the chip), then other masters cannot program the SIUL2 register. However, if these non-HSE_B masters program SIUL2 registers using other PDAC slots, a transfer error occurs. Any core can act as master core.
- Protects the "ConfigB-D" paths to SIUL2. You must configure VIRT_WRAPPER to define the SIUL2 R/W control registers that you can access through the "ConfigB-D" VIRT_WRAPPER access paths.
- The Virtualization Wrapper shall be configured using "IPS register interface". The Virtualization Wrapper access paths are accessible using IPS register interface.

9.1.4 Additional VIRT_WRAPPER details

See the "SIUL2 memory map overview and protection" table, later in this chapter, for an overview of the SIUL2 registers and their protection attributes.

By default, any core can access SIUL2 registers through PDAC0. This process is called ANY_MASTER. However, after XRDC configuration, HSE_B:

- Assigns PDACs to the respective cores or domain IDs
- Locks the XRDC configuration
- Programs VIRT_WRAPPER registers for pad assignments

HSE_B accesses PDAC0 solely for the pads having 11b as the value in their corresponding registers. Also, HSE_B retains the same value for the pads that it stores. In case HSE_B needs to take control of some pins, it can still configure PDAC*n* to be accessible not only from a specific core master, but also from HSE_B, before it allows other cores to execute.

PDAC3 protects the IPS register portal of VIRT_WRAPPER configuration registers that any master can access but only via the memory slot assigned to PDAC3. The configuration via IPS register portal can be locked to prevent any further changes until the next functional reset. PDAC3 implements this locking.

The protection information of the parallel port output registers is inherited from the protection information of the included pads, according to these rules:

- When all the pads assigned to a GPIO port share the same protection information, the corresponding parallel port output register for this port inherits the same protection information.
- When at least one of the pads assigned to a GPIO port has different protection information than the other pads assigned to this GPIO port, access to the corresponding parallel port output register for this port is disabled for all the masters. Because of the aforementioned protection group mapping, the data bits of a register encode the protection control information related to one full GPIO port, or a chunk of 16 pads.
- You can assign the following SIUL2 registers to individual PDAC control:
  - DISR0
  - DIRER0
  - DIRSR0
  - IREER0
  - IFEER0
  - IFER0
  - IFMCR0–31
  - IFCPR

  Then, you can access all these registers only through the assigned PDAC because the VIRT_WRAPPER_REG_C1039_1024 register controls the SIUL2 interrupt registers.
- You cannot access the SIUL2 R/W control registers configured through PDAC3 to be accessible through the ConfigA VIRT_WRAPPER access path, using the ConfigB VIRT_WRAPPER access path.

- You cannot access any of the SIUL2 R/W control registers through ConfigB-C VIRT_WRAPPER access paths prior to VIRT_WRAPPER configuration. By default, any core can access SIUL2 registers through PDAC0 after reset.
- The same PDAC that accesses the main SIUL2 registers is used to access the SIUL2-mirrored registers and soft-lock-bit registers. This is because these registers are a part of the same 16 KB space.
- Access to mirrored SIUL2 registers sets the bit in Soft lock bit as an inherent property of register protection. See the "Register Protection" chapter for details on the soft lock bit behavior, when accessing the mirrored region.
- You cannot access any of the SIUL2 R/W control registers through ConfigB-D VIRT_WRAPPER access paths prior to VIRT_WRAPPER configuration. By default, any core can access SIUL2 registers through PDAC0 after reset. Once Virtual wrapper is configured, each attempt to write to SIUL R/W register through any of unassigned paths shall result in a transfer error.

9.2 Introduction

Virtualization refers to the various techniques, methods, or approaches of creating a virtual (rather than actual) version of something, such as a virtual hardware platform, operating system (OS), storage device, or network resources. The term "hardware virtualization" refers to the creation of a virtual machine that acts like a real computer with an operating system. The underlying hardware resources separate the software executed on these virtual machines (named "guest program"). In many cases, the specifically modified guest programs are required to run in such a virtual environment.

There are different types of hardware virtualization. One of them is para-virtualization. The para-virtualization is a non-simulated hardware environment. However, a guest program is executed in its own isolated domain, as if it is running on a separate system. Such a behavior is especially beneficial for the software targeted toward functional safety, because it allows freedom of interference for certain aspects of this software.

The hardware-assisted virtualization is a way of improving the efficiency of hardware virtualization. It involves employing specially designed CPUs and hardware components that help improve the performance of a guest environment. VIRT_WRAPPER, described in this chapter, is such a hardware component.

9.2.1 Overview

Virtualization is implemented by protecting (for example, granting or inhibiting) the access to a register within the virtualized module, dependent on the specific criteria. Virtualized module is the module instance associated with VIRT_WRAPPER. In this chip, SIUL2 is the virtualized module. Registers within virtualized module are virtualized by granting or inhibiting the access to different PDAC slots as encoded within virtualization pad assignments in VIRT_WRAPPER configuration registers. For this purpose, the virtualization information is programmed into VIRT_WRAPPER configuration registers that specify the grouping of registers within the virtualized module. Upon an access to virtualized module through the peripheral interface, the grouping information is exercised to identify whether the corresponding transaction should be granted or inhibited. The following figure depicts the usage of VIRT_WRAPPER in combination with the virtualized module.

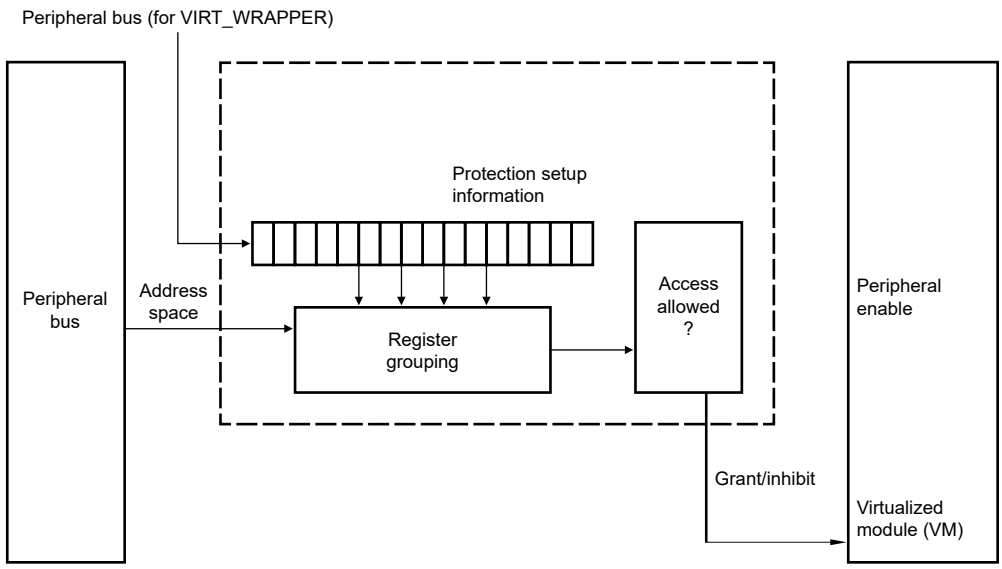


Figure 22. VIRT_WRAPPER block diagram

9.2.2 Features

VIRT_WRAPPER includes these features:

- A programmable chip-specific virtualization setup, capable to select a subset of the available masters in combination with up to four available address spaces
- Virtualizing accesses to the registers within a VM by inhibiting accesses to a subset of these registers under control of the specified virtualization information. Only write accesses can be inhibited for a specific master or set of masters.

9.2.3 Modes of operation

VIRT_WRAPPER is operable when the VM is operable. For details about the availability of the VM, see the chapter of the corresponding module. When there is no virtualization information specified for the VIRT_WRAPPER module, if XRDC is not configured, PDAC slot 0 is open as the default value of configuration register is 11b that is assigned with PDAC slot 0 and any master can access through it.

9.3 Functional description

This section describes the following topics:

- PDAC-based protection scheme
- Register group mapping considering SIUL2
- Access errors

9.3.1 PDAC-based protection scheme

VIRT_WRAPPER uses a protection scheme of the PDACs sub-blocks of XRDC. Each PDAC slot is assigned to one or more cores (by domain ID assignment in XRDC) as shown in Chip-specific VIRT_WRAPPER information section. Only the write accesses are protected by the virtualization feature. Read accesses are not impacted.

For any protection group defined within the register, two bits specify the protection control information according to the following table.

Table 42. Protection control information for a single group

Value	Mnemonic	Protection	Description
00	PDAC1	Restricted access of registers, MSCR, IMCR, to PDAC slot 1	Protection setup information for PDAC slot 1 is used.
01	PDAC2	Restricted access of registers, MSCR, IMCR, to PDAC slot 2	Protection setup information for PDAC slot 2 is used.
10	PDAC3	Restricted access of registers, MSCR, IMCR, to PDAC slot 3	Protection setup information for PDAC slot 3 is used.
11	PDAC0	By default, any core can access SIUL2 registers through PDAC slot 0. This process is called ANY_MASTER.	Protection setup information for PDAC slot 0 is used.

9.3.2 Clocking

This module has no major clocking considerations.

9.3.3 Register group mapping for SIUL2

For the SIUL2 instance, the corresponding virtualization information within the VIRT_WRAPPER module is defined on a per-pad basis. Additionally, the virtualization information protects the control registers (controlling the input multiplexing scheme).

9.3.3.1 Virtualization of pad output registers

Any functional pad (controlled by the SIUL2 instance) is assigned to a separate protection group. This scheme excludes the pads that an MSCR register does not control within an SIUL2 instance (for example, power supply pads). As a result, the protection granularity defined by the virtualization information is a single pad. The control information associated with this protection group affects all the registers related to this pad.

The protection control information for pad *i* is specified within the protection group *i*; thus, it enables a very simple assignment scheme for the related protection control data that is hard-coded. The protection control information for pad *i* affects all the registers related to this pad, in the MSCR*i* and the GPDO*i* registers (in case such registers exist).

SIUL2 can control up to 512 pads that the MSCR, GPDO, and GPDI registers can access individually. The actual number of pads associated with a SIUL2 instance is specific to implementation. GPIO pads are also organized in GPIO ports consisting of a maximum of 16 pads that the parallel port registers (PGPDO, PGPDI, MPGPDO) can access.

Only the pad control and pad output registers are protected. Accesses to the read-only registers GPDI and PGPDI are not affected by virtualization.

As the protection granularity is a single pad, any virtualization information associated with this pad affects the corresponding MCSR and GPDO registers directly.

The parallel port output registers (PGPDO, MPGPDO) inherit the protection information from the protection information of the included pads according to the following rules:

- When all pads assigned to a GPIO port share the same protection information, the corresponding parallel port output register for this port inherits the same protection information.
- When at least one of the pads assigned to a GPIO port has a different protection information than the other pads assigned to this GPIO port, the write access to the corresponding parallel port output register (PGPDO and MPGPDO) for this port is disabled for all the masters through all the PDAC slots and a transfer error is generated.

Due to the aforementioned protection group mapping, the data bits of a register encode the protection control information related to one full GPIO port, or a chunk of 16 pads.

9.3.3.2 Virtualization of input multiplexing control registers

Additionally, the SIUL2 instance supports the control of an input multiplexer (INMUX) scheme. This scheme provides the capability to select one of a set of the input pads to be the source of an input function for some of the peripheral modules. Any INMUX controlled by the SIUL2 instance is assigned to a separate protection group. It therefore provides an equivalent protection granularity within the virtualization information.

The protection group 512+*i* specifies the protection control information for the input multiplexer INMUX *i*. This enables a very simple assignment scheme for the related protection control data that is hard-coded. The protection control information for INMUX *i* affects the input multiplexing control register related to this input—namely, IMCR*i*.

SIUL2 can control up to 512 MSCRs and 512 IMCRs input multiplexers that the IMCRs can access individually. GPIO inputs are not affected by the input multiplexer scheme. Therefore, the associated virtualization information does not affect the corresponding pad input registers.

As the protection granularity is a single INMUX, any virtualization information associated with this INMUX affects the corresponding IMCR directly.

9.3.3.3 Virtualization of interrupt control registers

Additionally, the complete set of interrupt control registers within SIUL2 (address offset range 0010h–00C3h) can be protected as a separate group (protection group #1024).

The following table shows an overview of the SIUL2 registers and their protection attributes.

Table 43. SIUL2 memory map overview and protection

Offset range	Register	Size (bits)	Protected	Description
0000–000Fh	MIDR1, MIDR2	32	N	Read-only registers, not protected
0010–00C3h	Interrupt registers	32	Y	Protected as a single group—no individual protection of related registers
0240–A3Fh [1]	MSCR	32	Y	Amount of registers defines maximum number of PADs to be controlled and protected
0A40–0123Fh	IMCR	32	Y	Amount of registers defines maximum number of INMUXes to be controlled and protected
1300–14FFh	GPDO GPDO[0] (8-bit) register controls single PAD[0] pad means that if PAD[0] is assigned to PDAC slot 1 then PDAC slot 1 can write to GPDO[0] (8-bit).	8	Y	There are fewer GPDO registers than MSCR registers, as some PADs are not made available as GPIO PADs, but their electrical characteristics can still be programmed
1500–16FFh	GPDI	8	N	PAD input register, not protected
1700–173Fh	PGPDO PGPDO[0] (16-bit) register controls PAD[0-15] pads meaning if only all the PADs[0-15] are assigned to PDAC slot 1, then PDAC slot 1 can	16	Y	Writable with 8-, 16-, and as a pair with 32-bit accesses

Table continues on the next page...

Table 43. SIUL2 memory map overview and protection (continued)

Offset range	Register	Size (bits)	Protected	Description
	write to PGPDO[0] (16-bit).			
1740–177Fh	PGPDI	16	N	PAD input register, not protected
1780–17FFh	MPGPDO MPGPDO[0] (32-bit) register controls PAD[0-15] pads meaning if only all the PADs[0-15] are assigned to PDAC slot 1, then PDAC slot 1 can write to MPGPDO[0] (32-bit).	32	Y	Only writable with 32-bit accesses
2010–20C3h	Reserved	—	—	—
2240–2A3Fh	Reserved	—	—	—
2A40–323Fh	Reserved	—	—	—
3300–34FFh	Reserved	—	—	—
4000–47FFh	Reserved	—	—	—
4800–48FFh	Reserved	—	—	—

1. The SIUL2 block guide specifies the support of a maximum of 512 MSCRs.

9.4 External signals

This module has no external signals.

9.5 Initialization

This module does not require initialization.

9.6 Application Information

This module supports virtualization of accesses to the registers within a SIUL2 by inhibiting or granting accesses to a subset of these registers under control of the specified master(core) information.

9.7 VIRT_WRAPPER memory map register descriptions

NOTE

Access to reserved spaces outside the register bank and holes (unimplemented registers) within register bank generates the transfer error. Access to offset 104h does not generate any transfer error.

9.7.1 VIRT_WRAPPER memory map

VIRTUAL_WRAPPER base address: 402A_8000h

Offset	Register	Width (In bits)	Access	Reset value
0h - 4h	Parameter_n Register (REG_A0 - REG_A1)	32	RW	FFFF_FFFFh
8h	Parameter_n Register (REG_A2)	32	RW	FFFF_FFFFh
Ch - 34h	Parameter_n Register (REG_A3 - REG_A13)	32	RW	FFFF_FFFFh
38h	Parameter_n Register (REG_A14)	32	RW	FFFF_FFFFh
80h	Parameter_n Register (REG_B0)	32	RW	FFFF_FFFFh
84h - 8Ch	Parameter_n Register (REG_B1 - REG_B3)	32	RW	FFFF_FFFFh
90h	Parameter_n Register (REG_B4)	32	RW	FFFF_FFFFh
94h	Parameter_n Register (REG_B5)	32	RW	FFFF_FFFFh
98h	Parameter_n Register (REG_B6)	32	RW	FFFF_FFFFh
9Ch	Parameter_n Register (REG_B7)	32	RW	FFFF_FFFFh
A0h	Parameter_n Register (REG_B8)	32	RW	FFFF_FFFFh
A4h	Parameter_n Register (REG_B9)	32	RW	FFFF_FFFFh
A8h - ACh	Parameter_n Register (REG_B10 - REG_B11)	32	RW	FFFF_FFFFh
B0h	Parameter_n Register (REG_B12)	32	RW	FFFF_FFFFh
B4h	Parameter_n Register (REG_B13)	32	RW	FFFF_FFFFh
B8h - BCh	Parameter_n Register (REG_B14 - REG_B15)	32	RW	FFFF_FFFFh
C0h	Parameter_n Register (REG_B16)	32	RW	FFFF_FFFFh
C8h	Parameter_n Register (REG_B18)	32	RW	FFFF_FFFFh
CCh	Parameter_n Register (REG_B19)	32	RW	FFFF_FFFFh
D0h	Parameter_n Register (REG_B20)	32	RW	FFFF_FFFFh
D4h	Parameter_n Register (REG_B21)	32	RW	FFFF_FFFFh
D8h	Parameter_n Register (REG_B22)	32	RW	FFFF_FFFFh
DCh	Parameter_n Register (REG_B23)	32	RW	FFFF_FFFFh
E0h	Parameter_n Register (REG_B24)	32	RW	FFFF_FFFFh
E4h	Parameter_n Register (REG_B25)	32	RW	FFFF_FFFFh
E8h	Parameter_n Register (REG_B26)	32	RW	FFFF_FFFFh
ECh	Parameter_n Register (REG_B27)	32	RW	FFFF_FFFFh
100h	Parameter_n Register (REG_C1039_1024)	32	RW	FFFF_FFFFh

9.7.2 Parameter_n Register (REG_A0 - REG_A1)

Offset

Register	Offset
REG_A0	0h
REG_A1	4h

Function

This register set is for PAD0-511. They control MSCR, GPDO, PGPDO, and MPGPDO. Two bits assigned per PDAC slot have attributes of one of the implemented PDAC slots:

- 00-SIUL2_VIRTWRAPPER_PDAC1
- 01-SIUL2_VIRTWRAPPER_PDAC2
- 10-SIUL2_VIRTWRAPPER_PDAC4
- 11-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any master can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any master can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PAD_15		PAD_14		PAD_13		PAD_12		PAD_11		PAD_10		PAD_9		PAD_8	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PAD_7		PAD_6		PAD_5		PAD_4		PAD_3		PAD_2		PAD_1		PAD_0	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-30 PAD_15	PAD_15 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4

Table continues on the next page...

Table continued from the previous page...

Field	Function
	11b - SIUL2_VIRTWRAPPER_PDAC0
29-28 PAD_14	PAD_14 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
27-26 PAD_13	PAD_13 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
25-24 PAD_12	PAD_12 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
23-22 PAD_11	PAD_11 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
21-20 PAD_10	PAD_10 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
19-18 PAD_9	PAD_9 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
17-16	PAD_8

Table continues on the next page...

Table continued from the previous page...

Field	Function
PAD_8	00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
15-14 PAD_7	PAD_7 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
13-12 PAD_6	PAD_6 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
11-10 PAD_5	PAD_5 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
9-8 PAD_4	PAD_4 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
7-6 PAD_3	PAD_3 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
5-4 PAD_2	PAD_2 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
3-2 PAD_1	PAD_1 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
1-0 PAD_0	PAD_0 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0

9.7.3 Parameter_n Register (REG_A2)

Offset

Register	Offset
REG_A2	8h

Function

This register set is for PAD0-511. They control MSCR, GPDO, PGPDO, and MPGPDO. Two bits assigned per PDAC slot have attributes of one of the implemented PDAC slots:

- 00-SIUL2_VIRTWRAPPER_PDAC1
- 01-SIUL2_VIRTWRAPPER_PDAC2
- 10-SIUL2_VIRTWRAPPER_PDAC4
- 11-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any master can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any master can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PAD_15		PAD_14		PAD_13		PAD_12		PAD_11		PAD_10		PAD_9		PAD_8	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		Reserved		PAD_5		PAD_4		PAD_3		PAD_2		PAD_1		PAD_0	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-30 PAD_15	PAD_15 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
29-28 PAD_14	PAD_14 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
27-26 PAD_13	PAD_13 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
25-24 PAD_12	PAD_12 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
23-22 PAD_11	PAD_11 00b - SIUL2_VIRTWRAPPER_PDAC1

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
21-20 PAD_10	PAD_10 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
19-18 PAD_9	PAD_9 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
17-16 PAD_8	PAD_8 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
15-14 —	Reserved
13-12 —	Reserved
11-10 PAD_5	PAD_5 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
9-8 PAD_4	PAD_4 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-6 PAD_3	PAD_3 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
5-4 PAD_2	PAD_2 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
3-2 PAD_1	PAD_1 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
1-0 PAD_0	PAD_0 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0

9.7.4 Parameter_n Register (REG_A3 - REG_A13)

Offset

For a = 3 to 13:

Register	Offset
REG_Aa	0h + (a × 4h)

Function

This register set is for PAD0-511. They control MSCR, GPDO, PGPDO, and MPGPDO. Two bits assigned per PDAC slot have attributes of one of the implemented PDAC slots:

- 00-SIUL2_VIRTWRAPPER_PDAC1
- 01-SIUL2_VIRTWRAPPER_PDAC2
- 10-SIUL2_VIRTWRAPPER_PDAC4

- 11-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any master can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any master can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PAD_15		PAD_14		PAD_13		PAD_12		PAD_11		PAD_10		PAD_9		PAD_8	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PAD_7		PAD_6		PAD_5		PAD_4		PAD_3		PAD_2		PAD_1		PAD_0	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-30 PAD_15	PAD_15 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
29-28 PAD_14	PAD_14 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
27-26 PAD_13	PAD_13 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
25-24 PAD_12	PAD_12

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
23-22 PAD_11	PAD_11 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
21-20 PAD_10	PAD_10 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
19-18 PAD_9	PAD_9 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
17-16 PAD_8	PAD_8 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
15-14 PAD_7	PAD_7 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
13-12 PAD_6	PAD_6 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
11-10 PAD_5	PAD_5 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
9-8 PAD_4	PAD_4 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
7-6 PAD_3	PAD_3 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
5-4 PAD_2	PAD_2 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
3-2 PAD_1	PAD_1 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
1-0 PAD_0	PAD_0 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0

9.7.5 Parameter_n Register (REG_A14)

Offset

Register	Offset
REG_A14	38h

Function

This register set is for PAD0-511. They control MSCR, GPDO, PGPDO, and MPGPDO. Two bits assigned per PDAC slot have attributes of one of the implemented PDAC slots:

- 00-SIUL2_VIRTWRAPPER_PDAC1
- 01-SIUL2_VIRTWRAPPER_PDAC2
- 10-SIUL2_VIRTWRAPPER_PDAC4
- 11-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any master can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any master can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved		Reserved		Reserved		PAD_12		PAD_11		PAD_10		PAD_9		PAD_8	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PAD_7		PAD_6		PAD_5		PAD_4		PAD_3		PAD_2		PAD_1		PAD_0	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-30 —	Reserved
29-28 —	Reserved
27-26	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
25-24 PAD_12	PAD_12 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
23-22 PAD_11	PAD_11 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
21-20 PAD_10	PAD_10 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
19-18 PAD_9	PAD_9 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
17-16 PAD_8	PAD_8 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
15-14 PAD_7	PAD_7 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
13-12	PAD_6

Table continues on the next page...

Table continued from the previous page...

Field	Function
PAD_6	00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
11-10 PAD_5	PAD_5 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
9-8 PAD_4	PAD_4 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
7-6 PAD_3	PAD_3 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
5-4 PAD_2	PAD_2 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
3-2 PAD_1	PAD_1 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
1-0 PAD_0	PAD_0 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0

9.7.6 Parameter_n Register (REG_B0)

Offset

Register	Offset
REG_B0	80h

Function

Controls access to pads 0 to 255 specific for IMCR registers. Two bits assigned per IMCR register have attributes of one of the implemented PDAC slots:

- 00-SIUL2_VIRTWRAPPER_PDAC1
- 01-SIUL2_VIRTWRAPPER_PDAC2
- 10-SIUL2_VIRTWRAPPER_PDAC4
- 11-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any master can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any master can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		Reserved		INMUX_5		INMUX_4		INMUX_3		INMUX_2		INMUX_1		INMUX_0	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-30 —	Reserved
29-28 —	Reserved
27-26 —	Reserved
25-24 —	Reserved
23-22 —	Reserved
21-20 —	Reserved
19-18 —	Reserved
17-16 —	Reserved
15-14 —	Reserved
13-12 —	Reserved
11-10 INMUX_5	INMUX_5 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
9-8 INMUX_4	INMUX_4 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-6 INMUX_3	INMUX_3 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
5-4 INMUX_2	INMUX_2 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
3-2 INMUX_1	INMUX_1 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
1-0 INMUX_0	INMUX_0 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0

9.7.7 Parameter_n Register (REG_B1 - REG_B3)

Offset

Register	Offset
REG_B1	84h
REG_B2	88h
REG_B3	8Ch

Function

Controls access to pads 0 to 255 specific for IMCR registers. Two bits assigned per IMCR register have attributes of one of the implemented PDAC slots:

- 00-SIUL2_VIRTWRAPPER_PDAC1
- 01-SIUL2_VIRTWRAPPER_PDAC2

- 10-SIUL2_VIRTWRAPPER_PDAC4
- 11-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any master can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any master can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	INMUX_15		INMUX_14		INMUX_13		INMUX_12		INMUX_11		INMUX_10		INMUX_9		INMUX_8	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	INMUX_7		INMUX_6		INMUX_5		INMUX_4		INMUX_3		INMUX_2		INMUX_1		INMUX_0	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-30 INMUX_15	INMUX_15 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
29-28 INMUX_14	INMUX_14 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
27-26 INMUX_13	INMUX_13 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
25-24	INMUX_12

Table continues on the next page...

Table continued from the previous page...

Field	Function
INMUX_12	00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
23-22 INMUX_11	INMUX_11 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
21-20 INMUX_10	INMUX_10 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
19-18 INMUX_9	INMUX_9 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
17-16 INMUX_8	INMUX_8 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
15-14 INMUX_7	INMUX_7 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
13-12 INMUX_6	INMUX_6 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
11-10 INMUX_5	INMUX_5 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
9-8 INMUX_4	INMUX_4 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
7-6 INMUX_3	INMUX_3 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
5-4 INMUX_2	INMUX_2 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
3-2 INMUX_1	INMUX_1 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
1-0 INMUX_0	INMUX_0 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0

9.7.8 Parameter_n Register (REG_B4)

Offset

Register	Offset
REG_B4	90h

Function

Controls access to pads 0 to 255 specific for IMCR registers. Two bits assigned per IMCR register have attributes of one of the implemented PDAC slots:

- 00-SIUL2_VIRTWRAPPER_PDAC1
- 01-SIUL2_VIRTWRAPPER_PDAC2
- 10-SIUL2_VIRTWRAPPER_PDAC4
- 11-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any master can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any master can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	INMUX_7		INMUX_6		INMUX_5		INMUX_4		INMUX_3		INMUX_2		INMUX_1		INMUX_0	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-30 —	Reserved
29-28 —	Reserved
27-26	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
25-24 —	Reserved
23-22 —	Reserved
21-20 —	Reserved
19-18 —	Reserved
17-16 —	Reserved
15-14 INMUX_7	INMUX_7 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
13-12 INMUX_6	INMUX_6 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
11-10 INMUX_5	INMUX_5 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
9-8 INMUX_4	INMUX_4 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-6 INMUX_3	INMUX_3 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
5-4 INMUX_2	INMUX_2 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
3-2 INMUX_1	INMUX_1 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
1-0 INMUX_0	INMUX_0 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0

9.7.9 Parameter_n Register (REG_B5)

Offset

Register	Offset
REG_B5	94h

Function

Controls access to pads 0 to 255 specific for IMCR registers. Two bits assigned per IMCR register have attributes of one of the implemented PDAC slots:

- 00-SIUL2_VIRTWRAPPER_PDAC1
- 01-SIUL2_VIRTWRAPPER_PDAC2
- 10-SIUL2_VIRTWRAPPER_PDAC4
- 11-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any master can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any master can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	INMUX_15		INMUX_14		INMUX_13		INMUX_12		INMUX_11		INMUX_10		INMUX_9		INMUX_8	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	INMUX_7		INMUX_6		INMUX_5		INMUX_4		INMUX_3		INMUX_2		INMUX_1		INMUX_0	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-30 INMUX_15	INMUX_15 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
29-28 INMUX_14	INMUX_14 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
27-26 INMUX_13	INMUX_13 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
25-24 INMUX_12	INMUX_12 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
23-22 INMUX_11	INMUX_11 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
21-20 INMUX_10	INMUX_10 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
19-18 INMUX_9	INMUX_9 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
17-16 INMUX_8	INMUX_8 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
15-14 INMUX_7	INMUX_7 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
13-12 INMUX_6	INMUX_6 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0

Table continues on the next page...

Table continued from the previous page...

Field	Function
11-10 INMUX_5	INMUX_5 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
9-8 INMUX_4	INMUX_4 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
7-6 INMUX_3	INMUX_3 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
5-4 INMUX_2	INMUX_2 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
3-2 INMUX_1	INMUX_1 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
1-0 INMUX_0	INMUX_0 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0

9.7.10 Parameter_n Register (REG_B6)

Offset

Register	Offset
REG_B6	98h

Function

Controls access to pads 0 to 255 specific for IMCR registers. Two bits assigned per IMCR register have attributes of one of the implemented PDAC slots:

- 00-SIUL2_VIRTWRAPPER_PDAC1
- 01-SIUL2_VIRTWRAPPER_PDAC2
- 10-SIUL2_VIRTWRAPPER_PDAC4
- 11-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any master can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any master can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	INMUX_7		INMUX_6		INMUX_5		INMUX_4		INMUX_3		INMUX_2		INMUX_1		INMUX_0	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-30 —	Reserved
29-28 —	Reserved
27-26	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
25-24 —	Reserved
23-22 —	Reserved
21-20 —	Reserved
19-18 —	Reserved
17-16 —	Reserved
15-14 INMUX_7	INMUX_7 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
13-12 INMUX_6	INMUX_6 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
11-10 INMUX_5	INMUX_5 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
9-8 INMUX_4	INMUX_4 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-6 INMUX_3	INMUX_3 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
5-4 INMUX_2	INMUX_2 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
3-2 INMUX_1	INMUX_1 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
1-0 INMUX_0	INMUX_0 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0

9.7.11 Parameter_n Register (REG_B7)

Offset

Register	Offset
REG_B7	9Ch

Function

Controls access to pads 0 to 255 specific for IMCR registers. Two bits assigned per IMCR register have attributes of one of the implemented PDAC slots:

- 00-SIUL2_VIRTWRAPPER_PDAC1
- 01-SIUL2_VIRTWRAPPER_PDAC2
- 10-SIUL2_VIRTWRAPPER_PDAC4
- 11-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any master can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any master can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	INMUX_15		INMUX_14		INMUX_13		INMUX_12		INMUX_11		INMUX_10		INMUX_9		INMUX_8	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	INMUX_7		INMUX_6		INMUX_5		INMUX_4		INMUX_3		INMUX_2		INMUX_1		INMUX_0	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-30 INMUX_15	INMUX_15 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
29-28 INMUX_14	INMUX_14 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
27-26 INMUX_13	INMUX_13 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
25-24 INMUX_12	INMUX_12 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
23-22 INMUX_11	INMUX_11 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
21-20 INMUX_10	INMUX_10 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
19-18 INMUX_9	INMUX_9 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
17-16 INMUX_8	INMUX_8 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
15-14 INMUX_7	INMUX_7 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
13-12 INMUX_6	INMUX_6 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0

Table continues on the next page...

Table continued from the previous page...

Field	Function
11-10 INMUX_5	INMUX_5 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
9-8 INMUX_4	INMUX_4 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
7-6 INMUX_3	INMUX_3 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
5-4 INMUX_2	INMUX_2 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
3-2 INMUX_1	INMUX_1 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
1-0 INMUX_0	INMUX_0 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0

9.7.12 Parameter_n Register (REG_B8)

Offset

Register	Offset
REG_B8	A0h

Function

Controls access to pads 0 to 255 specific for IMCR registers. Two bits assigned per IMCR register have attributes of one of the implemented PDAC slots:

- 00-SIUL2_VIRTWRAPPER_PDAC1
- 01-SIUL2_VIRTWRAPPER_PDAC2
- 10-SIUL2_VIRTWRAPPER_PDAC4
- 11-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any master can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any master can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	INMUX_7		INMUX_6		INMUX_5		INMUX_4		INMUX_3		INMUX_2		INMUX_1		INMUX_0	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-30 —	Reserved
29-28 —	Reserved
27-26	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
25-24 —	Reserved
23-22 —	Reserved
21-20 —	Reserved
19-18 —	Reserved
17-16 —	Reserved
15-14 INMUX_7	INMUX_7 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
13-12 INMUX_6	INMUX_6 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
11-10 INMUX_5	INMUX_5 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
9-8 INMUX_4	INMUX_4 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-6 INMUX_3	INMUX_3 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
5-4 INMUX_2	INMUX_2 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
3-2 INMUX_1	INMUX_1 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
1-0 INMUX_0	INMUX_0 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0

9.7.13 Parameter_n Register (REG_B9)

Offset

Register	Offset
REG_B9	A4h

Function

Controls access to pads 0 to 255 specific for IMCR registers. Two bits assigned per IMCR register have attributes of one of the implemented PDAC slots:

- 00-SIUL2_VIRTWRAPPER_PDAC1
- 01-SIUL2_VIRTWRAPPER_PDAC2
- 10-SIUL2_VIRTWRAPPER_PDAC4
- 11-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any master can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any master can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	INMUX_15		INMUX_14		INMUX_13		INMUX_12		INMUX_11		INMUX_10		INMUX_9		INMUX_8	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		Reserved		INMUX_5		INMUX_4		INMUX_3		INMUX_2		INMUX_1		INMUX_0	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-30 INMUX_15	INMUX_15 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
29-28 INMUX_14	INMUX_14 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
27-26 INMUX_13	INMUX_13 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
25-24 INMUX_12	INMUX_12 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
23-22 INMUX_11	INMUX_11 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
21-20 INMUX_10	INMUX_10 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
19-18 INMUX_9	INMUX_9 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
17-16 INMUX_8	INMUX_8 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
15-14 —	Reserved
13-12 —	Reserved
11-10 INMUX_5	INMUX_5 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
9-8	INMUX_4

Table continues on the next page...

Table continued from the previous page...

Field	Function
INMUX_4	00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
7-6 INMUX_3	INMUX_3 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
5-4 INMUX_2	INMUX_2 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
3-2 INMUX_1	INMUX_1 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
1-0 INMUX_0	INMUX_0 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0

9.7.14 Parameter_n Register (REG_B10 - REG_B11)

Offset

Register	Offset
REG_B10	A8h
REG_B11	ACh

Function

Controls access to pads 0 to 255 specific for IMCR registers. Two bits assigned per IMCR register have attributes of one of the implemented PDAC slots:

- 00-SIUL2_VIRTWRAPPER_PDAC1
- 01-SIUL2_VIRTWRAPPER_PDAC2
- 10-SIUL2_VIRTWRAPPER_PDAC4
- 11-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any master can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any master can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	INMUX_15		INMUX_14		INMUX_13		INMUX_12		INMUX_11		INMUX_10		INMUX_9		INMUX_8	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	INMUX_7		INMUX_6		INMUX_5		INMUX_4		INMUX_3		INMUX_2		INMUX_1		INMUX_0	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-30 INMUX_15	INMUX_15 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
29-28 INMUX_14	INMUX_14 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
27-26	INMUX_13

Table continues on the next page...

Table continued from the previous page...

Field	Function
INMUX_13	00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
25-24 INMUX_12	INMUX_12 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
23-22 INMUX_11	INMUX_11 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
21-20 INMUX_10	INMUX_10 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
19-18 INMUX_9	INMUX_9 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
17-16 INMUX_8	INMUX_8 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
15-14 INMUX_7	INMUX_7 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
13-12 INMUX_6	INMUX_6 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
11-10 INMUX_5	INMUX_5 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
9-8 INMUX_4	INMUX_4 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
7-6 INMUX_3	INMUX_3 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
5-4 INMUX_2	INMUX_2 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
3-2 INMUX_1	INMUX_1 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0

Table continues on the next page...

Table continued from the previous page...

Field	Function
1-0 INMUX_0	INMUX_0 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0

9.7.15 Parameter_n Register (REG_B12)

Offset

Register	Offset
REG_B12	B0h

Function

Controls access to pads 0 to 255 specific for IMCR registers. Two bits assigned per IMCR register have attributes of one of the implemented PDAC slots:

- 00-SIUL2_VIRTWRAPPER_PDAC1
- 01-SIUL2_VIRTWRAPPER_PDAC2
- 10-SIUL2_VIRTWRAPPER_PDAC4
- 11-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any master can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any master can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved		Reserved		Reserved		Reserved		Reserved		INMUX_10		INMUX_9		INMUX_8	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	INMUX_7		INMUX_6		INMUX_5		INMUX_4		INMUX_3		INMUX_2		INMUX_1		INMUX_0	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-30 —	Reserved
29-28 —	Reserved
27-26 —	Reserved
25-24 —	Reserved
23-22 —	Reserved
21-20 INMUX_10	INMUX_10 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
19-18 INMUX_9	INMUX_9 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
17-16 INMUX_8	INMUX_8 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
15-14 INMUX_7	INMUX_7 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
13-12	INMUX_6

Table continues on the next page...

Table continued from the previous page...

Field	Function
INMUX_6	00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
11-10 INMUX_5	INMUX_5 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
9-8 INMUX_4	INMUX_4 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
7-6 INMUX_3	INMUX_3 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
5-4 INMUX_2	INMUX_2 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
3-2 INMUX_1	INMUX_1 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
1-0 INMUX_0	INMUX_0 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0

9.7.16 Parameter_n Register (REG_B13)

Offset

Register	Offset
REG_B13	B4h

Function

Controls access to pads 0 to 255 specific for IMCR registers. Two bits assigned per IMCR register have attributes of one of the implemented PDAC slots:

- 00-SIUL2_VIRTWRAPPER_PDAC1
- 01-SIUL2_VIRTWRAPPER_PDAC2
- 10-SIUL2_VIRTWRAPPER_PDAC4
- 11-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any master can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any master can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	INMUX_15		INMUX_14		INMUX_13		INMUX_12		INMUX_11		INMUX_10		INMUX_9		INMUX_8	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	INMUX_7		INMUX_6		INMUX_5		INMUX_4		INMUX_3		Reserved		Reserved		Reserved	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-30 INMUX_15	INMUX_15 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
29-28 INMUX_14	INMUX_14 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
27-26 INMUX_13	INMUX_13 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
25-24 INMUX_12	INMUX_12 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
23-22 INMUX_11	INMUX_11 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
21-20 INMUX_10	INMUX_10 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
19-18 INMUX_9	INMUX_9 00b - SIUL2_VIRTWRAPPER_PDAC1

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
17-16 INMUX_8	INMUX_8 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
15-14 INMUX_7	INMUX_7 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
13-12 INMUX_6	INMUX_6 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
11-10 INMUX_5	INMUX_5 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
9-8 INMUX_4	INMUX_4 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
7-6 INMUX_3	INMUX_3 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4

Table continues on the next page...

Table continued from the previous page...

Field	Function
	11b - SIUL2_VIRTWRAPPER_PDAC0
5-4 —	Reserved
3-2 —	Reserved
1-0 —	Reserved

9.7.17 Parameter_n Register (REG_B14 - REG_B15)

Offset

Register	Offset
REG_B14	B8h
REG_B15	BCh

Function

Controls access to pads 0 to 255 specific for IMCR registers. Two bits assigned per IMCR register have attributes of one of the implemented PDAC slots:

- 00-SIUL2_VIRTWRAPPER_PDAC1
- 01-SIUL2_VIRTWRAPPER_PDAC2
- 10-SIUL2_VIRTWRAPPER_PDAC4
- 11-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any master can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any master can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	INMUX_15		INMUX_14		INMUX_13		INMUX_12		INMUX_11		INMUX_10		INMUX_9		INMUX_8	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	INMUX_7		INMUX_6		INMUX_5		INMUX_4		INMUX_3		INMUX_2		INMUX_1		INMUX_0	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-30 INMUX_15	INMUX_15 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
29-28 INMUX_14	INMUX_14 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
27-26 INMUX_13	INMUX_13 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
25-24 INMUX_12	INMUX_12 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
23-22 INMUX_11	INMUX_11 00b - SIUL2_VIRTWRAPPER_PDAC1

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
21-20 INMUX_10	INMUX_10 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
19-18 INMUX_9	INMUX_9 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
17-16 INMUX_8	INMUX_8 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
15-14 INMUX_7	INMUX_7 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
13-12 INMUX_6	INMUX_6 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
11-10 INMUX_5	INMUX_5 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4

Table continues on the next page...

Table continued from the previous page...

Field	Function
	11b - SIUL2_VIRTWRAPPER_PDAC0
9-8 INMUX_4	INMUX_4 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
7-6 INMUX_3	INMUX_3 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
5-4 INMUX_2	INMUX_2 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
3-2 INMUX_1	INMUX_1 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
1-0 INMUX_0	INMUX_0 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0

9.7.18 Parameter_n Register (REG_B16)

Offset

Register	Offset
REG_B16	C0h

Function

Controls access to pads 0 to 255 specific for IMCR registers. Two bits assigned per IMCR register have attributes of one of the implemented PDAC slots:

- 00-SIUL2_VIRTWRAPPER_PDAC1
- 01-SIUL2_VIRTWRAPPER_PDAC2
- 10-SIUL2_VIRTWRAPPER_PDAC4
- 11-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any master can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any master can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved		Reserved		Reserved		INMUX_12		INMUX_11		INMUX_10		INMUX_9		INMUX_8	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	INMUX_7		INMUX_6		INMUX_5		INMUX_4		INMUX_3		INMUX_2		INMUX_1		INMUX_0	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-30 —	Reserved
29-28 —	Reserved
27-26 —	Reserved
25-24 INMUX_12	INMUX_12 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0

Table continues on the next page...

Table continued from the previous page...

Field	Function
23-22 INMUX_11	INMUX_11 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
21-20 INMUX_10	INMUX_10 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
19-18 INMUX_9	INMUX_9 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
17-16 INMUX_8	INMUX_8 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
15-14 INMUX_7	INMUX_7 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
13-12 INMUX_6	INMUX_6 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
11-10 INMUX_5	INMUX_5 00b - SIUL2_VIRTWRAPPER_PDAC1

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
9-8 INMUX_4	INMUX_4 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
7-6 INMUX_3	INMUX_3 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
5-4 INMUX_2	INMUX_2 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
3-2 INMUX_1	INMUX_1 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
1-0 INMUX_0	INMUX_0 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0

9.7.19 Parameter_n Register (REG_B18)

Offset

Register	Offset
REG_B18	C8h

Function

Controls access to pads 0 to 255 specific for IMCR registers. Two bits assigned per IMCR register have attributes of one of the implemented PDAC slots:

- 00-SIUL2_VIRTWRAPPER_PDAC1
- 01-SIUL2_VIRTWRAPPER_PDAC2
- 10-SIUL2_VIRTWRAPPER_PDAC4
- 11-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any master can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any master can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	INMUX_15		INMUX_14		INMUX_13		INMUX_12		INMUX_11		INMUX_10		INMUX_9		INMUX_8	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	INMUX_7		INMUX_6		INMUX_5		INMUX_4		INMUX_3		INMUX_2		INMUX_1		Reserved	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-30 INMUX_15	INMUX_15 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0

Table continues on the next page...

Table continued from the previous page...

Field	Function
29-28 INMUX_14	INMUX_14 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
27-26 INMUX_13	INMUX_13 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
25-24 INMUX_12	INMUX_12 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
23-22 INMUX_11	INMUX_11 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
21-20 INMUX_10	INMUX_10 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
19-18 INMUX_9	INMUX_9 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
17-16 INMUX_8	INMUX_8 00b - SIUL2_VIRTWRAPPER_PDAC1

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
15-14 INMUX_7	INMUX_7 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
13-12 INMUX_6	INMUX_6 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
11-10 INMUX_5	INMUX_5 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
9-8 INMUX_4	INMUX_4 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
7-6 INMUX_3	INMUX_3 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
5-4 INMUX_2	INMUX_2 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4

Table continues on the next page...

Table continued from the previous page...

Field	Function
	11b - SIUL2_VIRTWRAPPER_PDAC0
3-2 INMUX_1	INMUX_1 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
1-0 —	Reserved

9.7.20 Parameter_n Register (REG_B19)

Offset

Register	Offset
REG_B19	CCh

Function

Controls access to pads 0 to 255 specific for IMCR registers. Two bits assigned per IMCR register have attributes of one of the implemented PDAC slots:

- 00-SIUL2_VIRTWRAPPER_PDAC1
- 01-SIUL2_VIRTWRAPPER_PDAC2
- 10-SIUL2_VIRTWRAPPER_PDAC4
- 11-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any master can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any master can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	INMUX_15		INMUX_14		INMUX_13		INMUX_12		INMUX_11		Reserved		Reserved		Reserved	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		Reserved		INMUX_5		INMUX_4		INMUX_3		INMUX_2		INMUX_1		INMUX_0	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-30 INMUX_15	INMUX_15 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
29-28 INMUX_14	INMUX_14 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
27-26 INMUX_13	INMUX_13 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
25-24 INMUX_12	INMUX_12 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
23-22 INMUX_11	INMUX_11 00b - SIUL2_VIRTWRAPPER_PDAC1

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
21-20 —	Reserved
19-18 —	Reserved
17-16 —	Reserved
15-14 —	Reserved
13-12 —	Reserved
11-10 INMUX_5	INMUX_5 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
9-8 INMUX_4	INMUX_4 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
7-6 INMUX_3	INMUX_3 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
5-4 INMUX_2	INMUX_2 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
3-2 INMUX_1	INMUX_1 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
1-0 INMUX_0	INMUX_0 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0

9.7.21 Parameter_n Register (REG_B20)

Offset

Register	Offset
REG_B20	D0h

Function

Controls access to pads 0 to 255 specific for IMCR registers. Two bits assigned per IMCR register have attributes of one of the implemented PDAC slots:

- 00-SIUL2_VIRTWRAPPER_PDAC1
- 01-SIUL2_VIRTWRAPPER_PDAC2
- 10-SIUL2_VIRTWRAPPER_PDAC4
- 11-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any master can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any master can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		Reserved		INMUX_5		INMUX_4		INMUX_3		INMUX_2		INMUX_1		INMUX_0	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-30 —	Reserved
29-28 —	Reserved
27-26 —	Reserved
25-24 —	Reserved
23-22 —	Reserved
21-20 —	Reserved
19-18 —	Reserved
17-16 —	Reserved
15-14 —	Reserved
13-12 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
11-10 INMUX_5	INMUX_5 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
9-8 INMUX_4	INMUX_4 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
7-6 INMUX_3	INMUX_3 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
5-4 INMUX_2	INMUX_2 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
3-2 INMUX_1	INMUX_1 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
1-0 INMUX_0	INMUX_0 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0

9.7.22 Parameter_n Register (REG_B21)

Offset

Register	Offset
REG_B21	D4h

Function

Controls access to pads 0 to 255 specific for IMCR registers. Two bits assigned per IMCR register have attributes of one of the implemented PDAC slots:

- 00-SIUL2_VIRTWRAPPER_PDAC1
- 01-SIUL2_VIRTWRAPPER_PDAC2
- 10-SIUL2_VIRTWRAPPER_PDAC4
- 11-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any master can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any master can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	INMUX_15		INMUX_14		INMUX_13		INMUX_12		INMUX_11		INMUX_10		INMUX_9		INMUX_8	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	INMUX_7		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-30 INMUX_15	INMUX_15 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0

Table continues on the next page...

Table continued from the previous page...

Field	Function
29-28 INMUX_14	INMUX_14 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
27-26 INMUX_13	INMUX_13 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
25-24 INMUX_12	INMUX_12 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
23-22 INMUX_11	INMUX_11 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
21-20 INMUX_10	INMUX_10 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
19-18 INMUX_9	INMUX_9 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
17-16 INMUX_8	INMUX_8 00b - SIUL2_VIRTWRAPPER_PDAC1

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
15-14 INMUX_7	INMUX_7 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
13-12 —	Reserved
11-10 —	Reserved
9-8 —	Reserved
7-6 —	Reserved
5-4 —	Reserved
3-2 —	Reserved
1-0 —	Reserved

9.7.23 Parameter_n Register (REG_B22)

Offset

Register	Offset
REG_B22	D8h

Function

Controls access to pads 0 to 255 specific for IMCR registers. Two bits assigned per IMCR register have attributes of one of the implemented PDAC slots:

- 00-SIUL2_VIRTWRAPPER_PDAC1
- 01-SIUL2_VIRTWRAPPER_PDAC2
- 10-SIUL2_VIRTWRAPPER_PDAC4
- 11-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any master can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any master can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	INMUX_15		INMUX_14		INMUX_13		INMUX_12		INMUX_11		INMUX_10		INMUX_9		INMUX_8	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	INMUX_7		INMUX_6		INMUX_5		INMUX_4		INMUX_3		INMUX_2		INMUX_1		INMUX_0	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-30 INMUX_15	INMUX_15 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
29-28 INMUX_14	INMUX_14 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
27-26 INMUX_13	INMUX_13 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4

Table continues on the next page...

Table continued from the previous page...

Field	Function
	11b - SIUL2_VIRTWRAPPER_PDAC0
25-24 INMUX_12	INMUX_12 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
23-22 INMUX_11	INMUX_11 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
21-20 INMUX_10	INMUX_10 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
19-18 INMUX_9	INMUX_9 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
17-16 INMUX_8	INMUX_8 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
15-14 INMUX_7	INMUX_7 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
13-12	INMUX_6

Table continues on the next page...

Table continued from the previous page...

Field	Function
INMUX_6	00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
11-10 INMUX_5	INMUX_5 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
9-8 INMUX_4	INMUX_4 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
7-6 INMUX_3	INMUX_3 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
5-4 INMUX_2	INMUX_2 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
3-2 INMUX_1	INMUX_1 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
1-0 INMUX_0	INMUX_0 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0

9.7.24 Parameter_n Register (REG_B23)

Offset

Register	Offset
REG_B23	DCh

Function

Controls access to pads 0 to 255 specific for IMCR registers. Two bits assigned per IMCR register have attributes of one of the implemented PDAC slots:

- 00-SIUL2_VIRTWRAPPER_PDAC1
- 01-SIUL2_VIRTWRAPPER_PDAC2
- 10-SIUL2_VIRTWRAPPER_PDAC4
- 11-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any master can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any master can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved		Reserved		Reserved		Reserved		Reserved		INMUX_10		INMUX_9		INMUX_8	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	INMUX_7		INMUX_6		INMUX_5		INMUX_4		INMUX_3		INMUX_2		INMUX_1		INMUX_0	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-30 —	Reserved
29-28 —	Reserved
27-26 —	Reserved
25-24 —	Reserved
23-22 —	Reserved
21-20 INMUX_10	INMUX_10 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
19-18 INMUX_9	INMUX_9 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
17-16 INMUX_8	INMUX_8 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
15-14 INMUX_7	INMUX_7 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
13-12	INMUX_6

Table continues on the next page...

Table continued from the previous page...

Field	Function
INMUX_6	00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
11-10 INMUX_5	INMUX_5 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
9-8 INMUX_4	INMUX_4 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
7-6 INMUX_3	INMUX_3 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
5-4 INMUX_2	INMUX_2 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
3-2 INMUX_1	INMUX_1 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
1-0 INMUX_0	INMUX_0 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0

9.7.25 Parameter_n Register (REG_B24)

Offset

Register	Offset
REG_B24	E0h

Function

Controls access to pads 0 to 255 specific for IMCR registers. Two bits assigned per IMCR register have attributes of one of the implemented PDAC slots:

- 00-SIUL2_VIRTWRAPPER_PDAC1
- 01-SIUL2_VIRTWRAPPER_PDAC2
- 10-SIUL2_VIRTWRAPPER_PDAC4
- 11-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any master can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any master can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	INMUX_15		INMUX_14		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		Reserved		INMUX_5		Reserved		Reserved		Reserved		Reserved		Reserved	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-30 INMUX_15	INMUX_15 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
29-28 INMUX_14	INMUX_14 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
27-26 —	Reserved
25-24 —	Reserved
23-22 —	Reserved
21-20 —	Reserved
19-18 —	Reserved
17-16 —	Reserved
15-14 —	Reserved
13-12 —	Reserved
11-10 INMUX_5	INMUX_5 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0

Table continues on the next page...

Table continued from the previous page...

Field	Function
9-8 —	Reserved
7-6 —	Reserved
5-4 —	Reserved
3-2 —	Reserved
1-0 —	Reserved

9.7.26 Parameter_n Register (REG_B25)

Offset

Register	Offset
REG_B25	E4h

Function

Controls access to pads 0 to 255 specific for IMCR registers. Two bits assigned per IMCR register have attributes of one of the implemented PDAC slots:

- 00-SIUL2_VIRTWRAPPER_PDAC1
- 01-SIUL2_VIRTWRAPPER_PDAC2
- 10-SIUL2_VIRTWRAPPER_PDAC4
- 11-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any master can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any master can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	INMUX_15		INMUX_14		INMUX_13		INMUX_12		INMUX_11		INMUX_10		INMUX_9		Reserved	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-30 INMUX_15	INMUX_15 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
29-28 INMUX_14	INMUX_14 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
27-26 INMUX_13	INMUX_13 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
25-24 INMUX_12	INMUX_12 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
23-22 INMUX_11	INMUX_11 00b - SIUL2_VIRTWRAPPER_PDAC1

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
21-20 INMUX_10	INMUX_10 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
19-18 INMUX_9	INMUX_9 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
17-16 —	Reserved
15-14 —	Reserved
13-12 —	Reserved
11-10 —	Reserved
9-8 —	Reserved
7-6 —	Reserved
5-4 —	Reserved
3-2 —	Reserved
1-0	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	

9.7.27 Parameter_n Register (REG_B26)

Offset

Register	Offset
REG_B26	E8h

Function

Controls access to pads 0 to 255 specific for IMCR registers. Two bits assigned per IMCR register have attributes of one of the implemented PDAC slots:

- 00-SIUL2_VIRTWRAPPER_PDAC1
- 01-SIUL2_VIRTWRAPPER_PDAC2
- 10-SIUL2_VIRTWRAPPER_PDAC4
- 11-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any master can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any master can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	INMUX_15		INMUX_14		INMUX_13		INMUX_12		INMUX_11		INMUX_10		INMUX_9		INMUX_8	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	INMUX_7		INMUX_6		INMUX_5		INMUX_4		INMUX_3		INMUX_2		INMUX_1		INMUX_0	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-30	INMUX_15

Table continues on the next page...

Table continued from the previous page...

Field	Function
INMUX_15	00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
29-28 INMUX_14	INMUX_14 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
27-26 INMUX_13	INMUX_13 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
25-24 INMUX_12	INMUX_12 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
23-22 INMUX_11	INMUX_11 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
21-20 INMUX_10	INMUX_10 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
19-18 INMUX_9	INMUX_9 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
17-16 INMUX_8	INMUX_8 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
15-14 INMUX_7	INMUX_7 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
13-12 INMUX_6	INMUX_6 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
11-10 INMUX_5	INMUX_5 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
9-8 INMUX_4	INMUX_4 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
7-6 INMUX_3	INMUX_3 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0

Table continues on the next page...

Table continued from the previous page...

Field	Function
5-4 INMUX_2	INMUX_2 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
3-2 INMUX_1	INMUX_1 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
1-0 INMUX_0	INMUX_0 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0

9.7.28 Parameter_n Register (REG_B27)

Offset

Register	Offset
REG_B27	ECh

Function

Controls access to pads 0 to 255 specific for IMCR registers. Two bits assigned per IMCR register have attributes of one of the implemented PDAC slots:

- 00-SIUL2_VIRTWRAPPER_PDAC1
- 01-SIUL2_VIRTWRAPPER_PDAC2
- 10-SIUL2_VIRTWRAPPER_PDAC4
- 11-SIUL2_VIRTWRAPPER_PDAC0

NOTE

- Any master can access holes (unimplemented registers) of SIUL2 through PDAC0.
- After reset, when XRDC is not configured, any master can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		INMUX_8	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		Reserved		INMUX_5		Reserved		Reserved		INMUX_2		INMUX_1		INMUX_0	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-30 —	Reserved
29-28 —	Reserved
27-26 —	Reserved
25-24 —	Reserved
23-22 —	Reserved
21-20 —	Reserved
19-18 —	Reserved
17-16 INMUX_8	INMUX_8 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
15-14 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
13-12 —	Reserved
11-10 INMUX_5	INMUX_5 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
9-8 —	Reserved
7-6 —	Reserved
5-4 INMUX_2	INMUX_2 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
3-2 INMUX_1	INMUX_1 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0
1-0 INMUX_0	INMUX_0 00b - SIUL2_VIRTWRAPPER_PDAC1 01b - SIUL2_VIRTWRAPPER_PDAC2 10b - SIUL2_VIRTWRAPPER_PDAC4 11b - SIUL2_VIRTWRAPPER_PDAC0

9.7.29 Parameter_n Register (REG_C1039_1024)

Offset

Register	Offset
REG_C1039_1024	100h

Function

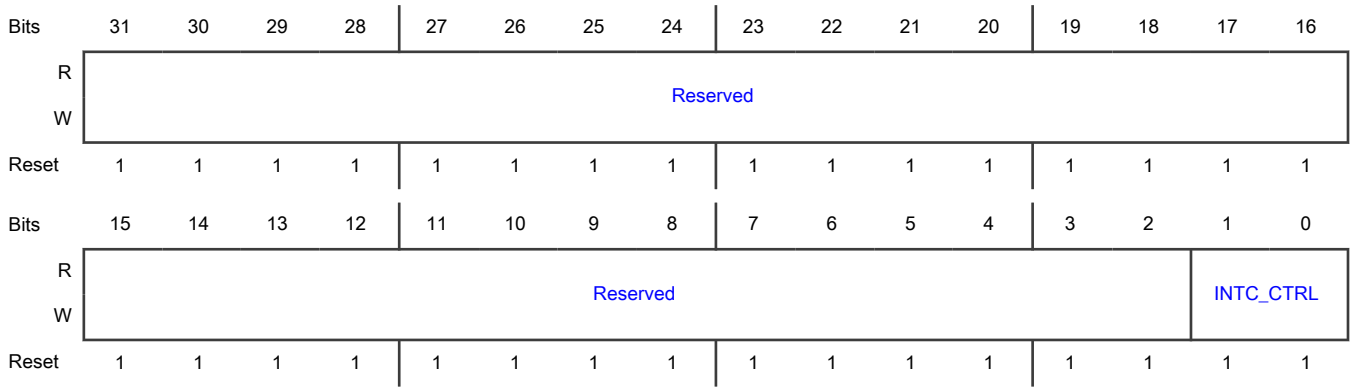
Controls access to DISR0, DIRER0, DIRSR0, IREER0, IFEER0, IFER0, IFMCR, and IFCPR0 interrupt registers. Eight bits assigned per PDAC slot have attributes of one of the implemented PDAC slots:

- 00-SIUL2_VIRTWRAPPER_PDAC1
- 01-SIUL2_VIRTWRAPPER_PDAC2
- 10-SIUL2_VIRTWRAPPER_PDAC4
- 11-SIUL2_VIRTWRAPPER_PDAC0

NOTE

After reset, when XRDC is not configured, any master can access SIUL2 registers through PDAC0 without getting transfer error. However, if SIUL2 registers are accessed through other PDAC slots, a transfer error will be generated.

Diagram



Fields

Field	Function
31-2 —	Reserved
1-0 INTC_CTRL	Interrupt register control This bit controls the interrupt register.

9.8 Glossary

VM Virtualized module, such as SIUL2.

PDAC Peripheral domain access control. PDAC and PDAC slot are interchangeable terms.

Chapter 10
System Integration Unit Lite2 (SIUL2)

10.1 Chip-specific SIUL2 information

10.1.1 Feature availability

This chip:

- Supports input filter enable (MSCR5[IFE]) only for the reset pad (PTA5). For details, see the "Pad signal description" table in the "Signal Multiplexing" chapter.
- Does not implement the open-drain feature. LPI2C directly configures only the pads related to the I2C and LPUART functions in pseudo open-drain when these functions are muxed to the pads. See the LPI2C and LPUART chapters for more information.
- Reserves GPDO[25:24] and PGPDO1[7:6] because PTA24 and PTA25 are input-only pins.

NOTE

- EIRQ[0-15] can be used either for interrupt or DMA request. EIRQ[16-31] can only be used for interrupt request.
- When a pad is used with IBE=1, the PAD must be actively driven. Otherwise, IO states are not deterministic.

10.1.2 Mapping of MSCR and IMCR instances

- CR numbers 0-511 correspond to the MSCR instances.
- CR numbers 512-1023 correspond to the IMCR instances. IMCRs defined in the attached IOMUX file have an offset of 512 with respect to the IMCR number defined in SIUL2 memory map section.

NOTE

IMCR register only supports 32 bit access, any other access might result in unexpected data. See IOMUX file attached to this document for the reset value and exact number of IMCR and MSCR register instances.

NOTE

SIUL2 registers description in the "memory map" section is generic. The valid fields for MSCR0- MSCR323 must be checked from the IOMUX file attached to this document.

10.2 Overview

SIUL2 provides control over all electrical pin controls and ports with 16 bits of bidirectional, general-purpose input and output (I/O) signals. It also performs the following functions:

- Enables you to select the functions and electrical characteristics that appear on external chip pins
- Controls the multiplexing of internal signals from one module to another and controls the chip I/O
- Supports a maximum of 32 external interrupts with trigger event configuration

10.2.1 Block diagram

SIUL2 provides a dedicated pad control to general-purpose pads that you can configure as either inputs or outputs. It provides registers for you to read values from GPIO pads configured as inputs and to write values to GPIO pads configured as outputs. Based on the configuration of GPIO, you can:

- Write to an internal register to control the state driven on the associated output pad (when configured as output).
- Detect the state of the associated pad by reading the value from an internal register (when configured as input).

- Read back the pad value to check whether the written value appears on the pad (when configured as input and output).

Access the GPIO data registers in the following ways to allow port access and bit manipulation without read-modify-write operations:

- Access to two 16-bit ports in one access
- Read/write access to a single bit
- 16-bit port write with a bit mask using a single 32-bit access

You can configure:

- External interrupt sources at the chip level to be used with any chip pad.
- Interrupt sources to have a digital filter to reject short glitches on the inputs.

The external interrupt or DMA requests map to the interrupt request pins (REQ) in the chip packages. The user-defined registers (UDRs) contain miscellaneous control and status bits.

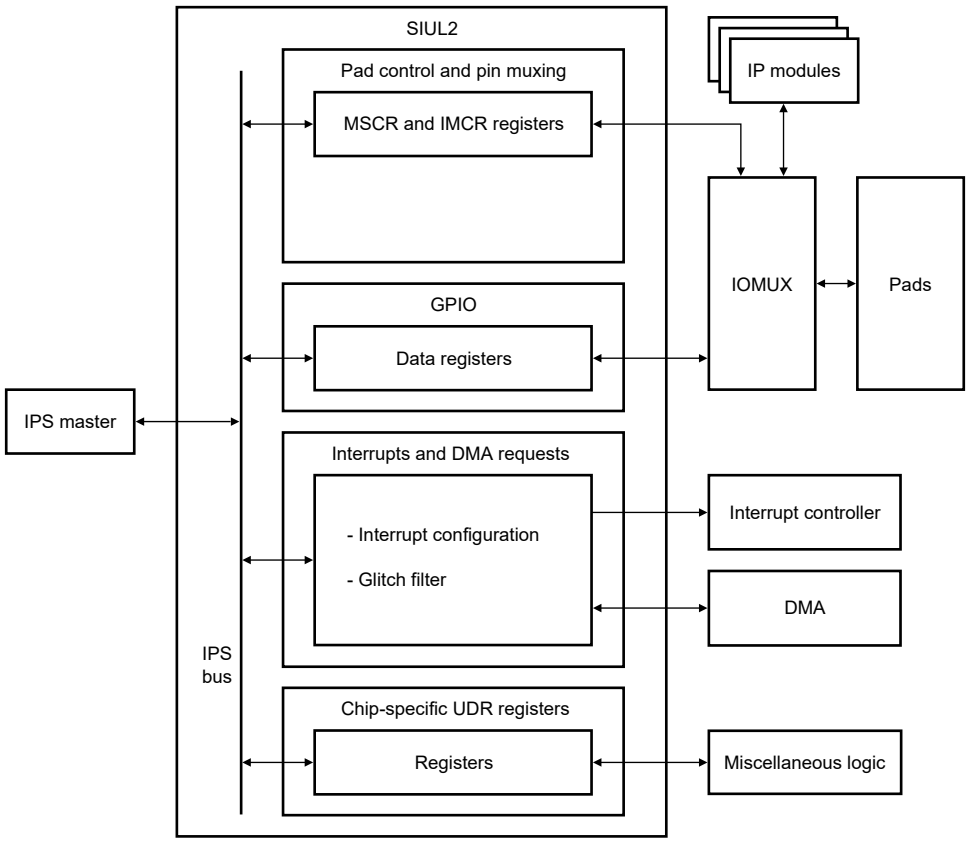


Figure 23. Block diagram

10.2.2 Features

SIUL2 supports the following:

- 1 to 32 GPIO ports with data control:
  - Drives data to as many as 16 independent I/O channels
  - Samples data from as many as 16 independent I/O channels
- Read or write of two 16-bit registers with one access for a 32-bit port

- External interrupt and DMA requests:
  - 1 to 32 programmable digital glitch filters, one for each interrupt REQ pin
  - Edge detection
- Multiplexed Signal Configuration Registers (MSCR) to configure the electrical parameters and settings for as many as 512 functional pads.

See the interrupt map file and the DMAMUX map file attached to this document for mapping of interrupt and DMA sources to interrupt vectors and DMA channels.

10.3 Functional description

10.3.1 Pad control

SIUL2 controls the electrical characteristics of around 512 pads. It provides a consistent interface for all pads, both on a by-port and a by-bit basis.

The setting of each pad out of reset is fixed per chip but you can configure it individually. This way you can select special pull settings or peripheral pad ownership.

You can configure each pad independently of all other pads on the chip or other pads grouped within a single port, therefore allowing grouping of different pad types together in ports and operating the pads individually. Grouping the various functions for each pad into a single register allows configuration of each pad with a single write to a register, which further allows you to duplicate software for similar pads with index changes.

10.3.2 General-purpose input and output pads (GPIO)

SIUL2 allows each pad to be configured as either of the following:

- General-purpose input or output pad (GPIO)
- A pad for one or more alternate functions (input or output) determined by the peripheral that uses the pad

You can also implement the GPIO pads without any alternate function.

SIUL2 manages as many as 512 GPIO pads organized as ports that can be accessed for data reads and writes as 8-bit, 16-bit, or 32-bit.

All port accesses are identical, with each read or write performed only at a different location to access a different port width.

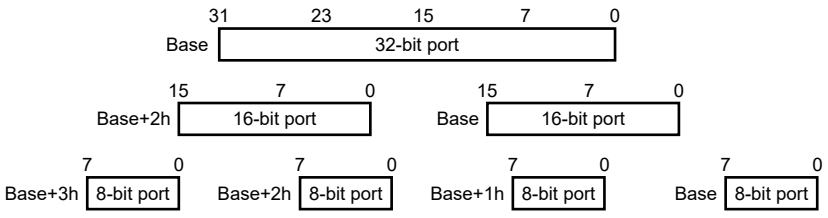


Figure 24. Data port example arrangement showing configuration for different port width accesses

SIUL2 has separate data input and data output registers for all pads. You can therefore directly read back an input or output value of a pad to validate what is present on the pad instead of confirming the value that was written to the data input registers. The data output registers support both read and write operations whereas the data input registers support only the read access.

When you configure a pad for using one of its alternate functions, the data input values reflect the respective value of the pad. If you perform a write operation to the data output register for a pad configured as an alternate function (non-GPIO), this write is not reflected by the pad value until reconfigured to GPIO. All general-purpose pads are implemented as bidirectional.

If a bidirectional operation impacts the performance or is not needed for a pad function, you can limit the functionality of the pad to input-only.

10.3.3 Clocking

This module has no clocking considerations.

10.3.4 External interrupts

SIUL2 supports 1–32 external interrupts allocated to pads by the chip.

See the interrupt map file and the DMAMUX map file attached to this document (device reference manual) for mapping of interrupt and DMA sources to interrupt vectors and DMA channels.

SIUL2 supports 1–4 interrupt vectors to the interrupt controller of the chip. Each interrupt vector can support as many as eight external interrupt sources from the chip pads.

All the external interrupt pads within a single group have equal priority. It is your responsibility to search for the application through a group of sources in an appropriate way.

NOTE

Glitch filters applied to external interrupts require a running internal oscillator clock. If such a clock is not available, enabling the glitch filter on an external interrupt disables the interrupt.

The external interrupt signals from a pad have internal synchronizers. Therefore, the width of the interrupt signals should be at least 2.5 or 3 times the internal RC oscillator (IRC) clock cycles to correctly capture the interrupts.

10.3.4.1 External interrupt initialization

Perform the following procedure to enable external interrupts (if you do not perform these steps, you may get a false interrupt flag during interrupt initialization):

1. Write 1 to the appropriate IFER[IFE*n*] fields to enable the glitch filter.
2. Write 0 to DIRER0[EIRE*n*] to mask interrupts.
3. Write 1 to the appropriate IREE*n* fields in Interrupt Rising-Edge Event Enable 0 (IREER0) and IFEE*n* fields in Interrupt Falling-Edge Event Enable 0 (IFEER0) as needed to select the pin polarity.
4. Configure the appropriate fields in MSCR for the external interrupt pins:
   a. Write 0 to the OBE and ODE to disable the output.
   b. Write 1 to IBE field to enable the input buffer of the pin.
   c. If you are using the internal pull-up or pull-down resistors, configure the appropriate PUE and PUS fields.

   NOTE

   If you select external interrupt inputs for external interrupt pins, do not configure them as outputs (that is, MSCR[OBE] must not be 0) because it can cause false interrupt detection (such as from a GPIO configuration).
5. Write to the appropriate DIRSR0[DIRS*n*] fields to select a request between DMA or interrupt.
6. Select the desired glitch filter setup for the pins:
   a. Write an appropriate value to IFMCR0[MAXCNT] for the respective external interrupt to the filter counter.
   b. Write an appropriate value to IFCPR[IFCP] to set the filter clock prescaler.
   c. Write to the appropriate IFER0[IFE*n*] fields to enable the glitch filter for the external interrupt pins.
7. Set the appropriate DISR0[EIF*n*] to clear any flags.
8. Write to the appropriate DIRER0[EIRE*n*] fields to enable the interrupt pins.

NOTE

If you do not follow these steps you may get a false interrupt flag during interrupt initialization.

10.3.4.2 External interrupt management

You can enable or disable each interrupt independently using a single rolled up register, DIRER0.

You can configure a pad defined as an external interrupt to recognize interrupts with an active rising edge, an active falling edge, or both, using Interrupt Rising-Edge Event Enable 0 (IREER0) and Interrupt Falling-Edge Event Enable 0 (IFEER0).

NOTE

You cannot disable both edge events of a given interrupt.

Each external interrupt has an individual flag, held in DMA or Interrupt Status Flag 0 (DISR0). DISR0 is a clear-by-write-1 register, which prevents inadvertent overwriting of other flags in the same register.

Figure 25 provides an overview of the external interrupt implementation.

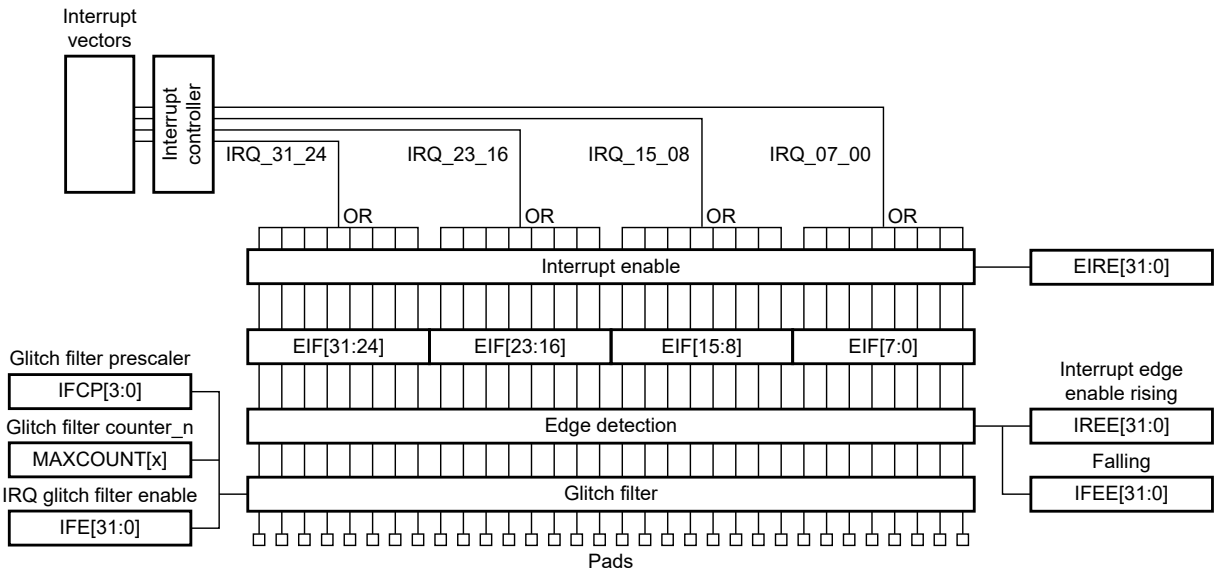


Figure 25. External interrupt pad diagram

10.3.4.3 External interrupt request

The REQ input pins on the chip are the sources for interrupt or DMA requests. The chip provides one possible interrupt vector for SIUL2. The 32 interrupt request sources map to vectors and channels as shown in Table 44.

Table 44. Interrupt source mapping to SIUL2 interrupt request output for 32 interrupt sources

Vector or channel #	Interrupt vector source
0	REQ[07] | REQ[06] | REQ[05] | REQ[04] | REQ[03] | REQ[02] | REQ[01] | REQ[00]
1	REQ[15] | REQ[14] | REQ[13] | REQ[12] | REQ[11] | REQ[10] | REQ[09] | REQ[08]
2	REQ[23] | REQ[22] | REQ[21] | REQ[20] | REQ[19] | REQ[18] | REQ[17] | REQ[16]
3	REQ[31] | REQ[30] | REQ[29] | REQ[28] | REQ[27] | REQ[26] | REQ[25] | REQ[24]

10.3.5 DMA requests

The REQ pins on the chip map to the independent DMA request channels in the DMA controller. See the DMAMUX map file attached to this document for the mapping of DMA sources to DMA channels.

DISR0 and DIRSR0 manage DMA requests in the following way:

- Service a DMA request to clear the DISR0 register flags.
- Write to the appropriate DIRSR0[DIRS*n*] to select a request between DMA or interrupt.
- If you select DMA in DIRSR0 and set the corresponding DISR0 flag for it, SIUL2 sends a DMA request signal as an output.

10.4 External signal description

See the IOMUX file attached to this document for more information.

10.5 Initialization

This module does not require initialization.

10.6 SIUL2 register descriptions

This section describes the SIUL2 registers.

- Undocumented register spaces in the SIUL2 memory map, including addresses shown as blanks, are reserved:
  - Reserved registers or spaces are read as 0.
  - Writes to reserved registers or spaces generate a transfer error.
- Writes to read-only registers generate a transfer error.

NOTE

- For the array of 8-bit registers, GPDO*n* and GPDI*n*:
  - An 8-bit access to an unimplemented address (a "hole") within the array region generates a transfer error.
  - However, when you perform a 16-bit or 32-bit access and if any register instance is implemented within the accessed range, a transfer error is not generated even if the range includes a hole.
- For the array of 16-bit registers, PGPDO*n* and PGPDI*n*:
  - A 16-bit access to an unimplemented address (a "hole") within the array region generates a transfer error.
  - However, a 32-bit access does not generate a transfer error for a hole irrespective of whether the other 16-bit range includes a register instance.

10.6.1 SIUL2 memory map

SIUL2 base address: 4029_0000h

Offset	Register	Width (In bits)	Access	Reset value
4h	SIUL2 MCU ID Register #1 (MIDR1)	32	R	See section
8h	SIUL2 MCU ID Register #2 (MIDR2)	32	R	See section
10h	DMA or Interrupt Status Flag 0 (DISR0)	32	RW	0000_0000h
18h	DMA or Interrupt Request Enable 0 (DIRER0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
20h	DMA or Interrupt Request Select 0 (DIRSR0)	32	RW	0000_0000h
28h	Interrupt Rising-Edge Event Enable 0 (IREER0)	32	RW	0000_0000h
30h	Interrupt Falling-Edge Event Enable 0 (IFEER0)	32	RW	0000_0000h
38h	Interrupt Filter Enable 0 (IFER0)	32	RW	0000_0000h
40h - BCh	Interrupt Filter Maximum Counter (IFMCR0 - IFMCR31)	32	RW	0000_0000h
C0h	Interrupt Filter Clock Prescaler (IFCPR)	32	RW	0000_0000h
100h	MUX0 EMIOS ENABLE 1 (MUX0_EMIOS_EN1)	32	RW	See section
104h	MUX0 MISC ENABLE (MUX0_MISC_EN)	32	RW	See section
108h	MUX1 EMIOS ENABLE (MUX1_EMIOS_EN)	32	RW	See section
10Ch	MUX1 MISC ENABLE (MUX1_MISC_EN)	32	RW	See section
110h	MUX2 EMIOS ENABLE (MUX2_EMIOS_EN)	32	RW	See section
114h	MUX2 MISC ENABLE (MUX2_MISC_EN)	32	RW	See section
200h	SIUL2 MCU ID Register #3 (MIDR3)	32	R	See section
204h	SIUL2 MCU ID Register #4 (MIDR4)	32	R	See section
240h	Multiplexed Signal Configuration (MSCR0)	32	RW	0000_0000h
244h	Multiplexed Signal Configuration (MSCR1)	32	RW	0000_0000h
248h	Multiplexed Signal Configuration (MSCR2)	32	RW	0000_0000h
24Ch	Multiplexed Signal Configuration (MSCR3)	32	RW	0000_0000h
250h	Multiplexed Signal Configuration (MSCR4)	32	RW	0008_2827h
254h	Multiplexed Signal Configuration (MSCR5)	32	RW	0000_0000h
258h	Multiplexed Signal Configuration (MSCR6)	32	RW	0000_0000h
25Ch	Multiplexed Signal Configuration (MSCR7)	32	RW	0000_0000h
260h	Multiplexed Signal Configuration (MSCR8)	32	RW	0000_0000h
264h	Multiplexed Signal Configuration (MSCR9)	32	RW	0000_0000h
268h	Multiplexed Signal Configuration (MSCR10)	32	RW	0000_0127h
26Ch	Multiplexed Signal Configuration (MSCR11)	32	RW	0000_0000h
270h	Multiplexed Signal Configuration (MSCR12)	32	RW	0000_0003h
274h	Multiplexed Signal Configuration (MSCR13)	32	RW	0000_0000h
278h	Multiplexed Signal Configuration (MSCR14)	32	RW	0000_0000h
27Ch	Multiplexed Signal Configuration (MSCR15)	32	RW	0000_0000h
280h	Multiplexed Signal Configuration (MSCR16)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
284h	Multiplexed Signal Configuration (MSCR17)	32	RW	0000_0000h
288h	Multiplexed Signal Configuration (MSCR18)	32	RW	0000_0000h
28Ch	Multiplexed Signal Configuration (MSCR19)	32	RW	0000_0000h
290h	Multiplexed Signal Configuration (MSCR20)	32	RW	0000_0000h
294h	Multiplexed Signal Configuration (MSCR21)	32	RW	0000_0000h
298h	Multiplexed Signal Configuration (MSCR22)	32	RW	0000_0000h
29Ch	Multiplexed Signal Configuration (MSCR23)	32	RW	0000_0000h
2A0h	Multiplexed Signal Configuration (MSCR24)	32	RW	0000_0000h
2A4h	Multiplexed Signal Configuration (MSCR25)	32	RW	0000_0000h
2A8h	Multiplexed Signal Configuration (MSCR26)	32	RW	0000_0000h
2ACh	Multiplexed Signal Configuration (MSCR27)	32	RW	0000_0000h
2B0h	Multiplexed Signal Configuration (MSCR28)	32	RW	0000_0000h
2B4h	Multiplexed Signal Configuration (MSCR29)	32	RW	0000_0000h
2B8h	Multiplexed Signal Configuration (MSCR30)	32	RW	0000_0000h
2BCh	Multiplexed Signal Configuration (MSCR31)	32	RW	0000_0000h
2C0h	Multiplexed Signal Configuration (MSCR32)	32	RW	0000_0000h
2C4h	Multiplexed Signal Configuration (MSCR33)	32	RW	0000_0000h
2C8h	Multiplexed Signal Configuration (MSCR34)	32	RW	0000_0000h
2CCh	Multiplexed Signal Configuration (MSCR35)	32	RW	0000_0000h
2D0h	Multiplexed Signal Configuration (MSCR36)	32	RW	0000_0000h
2D4h	Multiplexed Signal Configuration (MSCR37)	32	RW	0000_0000h
2E0h	Multiplexed Signal Configuration (MSCR40)	32	RW	0000_0000h
2E4h	Multiplexed Signal Configuration (MSCR41)	32	RW	0000_0000h
2E8h	Multiplexed Signal Configuration (MSCR42)	32	RW	0000_0000h
2ECh	Multiplexed Signal Configuration (MSCR43)	32	RW	0000_0000h
2F0h	Multiplexed Signal Configuration (MSCR44)	32	RW	0000_0000h
2F4h	Multiplexed Signal Configuration (MSCR45)	32	RW	0000_0000h
2F8h	Multiplexed Signal Configuration (MSCR46)	32	RW	0000_0000h
2FCh	Multiplexed Signal Configuration (MSCR47)	32	RW	0000_0000h
300h	Multiplexed Signal Configuration (MSCR48)	32	RW	0000_0000h
304h	Multiplexed Signal Configuration (MSCR49)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
308h	Multiplexed Signal Configuration (MSCR50)	32	RW	0000_0000h
30Ch	Multiplexed Signal Configuration (MSCR51)	32	RW	0000_0000h
310h	Multiplexed Signal Configuration (MSCR52)	32	RW	0000_0000h
314h	Multiplexed Signal Configuration (MSCR53)	32	RW	0000_0000h
318h	Multiplexed Signal Configuration (MSCR54)	32	RW	0000_0000h
31Ch	Multiplexed Signal Configuration (MSCR55)	32	RW	0000_0000h
320h	Multiplexed Signal Configuration (MSCR56)	32	RW	0000_0000h
324h	Multiplexed Signal Configuration (MSCR57)	32	RW	0000_0000h
328h	Multiplexed Signal Configuration (MSCR58)	32	RW	0000_0000h
32Ch	Multiplexed Signal Configuration (MSCR59)	32	RW	0000_0000h
330h	Multiplexed Signal Configuration (MSCR60)	32	RW	0000_0000h
334h	Multiplexed Signal Configuration (MSCR61)	32	RW	0000_0000h
338h	Multiplexed Signal Configuration (MSCR62)	32	RW	0000_0000h
33Ch	Multiplexed Signal Configuration (MSCR63)	32	RW	0000_0000h
340h	Multiplexed Signal Configuration (MSCR64)	32	RW	0000_0000h
344h	Multiplexed Signal Configuration (MSCR65)	32	RW	0000_0000h
348h	Multiplexed Signal Configuration (MSCR66)	32	RW	0000_4000h
34Ch	Multiplexed Signal Configuration (MSCR67)	32	RW	0000_4000h
350h	Multiplexed Signal Configuration (MSCR68)	32	RW	0008_2000h
354h	Multiplexed Signal Configuration (MSCR69)	32	RW	0008_2800h
358h	Multiplexed Signal Configuration (MSCR70)	32	RW	0000_0000h
35Ch	Multiplexed Signal Configuration (MSCR71)	32	RW	0000_0000h
360h	Multiplexed Signal Configuration (MSCR72)	32	RW	0000_0000h
364h	Multiplexed Signal Configuration (MSCR73)	32	RW	0000_0000h
368h	Multiplexed Signal Configuration (MSCR74)	32	RW	0000_0000h
36Ch	Multiplexed Signal Configuration (MSCR75)	32	RW	0000_0000h
370h	Multiplexed Signal Configuration (MSCR76)	32	RW	0000_4000h
374h	Multiplexed Signal Configuration (MSCR77)	32	RW	0000_0000h
378h	Multiplexed Signal Configuration (MSCR78)	32	RW	0000_0000h
37Ch	Multiplexed Signal Configuration (MSCR79)	32	RW	0000_0000h
380h	Multiplexed Signal Configuration (MSCR80)	32	RW	0000_4000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
384h	Multiplexed Signal Configuration (MSCR81)	32	RW	0000_0000h
388h	Multiplexed Signal Configuration (MSCR82)	32	RW	0000_0000h
38Ch	Multiplexed Signal Configuration (MSCR83)	32	RW	0000_0000h
390h	Multiplexed Signal Configuration (MSCR84)	32	RW	0000_0000h
394h	Multiplexed Signal Configuration (MSCR85)	32	RW	0000_0000h
398h	Multiplexed Signal Configuration (MSCR86)	32	RW	0000_0000h
39Ch	Multiplexed Signal Configuration (MSCR87)	32	RW	0000_0000h
3A0h	Multiplexed Signal Configuration (MSCR88)	32	RW	0000_0000h
3A4h	Multiplexed Signal Configuration (MSCR89)	32	RW	0000_0000h
3A8h	Multiplexed Signal Configuration (MSCR90)	32	RW	0000_0000h
3ACh	Multiplexed Signal Configuration (MSCR91)	32	RW	0000_0000h
3B0h	Multiplexed Signal Configuration (MSCR92)	32	RW	0000_0000h
3B4h	Multiplexed Signal Configuration (MSCR93)	32	RW	0000_0000h
3B8h	Multiplexed Signal Configuration (MSCR94)	32	RW	0000_0000h
3BCh	Multiplexed Signal Configuration (MSCR95)	32	RW	0000_0000h
3C0h	Multiplexed Signal Configuration (MSCR96)	32	RW	0000_0000h
3C4h	Multiplexed Signal Configuration (MSCR97)	32	RW	0000_0000h
3C8h	Multiplexed Signal Configuration (MSCR98)	32	RW	0000_0000h
3CCh	Multiplexed Signal Configuration (MSCR99)	32	RW	0000_0000h
3D0h	Multiplexed Signal Configuration (MSCR100)	32	RW	0000_0000h
3D4h	Multiplexed Signal Configuration (MSCR101)	32	RW	0000_4000h
3D8h	Multiplexed Signal Configuration (MSCR102)	32	RW	0000_4000h
3DCh	Multiplexed Signal Configuration (MSCR103)	32	RW	0000_4000h
3E0h	Multiplexed Signal Configuration (MSCR104)	32	RW	0000_0000h
3E4h	Multiplexed Signal Configuration (MSCR105)	32	RW	0000_0000h
3E8h	Multiplexed Signal Configuration (MSCR106)	32	RW	0000_4000h
3ECh	Multiplexed Signal Configuration (MSCR107)	32	RW	0000_4000h
3F0h	Multiplexed Signal Configuration (MSCR108)	32	RW	0000_4000h
3F4h	Multiplexed Signal Configuration (MSCR109)	32	RW	0000_0000h
3F8h	Multiplexed Signal Configuration (MSCR110)	32	RW	0000_0000h
3FCh	Multiplexed Signal Configuration (MSCR111)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
400h	Multiplexed Signal Configuration (MSCR112)	32	RW	0000_0000h
404h	Multiplexed Signal Configuration (MSCR113)	32	RW	0000_0000h
408h	Multiplexed Signal Configuration (MSCR114)	32	RW	0000_0000h
40Ch	Multiplexed Signal Configuration (MSCR115)	32	RW	0000_0000h
410h	Multiplexed Signal Configuration (MSCR116)	32	RW	0000_0000h
414h	Multiplexed Signal Configuration (MSCR117)	32	RW	0000_0000h
418h	Multiplexed Signal Configuration (MSCR118)	32	RW	0000_0000h
41Ch	Multiplexed Signal Configuration (MSCR119)	32	RW	0000_0000h
420h	Multiplexed Signal Configuration (MSCR120)	32	RW	0000_0000h
424h	Multiplexed Signal Configuration (MSCR121)	32	RW	0000_0000h
428h	Multiplexed Signal Configuration (MSCR122)	32	RW	0000_0000h
42Ch	Multiplexed Signal Configuration (MSCR123)	32	RW	0000_0000h
430h	Multiplexed Signal Configuration (MSCR124)	32	RW	0000_0000h
434h	Multiplexed Signal Configuration (MSCR125)	32	RW	0000_0000h
438h	Multiplexed Signal Configuration (MSCR126)	32	RW	0000_0000h
43Ch	Multiplexed Signal Configuration (MSCR127)	32	RW	0000_0000h
440h	Multiplexed Signal Configuration (MSCR128)	32	RW	0000_0000h
444h	Multiplexed Signal Configuration (MSCR129)	32	RW	0000_0000h
448h	Multiplexed Signal Configuration (MSCR130)	32	RW	0000_0000h
44Ch	Multiplexed Signal Configuration (MSCR131)	32	RW	0000_0000h
450h	Multiplexed Signal Configuration (MSCR132)	32	RW	0000_0000h
454h	Multiplexed Signal Configuration (MSCR133)	32	RW	0000_0000h
458h	Multiplexed Signal Configuration (MSCR134)	32	RW	0000_0000h
45Ch	Multiplexed Signal Configuration (MSCR135)	32	RW	0000_0000h
460h	Multiplexed Signal Configuration (MSCR136)	32	RW	0000_4000h
464h	Multiplexed Signal Configuration (MSCR137)	32	RW	0000_0000h
468h	Multiplexed Signal Configuration (MSCR138)	32	RW	0000_0000h
46Ch	Multiplexed Signal Configuration (MSCR139)	32	RW	0000_0000h
470h	Multiplexed Signal Configuration (MSCR140)	32	RW	0000_0000h
478h	Multiplexed Signal Configuration (MSCR142)	32	RW	0000_0000h
47Ch	Multiplexed Signal Configuration (MSCR143)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
480h	Multiplexed Signal Configuration (MSCR144)	32	RW	0000_0000h
484h	Multiplexed Signal Configuration (MSCR145)	32	RW	0000_0000h
488h	Multiplexed Signal Configuration (MSCR146)	32	RW	0000_0000h
48Ch	Multiplexed Signal Configuration (MSCR147)	32	RW	0000_0000h
490h	Multiplexed Signal Configuration (MSCR148)	32	RW	0000_0000h
494h	Multiplexed Signal Configuration (MSCR149)	32	RW	0000_0000h
498h	Multiplexed Signal Configuration (MSCR150)	32	RW	0000_0000h
49Ch	Multiplexed Signal Configuration (MSCR151)	32	RW	0000_0000h
4A0h	Multiplexed Signal Configuration (MSCR152)	32	RW	0000_0000h
4A4h	Multiplexed Signal Configuration (MSCR153)	32	RW	0000_0000h
4A8h	Multiplexed Signal Configuration (MSCR154)	32	RW	0000_0000h
4ACh	Multiplexed Signal Configuration (MSCR155)	32	RW	0000_0000h
4B0h	Multiplexed Signal Configuration (MSCR156)	32	RW	0000_0000h
4B4h	Multiplexed Signal Configuration (MSCR157)	32	RW	0000_0000h
4B8h	Multiplexed Signal Configuration (MSCR158)	32	RW	0000_0000h
4BCh	Multiplexed Signal Configuration (MSCR159)	32	RW	0000_0000h
4C0h	Multiplexed Signal Configuration (MSCR160)	32	RW	0000_0000h
4C4h	Multiplexed Signal Configuration (MSCR161)	32	RW	0000_0000h
4C8h	Multiplexed Signal Configuration (MSCR162)	32	RW	0000_0000h
4CCh	Multiplexed Signal Configuration (MSCR163)	32	RW	0000_0000h
4D0h	Multiplexed Signal Configuration (MSCR164)	32	RW	0000_0000h
4D4h	Multiplexed Signal Configuration (MSCR165)	32	RW	0000_0000h
4D8h	Multiplexed Signal Configuration (MSCR166)	32	RW	0000_0000h
4DCh	Multiplexed Signal Configuration (MSCR167)	32	RW	0000_0000h
4E0h	Multiplexed Signal Configuration (MSCR168)	32	RW	0000_0000h
4E4h	Multiplexed Signal Configuration (MSCR169)	32	RW	0000_0000h
4E8h	Multiplexed Signal Configuration (MSCR170)	32	RW	0000_0000h
4ECh	Multiplexed Signal Configuration (MSCR171)	32	RW	0000_0000h
4F0h	Multiplexed Signal Configuration (MSCR172)	32	RW	0000_0000h
4F4h	Multiplexed Signal Configuration (MSCR173)	32	RW	0000_0000h
4F8h	Multiplexed Signal Configuration (MSCR174)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
4FCh	Multiplexed Signal Configuration (MSCR175)	32	RW	0000_0000h
500h	Multiplexed Signal Configuration (MSCR176)	32	RW	0000_0000h
504h	Multiplexed Signal Configuration (MSCR177)	32	RW	0000_0000h
508h	Multiplexed Signal Configuration (MSCR178)	32	RW	0000_0000h
50Ch	Multiplexed Signal Configuration (MSCR179)	32	RW	0000_0000h
510h	Multiplexed Signal Configuration (MSCR180)	32	RW	0000_0000h
514h	Multiplexed Signal Configuration (MSCR181)	32	RW	0000_0000h
518h	Multiplexed Signal Configuration (MSCR182)	32	RW	0000_0000h
51Ch	Multiplexed Signal Configuration (MSCR183)	32	RW	0000_0000h
520h	Multiplexed Signal Configuration (MSCR184)	32	RW	0000_0000h
524h	Multiplexed Signal Configuration (MSCR185)	32	RW	0000_0000h
528h	Multiplexed Signal Configuration (MSCR186)	32	RW	0000_0000h
52Ch	Multiplexed Signal Configuration (MSCR187)	32	RW	0000_0000h
530h	Multiplexed Signal Configuration (MSCR188)	32	RW	0000_0000h
534h	Multiplexed Signal Configuration (MSCR189)	32	RW	0000_0000h
538h	Multiplexed Signal Configuration (MSCR190)	32	RW	0000_0000h
53Ch	Multiplexed Signal Configuration (MSCR191)	32	RW	0000_0000h
540h	Multiplexed Signal Configuration (MSCR192)	32	RW	0000_0000h
544h	Multiplexed Signal Configuration (MSCR193)	32	RW	0000_0000h
548h	Multiplexed Signal Configuration (MSCR194)	32	RW	0000_0000h
54Ch	Multiplexed Signal Configuration (MSCR195)	32	RW	0000_0000h
550h	Multiplexed Signal Configuration (MSCR196)	32	RW	0000_0000h
554h	Multiplexed Signal Configuration (MSCR197)	32	RW	0000_0000h
558h	Multiplexed Signal Configuration (MSCR198)	32	RW	0000_0000h
55Ch	Multiplexed Signal Configuration (MSCR199)	32	RW	0000_0000h
560h	Multiplexed Signal Configuration (MSCR200)	32	RW	0000_0000h
564h	Multiplexed Signal Configuration (MSCR201)	32	RW	0000_0000h
568h	Multiplexed Signal Configuration (MSCR202)	32	RW	0000_0000h
56Ch	Multiplexed Signal Configuration (MSCR203)	32	RW	0000_0000h
570h	Multiplexed Signal Configuration (MSCR204)	32	RW	0000_0000h
574h	Multiplexed Signal Configuration (MSCR205)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
578h	Multiplexed Signal Configuration (MSCR206)	32	RW	0000_0000h
57Ch	Multiplexed Signal Configuration (MSCR207)	32	RW	0000_0000h
580h	Multiplexed Signal Configuration (MSCR208)	32	RW	0000_0000h
584h	Multiplexed Signal Configuration (MSCR209)	32	RW	0000_0000h
588h	Multiplexed Signal Configuration (MSCR210)	32	RW	0000_0000h
58Ch	Multiplexed Signal Configuration (MSCR211)	32	RW	0000_0000h
590h	Multiplexed Signal Configuration (MSCR212)	32	RW	0000_0000h
594h	Multiplexed Signal Configuration (MSCR213)	32	RW	0000_0000h
598h	Multiplexed Signal Configuration (MSCR214)	32	RW	0000_0000h
59Ch	Multiplexed Signal Configuration (MSCR215)	32	RW	0000_0000h
5A0h	Multiplexed Signal Configuration (MSCR216)	32	RW	0000_0000h
5A4h	Multiplexed Signal Configuration (MSCR217)	32	RW	0000_0000h
5A8h	Multiplexed Signal Configuration (MSCR218)	32	RW	0000_0000h
5ACh	Multiplexed Signal Configuration (MSCR219)	32	RW	0000_0000h
5B0h	Multiplexed Signal Configuration (MSCR220)	32	RW	0000_0000h
5B4h	Multiplexed Signal Configuration (MSCR221)	32	RW	0000_0000h
5B8h	Multiplexed Signal Configuration (MSCR222)	32	RW	0000_0000h
5BCh	Multiplexed Signal Configuration (MSCR223)	32	RW	0000_0000h
5C0h	Multiplexed Signal Configuration (MSCR224)	32	RW	0000_0000h
5C4h	Multiplexed Signal Configuration (MSCR225)	32	RW	0000_0000h
5C8h	Multiplexed Signal Configuration (MSCR226)	32	RW	0000_0000h
5CCh	Multiplexed Signal Configuration (MSCR227)	32	RW	0000_0000h
5D0h	Multiplexed Signal Configuration (MSCR228)	32	RW	0000_0000h
5D4h	Multiplexed Signal Configuration (MSCR229)	32	RW	0000_0000h
5D8h	Multiplexed Signal Configuration (MSCR230)	32	RW	0000_0000h
5DCh	Multiplexed Signal Configuration (MSCR231)	32	RW	0000_0000h
5E0h	Multiplexed Signal Configuration (MSCR232)	32	RW	0000_0000h
5E4h	Multiplexed Signal Configuration (MSCR233)	32	RW	0000_0000h
5E8h	Multiplexed Signal Configuration (MSCR234)	32	RW	0000_0000h
5ECh	Multiplexed Signal Configuration (MSCR235)	32	RW	0000_0000h
5F0h	Multiplexed Signal Configuration (MSCR236)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
5F4h	Multiplexed Signal Configuration (MSCR237)	32	RW	0000_0000h
5F8h	Multiplexed Signal Configuration (MSCR238)	32	RW	0000_0000h
5FCh	Multiplexed Signal Configuration (MSCR239)	32	RW	0000_0000h
600h	Multiplexed Signal Configuration (MSCR240)	32	RW	0000_0000h
604h	Multiplexed Signal Configuration (MSCR241)	32	RW	0000_0000h
608h	Multiplexed Signal Configuration (MSCR242)	32	RW	0000_0000h
60Ch	Multiplexed Signal Configuration (MSCR243)	32	RW	0000_0000h
610h	Multiplexed Signal Configuration (MSCR244)	32	RW	0000_0000h
614h	Multiplexed Signal Configuration (MSCR245)	32	RW	0000_0000h
618h	Multiplexed Signal Configuration (MSCR246)	32	RW	0000_0000h
61Ch	Multiplexed Signal Configuration (MSCR247)	32	RW	0000_0000h
620h	Multiplexed Signal Configuration (MSCR248)	32	RW	0000_0000h
624h	Multiplexed Signal Configuration (MSCR249)	32	RW	0000_0000h
628h	Multiplexed Signal Configuration (MSCR250)	32	RW	0000_0000h
62Ch	Multiplexed Signal Configuration (MSCR251)	32	RW	0000_0000h
630h	Multiplexed Signal Configuration (MSCR252)	32	RW	0000_0000h
634h	Multiplexed Signal Configuration (MSCR253)	32	RW	0000_0000h
638h	Multiplexed Signal Configuration (MSCR254)	32	RW	0000_0000h
63Ch	Multiplexed Signal Configuration (MSCR255)	32	RW	0000_0000h
640h	Multiplexed Signal Configuration (MSCR256)	32	RW	0000_0000h
644h	Multiplexed Signal Configuration (MSCR257)	32	RW	0000_0000h
648h	Multiplexed Signal Configuration (MSCR258)	32	RW	0000_0000h
64Ch	Multiplexed Signal Configuration (MSCR259)	32	RW	0000_0000h
650h	Multiplexed Signal Configuration (MSCR260)	32	RW	0000_0000h
654h	Multiplexed Signal Configuration (MSCR261)	32	RW	0000_0000h
658h	Multiplexed Signal Configuration (MSCR262)	32	RW	0000_0000h
65Ch	Multiplexed Signal Configuration (MSCR263)	32	RW	0000_0000h
660h	Multiplexed Signal Configuration (MSCR264)	32	RW	0000_0000h
664h	Multiplexed Signal Configuration (MSCR265)	32	RW	0000_0000h
668h	Multiplexed Signal Configuration (MSCR266)	32	RW	0000_0000h
66Ch	Multiplexed Signal Configuration (MSCR267)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
670h	Multiplexed Signal Configuration (MSCR268)	32	RW	0000_0000h
674h	Multiplexed Signal Configuration (MSCR269)	32	RW	0000_0000h
678h	Multiplexed Signal Configuration (MSCR270)	32	RW	0000_0000h
67Ch	Multiplexed Signal Configuration (MSCR271)	32	RW	0000_0000h
680h	Multiplexed Signal Configuration (MSCR272)	32	RW	0000_0000h
684h	Multiplexed Signal Configuration (MSCR273)	32	RW	0000_0000h
688h	Multiplexed Signal Configuration (MSCR274)	32	RW	0000_0000h
68Ch	Multiplexed Signal Configuration (MSCR275)	32	RW	0000_0000h
690h	Multiplexed Signal Configuration (MSCR276)	32	RW	0000_0000h
694h	Multiplexed Signal Configuration (MSCR277)	32	RW	0000_0000h
698h	Multiplexed Signal Configuration (MSCR278)	32	RW	0000_0000h
69Ch	Multiplexed Signal Configuration (MSCR279)	32	RW	0000_0000h
6A0h	Multiplexed Signal Configuration (MSCR280)	32	RW	0000_0000h
6A4h	Multiplexed Signal Configuration (MSCR281)	32	RW	0000_0000h
6A8h	Multiplexed Signal Configuration (MSCR282)	32	RW	0000_0000h
6ACh	Multiplexed Signal Configuration (MSCR283)	32	RW	0000_0000h
6B0h	Multiplexed Signal Configuration (MSCR284)	32	RW	0000_0000h
6B4h	Multiplexed Signal Configuration (MSCR285)	32	RW	0000_0000h
6B8h	Multiplexed Signal Configuration (MSCR286)	32	RW	0000_0000h
6BCh	Multiplexed Signal Configuration (MSCR287)	32	RW	0000_0000h
6C0h	Multiplexed Signal Configuration (MSCR288)	32	RW	0000_0000h
6C4h	Multiplexed Signal Configuration (MSCR289)	32	RW	0000_0000h
6C8h	Multiplexed Signal Configuration (MSCR290)	32	RW	0000_0000h
6CCh	Multiplexed Signal Configuration (MSCR291)	32	RW	0000_0000h
6D0h	Multiplexed Signal Configuration (MSCR292)	32	RW	0000_0000h
6D4h	Multiplexed Signal Configuration (MSCR293)	32	RW	0000_0000h
6D8h	Multiplexed Signal Configuration (MSCR294)	32	RW	0000_0000h
6DCh	Multiplexed Signal Configuration (MSCR295)	32	RW	0000_0000h
6E0h	Multiplexed Signal Configuration (MSCR296)	32	RW	0000_0000h
6E4h	Multiplexed Signal Configuration (MSCR297)	32	RW	0000_0000h
6E8h	Multiplexed Signal Configuration (MSCR298)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
6ECh	Multiplexed Signal Configuration (MSCR299)	32	RW	0000_0000h
6F0h	Multiplexed Signal Configuration (MSCR300)	32	RW	0000_0000h
6F4h	Multiplexed Signal Configuration (MSCR301)	32	RW	0000_0000h
6F8h	Multiplexed Signal Configuration (MSCR302)	32	RW	0000_0000h
6FCh	Multiplexed Signal Configuration (MSCR303)	32	RW	0000_0000h
700h	Multiplexed Signal Configuration (MSCR304)	32	RW	0000_0000h
704h	Multiplexed Signal Configuration (MSCR305)	32	RW	0000_0000h
708h	Multiplexed Signal Configuration (MSCR306)	32	RW	0000_0000h
70Ch	Multiplexed Signal Configuration (MSCR307)	32	RW	0000_0000h
710h	Multiplexed Signal Configuration (MSCR308)	32	RW	0000_0000h
714h	Multiplexed Signal Configuration (MSCR309)	32	RW	0000_0000h
718h	Multiplexed Signal Configuration (MSCR310)	32	RW	0000_0000h
71Ch	Multiplexed Signal Configuration (MSCR311)	32	RW	0000_0000h
720h	Multiplexed Signal Configuration (MSCR312)	32	RW	0000_0000h
724h	Multiplexed Signal Configuration (MSCR313)	32	RW	0000_0000h
728h	Multiplexed Signal Configuration (MSCR314)	32	RW	0000_0000h
72Ch	Multiplexed Signal Configuration (MSCR315)	32	RW	0000_0000h
730h	Multiplexed Signal Configuration (MSCR316)	32	RW	0000_0000h
734h	Multiplexed Signal Configuration (MSCR317)	32	RW	0000_0000h
738h	Multiplexed Signal Configuration (MSCR318)	32	RW	0000_0000h
73Ch	Multiplexed Signal Configuration (MSCR319)	32	RW	0000_0000h
740h	Multiplexed Signal Configuration (MSCR320)	32	RW	0000_0000h
744h	Multiplexed Signal Configuration (MSCR321)	32	RW	0000_0000h
748h	Multiplexed Signal Configuration (MSCR322)	32	RW	0000_0000h
74Ch	Multiplexed Signal Configuration (MSCR323)	32	RW	0000_0000h
A40h	Input Multiplexed Signal Configuration (IMCR0)	32	RW	0000_0000h
A44h	Input Multiplexed Signal Configuration (IMCR1)	32	RW	0000_0000h
A48h	Input Multiplexed Signal Configuration (IMCR2)	32	RW	0000_0000h
A4Ch	Input Multiplexed Signal Configuration (IMCR3)	32	RW	0000_0000h
A50h	Input Multiplexed Signal Configuration (IMCR4)	32	RW	0000_0000h
A54h	Input Multiplexed Signal Configuration (IMCR5)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
A80h	Input Multiplexed Signal Configuration (IMCR16)	32	RW	0000_0000h
A84h	Input Multiplexed Signal Configuration (IMCR17)	32	RW	0000_0000h
A88h	Input Multiplexed Signal Configuration (IMCR18)	32	RW	0000_0000h
A8Ch	Input Multiplexed Signal Configuration (IMCR19)	32	RW	0000_0000h
A90h	Input Multiplexed Signal Configuration (IMCR20)	32	RW	0000_0000h
A94h	Input Multiplexed Signal Configuration (IMCR21)	32	RW	0000_0000h
A98h	Input Multiplexed Signal Configuration (IMCR22)	32	RW	0000_0000h
A9Ch	Input Multiplexed Signal Configuration (IMCR23)	32	RW	0000_0000h
AA0h	Input Multiplexed Signal Configuration (IMCR24)	32	RW	0000_0000h
AA4h	Input Multiplexed Signal Configuration (IMCR25)	32	RW	0000_0000h
AA8h	Input Multiplexed Signal Configuration (IMCR26)	32	RW	0000_0000h
AACh	Input Multiplexed Signal Configuration (IMCR27)	32	RW	0000_0000h
AB0h	Input Multiplexed Signal Configuration (IMCR28)	32	RW	0000_0000h
AB4h	Input Multiplexed Signal Configuration (IMCR29)	32	RW	0000_0000h
AB8h	Input Multiplexed Signal Configuration (IMCR30)	32	RW	0000_0000h
ABCh	Input Multiplexed Signal Configuration (IMCR31)	32	RW	0000_0000h
AC0h	Input Multiplexed Signal Configuration (IMCR32)	32	RW	0000_0000h
AC4h	Input Multiplexed Signal Configuration (IMCR33)	32	RW	0000_0000h
AC8h	Input Multiplexed Signal Configuration (IMCR34)	32	RW	0000_0000h
ACCh	Input Multiplexed Signal Configuration (IMCR35)	32	RW	0000_0000h
AD0h	Input Multiplexed Signal Configuration (IMCR36)	32	RW	0000_0000h
AD4h	Input Multiplexed Signal Configuration (IMCR37)	32	RW	0000_0000h
AD8h	Input Multiplexed Signal Configuration (IMCR38)	32	RW	0000_0000h
ADCh	Input Multiplexed Signal Configuration (IMCR39)	32	RW	0000_0000h
AE0h	Input Multiplexed Signal Configuration (IMCR40)	32	RW	0000_0000h
AE4h	Input Multiplexed Signal Configuration (IMCR41)	32	RW	0000_0000h
AE8h	Input Multiplexed Signal Configuration (IMCR42)	32	RW	0000_0000h
AECh	Input Multiplexed Signal Configuration (IMCR43)	32	RW	0000_0000h
AF0h	Input Multiplexed Signal Configuration (IMCR44)	32	RW	0000_0000h
AF4h	Input Multiplexed Signal Configuration (IMCR45)	32	RW	0000_0000h
AF8h	Input Multiplexed Signal Configuration (IMCR46)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
AFCh	Input Multiplexed Signal Configuration (IMCR47)	32	RW	0000_0000h
B00h	Input Multiplexed Signal Configuration (IMCR48)	32	RW	0000_0000h
B04h	Input Multiplexed Signal Configuration (IMCR49)	32	RW	0000_0000h
B08h	Input Multiplexed Signal Configuration (IMCR50)	32	RW	0000_0000h
B0Ch	Input Multiplexed Signal Configuration (IMCR51)	32	RW	0000_0000h
B10h	Input Multiplexed Signal Configuration (IMCR52)	32	RW	0000_0000h
B14h	Input Multiplexed Signal Configuration (IMCR53)	32	RW	0000_0000h
B18h	Input Multiplexed Signal Configuration (IMCR54)	32	RW	0000_0000h
B1Ch	Input Multiplexed Signal Configuration (IMCR55)	32	RW	0000_0000h
B20h	Input Multiplexed Signal Configuration (IMCR56)	32	RW	0000_0000h
B24h	Input Multiplexed Signal Configuration (IMCR57)	32	RW	0000_0000h
B28h	Input Multiplexed Signal Configuration (IMCR58)	32	RW	0000_0000h
B2Ch	Input Multiplexed Signal Configuration (IMCR59)	32	RW	0000_0000h
B30h	Input Multiplexed Signal Configuration (IMCR60)	32	RW	0000_0000h
B34h	Input Multiplexed Signal Configuration (IMCR61)	32	RW	0000_0000h
B38h	Input Multiplexed Signal Configuration (IMCR62)	32	RW	0000_0000h
B3Ch	Input Multiplexed Signal Configuration (IMCR63)	32	RW	0000_0000h
B40h	Input Multiplexed Signal Configuration (IMCR64)	32	RW	0000_0000h
B44h	Input Multiplexed Signal Configuration (IMCR65)	32	RW	0000_0000h
B48h	Input Multiplexed Signal Configuration (IMCR66)	32	RW	0000_0000h
B4Ch	Input Multiplexed Signal Configuration (IMCR67)	32	RW	0000_0000h
B50h	Input Multiplexed Signal Configuration (IMCR68)	32	RW	0000_0000h
B54h	Input Multiplexed Signal Configuration (IMCR69)	32	RW	0000_0000h
B58h	Input Multiplexed Signal Configuration (IMCR70)	32	RW	0000_0000h
B5Ch	Input Multiplexed Signal Configuration (IMCR71)	32	RW	0000_0000h
B80h	Input Multiplexed Signal Configuration (IMCR80)	32	RW	0000_0000h
B84h	Input Multiplexed Signal Configuration (IMCR81)	32	RW	0000_0000h
B88h	Input Multiplexed Signal Configuration (IMCR82)	32	RW	0000_0000h
B8Ch	Input Multiplexed Signal Configuration (IMCR83)	32	RW	0000_0000h
B90h	Input Multiplexed Signal Configuration (IMCR84)	32	RW	0000_0000h
B94h	Input Multiplexed Signal Configuration (IMCR85)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
B98h	Input Multiplexed Signal Configuration (IMCR86)	32	RW	0000_0000h
B9Ch	Input Multiplexed Signal Configuration (IMCR87)	32	RW	0000_0000h
BA0h	Input Multiplexed Signal Configuration (IMCR88)	32	RW	0000_0000h
BA4h	Input Multiplexed Signal Configuration (IMCR89)	32	RW	0000_0000h
BA8h	Input Multiplexed Signal Configuration (IMCR90)	32	RW	0000_0000h
BACh	Input Multiplexed Signal Configuration (IMCR91)	32	RW	0000_0000h
BB0h	Input Multiplexed Signal Configuration (IMCR92)	32	RW	0000_0000h
BB4h	Input Multiplexed Signal Configuration (IMCR93)	32	RW	0000_0000h
BB8h	Input Multiplexed Signal Configuration (IMCR94)	32	RW	0000_0000h
BBCh	Input Multiplexed Signal Configuration (IMCR95)	32	RW	0000_0000h
BC0h	Input Multiplexed Signal Configuration (IMCR96)	32	RW	0000_0000h
BC4h	Input Multiplexed Signal Configuration (IMCR97)	32	RW	0000_0000h
BC8h	Input Multiplexed Signal Configuration (IMCR98)	32	RW	0000_0000h
BCCh	Input Multiplexed Signal Configuration (IMCR99)	32	RW	0000_0000h
BD0h	Input Multiplexed Signal Configuration (IMCR100)	32	RW	0000_0000h
BD4h	Input Multiplexed Signal Configuration (IMCR101)	32	RW	0000_0000h
BD8h	Input Multiplexed Signal Configuration (IMCR102)	32	RW	0000_0000h
BDCh	Input Multiplexed Signal Configuration (IMCR103)	32	RW	0000_0000h
C00h	Input Multiplexed Signal Configuration (IMCR112)	32	RW	0000_0000h
C04h	Input Multiplexed Signal Configuration (IMCR113)	32	RW	0000_0000h
C08h	Input Multiplexed Signal Configuration (IMCR114)	32	RW	0000_0000h
C0Ch	Input Multiplexed Signal Configuration (IMCR115)	32	RW	0000_0000h
C10h	Input Multiplexed Signal Configuration (IMCR116)	32	RW	0000_0000h
C14h	Input Multiplexed Signal Configuration (IMCR117)	32	RW	0000_0000h
C18h	Input Multiplexed Signal Configuration (IMCR118)	32	RW	0000_0000h
C1Ch	Input Multiplexed Signal Configuration (IMCR119)	32	RW	0000_0000h
C20h	Input Multiplexed Signal Configuration (IMCR120)	32	RW	0000_0000h
C24h	Input Multiplexed Signal Configuration (IMCR121)	32	RW	0000_0000h
C28h	Input Multiplexed Signal Configuration (IMCR122)	32	RW	0000_0000h
C2Ch	Input Multiplexed Signal Configuration (IMCR123)	32	RW	0000_0000h
C30h	Input Multiplexed Signal Configuration (IMCR124)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
C34h	Input Multiplexed Signal Configuration (IMCR125)	32	RW	0000_0000h
C38h	Input Multiplexed Signal Configuration (IMCR126)	32	RW	0000_0000h
C3Ch	Input Multiplexed Signal Configuration (IMCR127)	32	RW	0000_0000h
C40h	Input Multiplexed Signal Configuration (IMCR128)	32	RW	0000_0000h
C44h	Input Multiplexed Signal Configuration (IMCR129)	32	RW	0000_0000h
C48h	Input Multiplexed Signal Configuration (IMCR130)	32	RW	0000_0000h
C4Ch	Input Multiplexed Signal Configuration (IMCR131)	32	RW	0000_0000h
C50h	Input Multiplexed Signal Configuration (IMCR132)	32	RW	0000_0000h
C54h	Input Multiplexed Signal Configuration (IMCR133)	32	RW	0000_0000h
C58h	Input Multiplexed Signal Configuration (IMCR134)	32	RW	0000_0000h
C5Ch	Input Multiplexed Signal Configuration (IMCR135)	32	RW	0000_0000h
C80h	Input Multiplexed Signal Configuration (IMCR144)	32	RW	0000_0000h
C84h	Input Multiplexed Signal Configuration (IMCR145)	32	RW	0000_0000h
C88h	Input Multiplexed Signal Configuration (IMCR146)	32	RW	0000_0000h
C8Ch	Input Multiplexed Signal Configuration (IMCR147)	32	RW	0000_0000h
C90h	Input Multiplexed Signal Configuration (IMCR148)	32	RW	0000_0000h
C94h	Input Multiplexed Signal Configuration (IMCR149)	32	RW	0000_0000h
CA0h	Input Multiplexed Signal Configuration (IMCR152)	32	RW	0000_0000h
CA4h	Input Multiplexed Signal Configuration (IMCR153)	32	RW	0000_0000h
CA8h	Input Multiplexed Signal Configuration (IMCR154)	32	RW	0000_0000h
CACh	Input Multiplexed Signal Configuration (IMCR155)	32	RW	0000_0000h
CB0h	Input Multiplexed Signal Configuration (IMCR156)	32	RW	0000_0000h
CB4h	Input Multiplexed Signal Configuration (IMCR157)	32	RW	0000_0000h
CB8h	Input Multiplexed Signal Configuration (IMCR158)	32	RW	0000_0000h
CBCh	Input Multiplexed Signal Configuration (IMCR159)	32	RW	0000_0000h
CC0h	Input Multiplexed Signal Configuration (IMCR160)	32	RW	0000_0000h
CC4h	Input Multiplexed Signal Configuration (IMCR161)	32	RW	0000_0000h
CC8h	Input Multiplexed Signal Configuration (IMCR162)	32	RW	0000_0000h
CCCh	Input Multiplexed Signal Configuration (IMCR163)	32	RW	0000_0000h
CD0h	Input Multiplexed Signal Configuration (IMCR164)	32	RW	0000_0000h
CD4h	Input Multiplexed Signal Configuration (IMCR165)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
CD8h	Input Multiplexed Signal Configuration (IMCR166)	32	RW	0000_0000h
CDCh	Input Multiplexed Signal Configuration (IMCR167)	32	RW	0000_0000h
CE0h	Input Multiplexed Signal Configuration (IMCR168)	32	RW	0000_0000h
CE4h	Input Multiplexed Signal Configuration (IMCR169)	32	RW	0000_0000h
CE8h	Input Multiplexed Signal Configuration (IMCR170)	32	RW	0000_0000h
CECh	Input Multiplexed Signal Configuration (IMCR171)	32	RW	0000_0000h
CF0h	Input Multiplexed Signal Configuration (IMCR172)	32	RW	0000_0000h
CF4h	Input Multiplexed Signal Configuration (IMCR173)	32	RW	0000_0000h
CF8h	Input Multiplexed Signal Configuration (IMCR174)	32	RW	0000_0000h
CFCh	Input Multiplexed Signal Configuration (IMCR175)	32	RW	0000_0000h
D00h	Input Multiplexed Signal Configuration (IMCR176)	32	RW	0000_0000h
D04h	Input Multiplexed Signal Configuration (IMCR177)	32	RW	0000_0000h
D08h	Input Multiplexed Signal Configuration (IMCR178)	32	RW	0000_0000h
D0Ch	Input Multiplexed Signal Configuration (IMCR179)	32	RW	0000_0000h
D10h	Input Multiplexed Signal Configuration (IMCR180)	32	RW	0000_0000h
D14h	Input Multiplexed Signal Configuration (IMCR181)	32	RW	0000_0000h
D18h	Input Multiplexed Signal Configuration (IMCR182)	32	RW	0000_0000h
D1Ch	Input Multiplexed Signal Configuration (IMCR183)	32	RW	0000_0000h
D20h	Input Multiplexed Signal Configuration (IMCR184)	32	RW	0000_0000h
D24h	Input Multiplexed Signal Configuration (IMCR185)	32	RW	0000_0000h
D28h	Input Multiplexed Signal Configuration (IMCR186)	32	RW	0000_0000h
D2Ch	Input Multiplexed Signal Configuration (IMCR187)	32	RW	0000_0000h
D30h	Input Multiplexed Signal Configuration (IMCR188)	32	RW	0000_0000h
D34h	Input Multiplexed Signal Configuration (IMCR189)	32	RW	0000_0000h
D38h	Input Multiplexed Signal Configuration (IMCR190)	32	RW	0000_0000h
D3Ch	Input Multiplexed Signal Configuration (IMCR191)	32	RW	0000_0000h
D40h	Input Multiplexed Signal Configuration (IMCR192)	32	RW	0000_0000h
D44h	Input Multiplexed Signal Configuration (IMCR193)	32	RW	0000_0000h
D48h	Input Multiplexed Signal Configuration (IMCR194)	32	RW	0000_0000h
D4Ch	Input Multiplexed Signal Configuration (IMCR195)	32	RW	0000_0000h
D50h	Input Multiplexed Signal Configuration (IMCR196)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
D54h	Input Multiplexed Signal Configuration (IMCR197)	32	RW	0000_0000h
D58h	Input Multiplexed Signal Configuration (IMCR198)	32	RW	0000_0000h
D5Ch	Input Multiplexed Signal Configuration (IMCR199)	32	RW	0000_0000h
D60h	Input Multiplexed Signal Configuration (IMCR200)	32	RW	0000_0000h
D64h	Input Multiplexed Signal Configuration (IMCR201)	32	RW	0000_0000h
D68h	Input Multiplexed Signal Configuration (IMCR202)	32	RW	0000_0000h
D8Ch	Input Multiplexed Signal Configuration (IMCR211)	32	RW	0000_0000h
D90h	Input Multiplexed Signal Configuration (IMCR212)	32	RW	0000_0000h
D94h	Input Multiplexed Signal Configuration (IMCR213)	32	RW	0000_0000h
D98h	Input Multiplexed Signal Configuration (IMCR214)	32	RW	0000_0000h
D9Ch	Input Multiplexed Signal Configuration (IMCR215)	32	RW	0000_0000h
DA0h	Input Multiplexed Signal Configuration (IMCR216)	32	RW	0000_0000h
DA4h	Input Multiplexed Signal Configuration (IMCR217)	32	RW	0000_0000h
DA8h	Input Multiplexed Signal Configuration (IMCR218)	32	RW	0000_0000h
DACh	Input Multiplexed Signal Configuration (IMCR219)	32	RW	0000_0000h
DB0h	Input Multiplexed Signal Configuration (IMCR220)	32	RW	0000_0000h
DB4h	Input Multiplexed Signal Configuration (IMCR221)	32	RW	0000_0000h
DB8h	Input Multiplexed Signal Configuration (IMCR222)	32	RW	0000_0000h
DBCh	Input Multiplexed Signal Configuration (IMCR223)	32	RW	0000_0000h
DC0h	Input Multiplexed Signal Configuration (IMCR224)	32	RW	0000_0000h
DC4h	Input Multiplexed Signal Configuration (IMCR225)	32	RW	0000_0000h
DC8h	Input Multiplexed Signal Configuration (IMCR226)	32	RW	0000_0000h
DCCh	Input Multiplexed Signal Configuration (IMCR227)	32	RW	0000_0000h
DD0h	Input Multiplexed Signal Configuration (IMCR228)	32	RW	0000_0000h
DD4h	Input Multiplexed Signal Configuration (IMCR229)	32	RW	0000_0000h
DD8h	Input Multiplexed Signal Configuration (IMCR230)	32	RW	0000_0000h
DDCh	Input Multiplexed Signal Configuration (IMCR231)	32	RW	0000_0000h
DE0h	Input Multiplexed Signal Configuration (IMCR232)	32	RW	0000_0000h
DE4h	Input Multiplexed Signal Configuration (IMCR233)	32	RW	0000_0000h
DE8h	Input Multiplexed Signal Configuration (IMCR234)	32	RW	0000_0000h
DECh	Input Multiplexed Signal Configuration (IMCR235)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
DF0h	Input Multiplexed Signal Configuration (IMCR236)	32	RW	0000_0000h
DF4h	Input Multiplexed Signal Configuration (IMCR237)	32	RW	0000_0000h
DF8h	Input Multiplexed Signal Configuration (IMCR238)	32	RW	0000_0000h
DFCh	Input Multiplexed Signal Configuration (IMCR239)	32	RW	0000_0000h
E00h	Input Multiplexed Signal Configuration (IMCR240)	32	RW	0000_0000h
E04h	Input Multiplexed Signal Configuration (IMCR241)	32	RW	0000_0000h
E08h	Input Multiplexed Signal Configuration (IMCR242)	32	RW	0000_0000h
E0Ch	Input Multiplexed Signal Configuration (IMCR243)	32	RW	0000_0000h
E10h	Input Multiplexed Signal Configuration (IMCR244)	32	RW	0000_0000h
E14h	Input Multiplexed Signal Configuration (IMCR245)	32	RW	0000_0000h
E18h	Input Multiplexed Signal Configuration (IMCR246)	32	RW	0000_0000h
E1Ch	Input Multiplexed Signal Configuration (IMCR247)	32	RW	0000_0000h
E20h	Input Multiplexed Signal Configuration (IMCR248)	32	RW	0000_0000h
E24h	Input Multiplexed Signal Configuration (IMCR249)	32	RW	0000_0000h
E28h	Input Multiplexed Signal Configuration (IMCR250)	32	RW	0000_0000h
E2Ch	Input Multiplexed Signal Configuration (IMCR251)	32	RW	0000_0000h
E30h	Input Multiplexed Signal Configuration (IMCR252)	32	RW	0000_0000h
E34h	Input Multiplexed Signal Configuration (IMCR253)	32	RW	0000_0000h
E38h	Input Multiplexed Signal Configuration (IMCR254)	32	RW	0000_0000h
E3Ch	Input Multiplexed Signal Configuration (IMCR255)	32	RW	0000_0000h
E40h	Input Multiplexed Signal Configuration (IMCR256)	32	RW	0000_0000h
E44h	Input Multiplexed Signal Configuration (IMCR257)	32	RW	0000_0000h
E48h	Input Multiplexed Signal Configuration (IMCR258)	32	RW	0000_0000h
E4Ch	Input Multiplexed Signal Configuration (IMCR259)	32	RW	0000_0000h
E50h	Input Multiplexed Signal Configuration (IMCR260)	32	RW	0000_0000h
E54h	Input Multiplexed Signal Configuration (IMCR261)	32	RW	0000_0000h
E58h	Input Multiplexed Signal Configuration (IMCR262)	32	RW	0000_0000h
E5Ch	Input Multiplexed Signal Configuration (IMCR263)	32	RW	0000_0000h
E60h	Input Multiplexed Signal Configuration (IMCR264)	32	RW	0000_0000h
E64h	Input Multiplexed Signal Configuration (IMCR265)	32	RW	0000_0000h
E68h	Input Multiplexed Signal Configuration (IMCR266)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
E6Ch	Input Multiplexed Signal Configuration (IMCR267)	32	RW	0000_0000h
E70h	Input Multiplexed Signal Configuration (IMCR268)	32	RW	0000_0000h
EC4h	Input Multiplexed Signal Configuration (IMCR289)	32	RW	0000_0000h
EC8h	Input Multiplexed Signal Configuration (IMCR290)	32	RW	0000_0000h
ECCh	Input Multiplexed Signal Configuration (IMCR291)	32	RW	0000_0000h
ED0h	Input Multiplexed Signal Configuration (IMCR292)	32	RW	0000_0000h
ED4h	Input Multiplexed Signal Configuration (IMCR293)	32	RW	0000_0000h
ED8h	Input Multiplexed Signal Configuration (IMCR294)	32	RW	0000_0000h
EDCh	Input Multiplexed Signal Configuration (IMCR295)	32	RW	0000_0000h
EE0h	Input Multiplexed Signal Configuration (IMCR296)	32	RW	0000_0000h
EE4h	Input Multiplexed Signal Configuration (IMCR297)	32	RW	0000_0000h
EE8h	Input Multiplexed Signal Configuration (IMCR298)	32	RW	0000_0000h
EECh	Input Multiplexed Signal Configuration (IMCR299)	32	RW	0000_0000h
EF0h	Input Multiplexed Signal Configuration (IMCR300)	32	RW	0000_0000h
EF4h	Input Multiplexed Signal Configuration (IMCR301)	32	RW	0000_0000h
EF8h	Input Multiplexed Signal Configuration (IMCR302)	32	RW	0000_0000h
EFCh	Input Multiplexed Signal Configuration (IMCR303)	32	RW	0000_0000h
F00h	Input Multiplexed Signal Configuration (IMCR304)	32	RW	0000_0000h
F04h	Input Multiplexed Signal Configuration (IMCR305)	32	RW	0000_0000h
F08h	Input Multiplexed Signal Configuration (IMCR306)	32	RW	0000_0000h
F0Ch	Input Multiplexed Signal Configuration (IMCR307)	32	RW	0000_0000h
F10h	Input Multiplexed Signal Configuration (IMCR308)	32	RW	0000_0000h
F14h	Input Multiplexed Signal Configuration (IMCR309)	32	RW	0000_0000h
F2Ch	Input Multiplexed Signal Configuration (IMCR315)	32	RW	0000_0000h
F30h	Input Multiplexed Signal Configuration (IMCR316)	32	RW	0000_0000h
F34h	Input Multiplexed Signal Configuration (IMCR317)	32	RW	0000_0000h
F38h	Input Multiplexed Signal Configuration (IMCR318)	32	RW	0000_0000h
F3Ch	Input Multiplexed Signal Configuration (IMCR319)	32	RW	0000_0000h
F40h	Input Multiplexed Signal Configuration (IMCR320)	32	RW	0000_0000h
F44h	Input Multiplexed Signal Configuration (IMCR321)	32	RW	0000_0000h
F48h	Input Multiplexed Signal Configuration (IMCR322)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
F4Ch	Input Multiplexed Signal Configuration (IMCR323)	32	RW	0000_0000h
F50h	Input Multiplexed Signal Configuration (IMCR324)	32	RW	0000_0000h
F54h	Input Multiplexed Signal Configuration (IMCR325)	32	RW	0000_0000h
F9Ch	Input Multiplexed Signal Configuration (IMCR343)	32	RW	0000_0000h
FA0h	Input Multiplexed Signal Configuration (IMCR344)	32	RW	0000_0000h
FA4h	Input Multiplexed Signal Configuration (IMCR345)	32	RW	0000_0000h
FA8h	Input Multiplexed Signal Configuration (IMCR346)	32	RW	0000_0000h
FACh	Input Multiplexed Signal Configuration (IMCR347)	32	RW	0000_0000h
FB0h	Input Multiplexed Signal Configuration (IMCR348)	32	RW	0000_0000h
FB4h	Input Multiplexed Signal Configuration (IMCR349)	32	RW	0000_0000h
FB8h	Input Multiplexed Signal Configuration (IMCR350)	32	RW	0000_0000h
FBCh	Input Multiplexed Signal Configuration (IMCR351)	32	RW	0000_0000h
FC0h	Input Multiplexed Signal Configuration (IMCR352)	32	RW	0000_0000h
FC4h	Input Multiplexed Signal Configuration (IMCR353)	32	RW	0000_0000h
FC8h	Input Multiplexed Signal Configuration (IMCR354)	32	RW	0000_0000h
FCCh	Input Multiplexed Signal Configuration (IMCR355)	32	RW	0000_0000h
FD0h	Input Multiplexed Signal Configuration (IMCR356)	32	RW	0000_0000h
FD4h	Input Multiplexed Signal Configuration (IMCR357)	32	RW	0000_0000h
FD8h	Input Multiplexed Signal Configuration (IMCR358)	32	RW	0000_0000h
FDCh	Input Multiplexed Signal Configuration (IMCR359)	32	RW	0000_0000h
FE0h	Input Multiplexed Signal Configuration (IMCR360)	32	RW	0000_0000h
FE4h	Input Multiplexed Signal Configuration (IMCR361)	32	RW	0000_0000h
FE8h	Input Multiplexed Signal Configuration (IMCR362)	32	RW	0000_0000h
FECh	Input Multiplexed Signal Configuration (IMCR363)	32	RW	0000_0000h
FF0h	Input Multiplexed Signal Configuration (IMCR364)	32	RW	0000_0000h
FF4h	Input Multiplexed Signal Configuration (IMCR365)	32	RW	0000_0000h
FF8h	Input Multiplexed Signal Configuration (IMCR366)	32	RW	0000_0000h
FFCh	Input Multiplexed Signal Configuration (IMCR367)	32	RW	0000_0000h
1000h	Input Multiplexed Signal Configuration (IMCR368)	32	RW	0000_0000h
1004h	Input Multiplexed Signal Configuration (IMCR369)	32	RW	0000_0000h
1008h	Input Multiplexed Signal Configuration (IMCR370)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1014h	Input Multiplexed Signal Configuration (IMCR373)	32	RW	0000_0000h
1018h	Input Multiplexed Signal Configuration (IMCR374)	32	RW	0000_0000h
101Ch	Input Multiplexed Signal Configuration (IMCR375)	32	RW	0000_0000h
1020h	Input Multiplexed Signal Configuration (IMCR376)	32	RW	0000_0000h
1024h	Input Multiplexed Signal Configuration (IMCR377)	32	RW	0000_0000h
1028h	Input Multiplexed Signal Configuration (IMCR378)	32	RW	0000_0000h
1054h	Input Multiplexed Signal Configuration (IMCR389)	32	RW	0000_0000h
1078h	Input Multiplexed Signal Configuration (IMCR398)	32	RW	0000_0000h
107Ch	Input Multiplexed Signal Configuration (IMCR399)	32	RW	0000_0000h
10A4h	Input Multiplexed Signal Configuration (IMCR409)	32	RW	0000_0000h
10A8h	Input Multiplexed Signal Configuration (IMCR410)	32	RW	0000_0000h
10ACh	Input Multiplexed Signal Configuration (IMCR411)	32	RW	0000_0000h
10B0h	Input Multiplexed Signal Configuration (IMCR412)	32	RW	0000_0000h
10B4h	Input Multiplexed Signal Configuration (IMCR413)	32	RW	0000_0000h
10B8h	Input Multiplexed Signal Configuration (IMCR414)	32	RW	0000_0000h
10BCh	Input Multiplexed Signal Configuration (IMCR415)	32	RW	0000_0000h
10C0h	Input Multiplexed Signal Configuration (IMCR416)	32	RW	0000_0000h
10C4h	Input Multiplexed Signal Configuration (IMCR417)	32	RW	0000_0000h
10C8h	Input Multiplexed Signal Configuration (IMCR418)	32	RW	0000_0000h
1120h	Input Multiplexed Signal Configuration (IMCR440)	32	RW	0000_0000h
1140h	Input Multiplexed Signal Configuration (IMCR448)	32	RW	0000_0000h
1144h	Input Multiplexed Signal Configuration (IMCR449)	32	RW	0000_0000h
1148h	Input Multiplexed Signal Configuration (IMCR450)	32	RW	0000_0000h
114Ch	Input Multiplexed Signal Configuration (IMCR451)	32	RW	0000_0000h
1150h	Input Multiplexed Signal Configuration (IMCR452)	32	RW	0000_0000h
1154h	Input Multiplexed Signal Configuration (IMCR453)	32	RW	0000_0000h
1158h	Input Multiplexed Signal Configuration (IMCR454)	32	RW	0000_0000h
115Ch	Input Multiplexed Signal Configuration (IMCR455)	32	RW	0000_0000h
1160h	Input Multiplexed Signal Configuration (IMCR456)	32	RW	0000_0000h
1164h	Input Multiplexed Signal Configuration (IMCR457)	32	RW	0000_0000h
1168h	Input Multiplexed Signal Configuration (IMCR458)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
116Ch	Input Multiplexed Signal Configuration (IMCR459)	32	RW	0000_0000h
1170h	Input Multiplexed Signal Configuration (IMCR460)	32	RW	0000_0000h
1174h	Input Multiplexed Signal Configuration (IMCR461)	32	RW	0000_0000h
1178h	Input Multiplexed Signal Configuration (IMCR462)	32	RW	0000_0000h
117Ch	Input Multiplexed Signal Configuration (IMCR463)	32	RW	0000_0000h
1180h	Input Multiplexed Signal Configuration (IMCR464)	32	RW	0000_0000h
1184h	Input Multiplexed Signal Configuration (IMCR465)	32	RW	0000_0000h
1188h	Input Multiplexed Signal Configuration (IMCR466)	32	RW	0000_0000h
118Ch	Input Multiplexed Signal Configuration (IMCR467)	32	RW	0000_0000h
1190h	Input Multiplexed Signal Configuration (IMCR468)	32	RW	0000_0000h
1194h	Input Multiplexed Signal Configuration (IMCR469)	32	RW	0000_0000h
1198h	Input Multiplexed Signal Configuration (IMCR470)	32	RW	0000_0000h
119Ch	Input Multiplexed Signal Configuration (IMCR471)	32	RW	0000_0000h
11A0h	Input Multiplexed Signal Configuration (IMCR472)	32	RW	0000_0000h
11A4h	Input Multiplexed Signal Configuration (IMCR473)	32	RW	0000_0000h
1300h	GPIO Pad Data Output (GPDO3)	8	RW	00h
1301h	GPIO Pad Data Output (GPDO2)	8	RW	00h
1302h	GPIO Pad Data Output (GPDO1)	8	RW	00h
1303h	GPIO Pad Data Output (GPDO0)	8	RW	00h
1304h	GPIO Pad Data Output (GPDO7)	8	RW	00h
1305h	GPIO Pad Data Output (GPDO6)	8	RW	00h
1306h	GPIO Pad Data Output (GPDO5)	8	RW	00h
1307h	GPIO Pad Data Output (GPDO4)	8	RW	00h
1308h	GPIO Pad Data Output (GPDO11)	8	RW	00h
1309h	GPIO Pad Data Output (GPDO10)	8	RW	00h
130Ah	GPIO Pad Data Output (GPDO9)	8	RW	00h
130Bh	GPIO Pad Data Output (GPDO8)	8	RW	00h
130Ch	GPIO Pad Data Output (GPDO15)	8	RW	00h
130Dh	GPIO Pad Data Output (GPDO14)	8	RW	00h
130Eh	GPIO Pad Data Output (GPDO13)	8	RW	00h
130Fh	GPIO Pad Data Output (GPDO12)	8	RW	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1310h	GPIO Pad Data Output (GPDO19)	8	RW	00h
1311h	GPIO Pad Data Output (GPDO18)	8	RW	00h
1312h	GPIO Pad Data Output (GPDO17)	8	RW	00h
1313h	GPIO Pad Data Output (GPDO16)	8	RW	00h
1314h	GPIO Pad Data Output (GPDO23)	8	RW	00h
1315h	GPIO Pad Data Output (GPDO22)	8	RW	00h
1316h	GPIO Pad Data Output (GPDO21)	8	RW	00h
1317h	GPIO Pad Data Output (GPDO20)	8	RW	00h
1318h	GPIO Pad Data Output (GPDO27)	8	RW	00h
1319h	GPIO Pad Data Output (GPDO26)	8	RW	00h
131Ah	GPIO Pad Data Output (GPDO25)	8	RW	00h
131Bh	GPIO Pad Data Output (GPDO24)	8	RW	00h
131Ch	GPIO Pad Data Output (GPDO31)	8	RW	00h
131Dh	GPIO Pad Data Output (GPDO30)	8	RW	00h
131Eh	GPIO Pad Data Output (GPDO29)	8	RW	00h
131Fh	GPIO Pad Data Output (GPDO28)	8	RW	00h
1320h	GPIO Pad Data Output (GPDO35)	8	RW	00h
1321h	GPIO Pad Data Output (GPDO34)	8	RW	00h
1322h	GPIO Pad Data Output (GPDO33)	8	RW	00h
1323h	GPIO Pad Data Output (GPDO32)	8	RW	00h
1326h	GPIO Pad Data Output (GPDO37)	8	RW	00h
1327h	GPIO Pad Data Output (GPDO36)	8	RW	00h
1328h	GPIO Pad Data Output (GPDO43)	8	RW	00h
1329h	GPIO Pad Data Output (GPDO42)	8	RW	00h
132Ah	GPIO Pad Data Output (GPDO41)	8	RW	00h
132Bh	GPIO Pad Data Output (GPDO40)	8	RW	00h
132Ch	GPIO Pad Data Output (GPDO47)	8	RW	00h
132Dh	GPIO Pad Data Output (GPDO46)	8	RW	00h
132Eh	GPIO Pad Data Output (GPDO45)	8	RW	00h
132Fh	GPIO Pad Data Output (GPDO44)	8	RW	00h
1330h	GPIO Pad Data Output (GPDO51)	8	RW	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1331h	GPIO Pad Data Output (GPDO50)	8	RW	00h
1332h	GPIO Pad Data Output (GPDO49)	8	RW	00h
1333h	GPIO Pad Data Output (GPDO48)	8	RW	00h
1334h	GPIO Pad Data Output (GPDO55)	8	RW	00h
1335h	GPIO Pad Data Output (GPDO54)	8	RW	00h
1336h	GPIO Pad Data Output (GPDO53)	8	RW	00h
1337h	GPIO Pad Data Output (GPDO52)	8	RW	00h
1338h	GPIO Pad Data Output (GPDO59)	8	RW	00h
1339h	GPIO Pad Data Output (GPDO58)	8	RW	00h
133Ah	GPIO Pad Data Output (GPDO57)	8	RW	00h
133Bh	GPIO Pad Data Output (GPDO56)	8	RW	00h
133Ch	GPIO Pad Data Output (GPDO63)	8	RW	00h
133Dh	GPIO Pad Data Output (GPDO62)	8	RW	00h
133Eh	GPIO Pad Data Output (GPDO61)	8	RW	00h
133Fh	GPIO Pad Data Output (GPDO60)	8	RW	00h
1340h	GPIO Pad Data Output (GPDO67)	8	RW	00h
1341h	GPIO Pad Data Output (GPDO66)	8	RW	00h
1342h	GPIO Pad Data Output (GPDO65)	8	RW	00h
1343h	GPIO Pad Data Output (GPDO64)	8	RW	00h
1344h	GPIO Pad Data Output (GPDO71)	8	RW	00h
1345h	GPIO Pad Data Output (GPDO70)	8	RW	00h
1346h	GPIO Pad Data Output (GPDO69)	8	RW	00h
1347h	GPIO Pad Data Output (GPDO68)	8	RW	00h
1348h	GPIO Pad Data Output (GPDO75)	8	RW	00h
1349h	GPIO Pad Data Output (GPDO74)	8	RW	00h
134Ah	GPIO Pad Data Output (GPDO73)	8	RW	00h
134Bh	GPIO Pad Data Output (GPDO72)	8	RW	00h
134Ch	GPIO Pad Data Output (GPDO79)	8	RW	00h
134Dh	GPIO Pad Data Output (GPDO78)	8	RW	00h
134Eh	GPIO Pad Data Output (GPDO77)	8	RW	00h
134Fh	GPIO Pad Data Output (GPDO76)	8	RW	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1350h	GPIO Pad Data Output (GPDO83)	8	RW	00h
1351h	GPIO Pad Data Output (GPDO82)	8	RW	00h
1352h	GPIO Pad Data Output (GPDO81)	8	RW	00h
1353h	GPIO Pad Data Output (GPDO80)	8	RW	00h
1354h	GPIO Pad Data Output (GPDO87)	8	RW	00h
1355h	GPIO Pad Data Output (GPDO86)	8	RW	00h
1356h	GPIO Pad Data Output (GPDO85)	8	RW	00h
1357h	GPIO Pad Data Output (GPDO84)	8	RW	00h
1358h	GPIO Pad Data Output (GPDO91)	8	RW	00h
1359h	GPIO Pad Data Output (GPDO90)	8	RW	00h
135Ah	GPIO Pad Data Output (GPDO89)	8	RW	00h
135Bh	GPIO Pad Data Output (GPDO88)	8	RW	00h
135Ch	GPIO Pad Data Output (GPDO95)	8	RW	00h
135Dh	GPIO Pad Data Output (GPDO94)	8	RW	00h
135Eh	GPIO Pad Data Output (GPDO93)	8	RW	00h
135Fh	GPIO Pad Data Output (GPDO92)	8	RW	00h
1360h	GPIO Pad Data Output (GPDO99)	8	RW	00h
1361h	GPIO Pad Data Output (GPDO98)	8	RW	00h
1362h	GPIO Pad Data Output (GPDO97)	8	RW	00h
1363h	GPIO Pad Data Output (GPDO96)	8	RW	00h
1364h	GPIO Pad Data Output (GPDO103)	8	RW	00h
1365h	GPIO Pad Data Output (GPDO102)	8	RW	00h
1366h	GPIO Pad Data Output (GPDO101)	8	RW	00h
1367h	GPIO Pad Data Output (GPDO100)	8	RW	00h
1368h	GPIO Pad Data Output (GPDO107)	8	RW	00h
1369h	GPIO Pad Data Output (GPDO106)	8	RW	00h
136Ah	GPIO Pad Data Output (GPDO105)	8	RW	00h
136Bh	GPIO Pad Data Output (GPDO104)	8	RW	00h
136Ch	GPIO Pad Data Output (GPDO111)	8	RW	00h
136Dh	GPIO Pad Data Output (GPDO110)	8	RW	00h
136Eh	GPIO Pad Data Output (GPDO109)	8	RW	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
136Fh	GPIO Pad Data Output (GPDO108)	8	RW	00h
1370h	GPIO Pad Data Output (GPDO115)	8	RW	00h
1371h	GPIO Pad Data Output (GPDO114)	8	RW	00h
1372h	GPIO Pad Data Output (GPDO113)	8	RW	00h
1373h	GPIO Pad Data Output (GPDO112)	8	RW	00h
1374h	GPIO Pad Data Output (GPDO119)	8	RW	00h
1375h	GPIO Pad Data Output (GPDO118)	8	RW	00h
1376h	GPIO Pad Data Output (GPDO117)	8	RW	00h
1377h	GPIO Pad Data Output (GPDO116)	8	RW	00h
1378h	GPIO Pad Data Output (GPDO123)	8	RW	00h
1379h	GPIO Pad Data Output (GPDO122)	8	RW	00h
137Ah	GPIO Pad Data Output (GPDO121)	8	RW	00h
137Bh	GPIO Pad Data Output (GPDO120)	8	RW	00h
137Ch	GPIO Pad Data Output (GPDO127)	8	RW	00h
137Dh	GPIO Pad Data Output (GPDO126)	8	RW	00h
137Eh	GPIO Pad Data Output (GPDO125)	8	RW	00h
137Fh	GPIO Pad Data Output (GPDO124)	8	RW	00h
1380h	GPIO Pad Data Output (GPDO131)	8	RW	00h
1381h	GPIO Pad Data Output (GPDO130)	8	RW	00h
1382h	GPIO Pad Data Output (GPDO129)	8	RW	00h
1383h	GPIO Pad Data Output (GPDO128)	8	RW	00h
1384h	GPIO Pad Data Output (GPDO135)	8	RW	00h
1385h	GPIO Pad Data Output (GPDO134)	8	RW	00h
1386h	GPIO Pad Data Output (GPDO133)	8	RW	00h
1387h	GPIO Pad Data Output (GPDO132)	8	RW	00h
1388h	GPIO Pad Data Output (GPDO139)	8	RW	00h
1389h	GPIO Pad Data Output (GPDO138)	8	RW	00h
138Ah	GPIO Pad Data Output (GPDO137)	8	RW	00h
138Bh	GPIO Pad Data Output (GPDO136)	8	RW	00h
138Ch	GPIO Pad Data Output (GPDO143)	8	RW	00h
138Dh	GPIO Pad Data Output (GPDO142)	8	RW	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
138Fh	GPIO Pad Data Output (GPDO140)	8	RW	00h
1390h	GPIO Pad Data Output (GPDO147)	8	RW	00h
1391h	GPIO Pad Data Output (GPDO146)	8	RW	00h
1392h	GPIO Pad Data Output (GPDO145)	8	RW	00h
1393h	GPIO Pad Data Output (GPDO144)	8	RW	00h
1394h	GPIO Pad Data Output (GPDO151)	8	RW	00h
1395h	GPIO Pad Data Output (GPDO150)	8	RW	00h
1396h	GPIO Pad Data Output (GPDO149)	8	RW	00h
1397h	GPIO Pad Data Output (GPDO148)	8	RW	00h
1398h	GPIO Pad Data Output (GPDO155)	8	RW	00h
1399h	GPIO Pad Data Output (GPDO154)	8	RW	00h
139Ah	GPIO Pad Data Output (GPDO153)	8	RW	00h
139Bh	GPIO Pad Data Output (GPDO152)	8	RW	00h
139Ch	GPIO Pad Data Output (GPDO159)	8	RW	00h
139Dh	GPIO Pad Data Output (GPDO158)	8	RW	00h
139Eh	GPIO Pad Data Output (GPDO157)	8	RW	00h
139Fh	GPIO Pad Data Output (GPDO156)	8	RW	00h
13A0h	GPIO Pad Data Output (GPDO163)	8	RW	00h
13A1h	GPIO Pad Data Output (GPDO162)	8	RW	00h
13A2h	GPIO Pad Data Output (GPDO161)	8	RW	00h
13A3h	GPIO Pad Data Output (GPDO160)	8	RW	00h
13A4h	GPIO Pad Data Output (GPDO167)	8	RW	00h
13A5h	GPIO Pad Data Output (GPDO166)	8	RW	00h
13A6h	GPIO Pad Data Output (GPDO165)	8	RW	00h
13A7h	GPIO Pad Data Output (GPDO164)	8	RW	00h
13A8h	GPIO Pad Data Output (GPDO171)	8	RW	00h
13A9h	GPIO Pad Data Output (GPDO170)	8	RW	00h
13AAh	GPIO Pad Data Output (GPDO169)	8	RW	00h
13ABh	GPIO Pad Data Output (GPDO168)	8	RW	00h
13ACh	GPIO Pad Data Output (GPDO175)	8	RW	00h
13ADh	GPIO Pad Data Output (GPDO174)	8	RW	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
13AEh	GPIO Pad Data Output (GPDO173)	8	RW	00h
13AFh	GPIO Pad Data Output (GPDO172)	8	RW	00h
13B0h	GPIO Pad Data Output (GPDO179)	8	RW	00h
13B1h	GPIO Pad Data Output (GPDO178)	8	RW	00h
13B2h	GPIO Pad Data Output (GPDO177)	8	RW	00h
13B3h	GPIO Pad Data Output (GPDO176)	8	RW	00h
13B4h	GPIO Pad Data Output (GPDO183)	8	RW	00h
13B5h	GPIO Pad Data Output (GPDO182)	8	RW	00h
13B6h	GPIO Pad Data Output (GPDO181)	8	RW	00h
13B7h	GPIO Pad Data Output (GPDO180)	8	RW	00h
13B8h	GPIO Pad Data Output (GPDO187)	8	RW	00h
13B9h	GPIO Pad Data Output (GPDO186)	8	RW	00h
13BAh	GPIO Pad Data Output (GPDO185)	8	RW	00h
13BBh	GPIO Pad Data Output (GPDO184)	8	RW	00h
13BCh	GPIO Pad Data Output (GPDO191)	8	RW	00h
13BDh	GPIO Pad Data Output (GPDO190)	8	RW	00h
13BEh	GPIO Pad Data Output (GPDO189)	8	RW	00h
13BFh	GPIO Pad Data Output (GPDO188)	8	RW	00h
13C0h	GPIO Pad Data Output (GPDO195)	8	RW	00h
13C1h	GPIO Pad Data Output (GPDO194)	8	RW	00h
13C2h	GPIO Pad Data Output (GPDO193)	8	RW	00h
13C3h	GPIO Pad Data Output (GPDO192)	8	RW	00h
13C4h	GPIO Pad Data Output (GPDO199)	8	RW	00h
13C5h	GPIO Pad Data Output (GPDO198)	8	RW	00h
13C6h	GPIO Pad Data Output (GPDO197)	8	RW	00h
13C7h	GPIO Pad Data Output (GPDO196)	8	RW	00h
13C8h	GPIO Pad Data Output (GPDO203)	8	RW	00h
13C9h	GPIO Pad Data Output (GPDO202)	8	RW	00h
13CAh	GPIO Pad Data Output (GPDO201)	8	RW	00h
13CBh	GPIO Pad Data Output (GPDO200)	8	RW	00h
13CCh	GPIO Pad Data Output (GPDO207)	8	RW	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
13CDh	GPIO Pad Data Output (GPDO206)	8	RW	00h
13CEh	GPIO Pad Data Output (GPDO205)	8	RW	00h
13CFh	GPIO Pad Data Output (GPDO204)	8	RW	00h
13D0h	GPIO Pad Data Output (GPDO211)	8	RW	00h
13D1h	GPIO Pad Data Output (GPDO210)	8	RW	00h
13D2h	GPIO Pad Data Output (GPDO209)	8	RW	00h
13D3h	GPIO Pad Data Output (GPDO208)	8	RW	00h
13D4h	GPIO Pad Data Output (GPDO215)	8	RW	00h
13D5h	GPIO Pad Data Output (GPDO214)	8	RW	00h
13D6h	GPIO Pad Data Output (GPDO213)	8	RW	00h
13D7h	GPIO Pad Data Output (GPDO212)	8	RW	00h
13D8h	GPIO Pad Data Output (GPDO219)	8	RW	00h
13D9h	GPIO Pad Data Output (GPDO218)	8	RW	00h
13DAh	GPIO Pad Data Output (GPDO217)	8	RW	00h
13DBh	GPIO Pad Data Output (GPDO216)	8	RW	00h
13DCh	GPIO Pad Data Output (GPDO223)	8	RW	00h
13DDh	GPIO Pad Data Output (GPDO222)	8	RW	00h
13DEh	GPIO Pad Data Output (GPDO221)	8	RW	00h
13DFh	GPIO Pad Data Output (GPDO220)	8	RW	00h
13E0h	GPIO Pad Data Output (GPDO227)	8	RW	00h
13E1h	GPIO Pad Data Output (GPDO226)	8	RW	00h
13E2h	GPIO Pad Data Output (GPDO225)	8	RW	00h
13E3h	GPIO Pad Data Output (GPDO224)	8	RW	00h
13E4h	GPIO Pad Data Output (GPDO231)	8	RW	00h
13E5h	GPIO Pad Data Output (GPDO230)	8	RW	00h
13E6h	GPIO Pad Data Output (GPDO229)	8	RW	00h
13E7h	GPIO Pad Data Output (GPDO228)	8	RW	00h
13E8h	GPIO Pad Data Output (GPDO235)	8	RW	00h
13E9h	GPIO Pad Data Output (GPDO234)	8	RW	00h
13EAh	GPIO Pad Data Output (GPDO233)	8	RW	00h
13EBh	GPIO Pad Data Output (GPDO232)	8	RW	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
13ECh	GPIO Pad Data Output (GPDO239)	8	RW	00h
13EDh	GPIO Pad Data Output (GPDO238)	8	RW	00h
13EEh	GPIO Pad Data Output (GPDO237)	8	RW	00h
13EFh	GPIO Pad Data Output (GPDO236)	8	RW	00h
13F0h	GPIO Pad Data Output (GPDO243)	8	RW	00h
13F1h	GPIO Pad Data Output (GPDO242)	8	RW	00h
13F2h	GPIO Pad Data Output (GPDO241)	8	RW	00h
13F3h	GPIO Pad Data Output (GPDO240)	8	RW	00h
13F4h	GPIO Pad Data Output (GPDO247)	8	RW	00h
13F5h	GPIO Pad Data Output (GPDO246)	8	RW	00h
13F6h	GPIO Pad Data Output (GPDO245)	8	RW	00h
13F7h	GPIO Pad Data Output (GPDO244)	8	RW	00h
13F8h	GPIO Pad Data Output (GPDO251)	8	RW	00h
13F9h	GPIO Pad Data Output (GPDO250)	8	RW	00h
13FAh	GPIO Pad Data Output (GPDO249)	8	RW	00h
13FBh	GPIO Pad Data Output (GPDO248)	8	RW	00h
13FCh	GPIO Pad Data Output (GPDO255)	8	RW	00h
13FDh	GPIO Pad Data Output (GPDO254)	8	RW	00h
13FEh	GPIO Pad Data Output (GPDO253)	8	RW	00h
13FFh	GPIO Pad Data Output (GPDO252)	8	RW	00h
1400h	GPIO Pad Data Output (GPDO259)	8	RW	00h
1401h	GPIO Pad Data Output (GPDO258)	8	RW	00h
1402h	GPIO Pad Data Output (GPDO257)	8	RW	00h
1403h	GPIO Pad Data Output (GPDO256)	8	RW	00h
1404h	GPIO Pad Data Output (GPDO263)	8	RW	00h
1405h	GPIO Pad Data Output (GPDO262)	8	RW	00h
1406h	GPIO Pad Data Output (GPDO261)	8	RW	00h
1407h	GPIO Pad Data Output (GPDO260)	8	RW	00h
1408h	GPIO Pad Data Output (GPDO267)	8	RW	00h
1409h	GPIO Pad Data Output (GPDO266)	8	RW	00h
140Ah	GPIO Pad Data Output (GPDO265)	8	RW	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
140Bh	GPIO Pad Data Output (GPDO264)	8	RW	00h
140Ch	GPIO Pad Data Output (GPDO271)	8	RW	00h
140Dh	GPIO Pad Data Output (GPDO270)	8	RW	00h
140Eh	GPIO Pad Data Output (GPDO269)	8	RW	00h
140Fh	GPIO Pad Data Output (GPDO268)	8	RW	00h
1410h	GPIO Pad Data Output (GPDO275)	8	RW	00h
1411h	GPIO Pad Data Output (GPDO274)	8	RW	00h
1412h	GPIO Pad Data Output (GPDO273)	8	RW	00h
1413h	GPIO Pad Data Output (GPDO272)	8	RW	00h
1414h	GPIO Pad Data Output (GPDO279)	8	RW	00h
1415h	GPIO Pad Data Output (GPDO278)	8	RW	00h
1416h	GPIO Pad Data Output (GPDO277)	8	RW	00h
1417h	GPIO Pad Data Output (GPDO276)	8	RW	00h
1418h	GPIO Pad Data Output (GPDO283)	8	RW	00h
1419h	GPIO Pad Data Output (GPDO282)	8	RW	00h
141Ah	GPIO Pad Data Output (GPDO281)	8	RW	00h
141Bh	GPIO Pad Data Output (GPDO280)	8	RW	00h
141Ch	GPIO Pad Data Output (GPDO287)	8	RW	00h
141Dh	GPIO Pad Data Output (GPDO286)	8	RW	00h
141Eh	GPIO Pad Data Output (GPDO285)	8	RW	00h
141Fh	GPIO Pad Data Output (GPDO284)	8	RW	00h
1420h	GPIO Pad Data Output (GPDO291)	8	RW	00h
1421h	GPIO Pad Data Output (GPDO290)	8	RW	00h
1422h	GPIO Pad Data Output (GPDO289)	8	RW	00h
1423h	GPIO Pad Data Output (GPDO288)	8	RW	00h
1424h	GPIO Pad Data Output (GPDO295)	8	RW	00h
1425h	GPIO Pad Data Output (GPDO294)	8	RW	00h
1426h	GPIO Pad Data Output (GPDO293)	8	RW	00h
1427h	GPIO Pad Data Output (GPDO292)	8	RW	00h
1428h	GPIO Pad Data Output (GPDO299)	8	RW	00h
1429h	GPIO Pad Data Output (GPDO298)	8	RW	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
142Ah	GPIO Pad Data Output (GPDO297)	8	RW	00h
142Bh	GPIO Pad Data Output (GPDO296)	8	RW	00h
142Ch	GPIO Pad Data Output (GPDO303)	8	RW	00h
142Dh	GPIO Pad Data Output (GPDO302)	8	RW	00h
142Eh	GPIO Pad Data Output (GPDO301)	8	RW	00h
142Fh	GPIO Pad Data Output (GPDO300)	8	RW	00h
1430h	GPIO Pad Data Output (GPDO307)	8	RW	00h
1431h	GPIO Pad Data Output (GPDO306)	8	RW	00h
1432h	GPIO Pad Data Output (GPDO305)	8	RW	00h
1433h	GPIO Pad Data Output (GPDO304)	8	RW	00h
1434h	GPIO Pad Data Output (GPDO311)	8	RW	00h
1435h	GPIO Pad Data Output (GPDO310)	8	RW	00h
1436h	GPIO Pad Data Output (GPDO309)	8	RW	00h
1437h	GPIO Pad Data Output (GPDO308)	8	RW	00h
1438h	GPIO Pad Data Output (GPDO315)	8	RW	00h
1439h	GPIO Pad Data Output (GPDO314)	8	RW	00h
143Ah	GPIO Pad Data Output (GPDO313)	8	RW	00h
143Bh	GPIO Pad Data Output (GPDO312)	8	RW	00h
143Ch	GPIO Pad Data Output (GPDO319)	8	RW	00h
143Dh	GPIO Pad Data Output (GPDO318)	8	RW	00h
143Eh	GPIO Pad Data Output (GPDO317)	8	RW	00h
143Fh	GPIO Pad Data Output (GPDO316)	8	RW	00h
1440h	GPIO Pad Data Output (GPDO323)	8	RW	00h
1441h	GPIO Pad Data Output (GPDO322)	8	RW	00h
1442h	GPIO Pad Data Output (GPDO321)	8	RW	00h
1443h	GPIO Pad Data Output (GPDO320)	8	RW	00h
1500h	GPIO Pad Data Input (GPDI3)	8	R	00h
1501h	GPIO Pad Data Input (GPDI2)	8	R	00h
1502h	GPIO Pad Data Input (GPDI1)	8	R	00h
1503h	GPIO Pad Data Input (GPDI0)	8	R	00h
1504h	GPIO Pad Data Input (GPDI7)	8	R	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1505h	GPIO Pad Data Input (GPDI6)	8	R	00h
1506h	GPIO Pad Data Input (GPDI5)	8	R	00h
1507h	GPIO Pad Data Input (GPDI4)	8	R	00h
1508h	GPIO Pad Data Input (GPDI11)	8	R	00h
1509h	GPIO Pad Data Input (GPDI10)	8	R	00h
150Ah	GPIO Pad Data Input (GPDI9)	8	R	00h
150Bh	GPIO Pad Data Input (GPDI8)	8	R	00h
150Ch	GPIO Pad Data Input (GPDI15)	8	R	00h
150Dh	GPIO Pad Data Input (GPDI14)	8	R	00h
150Eh	GPIO Pad Data Input (GPDI13)	8	R	00h
150Fh	GPIO Pad Data Input (GPDI12)	8	R	00h
1510h	GPIO Pad Data Input (GPDI19)	8	R	00h
1511h	GPIO Pad Data Input (GPDI18)	8	R	00h
1512h	GPIO Pad Data Input (GPDI17)	8	R	00h
1513h	GPIO Pad Data Input (GPDI16)	8	R	00h
1514h	GPIO Pad Data Input (GPDI23)	8	R	00h
1515h	GPIO Pad Data Input (GPDI22)	8	R	00h
1516h	GPIO Pad Data Input (GPDI21)	8	R	00h
1517h	GPIO Pad Data Input (GPDI20)	8	R	00h
1518h	GPIO Pad Data Input (GPDI27)	8	R	00h
1519h	GPIO Pad Data Input (GPDI26)	8	R	00h
151Ah	GPIO Pad Data Input (GPDI25)	8	R	00h
151Bh	GPIO Pad Data Input (GPDI24)	8	R	00h
151Ch	GPIO Pad Data Input (GPDI31)	8	R	00h
151Dh	GPIO Pad Data Input (GPDI30)	8	R	00h
151Eh	GPIO Pad Data Input (GPDI29)	8	R	00h
151Fh	GPIO Pad Data Input (GPDI28)	8	R	00h
1520h	GPIO Pad Data Input (GPDI35)	8	R	00h
1521h	GPIO Pad Data Input (GPDI34)	8	R	00h
1522h	GPIO Pad Data Input (GPDI33)	8	R	00h
1523h	GPIO Pad Data Input (GPDI32)	8	R	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1526h	GPIO Pad Data Input (GPDI37)	8	R	00h
1527h	GPIO Pad Data Input (GPDI36)	8	R	00h
1528h	GPIO Pad Data Input (GPDI43)	8	R	00h
1529h	GPIO Pad Data Input (GPDI42)	8	R	00h
152Ah	GPIO Pad Data Input (GPDI41)	8	R	00h
152Bh	GPIO Pad Data Input (GPDI40)	8	R	00h
152Ch	GPIO Pad Data Input (GPDI47)	8	R	00h
152Dh	GPIO Pad Data Input (GPDI46)	8	R	00h
152Eh	GPIO Pad Data Input (GPDI45)	8	R	00h
152Fh	GPIO Pad Data Input (GPDI44)	8	R	00h
1530h	GPIO Pad Data Input (GPDI51)	8	R	00h
1531h	GPIO Pad Data Input (GPDI50)	8	R	00h
1532h	GPIO Pad Data Input (GPDI49)	8	R	00h
1533h	GPIO Pad Data Input (GPDI48)	8	R	00h
1534h	GPIO Pad Data Input (GPDI55)	8	R	00h
1535h	GPIO Pad Data Input (GPDI54)	8	R	00h
1536h	GPIO Pad Data Input (GPDI53)	8	R	00h
1537h	GPIO Pad Data Input (GPDI52)	8	R	00h
1538h	GPIO Pad Data Input (GPDI59)	8	R	00h
1539h	GPIO Pad Data Input (GPDI58)	8	R	00h
153Ah	GPIO Pad Data Input (GPDI57)	8	R	00h
153Bh	GPIO Pad Data Input (GPDI56)	8	R	00h
153Ch	GPIO Pad Data Input (GPDI63)	8	R	00h
153Dh	GPIO Pad Data Input (GPDI62)	8	R	00h
153Eh	GPIO Pad Data Input (GPDI61)	8	R	00h
153Fh	GPIO Pad Data Input (GPDI60)	8	R	00h
1540h	GPIO Pad Data Input (GPDI67)	8	R	00h
1541h	GPIO Pad Data Input (GPDI66)	8	R	00h
1542h	GPIO Pad Data Input (GPDI65)	8	R	00h
1543h	GPIO Pad Data Input (GPDI64)	8	R	00h
1544h	GPIO Pad Data Input (GPDI71)	8	R	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1545h	GPIO Pad Data Input (GPDI70)	8	R	00h
1546h	GPIO Pad Data Input (GPDI69)	8	R	00h
1547h	GPIO Pad Data Input (GPDI68)	8	R	00h
1548h	GPIO Pad Data Input (GPDI75)	8	R	00h
1549h	GPIO Pad Data Input (GPDI74)	8	R	00h
154Ah	GPIO Pad Data Input (GPDI73)	8	R	00h
154Bh	GPIO Pad Data Input (GPDI72)	8	R	00h
154Ch	GPIO Pad Data Input (GPDI79)	8	R	00h
154Dh	GPIO Pad Data Input (GPDI78)	8	R	00h
154Eh	GPIO Pad Data Input (GPDI77)	8	R	00h
154Fh	GPIO Pad Data Input (GPDI76)	8	R	00h
1550h	GPIO Pad Data Input (GPDI83)	8	R	00h
1551h	GPIO Pad Data Input (GPDI82)	8	R	00h
1552h	GPIO Pad Data Input (GPDI81)	8	R	00h
1553h	GPIO Pad Data Input (GPDI80)	8	R	00h
1554h	GPIO Pad Data Input (GPDI87)	8	R	00h
1555h	GPIO Pad Data Input (GPDI86)	8	R	00h
1556h	GPIO Pad Data Input (GPDI85)	8	R	00h
1557h	GPIO Pad Data Input (GPDI84)	8	R	00h
1558h	GPIO Pad Data Input (GPDI91)	8	R	00h
1559h	GPIO Pad Data Input (GPDI90)	8	R	00h
155Ah	GPIO Pad Data Input (GPDI89)	8	R	00h
155Bh	GPIO Pad Data Input (GPDI88)	8	R	00h
155Ch	GPIO Pad Data Input (GPDI95)	8	R	00h
155Dh	GPIO Pad Data Input (GPDI94)	8	R	00h
155Eh	GPIO Pad Data Input (GPDI93)	8	R	00h
155Fh	GPIO Pad Data Input (GPDI92)	8	R	00h
1560h	GPIO Pad Data Input (GPDI99)	8	R	00h
1561h	GPIO Pad Data Input (GPDI98)	8	R	00h
1562h	GPIO Pad Data Input (GPDI97)	8	R	00h
1563h	GPIO Pad Data Input (GPDI96)	8	R	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1564h	GPIO Pad Data Input (GPDI103)	8	R	00h
1565h	GPIO Pad Data Input (GPDI102)	8	R	00h
1566h	GPIO Pad Data Input (GPDI101)	8	R	00h
1567h	GPIO Pad Data Input (GPDI100)	8	R	00h
1568h	GPIO Pad Data Input (GPDI107)	8	R	00h
1569h	GPIO Pad Data Input (GPDI106)	8	R	00h
156Ah	GPIO Pad Data Input (GPDI105)	8	R	00h
156Bh	GPIO Pad Data Input (GPDI104)	8	R	00h
156Ch	GPIO Pad Data Input (GPDI111)	8	R	00h
156Dh	GPIO Pad Data Input (GPDI110)	8	R	00h
156Eh	GPIO Pad Data Input (GPDI109)	8	R	00h
156Fh	GPIO Pad Data Input (GPDI108)	8	R	00h
1570h	GPIO Pad Data Input (GPDI115)	8	R	00h
1571h	GPIO Pad Data Input (GPDI114)	8	R	00h
1572h	GPIO Pad Data Input (GPDI113)	8	R	00h
1573h	GPIO Pad Data Input (GPDI112)	8	R	00h
1574h	GPIO Pad Data Input (GPDI119)	8	R	00h
1575h	GPIO Pad Data Input (GPDI118)	8	R	00h
1576h	GPIO Pad Data Input (GPDI117)	8	R	00h
1577h	GPIO Pad Data Input (GPDI116)	8	R	00h
1578h	GPIO Pad Data Input (GPDI123)	8	R	00h
1579h	GPIO Pad Data Input (GPDI122)	8	R	00h
157Ah	GPIO Pad Data Input (GPDI121)	8	R	00h
157Bh	GPIO Pad Data Input (GPDI120)	8	R	00h
157Ch	GPIO Pad Data Input (GPDI127)	8	R	00h
157Dh	GPIO Pad Data Input (GPDI126)	8	R	00h
157Eh	GPIO Pad Data Input (GPDI125)	8	R	00h
157Fh	GPIO Pad Data Input (GPDI124)	8	R	00h
1580h	GPIO Pad Data Input (GPDI131)	8	R	00h
1581h	GPIO Pad Data Input (GPDI130)	8	R	00h
1582h	GPIO Pad Data Input (GPDI129)	8	R	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1583h	GPIO Pad Data Input (GPDI128)	8	R	00h
1584h	GPIO Pad Data Input (GPDI135)	8	R	00h
1585h	GPIO Pad Data Input (GPDI134)	8	R	00h
1586h	GPIO Pad Data Input (GPDI133)	8	R	00h
1587h	GPIO Pad Data Input (GPDI132)	8	R	00h
1588h	GPIO Pad Data Input (GPDI139)	8	R	00h
1589h	GPIO Pad Data Input (GPDI138)	8	R	00h
158Ah	GPIO Pad Data Input (GPDI137)	8	R	00h
158Bh	GPIO Pad Data Input (GPDI136)	8	R	00h
158Ch	GPIO Pad Data Input (GPDI143)	8	R	00h
158Dh	GPIO Pad Data Input (GPDI142)	8	R	00h
158Fh	GPIO Pad Data Input (GPDI140)	8	R	00h
1590h	GPIO Pad Data Input (GPDI147)	8	R	00h
1591h	GPIO Pad Data Input (GPDI146)	8	R	00h
1592h	GPIO Pad Data Input (GPDI145)	8	R	00h
1593h	GPIO Pad Data Input (GPDI144)	8	R	00h
1594h	GPIO Pad Data Input (GPDI151)	8	R	00h
1595h	GPIO Pad Data Input (GPDI150)	8	R	00h
1596h	GPIO Pad Data Input (GPDI149)	8	R	00h
1597h	GPIO Pad Data Input (GPDI148)	8	R	00h
1598h	GPIO Pad Data Input (GPDI155)	8	R	00h
1599h	GPIO Pad Data Input (GPDI154)	8	R	00h
159Ah	GPIO Pad Data Input (GPDI153)	8	R	00h
159Bh	GPIO Pad Data Input (GPDI152)	8	R	00h
159Ch	GPIO Pad Data Input (GPDI159)	8	R	00h
159Dh	GPIO Pad Data Input (GPDI158)	8	R	00h
159Eh	GPIO Pad Data Input (GPDI157)	8	R	00h
159Fh	GPIO Pad Data Input (GPDI156)	8	R	00h
15A0h	GPIO Pad Data Input (GPDI163)	8	R	00h
15A1h	GPIO Pad Data Input (GPDI162)	8	R	00h
15A2h	GPIO Pad Data Input (GPDI161)	8	R	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
15A3h	GPIO Pad Data Input (GPDI160)	8	R	00h
15A4h	GPIO Pad Data Input (GPDI167)	8	R	00h
15A5h	GPIO Pad Data Input (GPDI166)	8	R	00h
15A6h	GPIO Pad Data Input (GPDI165)	8	R	00h
15A7h	GPIO Pad Data Input (GPDI164)	8	R	00h
15A8h	GPIO Pad Data Input (GPDI171)	8	R	00h
15A9h	GPIO Pad Data Input (GPDI170)	8	R	00h
15AAh	GPIO Pad Data Input (GPDI169)	8	R	00h
15ABh	GPIO Pad Data Input (GPDI168)	8	R	00h
15ACh	GPIO Pad Data Input (GPDI175)	8	R	00h
15ADh	GPIO Pad Data Input (GPDI174)	8	R	00h
15AEh	GPIO Pad Data Input (GPDI173)	8	R	00h
15AFh	GPIO Pad Data Input (GPDI172)	8	R	00h
15B0h	GPIO Pad Data Input (GPDI179)	8	R	00h
15B1h	GPIO Pad Data Input (GPDI178)	8	R	00h
15B2h	GPIO Pad Data Input (GPDI177)	8	R	00h
15B3h	GPIO Pad Data Input (GPDI176)	8	R	00h
15B4h	GPIO Pad Data Input (GPDI183)	8	R	00h
15B5h	GPIO Pad Data Input (GPDI182)	8	R	00h
15B6h	GPIO Pad Data Input (GPDI181)	8	R	00h
15B7h	GPIO Pad Data Input (GPDI180)	8	R	00h
15B8h	GPIO Pad Data Input (GPDI187)	8	R	00h
15B9h	GPIO Pad Data Input (GPDI186)	8	R	00h
15BAh	GPIO Pad Data Input (GPDI185)	8	R	00h
15BBh	GPIO Pad Data Input (GPDI184)	8	R	00h
15BCh	GPIO Pad Data Input (GPDI191)	8	R	00h
15BDh	GPIO Pad Data Input (GPDI190)	8	R	00h
15BEh	GPIO Pad Data Input (GPDI189)	8	R	00h
15BFh	GPIO Pad Data Input (GPDI188)	8	R	00h
15C0h	GPIO Pad Data Input (GPDI195)	8	R	00h
15C1h	GPIO Pad Data Input (GPDI194)	8	R	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
15C2h	GPIO Pad Data Input (GPDI193)	8	R	00h
15C3h	GPIO Pad Data Input (GPDI192)	8	R	00h
15C4h	GPIO Pad Data Input (GPDI199)	8	R	00h
15C5h	GPIO Pad Data Input (GPDI198)	8	R	00h
15C6h	GPIO Pad Data Input (GPDI197)	8	R	00h
15C7h	GPIO Pad Data Input (GPDI196)	8	R	00h
15C8h	GPIO Pad Data Input (GPDI203)	8	R	00h
15C9h	GPIO Pad Data Input (GPDI202)	8	R	00h
15CAh	GPIO Pad Data Input (GPDI201)	8	R	00h
15CBh	GPIO Pad Data Input (GPDI200)	8	R	00h
15CCh	GPIO Pad Data Input (GPDI207)	8	R	00h
15CDh	GPIO Pad Data Input (GPDI206)	8	R	00h
15CEh	GPIO Pad Data Input (GPDI205)	8	R	00h
15CFh	GPIO Pad Data Input (GPDI204)	8	R	00h
15D0h	GPIO Pad Data Input (GPDI211)	8	R	00h
15D1h	GPIO Pad Data Input (GPDI210)	8	R	00h
15D2h	GPIO Pad Data Input (GPDI209)	8	R	00h
15D3h	GPIO Pad Data Input (GPDI208)	8	R	00h
15D4h	GPIO Pad Data Input (GPDI215)	8	R	00h
15D5h	GPIO Pad Data Input (GPDI214)	8	R	00h
15D6h	GPIO Pad Data Input (GPDI213)	8	R	00h
15D7h	GPIO Pad Data Input (GPDI212)	8	R	00h
15D8h	GPIO Pad Data Input (GPDI219)	8	R	00h
15D9h	GPIO Pad Data Input (GPDI218)	8	R	00h
15DAh	GPIO Pad Data Input (GPDI217)	8	R	00h
15DBh	GPIO Pad Data Input (GPDI216)	8	R	00h
15DCh	GPIO Pad Data Input (GPDI223)	8	R	00h
15DDh	GPIO Pad Data Input (GPDI222)	8	R	00h
15DEh	GPIO Pad Data Input (GPDI221)	8	R	00h
15DFh	GPIO Pad Data Input (GPDI220)	8	R	00h
15E0h	GPIO Pad Data Input (GPDI227)	8	R	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
15E1h	GPIO Pad Data Input (GPDI226)	8	R	00h
15E2h	GPIO Pad Data Input (GPDI225)	8	R	00h
15E3h	GPIO Pad Data Input (GPDI224)	8	R	00h
15E4h	GPIO Pad Data Input (GPDI231)	8	R	00h
15E5h	GPIO Pad Data Input (GPDI230)	8	R	00h
15E6h	GPIO Pad Data Input (GPDI229)	8	R	00h
15E7h	GPIO Pad Data Input (GPDI228)	8	R	00h
15E8h	GPIO Pad Data Input (GPDI235)	8	R	00h
15E9h	GPIO Pad Data Input (GPDI234)	8	R	00h
15EAh	GPIO Pad Data Input (GPDI233)	8	R	00h
15EBh	GPIO Pad Data Input (GPDI232)	8	R	00h
15ECh	GPIO Pad Data Input (GPDI239)	8	R	00h
15EDh	GPIO Pad Data Input (GPDI238)	8	R	00h
15EEh	GPIO Pad Data Input (GPDI237)	8	R	00h
15EFh	GPIO Pad Data Input (GPDI236)	8	R	00h
15F0h	GPIO Pad Data Input (GPDI243)	8	R	00h
15F1h	GPIO Pad Data Input (GPDI242)	8	R	00h
15F2h	GPIO Pad Data Input (GPDI241)	8	R	00h
15F3h	GPIO Pad Data Input (GPDI240)	8	R	00h
15F4h	GPIO Pad Data Input (GPDI247)	8	R	00h
15F5h	GPIO Pad Data Input (GPDI246)	8	R	00h
15F6h	GPIO Pad Data Input (GPDI245)	8	R	00h
15F7h	GPIO Pad Data Input (GPDI244)	8	R	00h
15F8h	GPIO Pad Data Input (GPDI251)	8	R	00h
15F9h	GPIO Pad Data Input (GPDI250)	8	R	00h
15FAh	GPIO Pad Data Input (GPDI249)	8	R	00h
15FBh	GPIO Pad Data Input (GPDI248)	8	R	00h
15FCh	GPIO Pad Data Input (GPDI255)	8	R	00h
15FDh	GPIO Pad Data Input (GPDI254)	8	R	00h
15FEh	GPIO Pad Data Input (GPDI253)	8	R	00h
15FFh	GPIO Pad Data Input (GPDI252)	8	R	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1600h	GPIO Pad Data Input (GPDI259)	8	R	00h
1601h	GPIO Pad Data Input (GPDI258)	8	R	00h
1602h	GPIO Pad Data Input (GPDI257)	8	R	00h
1603h	GPIO Pad Data Input (GPDI256)	8	R	00h
1604h	GPIO Pad Data Input (GPDI263)	8	R	00h
1605h	GPIO Pad Data Input (GPDI262)	8	R	00h
1606h	GPIO Pad Data Input (GPDI261)	8	R	00h
1607h	GPIO Pad Data Input (GPDI260)	8	R	00h
1608h	GPIO Pad Data Input (GPDI267)	8	R	00h
1609h	GPIO Pad Data Input (GPDI266)	8	R	00h
160Ah	GPIO Pad Data Input (GPDI265)	8	R	00h
160Bh	GPIO Pad Data Input (GPDI264)	8	R	00h
160Ch	GPIO Pad Data Input (GPDI271)	8	R	00h
160Dh	GPIO Pad Data Input (GPDI270)	8	R	00h
160Eh	GPIO Pad Data Input (GPDI269)	8	R	00h
160Fh	GPIO Pad Data Input (GPDI268)	8	R	00h
1610h	GPIO Pad Data Input (GPDI275)	8	R	00h
1611h	GPIO Pad Data Input (GPDI274)	8	R	00h
1612h	GPIO Pad Data Input (GPDI273)	8	R	00h
1613h	GPIO Pad Data Input (GPDI272)	8	R	00h
1614h	GPIO Pad Data Input (GPDI279)	8	R	00h
1615h	GPIO Pad Data Input (GPDI278)	8	R	00h
1616h	GPIO Pad Data Input (GPDI277)	8	R	00h
1617h	GPIO Pad Data Input (GPDI276)	8	R	00h
1618h	GPIO Pad Data Input (GPDI283)	8	R	00h
1619h	GPIO Pad Data Input (GPDI282)	8	R	00h
161Ah	GPIO Pad Data Input (GPDI281)	8	R	00h
161Bh	GPIO Pad Data Input (GPDI280)	8	R	00h
161Ch	GPIO Pad Data Input (GPDI287)	8	R	00h
161Dh	GPIO Pad Data Input (GPDI286)	8	R	00h
161Eh	GPIO Pad Data Input (GPDI285)	8	R	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
161Fh	GPIO Pad Data Input (GPDI284)	8	R	00h
1620h	GPIO Pad Data Input (GPDI291)	8	R	00h
1621h	GPIO Pad Data Input (GPDI290)	8	R	00h
1622h	GPIO Pad Data Input (GPDI289)	8	R	00h
1623h	GPIO Pad Data Input (GPDI288)	8	R	00h
1624h	GPIO Pad Data Input (GPDI295)	8	R	00h
1625h	GPIO Pad Data Input (GPDI294)	8	R	00h
1626h	GPIO Pad Data Input (GPDI293)	8	R	00h
1627h	GPIO Pad Data Input (GPDI292)	8	R	00h
1628h	GPIO Pad Data Input (GPDI299)	8	R	00h
1629h	GPIO Pad Data Input (GPDI298)	8	R	00h
162Ah	GPIO Pad Data Input (GPDI297)	8	R	00h
162Bh	GPIO Pad Data Input (GPDI296)	8	R	00h
162Ch	GPIO Pad Data Input (GPDI303)	8	R	00h
162Dh	GPIO Pad Data Input (GPDI302)	8	R	00h
162Eh	GPIO Pad Data Input (GPDI301)	8	R	00h
162Fh	GPIO Pad Data Input (GPDI300)	8	R	00h
1630h	GPIO Pad Data Input (GPDI307)	8	R	00h
1631h	GPIO Pad Data Input (GPDI306)	8	R	00h
1632h	GPIO Pad Data Input (GPDI305)	8	R	00h
1633h	GPIO Pad Data Input (GPDI304)	8	R	00h
1634h	GPIO Pad Data Input (GPDI311)	8	R	00h
1635h	GPIO Pad Data Input (GPDI310)	8	R	00h
1636h	GPIO Pad Data Input (GPDI309)	8	R	00h
1637h	GPIO Pad Data Input (GPDI308)	8	R	00h
1638h	GPIO Pad Data Input (GPDI315)	8	R	00h
1639h	GPIO Pad Data Input (GPDI314)	8	R	00h
163Ah	GPIO Pad Data Input (GPDI313)	8	R	00h
163Bh	GPIO Pad Data Input (GPDI312)	8	R	00h
163Ch	GPIO Pad Data Input (GPDI319)	8	R	00h
163Dh	GPIO Pad Data Input (GPDI318)	8	R	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
163Eh	GPIO Pad Data Input (GPDI317)	8	R	00h
163Fh	GPIO Pad Data Input (GPDI316)	8	R	00h
1640h	GPIO Pad Data Input (GPDI323)	8	R	00h
1641h	GPIO Pad Data Input (GPDI322)	8	R	00h
1642h	GPIO Pad Data Input (GPDI321)	8	R	00h
1643h	GPIO Pad Data Input (GPDI320)	8	R	00h
1700h - 1704h	Parallel GPIO Pad Data Output (PGPDO0 - PGPDO3) [1]	16	RW	0000h
1706h	Parallel GPIO Pad Data Output (PGPDO2)	16	RW	0000h
1708h - 1710h	Parallel GPIO Pad Data Output (PGPDO4 - PGPDO9) [1]	16	RW	0000h
1712h	Parallel GPIO Pad Data Output (PGPDO8)	16	RW	0000h
1714h - 1726h	Parallel GPIO Pad Data Output (PGPDO10 - PGPDO19) [1]	16	RW	0000h
1740h - 1744h	Parallel GPIO Pad Data Input (PGPDI0 - PGPDI3) [1]	16	R	0000h
1746h	Parallel GPIO Pad Data Input (PGPDI2)	16	R	0000h
1748h - 1750h	Parallel GPIO Pad Data Input (PGPDI4 - PGPDI9) [1]	16	R	0000h
1752h	Parallel GPIO Pad Data Input (PGPDI8)	16	R	0000h
1754h - 1766h	Parallel GPIO Pad Data Input (PGPDI10 - PGPDI19) [1]	16	R	0000h
1780h - 1784h	Masked Parallel GPIO Pad Data Output (MPGPDO0 - MPGPDO1)	32	RW	0000_0000h
1788h	Masked Parallel GPIO Pad Data Output (MPGPDO2)	32	RW	0000_0000h
178Ch - 179Ch	Masked Parallel GPIO Pad Data Output (MPGPDO3 - MPGPDO7)	32	RW	0000_0000h
17A0h	Masked Parallel GPIO Pad Data Output (MPGPDO8)	32	RW	0000_0000h
17A4h - 17CCh	Masked Parallel GPIO Pad Data Output (MPGPDO9 - MPGPDO19)	32	RW	0000_0000h

1. In this array, the index and offset values of the registers do not increment in direct alignment. For details, see the register description.

10.6.2 SIUL2 MCU ID Register #1 (MIDR1)

Offset

Register	Offset
MIDR1	4h

Function

This register holds identification information about the device.

NOTE

This register supports only 32-bit accesses. Byte and half-word accesses are not supported.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PRODUCT_LINE_LETTER								PART_NO							
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								MAJOR_MASK				MINOR_MASK			
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Fields

Field	Function
31-26 PRODUCT_LINE_LETTER	Product Line Letter Identified the ASCII character in MCU Part Number. This field specifies the part number suffix, and needs to be combined with MIDR1[PART_NO] to provide the full chip number. 0x0B K This value is set at the factory and cannot be changed. All other values are reserved.
25-16 PART_NO	MCU Part Number S32K310 - 0x136 S32K341 - 0x155 S32K311 - 0x137 S32K312 - 0x138 S32K314 - 0x13A S32K322 - 0x142

Table continues on the next page...

Table continued from the previous page...

Field	Function
	S32K324 - 0x144 S32K328 - 0x148 S32K338 - 0x152 S32K342 - 0x156 S32K344 - 0x158 S32K348 - 0x15C S32K356 - 0x164 S32K358 - 0x166 S32K388 - 0x184 S32K389 - 0x185
15-8 —	Reserved
7-4 MAJOR_MASK	Major Mask Revision For all variants - 0
3-0 MINOR_MASK	Minor Mask Revision S32K358 - 1 For other variants - 0

10.6.3 SIUL2 MCU ID Register #2 (MIDR2)

Offset

Register	Offset
MIDR2	8h

Function

NOTE

This register supports only 32-bit accesses. Byte and half-word accesses are not supported.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TECHNOLOGY				TEMPERATURE				PACKAGE				FREQUENCY			
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FLASH_CODE		FLASH_DATA		FLASH_SIZE_DATA				FLASH_SIZE_CODE							
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Fields

Field	Function
31-29 TECHNOLOGY	Technology Identifies the silicon technology. 001b - C40EFS3
28-26 TEMPERATURE	Temperature Identifies the ambient temperature range. 010b - V = 105C 100b - M = 125C
25-20 PACKAGE	Package This field can by read by software to determine the package type that is used for the particular device. 00_0011b - 257-MAPBGA 00_0100b - 289-MAPBGA 10_0010b - 100-HDQFP 10_0011b - 100-HDQFP 10_0101b - 172-HDQFP 10_0110b - 172-HDQFP 10_0111b - 172-HDQFP-EP
19-16 FREQUENCY	Frequency Identifies maximum core frequency. Qualified by Product Line Letter to provide wider range of frequencies. 0011b - 120 MHz 0100b - 160 MHz 0101b - 240 MHz 0110b - 320 MHz

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-14 FLASH_CODE	Flash Code Identifies the location of Code Flash, if any, within the package. 10b - Monolithic
13-12 FLASH_DATA	Flash Data Identifies the location of Data Flash, if any, within the package. 10b - Monolithic
11-8 FLASH_SIZE_DATA	Flash Size Data Identifies the Flash (EE) memory size. 0000b - 64KB 0001b - 128KB 0010b - 256KB
7-0 FLASH_SIZE_CODE	Flash Size Code Identifies the Flash (code) memory size. 0000_0010b - 512kB 0000_0100b - 1MB 0000_1000b - 2.00MB 0000_1100b - 3.00MB 0001_0000b - 4.00MB 0001_1000b - 6.00MB 0010_0000b - 8.00MB

10.6.4 DMA or Interrupt Status Flag 0 (DISR0)

Offset

Register	Offset
DISR0	10h

Function

Contains flags that record an event on the external IRQ pins. This register supports 8-, 16-, and 32-bit accesses.

When an event (as defined in Interrupt Rising-Edge Event Enable 0 (IREER0) and IFEER0) occurs, the corresponding flag is set. The IRQ flag is set regardless of the state of the corresponding DIRER0[EIRE*n*]. The IRQ flag remains set until you clear it or is cleared by servicing of a DMA request. The IRQ flags are cleared when you write 1 to them. A write of 0 has no effect.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	EIF31	EIF30	EIF29	EIF28	EIF27	EIF26	EIF25	EIF24	EIF23	EIF22	EIF21	EIF20	EIF19	EIF18	EIF17	EIF16
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EIF15	EIF14	EIF13	EIF12	EIF11	EIF10	EIF9	EIF8	EIF7	EIF6	EIF5	EIF4	EIF3	EIF2	EIF1	EIF0
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 EIF31	External Interrupt Status Flag 31 Indicates whether an interrupt event (as defined by IREER31 and IFEER31) has occurred. If this flag is set (DIRERR31 = 1), it causes an interrupt. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Interrupt event did not occur on the pad 1b - Interrupt event occurred When writing 0b - No effect 1b - Clear the flag
30 EIF30	External Interrupt Status Flag 30 Indicates whether an interrupt event (as defined by IREER30 and IFEER30) has occurred. If this flag is set (DIRERR30 = 1), it causes an interrupt. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Interrupt event did not occur on the pad 1b - Interrupt event occurred When writing 0b - No effect 1b - Clear the flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
29 EIF29	External Interrupt Status Flag 29 Indicates whether an interrupt event (as defined by IREER29 and IFEER29) has occurred. If this flag is set (DIRERR29 = 1), it causes an interrupt. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Interrupt event did not occur on the pad 1b - Interrupt event occurred When writing 0b - No effect 1b - Clear the flag
28 EIF28	External Interrupt Status Flag 28 Indicates whether an interrupt event (as defined by IREER28 and IFEER28) has occurred. If this flag is set (DIRERR28 = 1), it causes an interrupt. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Interrupt event did not occur on the pad 1b - Interrupt event occurred When writing 0b - No effect 1b - Clear the flag
27 EIF27	External Interrupt Status Flag 27 Indicates whether an interrupt event (as defined by IREER27 and IFEER27) has occurred. If this flag is set (DIRERR27 = 1), it causes an interrupt. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Interrupt event did not occur on the pad 1b - Interrupt event occurred When writing 0b - No effect

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Clear the flag
26 EIF26	External Interrupt Status Flag 26 Indicates whether an interrupt event (as defined by IREER26 and IFEER26) has occurred. If this flag is set (DIRERR26 = 1), it causes an interrupt. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Interrupt event did not occur on the pad 1b - Interrupt event occurred When writing 0b - No effect 1b - Clear the flag
25 EIF25	External Interrupt Status Flag 25 Indicates whether an interrupt event (as defined by IREER25 and IFEER25) has occurred. If this flag is set (DIRERR25 = 1), it causes an interrupt. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Interrupt event did not occur on the pad 1b - Interrupt event occurred When writing 0b - No effect 1b - Clear the flag
24 EIF24	External Interrupt Status Flag 24 Indicates whether an interrupt event (as defined by IREER24 and IFEER24) has occurred. If this flag is set (DIRERR24 = 1), it causes an interrupt. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Interrupt event did not occur on the pad 1b - Interrupt event occurred When writing

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No effect 1b - Clear the flag
23 EIF23	External Interrupt Status Flag 23 Indicates whether an interrupt event (as defined by IREER23 and IFEER23) has occurred. If this flag is set (DIRERR23 = 1), it causes an interrupt. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Interrupt event did not occur on the pad 1b - Interrupt event occurred When writing 0b - No effect 1b - Clear the flag
22 EIF22	External Interrupt Status Flag 22 Indicates whether an interrupt event (as defined by IREER22 and IFEER22) has occurred. If this flag is set (DIRERR22 = 1), it causes an interrupt. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Interrupt event did not occur on the pad 1b - Interrupt event occurred When writing 0b - No effect 1b - Clear the flag
21 EIF21	External Interrupt Status Flag 21 Indicates whether an interrupt event (as defined by IREER21 and IFEER21) has occurred. If this flag is set (DIRERR21 = 1), it causes an interrupt. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Interrupt event did not occur on the pad

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Interrupt event occurred When writing 0b - No effect 1b - Clear the flag
20 EIF20	External Interrupt Status Flag 20 Indicates whether an interrupt event (as defined by IREER20 and IFEER20) has occurred. If this flag is set (DIRERR20 = 1), it causes an interrupt. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Interrupt event did not occur on the pad 1b - Interrupt event occurred When writing 0b - No effect 1b - Clear the flag
19 EIF19	External Interrupt Status Flag 19 Indicates whether an interrupt event (as defined by IREER19 and IFEER19) has occurred. If this flag is set (DIRERR19 = 1), it causes an interrupt. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Interrupt event did not occur on the pad 1b - Interrupt event occurred When writing 0b - No effect 1b - Clear the flag
18 EIF18	External Interrupt Status Flag 18 Indicates whether an interrupt event (as defined by IREER18 and IFEER18) has occurred. If this flag is set (DIRERR18 = 1), it causes an interrupt. **NOTE** This field behaves differently for register reads and writes. When reading

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Interrupt event did not occur on the pad 1b - Interrupt event occurred When writing 0b - No effect 1b - Clear the flag
17 EIF17	External Interrupt Status Flag 17 Indicates whether an interrupt event (as defined by IREER17 and IFEER17) has occurred. If this flag is set (DIRERR17 = 1), it causes an interrupt. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Interrupt event did not occur on the pad 1b - Interrupt event occurred When writing 0b - No effect 1b - Clear the flag
16 EIF16	External Interrupt Status Flag 16 Indicates whether an interrupt event (as defined by IREER16 and IFEER16) has occurred. If this flag is set (DIRERR16 = 1), it causes an interrupt. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Interrupt event did not occur on the pad 1b - Interrupt event occurred When writing 0b - No effect 1b - Clear the flag
15 EIF15	External Interrupt Status Flag 15 Indicates whether an interrupt event (as defined by IREER15 and IFEER15) has occurred. If this flag is set (DIRERR15 = 1), it causes an interrupt or DMA request. **NOTE** This field behaves differently for register reads and writes.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When reading 0b - Interrupt event did not occur on the pad 1b - Interrupt event occurred When writing 0b - No effect 1b - Clear the flag
14 EIF14	External Interrupt Status Flag 14 Indicates whether an interrupt event (as defined by IREER14 and IFEER14) has occurred. If this flag is set (DIRERR14 = 1), it causes an interrupt or DMA request. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Interrupt event did not occur on the pad 1b - Interrupt event occurred When writing 0b - No effect 1b - Clear the flag
13 EIF13	External Interrupt Status Flag 13 Indicates whether an interrupt event (as defined by IREER13 and IFEER13) has occurred. If this flag is set (DIRERR13 = 1), it causes an interrupt or DMA request. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Interrupt event did not occur on the pad 1b - Interrupt event occurred When writing 0b - No effect 1b - Clear the flag
12 EIF12	External Interrupt Status Flag 12 Indicates whether an interrupt event (as defined by IREER12 and IFEER12) has occurred. If this flag is set (DIRERR12 = 1), it causes an interrupt or DMA request.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field behaves differently for register reads and writes. When reading 0b - Interrupt event did not occur on the pad 1b - Interrupt event occurred When writing 0b - No effect 1b - Clear the flag
11 EIF11	External Interrupt Status Flag 11 Indicates whether an interrupt event (as defined by IREER11 and IFEER11) has occurred. If this flag is set (DIRERR11 = 1), it causes an interrupt or DMA request. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Interrupt event did not occur on the pad 1b - Interrupt event occurred When writing 0b - No effect 1b - Clear the flag
10 EIF10	External Interrupt Status Flag 10 Indicates whether an interrupt event (as defined by IREER10 and IFEER10) has occurred. If this flag is set (DIRERR10 = 1), it causes an interrupt or DMA request. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Interrupt event did not occur on the pad 1b - Interrupt event occurred When writing 0b - No effect 1b - Clear the flag
9 EIF9	External Interrupt Status Flag 9 Indicates whether an interrupt event (as defined by IREER9 and IFEER9) has occurred.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	If this flag is set (DIRERR9 = 1), it causes an interrupt or DMA request. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Interrupt event did not occur on the pad 1b - Interrupt event occurred When writing 0b - No effect 1b - Clear the flag
8 EIF8	External Interrupt Status Flag 8 Indicates whether an interrupt event (as defined by IREER8 and IFEER8) has occurred. If this flag is set (DIRERR8 = 1), it causes an interrupt or DMA request. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Interrupt event did not occur on the pad 1b - Interrupt event occurred When writing 0b - No effect 1b - Clear the flag
7 EIF7	External Interrupt Status Flag 7 Indicates whether an interrupt event (as defined by IREER7 and IFEER7) has occurred. If this flag is set (DIRERR7 = 1), it causes an interrupt or DMA request. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Interrupt event did not occur on the pad 1b - Interrupt event occurred When writing 0b - No effect 1b - Clear the flag
6	External Interrupt Status Flag 6

Table continues on the next page...

Table continued from the previous page...

Field	Function
EIF6	Indicates whether an interrupt event (as defined by IREER6 and IFEER6) has occurred. If this flag is set (DIRERR6 = 1), it causes an interrupt or DMA request. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Interrupt event did not occur on the pad 1b - Interrupt event occurred When writing 0b - No effect 1b - Clear the flag
5 EIF5	External Interrupt Status Flag 5 Indicates whether an interrupt event (as defined by IREER5 and IFEER5) has occurred. If this flag is set (DIRERR5 = 1), it causes an interrupt or DMA request. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Interrupt event did not occur on the pad 1b - Interrupt event occurred When writing 0b - No effect 1b - Clear the flag
4 EIF4	External Interrupt Status Flag 4 Indicates whether an interrupt event (as defined by IREER4 and IFEER4) has occurred. If this flag is set (DIRERR4 = 1), it causes an interrupt or DMA request. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Interrupt event did not occur on the pad 1b - Interrupt event occurred When writing 0b - No effect 1b - Clear the flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
3 EIF3	External Interrupt Status Flag 3 Indicates whether an interrupt event (as defined by IREER3 and IFEER3) has occurred. If this flag is set (DIRERR3 = 1), it causes an interrupt or DMA request. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Interrupt event did not occur on the pad 1b - Interrupt event occurred When writing 0b - No effect 1b - Clear the flag
2 EIF2	External Interrupt Status Flag 2 Indicates whether an interrupt event (as defined by IREER2 and IFEER2) has occurred. If this flag is set (DIRERR2 = 1), it causes an interrupt or DMA request. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Interrupt event did not occur on the pad 1b - Interrupt event occurred When writing 0b - No effect 1b - Clear the flag
1 EIF1	External Interrupt Status Flag 1 Indicates whether an interrupt event (as defined by IREER1 and IFEER1) has occurred. If this flag is set (DIRERR1 = 1), it causes an interrupt or DMA request. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Interrupt event did not occur on the pad 1b - Interrupt event occurred When writing 0b - No effect

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Clear the flag
0 EIF0	External Interrupt Status Flag 0 Indicates whether an interrupt event (as defined by IREER0 and IFEER0) has occurred. If this flag is set (DIRERR0 = 1), it causes an interrupt or DMA request. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Interrupt event did not occur on the pad 1b - Interrupt event occurred When writing 0b - No effect 1b - Clear the flag

10.6.5 DMA or Interrupt Request Enable 0 (DIRER0)

Offset

Register	Offset
DIRER0	18h

Function

Enables the assertion of a DMA or interrupt request to the interrupt controller if the corresponding DISR0[EIF*n*] flag is 1. The type of request is determined by the corresponding DIRSR0[DIRSR*n*] field.

This register supports 8-, 16-, and 32-bit accesses.

NOTE

You cannot enable or disable DIRSR0 after it selects a DMA request.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	EIRE31	EIRE30	EIRE29	EIRE28	EIRE27	EIRE26	EIRE25	EIRE24	EIRE23	EIRE22	EIRE21	EIRE20	EIRE19	EIRE18	EIRE17	EIRE16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	EIRE15	EIRE14	EIRE13	EIRE12	EIRE11	EIRE10	EIRE9	EIRE8	EIRE7	EIRE6	EIRE5	EIRE4	EIRE3	EIRE2	EIRE1	EIRE0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 EIRE31	External Interrupt Request Enable Enables the interrupt requests from the corresponding pin. 0b - Disable 1b - Enable
30 EIRE30	External Interrupt Request Enable Enables the interrupt requests from the corresponding pin. 0b - Disable 1b - Enable
29 EIRE29	External Interrupt Request Enable Enables the interrupt requests from the corresponding pin. 0b - Disable 1b - Enable
28 EIRE28	External Interrupt Request Enable Enables the interrupt requests from the corresponding pin. 0b - Disable 1b - Enable
27 EIRE27	External Interrupt Request Enable Enables the interrupt requests from the corresponding pin. 0b - Disable 1b - Enable
26 EIRE26	External Interrupt Request Enable Enables the interrupt requests from the corresponding pin.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disable 1b - Enable
25 EIRE25	External Interrupt Request Enable Enables the interrupt requests from the corresponding pin. 0b - Disable 1b - Enable
24 EIRE24	External Interrupt Request Enable Enables the interrupt requests from the corresponding pin. 0b - Disable 1b - Enable
23 EIRE23	External Interrupt Request Enable Enables the interrupt requests from the corresponding pin. 0b - Disable 1b - Enable
22 EIRE22	External Interrupt Request Enable Enables the interrupt requests from the corresponding pin. 0b - Disable 1b - Enable
21 EIRE21	External Interrupt Request Enable Enables the interrupt requests from the corresponding pin. 0b - Disable 1b - Enable
20 EIRE20	External Interrupt Request Enable Enables the interrupt requests from the corresponding pin. 0b - Disable 1b - Enable
19 EIRE19	External Interrupt Request Enable Enables the interrupt requests from the corresponding pin. 0b - Disable 1b - Enable
18	External Interrupt Request Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
EIRE18	Enables the interrupt requests from the corresponding pin. 0b - Disable 1b - Enable
17 EIRE17	External Interrupt Request Enable Enables the interrupt requests from the corresponding pin. 0b - Disable 1b - Enable
16 EIRE16	External Interrupt Request Enable Enables the interrupt requests from the corresponding pin. 0b - Disable 1b - Enable
15 EIRE15	External Interrupt Request Enable Enables the interrupt requests from the corresponding pin. 0b - Disable 1b - Enable
14 EIRE14	External Interrupt Request Enable Enables the interrupt requests from the corresponding pin. 0b - Disable 1b - Enable
13 EIRE13	External Interrupt Request Enable Enables the interrupt requests from the corresponding pin. 0b - Disable 1b - Enable
12 EIRE12	External Interrupt Request Enable Enables the interrupt requests from the corresponding pin. 0b - Disable 1b - Enable
11 EIRE11	External Interrupt Request Enable Enables the interrupt requests from the corresponding pin. 0b - Disable 1b - Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
10 EIRE10	External Interrupt Request Enable Enables the interrupt requests from the corresponding pin. 0b - Disable 1b - Enable
9 EIRE9	External Interrupt Request Enable Enables the interrupt requests from the corresponding pin. 0b - Disable 1b - Enable
8 EIRE8	External Interrupt Request Enable Enables the interrupt requests from the corresponding pin. 0b - Disable 1b - Enable
7 EIRE7	External Interrupt Request Enable Enables the interrupt requests from the corresponding pin. 0b - Disable 1b - Enable
6 EIRE6	External Interrupt Request Enable Enables the interrupt requests from the corresponding pin. 0b - Disable 1b - Enable
5 EIRE5	External Interrupt Request Enable Enables the interrupt requests from the corresponding pin. 0b - Disable 1b - Enable
4 EIRE4	External Interrupt Request Enable Enables the interrupt requests from the corresponding pin. 0b - Disable 1b - Enable
3 EIRE3	External Interrupt Request Enable Enables the interrupt requests from the corresponding pin. 0b - Disable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Enable
2 EIRE2	External Interrupt Request Enable Enables the interrupt requests from the corresponding pin. 0b - Disable 1b - Enable
1 EIRE1	External Interrupt Request Enable Enables the interrupt requests from the corresponding pin. 0b - Disable 1b - Enable
0 EIRE0	External Interrupt Request Enable Enables the interrupt requests from the corresponding pin. 0b - Disable 1b - Enable

10.6.6 DMA or Interrupt Request Select 0 (DIRSR0)

Offset

Register	Offset
DIRSR0	20h

Function

Selects the type of request (DMA or interrupt request). This register supports 8-, 16-, and 32-bit accesses.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DIRSR 31	DIRSR 30	DIRSR 29	DIRSR 28	DIRSR 27	DIRSR 26	DIRSR 25	DIRSR 24	DIRSR 23	DIRSR 22	DIRSR 21	DIRSR 20	DIRSR 19	DIRSR 18	DIRSR 17	DIRSR 16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DIRSR 15	DIRSR 14	DIRSR 13	DIRSR 12	DIRSR 11	DIRSR 10	DIRSR 9	DIRSR 8	DIRSR 7	DIRSR 6	DIRSR 5	DIRSR 4	DIRSR 3	DIRSR 2	DIRSR 1	DIRSR 0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 DIRSR31	DMA or Interrupt Request Select Selects either a DMA request or an external interrupt request when an edge-triggered event occurs on the corresponding pin. This field determines whether a DMA request or an interrupt request asserts the corresponding DISR0[EIF*n*]. 0b - Interrupt request 1b - Reserved
30 DIRSR30	DMA or Interrupt Request Select Selects either a DMA request or an external interrupt request when an edge-triggered event occurs on the corresponding pin. This field determines whether a DMA request or an interrupt request asserts the corresponding DISR0[EIF*n*]. 0b - Interrupt request 1b - Reserved
29 DIRSR29	DMA or Interrupt Request Select Selects either a DMA request or an external interrupt request when an edge-triggered event occurs on the corresponding pin. This field determines whether a DMA request or an interrupt request asserts the corresponding DISR0[EIF*n*]. 0b - Interrupt request 1b - Reserved
28 DIRSR28	DMA or Interrupt Request Select Selects either a DMA request or an external interrupt request when an edge-triggered event occurs on the corresponding pin. This field determines whether a DMA request or an interrupt request asserts the corresponding DISR0[EIF*n*]. 0b - Interrupt request 1b - Reserved
27 DIRSR27	DMA or Interrupt Request Select Selects either a DMA request or an external interrupt request when an edge-triggered event occurs on the corresponding pin. This field determines whether a DMA request or an interrupt request asserts the corresponding DISR0[EIF*n*]. 0b - Interrupt request 1b - Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
26 DIRSR26	DMA or Interrupt Request Select Selects either a DMA request or an external interrupt request when an edge-triggered event occurs on the corresponding pin. This field determines whether a DMA request or an interrupt request asserts the corresponding DISR0[EIF*n*]. 0b - Interrupt request 1b - Reserved
25 DIRSR25	DMA or Interrupt Request Select Selects either a DMA request or an external interrupt request when an edge-triggered event occurs on the corresponding pin. This field determines whether a DMA request or an interrupt request asserts the corresponding DISR0[EIF*n*]. 0b - Interrupt request 1b - Reserved
24 DIRSR24	DMA or Interrupt Request Select Selects either a DMA request or an external interrupt request when an edge-triggered event occurs on the corresponding pin. This field determines whether a DMA request or an interrupt request asserts the corresponding DISR0[EIF*n*]. 0b - Interrupt request 1b - Reserved
23 DIRSR23	DMA or Interrupt Request Select Selects either a DMA request or an external interrupt request when an edge-triggered event occurs on the corresponding pin. This field determines whether a DMA request or an interrupt request asserts the corresponding DISR0[EIF*n*]. 0b - Interrupt request 1b - Reserved
22 DIRSR22	DMA or Interrupt Request Select Selects either a DMA request or an external interrupt request when an edge-triggered event occurs on the corresponding pin. This field determines whether a DMA request or an interrupt request asserts the corresponding DISR0[EIF*n*]. 0b - Interrupt request 1b - Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
21 DIRSR21	DMA or Interrupt Request Select Selects either a DMA request or an external interrupt request when an edge-triggered event occurs on the corresponding pin. This field determines whether a DMA request or an interrupt request asserts the corresponding DISR0[EIF*n*]. 0b - Interrupt request 1b - Reserved
20 DIRSR20	DMA or Interrupt Request Select Selects either a DMA request or an external interrupt request when an edge-triggered event occurs on the corresponding pin. This field determines whether a DMA request or an interrupt request asserts the corresponding DISR0[EIF*n*]. 0b - Interrupt request 1b - Reserved
19 DIRSR19	DMA or Interrupt Request Select Selects either a DMA request or an external interrupt request when an edge-triggered event occurs on the corresponding pin. This field determines whether a DMA request or an interrupt request asserts the corresponding DISR0[EIF*n*]. 0b - Interrupt request 1b - Reserved
18 DIRSR18	DMA or Interrupt Request Select Selects either a DMA request or an external interrupt request when an edge-triggered event occurs on the corresponding pin. This field determines whether a DMA request or an interrupt request asserts the corresponding DISR0[EIF*n*]. 0b - Interrupt request 1b - Reserved
17 DIRSR17	DMA or Interrupt Request Select Selects either a DMA request or an external interrupt request when an edge-triggered event occurs on the corresponding pin. This field determines whether a DMA request or an interrupt request asserts the corresponding DISR0[EIF*n*]. 0b - Interrupt request 1b - Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
16 DIRSR16	DMA or Interrupt Request Select Selects either a DMA request or an external interrupt request when an edge-triggered event occurs on the corresponding pin. This field determines whether a DMA request or an interrupt request asserts the corresponding DISR0[EIF*n*]. 0b - Interrupt request 1b - Reserved
15 DIRSR15	DMA or Interrupt Request Select Selects either a DMA request or an external interrupt request when an edge-triggered event occurs on the corresponding pin. This field determines whether a DMA request or an interrupt request asserts the corresponding DISR0[EIF*n*]. 0b - Interrupt request 1b - DMA request
14 DIRSR14	DMA or Interrupt Request Select Selects either a DMA request or an external interrupt request when an edge-triggered event occurs on the corresponding pin. This field determines whether a DMA request or an interrupt request asserts the corresponding DISR0[EIF*n*]. 0b - Interrupt request 1b - DMA request
13 DIRSR13	DMA or Interrupt Request Select Selects either a DMA request or an external interrupt request when an edge-triggered event occurs on the corresponding pin. This field determines whether a DMA request or an interrupt request asserts the corresponding DISR0[EIF*n*]. 0b - Interrupt request 1b - DMA request
12 DIRSR12	DMA or Interrupt Request Select Selects either a DMA request or an external interrupt request when an edge-triggered event occurs on the corresponding pin. This field determines whether a DMA request or an interrupt request asserts the corresponding DISR0[EIF*n*]. 0b - Interrupt request 1b - DMA request

Table continues on the next page...

Table continued from the previous page...

Field	Function
11 DIRSR11	DMA or Interrupt Request Select Selects either a DMA request or an external interrupt request when an edge-triggered event occurs on the corresponding pin. This field determines whether a DMA request or an interrupt request asserts the corresponding DISR0[EIF*n*]. 0b - Interrupt request 1b - DMA request
10 DIRSR10	DMA or Interrupt Request Select Selects either a DMA request or an external interrupt request when an edge-triggered event occurs on the corresponding pin. This field determines whether a DMA request or an interrupt request asserts the corresponding DISR0[EIF*n*]. 0b - Interrupt request 1b - DMA request
9 DIRSR9	DMA or Interrupt Request Select Selects either a DMA request or an external interrupt request when an edge-triggered event occurs on the corresponding pin. This field determines whether a DMA request or an interrupt request asserts the corresponding DISR0[EIF*n*]. 0b - Interrupt request 1b - DMA request
8 DIRSR8	DMA or Interrupt Request Select Selects either a DMA request or an external interrupt request when an edge-triggered event occurs on the corresponding pin. This field determines whether a DMA request or an interrupt request asserts the corresponding DISR0[EIF*n*]. 0b - Interrupt request 1b - DMA request
7 DIRSR7	DMA or Interrupt Request Select Selects either a DMA request or an external interrupt request when an edge-triggered event occurs on the corresponding pin. This field determines whether a DMA request or an interrupt request asserts the corresponding DISR0[EIF*n*]. 0b - Interrupt request 1b - DMA request

Table continues on the next page...

Table continued from the previous page...

Field	Function
6 DIRSR6	DMA or Interrupt Request Select Selects either a DMA request or an external interrupt request when an edge-triggered event occurs on the corresponding pin. This field determines whether a DMA request or an interrupt request asserts the corresponding DISR0[EIF*n*]. 0b - Interrupt request 1b - DMA request
5 DIRSR5	DMA or Interrupt Request Select Selects either a DMA request or an external interrupt request when an edge-triggered event occurs on the corresponding pin. This field determines whether a DMA request or an interrupt request asserts the corresponding DISR0[EIF*n*]. 0b - Interrupt request 1b - DMA request
4 DIRSR4	DMA or Interrupt Request Select Selects either a DMA request or an external interrupt request when an edge-triggered event occurs on the corresponding pin. This field determines whether a DMA request or an interrupt request asserts the corresponding DISR0[EIF*n*]. 0b - Interrupt request 1b - DMA request
3 DIRSR3	DMA or Interrupt Request Select Selects either a DMA request or an external interrupt request when an edge-triggered event occurs on the corresponding pin. This field determines whether a DMA request or an interrupt request asserts the corresponding DISR0[EIF*n*]. 0b - Interrupt request 1b - DMA request
2 DIRSR2	DMA or Interrupt Request Select Selects either a DMA request or an external interrupt request when an edge-triggered event occurs on the corresponding pin. This field determines whether a DMA request or an interrupt request asserts the corresponding DISR0[EIF*n*]. 0b - Interrupt request 1b - DMA request

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 DIRSR1	DMA or Interrupt Request Select Selects either a DMA request or an external interrupt request when an edge-triggered event occurs on the corresponding pin. This field determines whether a DMA request or an interrupt request asserts the corresponding DISR0[EIF*n*]. 0b - Interrupt request 1b - DMA request
0 DIRSR0	DMA or Interrupt Request Select Selects either a DMA request or an external interrupt request when an edge-triggered event occurs on the corresponding pin. This field determines whether a DMA request or an interrupt request asserts the corresponding DISR0[EIF*n*]. 0b - Interrupt request 1b - DMA request

10.6.7 Interrupt Rising-Edge Event Enable 0 (IREER0)

Offset

Register	Offset
IREER0	28h

Function

Enables the rising-edge triggered events on the corresponding interrupt pads. This register supports 8-, 16-, and 32-bit accesses.

NOTE

If you write 0 to both the IREE and IFEE fields for the same interrupt source, the interrupt status flag for the corresponding external interrupt never sets.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	IREE31	IREE30	IREE29	IREE28	IREE27	IREE26	IREE25	IREE24	IREE23	IREE22	IREE21	IREE20	IREE19	IREE18	IREE17	IREE16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	IREE15	IREE14	IREE13	IREE12	IREE11	IREE10	IREE9	IREE8	IREE7	IREE6	IREE5	IREE4	IREE3	IREE2	IREE1	IREE0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 IREE31	Interrupt Rising-Edge Event Enable 31 Enables the rising-edge events to set DISR0[EIF31]. 0b - Disable 1b - Enable
30 IREE30	Interrupt Rising-Edge Event Enable 30 Enables the rising-edge events to set DISR0[EIF30]. 0b - Disable 1b - Enable
29 IREE29	Interrupt Rising-Edge Event Enable 29 Enables the rising-edge events to set DISR0[EIF29]. 0b - Disable 1b - Enable
28 IREE28	Interrupt Rising-Edge Event Enable 28 Enables the rising-edge events to set DISR0[EIF28]. 0b - Disable 1b - Enable
27 IREE27	Interrupt Rising-Edge Event Enable 27 Enables the rising-edge events to set DISR0[EIF27]. 0b - Disable 1b - Enable
26 IREE26	Interrupt Rising-Edge Event Enable 26 Enables the rising-edge events to set DISR0[EIF26]. 0b - Disable 1b - Enable
25 IREE25	Interrupt Rising-Edge Event Enable 25 Enables the rising-edge events to set DISR0[EIF25]. 0b - Disable 1b - Enable
24 IREE24	Interrupt Rising-Edge Event Enable 24 Enables the rising-edge events to set DISR0[EIF24]. 0b - Disable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Enable
23 IREE23	Interrupt Rising-Edge Event Enable 23 Enables the rising-edge events to set DISR0[EIF23]. 0b - Disable 1b - Enable
22 IREE22	Interrupt Rising-Edge Event Enable 22 Enables the rising-edge events to set DISR0[EIF22]. 0b - Disable 1b - Enable
21 IREE21	Interrupt Rising-Edge Event Enable 21 Enables the rising-edge events to set DISR0[EIF21]. 0b - Disable 1b - Enable
20 IREE20	Interrupt Rising-Edge Event Enable 20 Enables the rising-edge events to set DISR0[EIF20]. 0b - Disable 1b - Enable
19 IREE19	Interrupt Rising-Edge Event Enable 19 Enables the rising-edge events to set DISR0[EIF19]. 0b - Disable 1b - Enable
18 IREE18	Interrupt Rising-Edge Event Enable 18 Enables the rising-edge events to set DISR0[EIF18]. 0b - Disable 1b - Enable
17 IREE17	Interrupt Rising-Edge Event Enable 17 Enables the rising-edge events to set DISR0[EIF17]. 0b - Disable 1b - Enable
16 IREE16	Interrupt Rising-Edge Event Enable 16 Enables the rising-edge events to set DISR0[EIF16].

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disable 1b - Enable
15 IREE15	Interrupt Rising-Edge Event Enable 15 Enables the rising-edge events to set DISR0[EIF15]. 0b - Disable 1b - Enable
14 IREE14	Interrupt Rising-Edge Event Enable 14 Enables the rising-edge events to set DISR0[EIF14]. 0b - Disable 1b - Enable
13 IREE13	Interrupt Rising-Edge Event Enable 13 Enables the rising-edge events to set DISR0[EIF13]. 0b - Disable 1b - Enable
12 IREE12	Interrupt Rising-Edge Event Enable 12 Enables the rising-edge events to set DISR0[EIF12]. 0b - Disable 1b - Enable
11 IREE11	Interrupt Rising-Edge Event Enable 11 Enables the rising-edge events to set DISR0[EIF11]. 0b - Disable 1b - Enable
10 IREE10	Interrupt Rising-Edge Event Enable 10 Enables the rising-edge events to set DISR0[EIF10]. 0b - Disable 1b - Enable
9 IREE9	Interrupt Rising-Edge Event Enable 9 Enables the rising-edge events to set DISR0[EIF9]. 0b - Disable 1b - Enable
8	Interrupt Rising-Edge Event Enable 8

Table continues on the next page...

Table continued from the previous page...

Field	Function
IREE8	Enables the rising-edge events to set DISR0[EIF8]. 0b - Disable 1b - Enable
7 IREE7	Interrupt Rising-Edge Event Enable 7 Enables the rising-edge events to set DISR0[EIF7]. 0b - Disable 1b - Enable
6 IREE6	Interrupt Rising-Edge Event Enable 6 Enables the rising-edge events to set DISR0[EIF6]. 0b - Disable 1b - Enable
5 IREE5	Interrupt Rising-Edge Event Enable 5 Enables the rising-edge events to set DISR0[EIF5]. 0b - Disable 1b - Enable
4 IREE4	Interrupt Rising-Edge Event Enable 4 Enables the rising-edge events to set DISR0[EIF4]. 0b - Disable 1b - Enable
3 IREE3	Interrupt Rising-Edge Event Enable 3 Enables the rising-edge events to set DISR0[EIF3]. 0b - Disable 1b - Enable
2 IREE2	Interrupt Rising-Edge Event Enable 2 Enables the rising-edge events to set DISR0[EIF2]. 0b - Disable 1b - Enable
1 IREE1	Interrupt Rising-Edge Event Enable 1 Enables the rising-edge events to set DISR0[EIF1]. 0b - Disable 1b - Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 IREE0	Interrupt Rising-Edge Event Enable 0 Enables the rising-edge events to set DISR0[EIF0]. 0b - Disable 1b - Enable

10.6.8 Interrupt Falling-Edge Event Enable 0 (IFEER0)

Offset

Register	Offset
IFEER0	30h

Function

Enables the falling-edge triggered events on the corresponding interrupt pads. This register supports 8-, 16-, and 32-bit accesses.

NOTE

If you write 0 to both the IREE and IFEE fields for the same interrupt source, the interrupt status flag for the corresponding external interrupt never sets.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	IFEE31	IFEE30	IFEE29	IFEE28	IFEE27	IFEE26	IFEE25	IFEE24	IFEE23	IFEE22	IFEE21	IFEE20	IFEE19	IFEE18	IFEE17	IFEE16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	IFEE15	IFEE14	IFEE13	IFEE12	IFEE11	IFEE10	IFEE9	IFEE8	IFEE7	IFEE6	IFEE5	IFEE4	IFEE3	IFEE2	IFEE1	IFEE0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 IFEE31	Interrupt Falling-Edge Event Enable 31 Enables the falling-edge events to set DISR0[EIF31]. 0b - Disable 1b - Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
30 IFEE30	Interrupt Falling-Edge Event Enable 30 Enables the falling-edge events to set DISR0[EIF30]. 0b - Disable 1b - Enable
29 IFEE29	Interrupt Falling-Edge Event Enable 29 Enables the falling-edge events to set DISR0[EIF29]. 0b - Disable 1b - Enable
28 IFEE28	Interrupt Falling-Edge Event Enable 28 Enables the falling-edge events to set DISR0[EIF28]. 0b - Disable 1b - Enable
27 IFEE27	Interrupt Falling-Edge Event Enable 27 Enables the falling-edge events to set DISR0[EIF27]. 0b - Disable 1b - Enable
26 IFEE26	Interrupt Falling-Edge Event Enable 26 Enables the falling-edge events to set DISR0[EIF26]. 0b - Disable 1b - Enable
25 IFEE25	Interrupt Falling-Edge Event Enable 25 Enables the falling-edge events to set DISR0[EIF25]. 0b - Disable 1b - Enable
24 IFEE24	Interrupt Falling-Edge Event Enable 24 Enables the falling-edge events to set DISR0[EIF24]. 0b - Disable 1b - Enable
23 IFEE23	Interrupt Falling-Edge Event Enable 23 Enables the falling-edge events to set DISR0[EIF23]. 0b - Disable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Enable
22 IFEE22	Interrupt Falling-Edge Event Enable 22 Enables the falling-edge events to set DISR0[EIF22]. 0b - Disable 1b - Enable
21 IFEE21	Interrupt Falling-Edge Event Enable 21 Enables the falling-edge events to set DISR0[EIF21]. 0b - Disable 1b - Enable
20 IFEE20	Interrupt Falling-Edge Event Enable 20 Enables the falling-edge events to set DISR0[EIF20]. 0b - Disable 1b - Enable
19 IFEE19	Interrupt Falling-Edge Event Enable 19 Enables the falling-edge events to set DISR0[EIF19]. 0b - Disable 1b - Enable
18 IFEE18	Interrupt Falling-Edge Event Enable 18 Enables the falling-edge events to set DISR0[EIF18]. 0b - Disable 1b - Enable
17 IFEE17	Interrupt Falling-Edge Event Enable 17 Enables the falling-edge events to set DISR0[EIF17]. 0b - Disable 1b - Enable
16 IFEE16	Interrupt Falling-Edge Event Enable 16 Enables the falling-edge events to set DISR0[EIF16]. 0b - Disable 1b - Enable
15 IFEE15	Interrupt Falling-Edge Event Enable 15 Enables the falling-edge events to set DISR0[EIF15].

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disable 1b - Enable
14 IFEE14	Interrupt Falling-Edge Event Enable 14 Enables the falling-edge events to set DISR0[EIF14]. 0b - Disable 1b - Enable
13 IFEE13	Interrupt Falling-Edge Event Enable 13 Enables the falling-edge events to set DISR0[EIF13]. 0b - Disable 1b - Enable
12 IFEE12	Interrupt Falling-Edge Event Enable 12 Enables the falling-edge events to set DISR0[EIF12]. 0b - Disable 1b - Enable
11 IFEE11	Interrupt Falling-Edge Event Enable 11 Enables the falling-edge events to set DISR0[EIF11]. 0b - Disable 1b - Enable
10 IFEE10	Interrupt Falling-Edge Event Enable 10 Enables the falling-edge events to set DISR0[EIF10]. 0b - Disable 1b - Enable
9 IFEE9	Interrupt Falling-Edge Event Enable 9 Enables the falling-edge events to set DISR0[EIF9]. 0b - Disable 1b - Enable
8 IFEE8	Interrupt Falling-Edge Event Enable 8 Enables the falling-edge events to set DISR0[EIF8]. 0b - Disable 1b - Enable
7	Interrupt Falling-Edge Event Enable 7

Table continues on the next page...

Table continued from the previous page...

Field	Function
IFEE7	Enables the falling-edge events to set DISR0[EIF7]. 0b - Disable 1b - Enable
6 IFEE6	Interrupt Falling-Edge Event Enable 6 Enables the falling-edge events to set DISR0[EIF6]. 0b - Disable 1b - Enable
5 IFEE5	Interrupt Falling-Edge Event Enable 5 Enables the falling-edge events to set DISR0[EIF5]. 0b - Disable 1b - Enable
4 IFEE4	Interrupt Falling-Edge Event Enable 4 Enables the falling-edge events to set DISR0[EIF4]. 0b - Disable 1b - Enable
3 IFEE3	Interrupt Falling-Edge Event Enable 3 Enables the falling-edge events to set DISR0[EIF3]. 0b - Disable 1b - Enable
2 IFEE2	Interrupt Falling-Edge Event Enable 2 Enables the falling-edge events to set DISR0[EIF2]. 0b - Disable 1b - Enable
1 IFEE1	Interrupt Falling-Edge Event Enable 1 Enables the falling-edge events to set DISR0[EIF1]. 0b - Disable 1b - Enable
0 IFEE0	Interrupt Falling-Edge Event Enable 0 Enables the falling-edge events to set DISR0[EIF0]. 0b - Disable 1b - Enable

10.6.9 Interrupt Filter Enable 0 (IFER0)

Offset

Register	Offset
IFER0	38h

Function

Enables a digital filter counter on the corresponding interrupt pads to filter out glitches on the inputs. This register supports 8-, 16-, and 32-bit accesses.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	IFE31	IFE30	IFE29	IFE28	IFE27	IFE26	IFE25	IFE24	IFE23	IFE22	IFE21	IFE20	IFE19	IFE18	IFE17	IFE16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	IFE15	IFE14	IFE13	IFE12	IFE11	IFE10	IFE9	IFE8	IFE7	IFE6	IFE5	IFE4	IFE3	IFE2	IFE1	IFE0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 IFE31	Interrupt Filter Enable 31 Enables the digital glitch filter on the interrupt pad input. 0b - Disable 1b - Enable
30 IFE30	Interrupt Filter Enable 30 Enables the digital glitch filter on the interrupt pad input. 0b - Disable 1b - Enable
29 IFE29	Interrupt Filter Enable 29 Enables the digital glitch filter on the interrupt pad input. 0b - Disable 1b - Enable
28	Interrupt Filter Enable 28

Table continues on the next page...

Table continued from the previous page...

Field	Function
IFE28	Enables the digital glitch filter on the interrupt pad input. 0b - Disable 1b - Enable
27 IFE27	Interrupt Filter Enable 27 Enables the digital glitch filter on the interrupt pad input. 0b - Disable 1b - Enable
26 IFE26	Interrupt Filter Enable 26 Enables the digital glitch filter on the interrupt pad input. 0b - Disable 1b - Enable
25 IFE25	Interrupt Filter Enable 25 Enables the digital glitch filter on the interrupt pad input. 0b - Disable 1b - Enable
24 IFE24	Interrupt Filter Enable 24 Enables the digital glitch filter on the interrupt pad input. 0b - Disable 1b - Enable
23 IFE23	Interrupt Filter Enable 23 Enables the digital glitch filter on the interrupt pad input. 0b - Disable 1b - Enable
22 IFE22	Interrupt Filter Enable 22 Enables the digital glitch filter on the interrupt pad input. 0b - Disable 1b - Enable
21 IFE21	Interrupt Filter Enable 21 Enables the digital glitch filter on the interrupt pad input. 0b - Disable 1b - Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
20 IFE20	Interrupt Filter Enable 20 Enables the digital glitch filter on the interrupt pad input. 0b - Disable 1b - Enable
19 IFE19	Interrupt Filter Enable 19 Enables the digital glitch filter on the interrupt pad input. 0b - Disable 1b - Enable
18 IFE18	Interrupt Filter Enable 18 Enables the digital glitch filter on the interrupt pad input. 0b - Disable 1b - Enable
17 IFE17	Interrupt Filter Enable 17 Enables the digital glitch filter on the interrupt pad input. 0b - Disable 1b - Enable
16 IFE16	Interrupt Filter Enable 16 Enables the digital glitch filter on the interrupt pad input. 0b - Disable 1b - Enable
15 IFE15	Interrupt Filter Enable 15 Enables the digital glitch filter on the interrupt pad input. 0b - Disable 1b - Enable
14 IFE14	Interrupt Filter Enable 14 Enables the digital glitch filter on the interrupt pad input. 0b - Disable 1b - Enable
13 IFE13	Interrupt Filter Enable 13 Enables the digital glitch filter on the interrupt pad input. 0b - Disable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Enable
12 IFE12	Interrupt Filter Enable 12 Enables the digital glitch filter on the interrupt pad input. 0b - Disable 1b - Enable
11 IFE11	Interrupt Filter Enable 11 Enables the digital glitch filter on the interrupt pad input. 0b - Disable 1b - Enable
10 IFE10	Interrupt Filter Enable 10 Enables the digital glitch filter on the interrupt pad input. 0b - Disable 1b - Enable
9 IFE9	Interrupt Filter Enable 9 Enables the digital glitch filter on the interrupt pad input. 0b - Disable 1b - Enable
8 IFE8	Interrupt Filter Enable 8 Enables the digital glitch filter on the interrupt pad input. 0b - Disable 1b - Enable
7 IFE7	Interrupt Filter Enable 7 Enables the digital glitch filter on the interrupt pad input. 0b - Disable 1b - Enable
6 IFE6	Interrupt Filter Enable 6 Enables the digital glitch filter on the interrupt pad input. 0b - Disable 1b - Enable
5 IFE5	Interrupt Filter Enable 5 Enables the digital glitch filter on the interrupt pad input.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disable 1b - Enable
4 IFE4	Interrupt Filter Enable 4 Enables the digital glitch filter on the interrupt pad input. 0b - Disable 1b - Enable
3 IFE3	Interrupt Filter Enable 3 Enables the digital glitch filter on the interrupt pad input. 0b - Disable 1b - Enable
2 IFE2	Interrupt Filter Enable 2 Enables the digital glitch filter on the interrupt pad input. 0b - Disable 1b - Enable
1 IFE1	Interrupt Filter Enable 1 Enables the digital glitch filter on the interrupt pad input. 0b - Disable 1b - Enable
0 IFE0	Interrupt Filter Enable 0 Enables the digital glitch filter on the interrupt pad input. 0b - Disable 1b - Enable

10.6.10 Interrupt Filter Maximum Counter (IFMCR0 - IFMCR31)

Offset

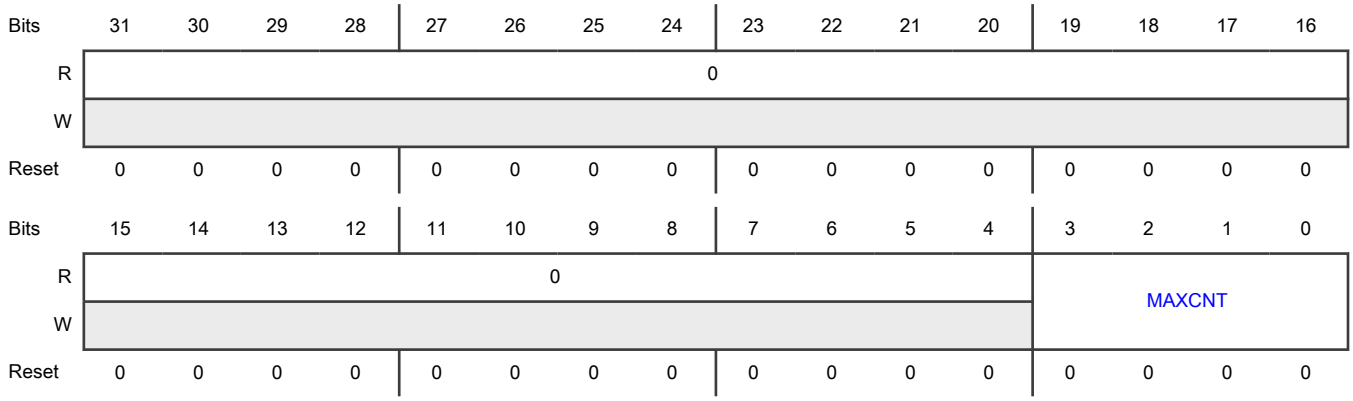
For a = 0 to 31:

Register	Offset
IFMCRa	40h + (a × 4h)

Function

Configures the filter counter associated with each of the digital glitch filter and supports only 32-bit accesses. It does not support byte and halfword accesses.

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 MAXCNT	Maximum Interrupt Filter Counter Setting Specifies the settings of the maximum interrupt filter counter. The value of this field varies between 0d – 15d. Based on its value, it exhibits the following settings: • A value of 0d, 1d, or 2d sets the filter as an all pass filter • A value of 3d to 15d sets the filter period to TCK × MAXCNT + *n* × TCK, where: — *n* is 1, 2, 3, or 4 (*n* accounts for the uncertainty factor in filter period calculation) — TCK is the prescaled filter clock period, which is the IRC clock prescaled to the IFCP value specified in IFCPR

10.6.11 Interrupt Filter Clock Prescaler (IFCPR)

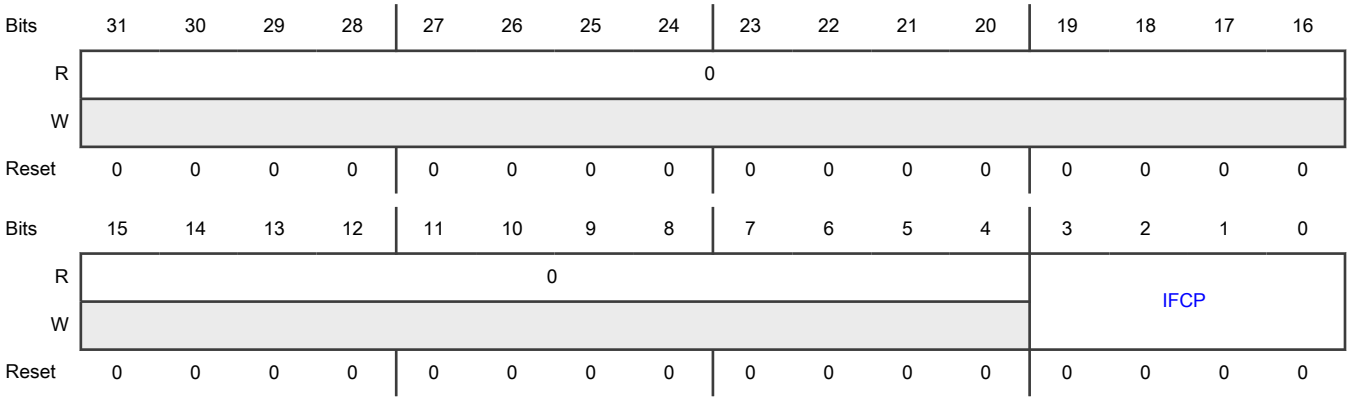
Offset

Register	Offset
IFCPR	C0h

Function

Configures the clock prescaler that selects the clock for all digital filters and supports only 32-bit accesses. Byte and half-word accesses are not supported. A prescaler is applied to the input clock to SIUL2, which is the peripheral clock counter in SIUL2.

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 IFCP	Interrupt Filter Clock Prescaler Setting Specifies the settings of the interrupt filter clock prescaler. The prescaled filter clock period is determined by T_{IRC} × (IFCP + 1), where: • T_{IRC} is the internal oscillator period • IFCP is 0 to 15

10.6.12 MUX0 EMIOS ENABLE 1 (MUX0_EMIOS_EN1)

Offset

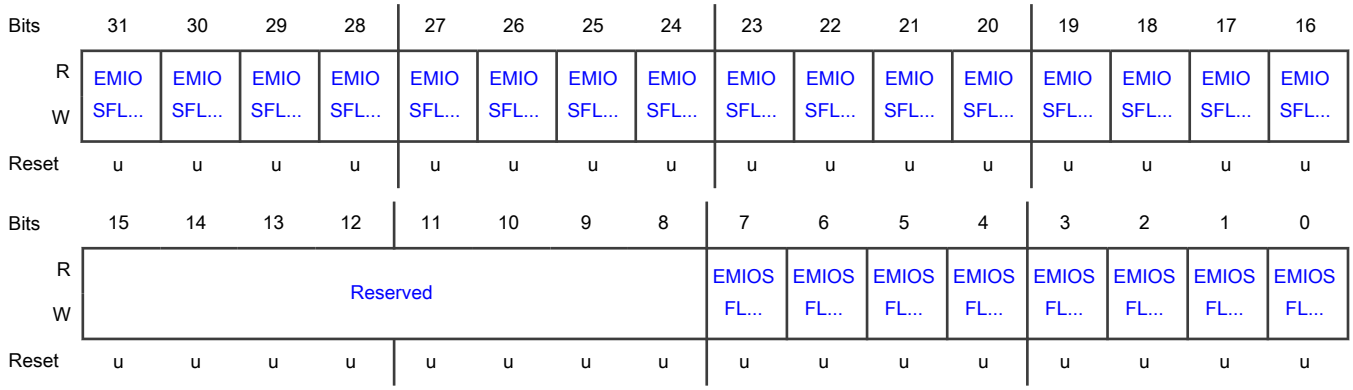
Register	Offset
MUX0_EMIOS_EN1	100h

Function

NOTE

This register is reserved for S32K3x4.

Diagram



Fields

Field	Function
31 EMIOSFLG15_EN	EMIOS0 Output Flag 15 Monitor Enable
30 EMIOSFLG14_EN	EMIOS0 Output Flag 14 Monitor Enable
29 EMIOSFLG13_EN	EMIOS0 Output Flag 13 Monitor Enable
28 EMIOSFLG12_EN	EMIOS0 Output Flag 12 Monitor Enable
27 EMIOSFLG11_EN	EMIOS0 Output Flag 11 Monitor Enable
26 EMIOSFLG10_EN	EMIOS0 Output Flag 10 Monitor Enable
25 EMIOSFLG9_EN	EMIOS0 Output Flag 9 Monitor Enable
24 EMIOSFLG8_EN	EMIOS0 Output Flag 8 Monitor Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
23 EMIOSFLG7_EN	EMIOS0 Output Flag 7 Monitor Enable
22 EMIOSFLG6_EN	EMIOS0 Output Flag 6 Monitor Enable
21 EMIOSFLG5_EN	EMIOS0 Output Flag 5 Monitor Enable
20 EMIOSFLG4_EN	EMIOS0 Output Flag 4 Monitor Enable
19 EMIOSFLG3_EN	EMIOS0 Output Flag 3 Monitor Enable
18 EMIOSFLG2_EN	EMIOS0 Output Flag 2 Monitor Enable
17 EMIOSFLG1_EN	EMIOS0 Output Flag 1 Monitor Enable
16 EMIOSFLG0_EN	EMIOS0 Output Flag 0 Monitor Enable
15-8 —	Reserved
7 EMIOSFLG23_EN	EMIOS0 Output Flag 23 Monitor Enable
6 EMIOSFLG22_EN	EMIOS0 Output Flag 22 Monitor Enable
5	EMIOS0 Output Flag 21 Monitor Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
EMIOSFLG21_EN	
4 EMIOSFLG20_EN	EMIOS0 Output Flag 20 Monitor Enable
3 EMIOSFLG19_EN	EMIOS0 Output Flag 19 Monitor Enable
2 EMIOSFLG18_EN	EMIOS0 Output Flag 18 Monitor Enable
1 EMIOSFLG17_EN	EMIOS0 Output Flag 17 Monitor Enable
0 EMIOSFLG16_EN	EMIOS0 Output Flag 16 Monitor Enable

10.6.13 MUX0 MISC ENABLE (MUX0_MISC_EN)

Offset

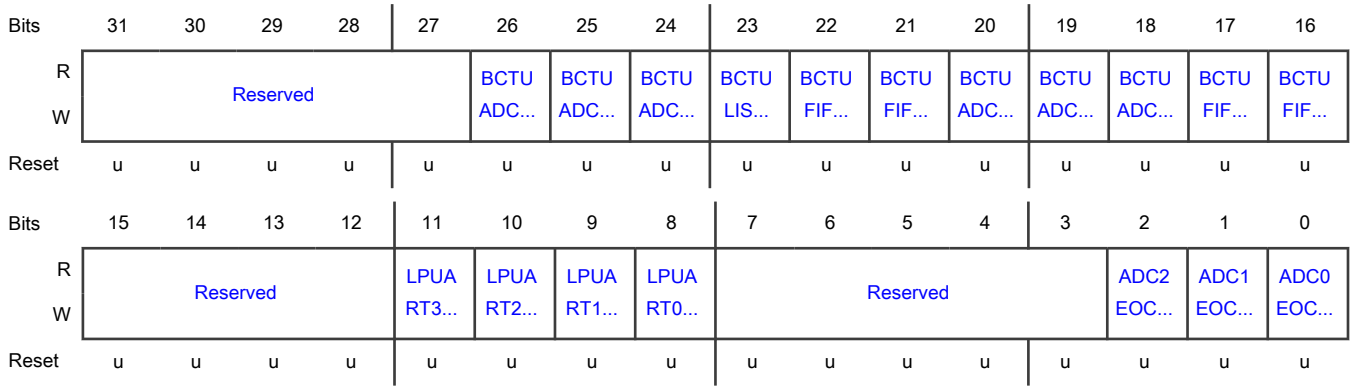
Register	Offset
MUX0_MISC_EN	104h

Function

NOTE

This register is reserved for S32K3x4.

Diagram



Fields

Field	Function
31-27 —	Reserved
26 BCTUADC2INT_EN	BCTU ADC2DR Interrupt Request Monitor Enable
25 BCTUADC1INT_EN	BCTU ADC1DR Interrupt Request Monitor Enable
24 BCTUADC0INT_EN	BCTU ADC0DR Interrupt Request Monitor Enable
23 BCTULISTINT_EN	BCTU Conversion List Interrupt Request Enable
22 BCTUFIFO1INT_EN	BCTU FIFO0 Interrupt Request Monitor Enable
21 BCTUFIFO0INT_EN	BCTU FIFO0 Interrupt Request Monitor Enable
20 BCTUADC2DMA_EN	BCTU ADC2DR DMA Request Monitor Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
19 BCTUADC1DMA_EN	BCTU ADC1DR DMA Request Monitor Enable
18 BCTUADC0DMA_EN	BCTU ADC0DR DMA Request Monitor Enable
17 BCTUFIFO1DMA_EN	BCTU FIFO1 DMA Request Monitor Enable
16 BCTUFIFO0DMA_EN	BCTU FIFO0 DMA Request Monitor Enable
15-12 —	Reserved
11 LPUART3TRG_EN	LPUART3 Output Trigger Monitor Enable
10 LPUART2TRG_EN	LPUART2 Output Trigger Monitor Enable
9 LPUART1TRG_EN	LPUART1 Output Trigger Monitor Enable
8 LPUART0TRG_EN	LPUART0 Output Trigger Monitor Enable
7-3 —	Reserved
2 ADC2EOC_EN	ADC2 End of Conversion Trigger Monitor
1 ADC1EOC_EN	ADC1 End of Conversion Trigger Monitor

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 ADC0EOC_EN	ADC0 End of Conversion Trigger Monitor

10.6.14 MUX1 EMIOS ENABLE (MUX1_EMIOS_EN)

Offset

Register	Offset
MUX1_EMIOS_EN	108h

Function

NOTE

This register is reserved for S32K3x4.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	EMIOSFL...	EMIOSFL...	EMIOSFL...	EMIOSFL...	EMIOSFL...	EMIOSFL...	EMIOSFL...	EMIOSFL...	EMIOSFL...	EMIOSFL...	EMIOSFL...	EMIOSFL...	EMIOSFL...	EMIOSFL...	EMIOSFL...	EMIOSFL...
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								EMIOSFL...	EMIOSFL...	EMIOSFL...	EMIOSFL...	EMIOSFL...	EMIOSFL...	EMIOSFL...	EMIOSFL...
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Fields

Field	Function
31 EMIOSFLG15_EN	EMIOS0 Output Flag 15 Monitor Enable
30 EMIOSFLG14_EN	EMIOS0 Output Flag 14 Monitor Enable
29	EMIOS0 Output Flag 13 Monitor Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
EMIOSFLG13_EN	
28 EMIOSFLG12_EN	EMIOS0 Output Flag 12 Monitor Enable
27 EMIOSFLG11_EN	EMIOS0 Output Flag 11 Monitor Enable
26 EMIOSFLG10_EN	EMIOS0 Output Flag 10 Monitor Enable
25 EMIOSFLG9_EN	EMIOS0 Output Flag 9 Monitor Enable
24 EMIOSFLG8_EN	EMIOS0 Output Flag 8 Monitor Enable
23 EMIOSFLG7_EN	EMIOS0 Output Flag 7 Monitor Enable
22 EMIOSFLG6_EN	EMIOS0 Output Flag 6 Monitor Enable
21 EMIOSFLG5_EN	EMIOS0 Output Flag 5 Monitor Enable
20 EMIOSFLG4_EN	EMIOS0 Output Flag 4 Monitor Enable
19 EMIOSFLG3_EN	EMIOS0 Output Flag 3 Monitor Enable
18	EMIOS0 Output Flag 2 Monitor Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
EMIOSFLG2_EN	
17 EMIOSFLG1_EN	EMIOS0 Output Flag 1 Monitor Enable
16 EMIOSFLG0_EN	EMIOS0 Output Flag 0 Monitor Enable
15-8 —	Reserved
7 EMIOSFLG23_EN	EMIOS0 Output Flag 23 Monitor Enable
6 EMIOSFLG22_EN	EMIOS0 Output Flag 22 Monitor Enable
5 EMIOSFLG21_EN	EMIOS0 Output Flag 21 Monitor Enable
4 EMIOSFLG20_EN	EMIOS0 Output Flag 20 Monitor Enable
3 EMIOSFLG19_EN	EMIOS0 Output Flag 19 Monitor Enable
2 EMIOSFLG18_EN	EMIOS0 Output Flag 18 Monitor Enable
1 EMIOSFLG17_EN	EMIOS0 Output Flag 17 Monitor Enable
0	EMIOS0 Output Flag 16 Monitor Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
EMIOSFLG16_EN	

10.6.15 MUX1 MISC ENABLE (MUX1_MISC_EN)

Offset

Register	Offset
MUX1_MISC_EN	10Ch

Function

NOTE

This register is reserved for S32K3x4.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved					BCTU ADC...	BCTU ADC...	BCTU ADC...	BCTU LIS...	BCTU FIF...	BCTU FIF...	BCTU ADC...	BCTU ADC...	BCTU ADC...	BCTU FIF...	BCTU FIF...
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved				LPUA RT3...	LPUA RT2...	LPUA RT1...	LPUA RT0...	Reserved					ADC2 EOC...	ADC1 EOC...	ADC0 EOC...
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Fields

Field	Function
31-27 —	Reserved
26 BCTUADC2INT_EN	BCTU ADC2DR Interrupt Request Monitor Enable
25 BCTUADC1INT_EN	BCTU ADC1DR Interrupt Request Monitor Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
24 BCTUADC0INT_EN	BCTU ADC0DR Interrupt Request Monitor Enable
23 BCTULISTINT_EN	BCTU Conversion List Interrupt Request Enable
22 BCTUFIFO1INT_EN	BCTU FIFO0 Interrupt Request Monitor Enable
21 BCTUFIFO0INT_EN	BCTU FIFO0 Interrupt Request Monitor Enable
20 BCTUADC2DMA_EN	BCTU ADC2DR DMA Request Monitor Enable
19 BCTUADC1DMA_EN	BCTU ADC1DR DMA Request Monitor Enable
18 BCTUADC0DMA_EN	BCTU ADC0DR DMA Request Monitor Enable
17 BCTUFIFO1DMA_EN	BCTU FIFO1 DMA Request Monitor Enable
16 BCTUFIFO0DMA_EN	BCTU FIFO0 DMA Request Monitor Enable
15-12 —	Reserved
11 LPUART3TRG_EN	LPUART3 Output Trigger Monitor Enable
10	LPUART2 Output Trigger Monitor Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
LPUART2TRG_EN	
9 LPUART1TRG_EN	LPUART1 Output Trigger Monitor Enable
8 LPUART0TRG_EN	LPUART0 Output Trigger Monitor Enable
7-3 —	Reserved
2 ADC2EOC_EN	ADC2 End of Conversion Trigger Monitor
1 ADC1EOC_EN	ADC1 End of Conversion Trigger Monitor
0 ADC0EOC_EN	ADC0 End of Conversion Trigger Monitor

10.6.16 MUX2 EMIOS ENABLE (MUX2_EMIOS_EN)

Offset

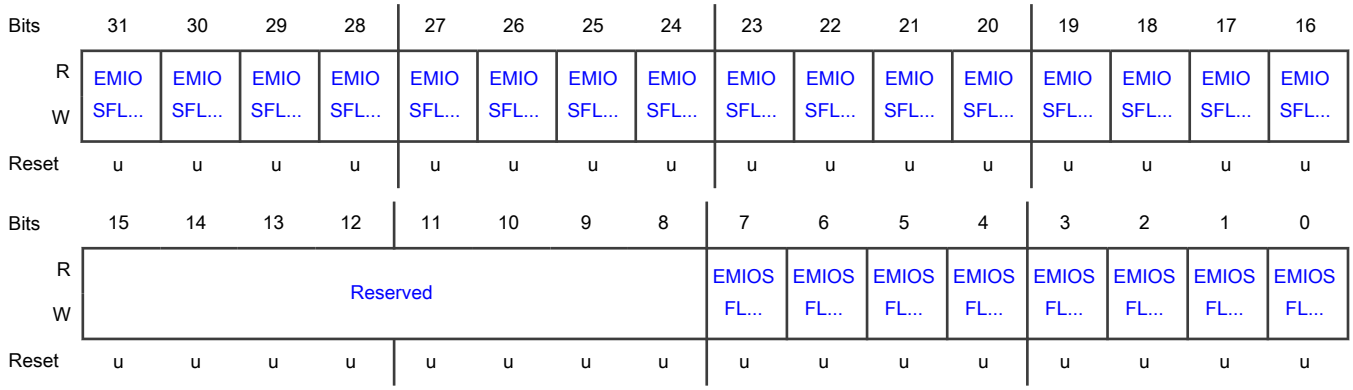
Register	Offset
MUX2_EMIOS_EN	110h

Function

NOTE

This register is reserved for S32K3x4, S32K310, S32K311, S32K312, S32K322, S32K341 and S32K342.

Diagram



Fields

Field	Function
31 EMIOSFLG15_EN	EMIOS0 Output Flag 15 Monitor Enable
30 EMIOSFLG14_EN	EMIOS0 Output Flag 14 Monitor Enable
29 EMIOSFLG13_EN	EMIOS0 Output Flag 13 Monitor Enable
28 EMIOSFLG12_EN	EMIOS0 Output Flag 12 Monitor Enable
27 EMIOSFLG11_EN	EMIOS0 Output Flag 11 Monitor Enable
26 EMIOSFLG10_EN	EMIOS0 Output Flag 10 Monitor Enable
25 EMIOSFLG9_EN	EMIOS0 Output Flag 9 Monitor Enable
24 EMIOSFLG8_EN	EMIOS0 Output Flag 8 Monitor Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
23 EMIOSFLG7_EN	EMIOS0 Output Flag 7 Monitor Enable
22 EMIOSFLG6_EN	EMIOS0 Output Flag 6 Monitor Enable
21 EMIOSFLG5_EN	EMIOS0 Output Flag 5 Monitor Enable
20 EMIOSFLG4_EN	EMIOS0 Output Flag 4 Monitor Enable
19 EMIOSFLG3_EN	EMIOS0 Output Flag 3 Monitor Enable
18 EMIOSFLG2_EN	EMIOS0 Output Flag 2 Monitor Enable
17 EMIOSFLG1_EN	EMIOS0 Output Flag 1 Monitor Enable
16 EMIOSFLG0_EN	EMIOS0 Output Flag 0 Monitor Enable
15-8 —	Reserved
7 EMIOSFLG23_EN	EMIOS0 Output Flag 23 Monitor Enable
6 EMIOSFLG22_EN	EMIOS0 Output Flag 22 Monitor Enable
5	EMIOS0 Output Flag 21 Monitor Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
EMIOSFLG21_EN	
4 EMIOSFLG20_EN	EMIOS0 Output Flag 20 Monitor Enable
3 EMIOSFLG19_EN	EMIOS0 Output Flag 19 Monitor Enable
2 EMIOSFLG18_EN	EMIOS0 Output Flag 18 Monitor Enable
1 EMIOSFLG17_EN	EMIOS0 Output Flag 17 Monitor Enable
0 EMIOSFLG16_EN	EMIOS0 Output Flag 16 Monitor Enable

10.6.17 MUX2 MISC ENABLE (MUX2_MISC_EN)

Offset

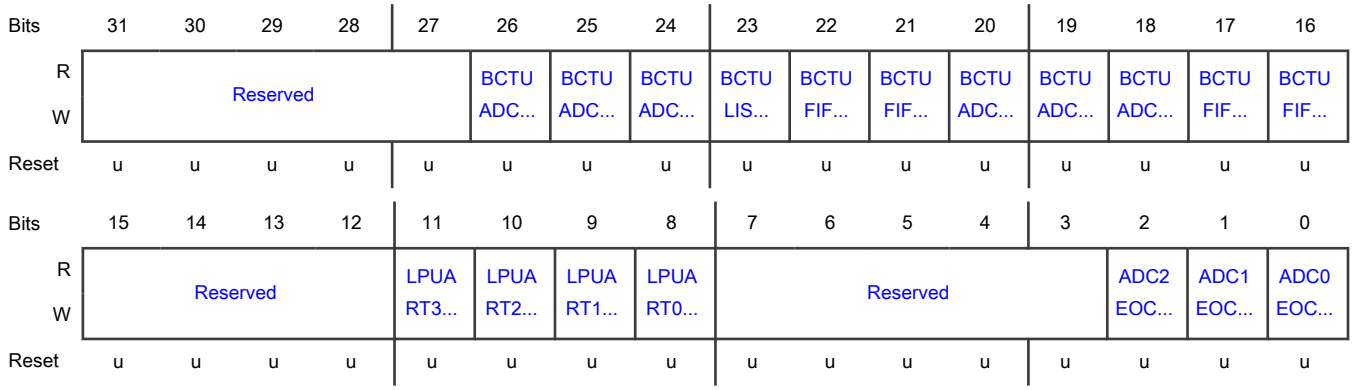
Register	Offset
MUX2_MISC_EN	114h

Function

NOTE

This register is reserved for S32K3x4, S32K310, S32K311, S32K312, S32K322, S32K341 and S32K342.

Diagram



Fields

Field	Function
31-27 —	Reserved
26 BCTUADC2INT_EN	BCTU ADC2DR Interrupt Request Monitor Enable
25 BCTUADC1INT_EN	BCTU ADC1DR Interrupt Request Monitor Enable
24 BCTUADC0INT_EN	BCTU ADC0DR Interrupt Request Monitor Enable
23 BCTULISTINT_EN	BCTU Conversion List Interrupt Request Enable
22 BCTUFIFO1INT_EN	BCTU FIFO0 Interrupt Request Monitor Enable
21 BCTUFIFO0INT_EN	BCTU FIFO0 Interrupt Request Monitor Enable
20 BCTUADC2DMA_EN	BCTU ADC2DR DMA Request Monitor Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
19 BCTUADC1DMA_EN	BCTU ADC1DR DMA Request Monitor Enable
18 BCTUADC0DMA_EN	BCTU ADC0DR DMA Request Monitor Enable
17 BCTUFIFO1DMA_EN	BCTU FIFO1 DMA Request Monitor Enable
16 BCTUFIFO0DMA_EN	BCTU FIFO0 DMA Request Monitor Enable
15-12 —	Reserved
11 LPUART3TRG_EN	LPUART3 Output Trigger Monitor Enable
10 LPUART2TRG_EN	LPUART2 Output Trigger Monitor Enable
9 LPUART1TRG_EN	LPUART1 Output Trigger Monitor Enable
8 LPUART0TRG_EN	LPUART0 Output Trigger Monitor Enable
7-3 —	Reserved
2 ADC2EOC_EN	ADC2 End of Conversion Trigger Monitor
1 ADC1EOC_EN	ADC1 End of Conversion Trigger Monitor

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 ADC0EOC_EN	ADC0 End of Conversion Trigger Monitor

10.6.18 SIUL2 MCU ID Register #3 (MIDR3)

Offset

Register	Offset
MIDR3	200h

Function

NOTE

This register supports only 32-bit accesses. Byte and half-word accesses are not supported.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PROD_FAM_LET						PROD_FAM_NO									
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PART_NO_SUF						0		SYS_RAM_SIZE							
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Fields

Field	Function
31-26 PROD_FAM_LET	Product Family Letter Identifies the product family letter. 01_0011b - S
25-16 PROD_FAM_NO	Product Family Number Identifies the product family number. 00_0010_0000b - 32
15-10	Part Number Suffix

Table continues on the next page...

Table continued from the previous page...

Field	Function
PART_NO_SUF	Describes the part number suffix. 00_0000b - None
9-6 —	Reserved
5-0 SYS_RAM_SIZE	System RAM Size Total RAM size in SoC, including TCMs. 00_0010b - 128 kB 00_0011b - 192 kB 00_0100b - 256 kB 00_0110b - 512 kB 00_1100b - 1152 kB

10.6.19 SIUL2 MCU ID Register #4 (MIDR4)

Offset

Register	Offset
MIDR4	204h

Function

NOTE

This register supports only 32-bit accesses. Byte and half-word accesses are not supported.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CORE_PLAT_FET_1		0											CORE_PLAT_FET_2		
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Fields

Field	Function
31-16 —	Reserved
15-14 CORE_PLAT_FET_1	Core Platform Options Feature Core platform options feature. **NOTE** The 5-bit field is a concatenation of bits 15-14 (CORE_PLAT_FET_1) and 2-0 (CORE_PLAT_FET_2).
13-3 —	Reserved
2-0 CORE_PLAT_FET_2	Core Platform Options Feature Core platform options feature. For field values see description of CORE_PLAT_FET_1

00000b	1x M7 core
00001b	2x M7 cores
00010b	1x M7 LS core
00011b	1x M7 LS core + 1x M7 core
00100b	3x M7 cores
01001b	2x M7 LS cores + 1x M7 core or 1x M7 LS core + 3x M7 cores

10.6.20 Multiplexed Signal Configuration (MSCR0 - MSCR323)

Offset

Register	Offset
MSCR0	240h
MSCR1	244h
MSCR2	248h
MSCR3	24Ch
MSCR4	250h
MSCR5	254h
MSCR6	258h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MSCR7	25Ch
MSCR8	260h
MSCR9	264h
MSCR10	268h
MSCR11	26Ch
MSCR12	270h
MSCR13	274h
MSCR14	278h
MSCR15	27Ch
MSCR16	280h
MSCR17	284h
MSCR18	288h
MSCR19	28Ch
MSCR20	290h
MSCR21	294h
MSCR22	298h
MSCR23	29Ch
MSCR24	2A0h
MSCR25	2A4h
MSCR26	2A8h
MSCR27	2ACh
MSCR28	2B0h
MSCR29	2B4h
MSCR30	2B8h
MSCR31	2BCh
MSCR32	2C0h
MSCR33	2C4h
MSCR34	2C8h
MSCR35	2CCh
MSCR36	2D0h
MSCR37	2D4h
MSCR40	2E0h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MSCR41	2E4h
MSCR42	2E8h
MSCR43	2ECh
MSCR44	2F0h
MSCR45	2F4h
MSCR46	2F8h
MSCR47	2FCh
MSCR48	300h
MSCR49	304h
MSCR50	308h
MSCR51	30Ch
MSCR52	310h
MSCR53	314h
MSCR54	318h
MSCR55	31Ch
MSCR56	320h
MSCR57	324h
MSCR58	328h
MSCR59	32Ch
MSCR60	330h
MSCR61	334h
MSCR62	338h
MSCR63	33Ch
MSCR64	340h
MSCR65	344h
MSCR66	348h
MSCR67	34Ch
MSCR68	350h
MSCR69	354h
MSCR70	358h
MSCR71	35Ch
MSCR72	360h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MSCR73	364h
MSCR74	368h
MSCR75	36Ch
MSCR76	370h
MSCR77	374h
MSCR78	378h
MSCR79	37Ch
MSCR80	380h
MSCR81	384h
MSCR82	388h
MSCR83	38Ch
MSCR84	390h
MSCR85	394h
MSCR86	398h
MSCR87	39Ch
MSCR88	3A0h
MSCR89	3A4h
MSCR90	3A8h
MSCR91	3ACh
MSCR92	3B0h
MSCR93	3B4h
MSCR94	3B8h
MSCR95	3BCh
MSCR96	3C0h
MSCR97	3C4h
MSCR98	3C8h
MSCR99	3CCh
MSCR100	3D0h
MSCR101	3D4h
MSCR102	3D8h
MSCR103	3DCh
MSCR104	3E0h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MSCR105	3E4h
MSCR106	3E8h
MSCR107	3ECh
MSCR108	3F0h
MSCR109	3F4h
MSCR110	3F8h
MSCR111	3FCh
MSCR112	400h
MSCR113	404h
MSCR114	408h
MSCR115	40Ch
MSCR116	410h
MSCR117	414h
MSCR118	418h
MSCR119	41Ch
MSCR120	420h
MSCR121	424h
MSCR122	428h
MSCR123	42Ch
MSCR124	430h
MSCR125	434h
MSCR126	438h
MSCR127	43Ch
MSCR128	440h
MSCR129	444h
MSCR130	448h
MSCR131	44Ch
MSCR132	450h
MSCR133	454h
MSCR134	458h
MSCR135	45Ch
MSCR136	460h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MSCR137	464h
MSCR138	468h
MSCR139	46Ch
MSCR140	470h
MSCR142	478h
MSCR143	47Ch
MSCR144	480h
MSCR145	484h
MSCR146	488h
MSCR147	48Ch
MSCR148	490h
MSCR149	494h
MSCR150	498h
MSCR151	49Ch
MSCR152	4A0h
MSCR153	4A4h
MSCR154	4A8h
MSCR155	4ACh
MSCR156	4B0h
MSCR157	4B4h
MSCR158	4B8h
MSCR159	4BCh
MSCR160	4C0h
MSCR161	4C4h
MSCR162	4C8h
MSCR163	4CCh
MSCR164	4D0h
MSCR165	4D4h
MSCR166	4D8h
MSCR167	4DCh
MSCR168	4E0h
MSCR169	4E4h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MSCR170	4E8h
MSCR171	4ECh
MSCR172	4F0h
MSCR173	4F4h
MSCR174	4F8h
MSCR175	4FCh
MSCR176	500h
MSCR177	504h
MSCR178	508h
MSCR179	50Ch
MSCR180	510h
MSCR181	514h
MSCR182	518h
MSCR183	51Ch
MSCR184	520h
MSCR185	524h
MSCR186	528h
MSCR187	52Ch
MSCR188	530h
MSCR189	534h
MSCR190	538h
MSCR191	53Ch
MSCR192	540h
MSCR193	544h
MSCR194	548h
MSCR195	54Ch
MSCR196	550h
MSCR197	554h
MSCR198	558h
MSCR199	55Ch
MSCR200	560h
MSCR201	564h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MSCR202	568h
MSCR203	56Ch
MSCR204	570h
MSCR205	574h
MSCR206	578h
MSCR207	57Ch
MSCR208	580h
MSCR209	584h
MSCR210	588h
MSCR211	58Ch
MSCR212	590h
MSCR213	594h
MSCR214	598h
MSCR215	59Ch
MSCR216	5A0h
MSCR217	5A4h
MSCR218	5A8h
MSCR219	5ACh
MSCR220	5B0h
MSCR221	5B4h
MSCR222	5B8h
MSCR223	5BCh
MSCR224	5C0h
MSCR225	5C4h
MSCR226	5C8h
MSCR227	5CCh
MSCR228	5D0h
MSCR229	5D4h
MSCR230	5D8h
MSCR231	5DCh
MSCR232	5E0h
MSCR233	5E4h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MSCR234	5E8h
MSCR235	5ECh
MSCR236	5F0h
MSCR237	5F4h
MSCR238	5F8h
MSCR239	5FCh
MSCR240	600h
MSCR241	604h
MSCR242	608h
MSCR243	60Ch
MSCR244	610h
MSCR245	614h
MSCR246	618h
MSCR247	61Ch
MSCR248	620h
MSCR249	624h
MSCR250	628h
MSCR251	62Ch
MSCR252	630h
MSCR253	634h
MSCR254	638h
MSCR255	63Ch
MSCR256	640h
MSCR257	644h
MSCR258	648h
MSCR259	64Ch
MSCR260	650h
MSCR261	654h
MSCR262	658h
MSCR263	65Ch
MSCR264	660h
MSCR265	664h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MSCR266	668h
MSCR267	66Ch
MSCR268	670h
MSCR269	674h
MSCR270	678h
MSCR271	67Ch
MSCR272	680h
MSCR273	684h
MSCR274	688h
MSCR275	68Ch
MSCR276	690h
MSCR277	694h
MSCR278	698h
MSCR279	69Ch
MSCR280	6A0h
MSCR281	6A4h
MSCR282	6A8h
MSCR283	6ACh
MSCR284	6B0h
MSCR285	6B4h
MSCR286	6B8h
MSCR287	6BCh
MSCR288	6C0h
MSCR289	6C4h
MSCR290	6C8h
MSCR291	6CCh
MSCR292	6D0h
MSCR293	6D4h
MSCR294	6D8h
MSCR295	6DCh
MSCR296	6E0h
MSCR297	6E4h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MSCR298	6E8h
MSCR299	6ECh
MSCR300	6F0h
MSCR301	6F4h
MSCR302	6F8h
MSCR303	6FCh
MSCR304	700h
MSCR305	704h
MSCR306	708h
MSCR307	70Ch
MSCR308	710h
MSCR309	714h
MSCR310	718h
MSCR311	71Ch
MSCR312	720h
MSCR313	724h
MSCR314	728h
MSCR315	72Ch
MSCR316	730h
MSCR317	734h
MSCR318	738h
MSCR319	73Ch
MSCR320	740h
MSCR321	744h
MSCR322	748h
MSCR323	74Ch

Function

Selects the source signal connected to the register's associated destination, which is a chip output pin or a chip pin that can be configured as an output. It also specifies the electrical properties of the associated pin.

This register supports only 32-bit accesses. Byte and half-word write accesses are not supported.

For chip-pin MSCR assignments and pin types, see the IOMUX file attached to this document.

NOTE

- Configure these registers only during application initialization; you must not modify them during application runtime.
- Accessing a reserved MSCR*n* register generates a transfer error.
- These registers are a part of the SIUL memory map but the physical implementation of these registers is a part of the IOMUX RTL.
- SIUL2 interprets accesses to MSCR*n* at the module level.

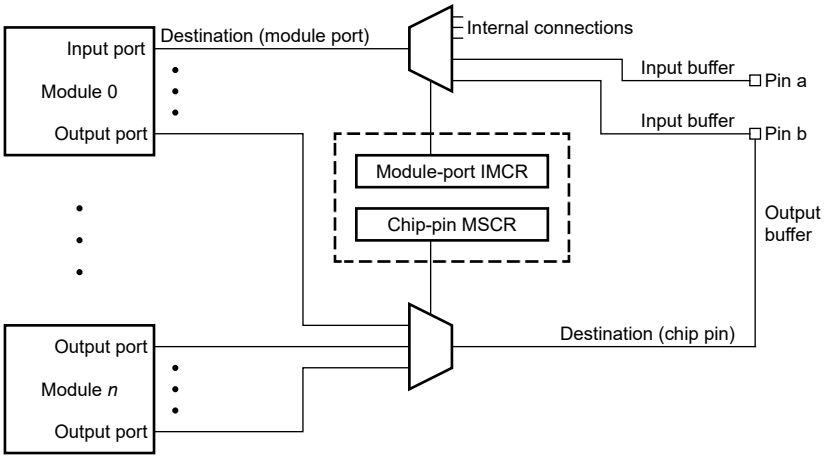


Figure 26. MSCR and IMCR port and pin connection

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0										OBE	0	IBE	0	INV	PKE
W																
Reset	See Register reset values.															

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	SRC	PUE	0	PUS	0		DSE	0	IFE	SMC	0	SSS_3	SSS_2	SSS_1	SSS_0
W																
Reset	See Register reset values.															

Register reset values

Register	Reset value
MSCR0–MSCR3	0000_0000h
MSCR4	0008_2827h
MSCR5–MSCR9	0000_0000h
MSCR10	0000_0127h

Table continues on the next page...

Table continued from the previous page...

Register	Reset value
MSCR11	0000_0000h
MSCR12	0000_0003h
MSCR13–MSCR65	0000_0000h
MSCR38–MSCR39	—
MSCR66–MSCR67	0000_4000h
MSCR68	0008_2000h
MSCR69	0008_2800h
MSCR70–MSCR75	0000_0000h
MSCR76	0000_4000h
MSCR77–MSCR79	0000_0000h
MSCR80	0000_4000h
MSCR81–MSCR100	0000_0000h
MSCR101–MSCR103	0000_4000h
MSCR104–MSCR105	0000_0000h
MSCR106–MSCR108	0000_4000h
MSCR109–MSCR135	0000_0000h
MSCR136	0000_4000h
MSCR137–MSCR323	0000_0000h
MSCR141	—

Fields

Field	Function
31-22 —	Reserved
21 OBE	GPIO Output Buffer Enable Applies only to digital pins. Otherwise this bit is reserved. 0b - Output driver disabled 1b - Output driver enabled
20 —	Reserved
19	Input Buffer Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
IBE	Used only when the associated destination is a chip pin. Enables the associated pin's input buffer. 0b - Disabled 1b - Enabled
18 —	Reserved
17 INV	Invert Inverts the signal selected by SSS before transmitting it to the associated destination (chip pin or module port). 0b - Don't invert 1b - Invert
16 PKE	Pad keeping enable Pad keeping enable 0b - Disabled 1b - Enabled
15 —	Reserved
14 SRC	Slew Rate Control 0b - Fastest setting 1b - Slowest setting
13 PUE	Pull Enable Enables the pull function. Used only when the associated destination is a chip pin. 0b - Disabled 1b - Enabled
12 —	Reserved
11 PUS	Pull Select Determines whether the pull function is a pullup or pulldown when the pull function is enabled by the PUE field. Used only when the associated destination is a chip pin. 0b - Pull down 1b - Pull up
10-9	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
8 DSE	DSE Drive strength enable 0b - Disabled 1b - Enabled
7 —	Reserved
6 IFE	IFE Input filter enable **NOTE** This field is supported for RESET pad only (PTA5). 0b - Disabled 1b - Enabled
5 SMC	Safe Mode Control Used only when the associated destination is a chip pin. Specifies whether the chip disables the pin's output buffer when the chip enters Safe mode. 0b - Disable (The output buffer returns to its previous state when the chip leaves Safe mode.) 1b - Don't disable
4 —	Reserved
3 SSS_3	Source Signal Select_3 Selects a function for the pad. Refer to "SSS" column of the 'IO Signal Table' tab of the IOMUX spreadsheet attachment.
2 SSS_2	Source Signal Select_2 Selects a function for the pad. Refer to "SSS" column of the 'IO Signal Table' tab of the IOMUX spreadsheet attachment.
1 SSS_1	Source Signal Select_1 Selects a function for the pad. Refer to "SSS" column of the 'IO Signal Table' tab of the IOMUX spreadsheet attachment.
0 SSS_0	Source Signal Select_0 Selects a function for the pad. Refer to "SSS" column of the 'IO Signal Table' tab of the IOMUX spreadsheet attachment.

10.6.21 Input Multiplexed Signal Configuration (IMCR0 - IMCR473)

Offset

Register	Offset
IMCR0	A40h
IMCR1	A44h
IMCR2	A48h
IMCR3	A4Ch
IMCR4	A50h
IMCR5	A54h
IMCR16	A80h
IMCR17	A84h
IMCR18	A88h
IMCR19	A8Ch
IMCR20	A90h
IMCR21	A94h
IMCR22	A98h
IMCR23	A9Ch
IMCR24	AA0h
IMCR25	AA4h
IMCR26	AA8h
IMCR27	AACh
IMCR28	AB0h
IMCR29	AB4h
IMCR30	AB8h
IMCR31	ABCh
IMCR32	AC0h
IMCR33	AC4h
IMCR34	AC8h
IMCR35	ACCh
IMCR36	AD0h
IMCR37	AD4h
IMCR38	AD8h
IMCR39	ADCh

Table continues on the next page...

Table continued from the previous page...

Register	Offset
IMCR40	AE0h
IMCR41	AE4h
IMCR42	AE8h
IMCR43	AECh
IMCR44	AF0h
IMCR45	AF4h
IMCR46	AF8h
IMCR47	AFCh
IMCR48	B00h
IMCR49	B04h
IMCR50	B08h
IMCR51	B0Ch
IMCR52	B10h
IMCR53	B14h
IMCR54	B18h
IMCR55	B1Ch
IMCR56	B20h
IMCR57	B24h
IMCR58	B28h
IMCR59	B2Ch
IMCR60	B30h
IMCR61	B34h
IMCR62	B38h
IMCR63	B3Ch
IMCR64	B40h
IMCR65	B44h
IMCR66	B48h
IMCR67	B4Ch
IMCR68	B50h
IMCR69	B54h
IMCR70	B58h
IMCR71	B5Ch

Table continues on the next page...

Table continued from the previous page...

Register	Offset
IMCR80	B80h
IMCR81	B84h
IMCR82	B88h
IMCR83	B8Ch
IMCR84	B90h
IMCR85	B94h
IMCR86	B98h
IMCR87	B9Ch
IMCR88	BA0h
IMCR89	BA4h
IMCR90	BA8h
IMCR91	BACh
IMCR92	BB0h
IMCR93	BB4h
IMCR94	BB8h
IMCR95	BBCh
IMCR96	BC0h
IMCR97	BC4h
IMCR98	BC8h
IMCR99	BCCh
IMCR100	BD0h
IMCR101	BD4h
IMCR102	BD8h
IMCR103	BDCh
IMCR112	C00h
IMCR113	C04h
IMCR114	C08h
IMCR115	C0Ch
IMCR116	C10h
IMCR117	C14h
IMCR118	C18h
IMCR119	C1Ch

Table continues on the next page...

Table continued from the previous page...

Register	Offset
IMCR120	C20h
IMCR121	C24h
IMCR122	C28h
IMCR123	C2Ch
IMCR124	C30h
IMCR125	C34h
IMCR126	C38h
IMCR127	C3Ch
IMCR128	C40h
IMCR129	C44h
IMCR130	C48h
IMCR131	C4Ch
IMCR132	C50h
IMCR133	C54h
IMCR134	C58h
IMCR135	C5Ch
IMCR144	C80h
IMCR145	C84h
IMCR146	C88h
IMCR147	C8Ch
IMCR148	C90h
IMCR149	C94h
IMCR152	CA0h
IMCR153	CA4h
IMCR154	CA8h
IMCR155	CACh
IMCR156	CB0h
IMCR157	CB4h
IMCR158	CB8h
IMCR159	CBCh
IMCR160	CC0h
IMCR161	CC4h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
IMCR162	CC8h
IMCR163	CCCh
IMCR164	CD0h
IMCR165	CD4h
IMCR166	CD8h
IMCR167	CDCh
IMCR168	CE0h
IMCR169	CE4h
IMCR170	CE8h
IMCR171	CECh
IMCR172	CF0h
IMCR173	CF4h
IMCR174	CF8h
IMCR175	CFCh
IMCR176	D00h
IMCR177	D04h
IMCR178	D08h
IMCR179	D0Ch
IMCR180	D10h
IMCR181	D14h
IMCR182	D18h
IMCR183	D1Ch
IMCR184	D20h
IMCR185	D24h
IMCR186	D28h
IMCR187	D2Ch
IMCR188	D30h
IMCR189	D34h
IMCR190	D38h
IMCR191	D3Ch
IMCR192	D40h
IMCR193	D44h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
IMCR194	D48h
IMCR195	D4Ch
IMCR196	D50h
IMCR197	D54h
IMCR198	D58h
IMCR199	D5Ch
IMCR200	D60h
IMCR201	D64h
IMCR202	D68h
IMCR211	D8Ch
IMCR212	D90h
IMCR213	D94h
IMCR214	D98h
IMCR215	D9Ch
IMCR216	DA0h
IMCR217	DA4h
IMCR218	DA8h
IMCR219	DACh
IMCR220	DB0h
IMCR221	DB4h
IMCR222	DB8h
IMCR223	DBCh
IMCR224	DC0h
IMCR225	DC4h
IMCR226	DC8h
IMCR227	DCCh
IMCR228	DD0h
IMCR229	DD4h
IMCR230	DD8h
IMCR231	DDCh
IMCR232	DE0h
IMCR233	DE4h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
IMCR234	DE8h
IMCR235	DECh
IMCR236	DF0h
IMCR237	DF4h
IMCR238	DF8h
IMCR239	DFCh
IMCR240	E00h
IMCR241	E04h
IMCR242	E08h
IMCR243	E0Ch
IMCR244	E10h
IMCR245	E14h
IMCR246	E18h
IMCR247	E1Ch
IMCR248	E20h
IMCR249	E24h
IMCR250	E28h
IMCR251	E2Ch
IMCR252	E30h
IMCR253	E34h
IMCR254	E38h
IMCR255	E3Ch
IMCR256	E40h
IMCR257	E44h
IMCR258	E48h
IMCR259	E4Ch
IMCR260	E50h
IMCR261	E54h
IMCR262	E58h
IMCR263	E5Ch
IMCR264	E60h
IMCR265	E64h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
IMCR266	E68h
IMCR267	E6Ch
IMCR268	E70h
IMCR289	EC4h
IMCR290	EC8h
IMCR291	ECCh
IMCR292	ED0h
IMCR293	ED4h
IMCR294	ED8h
IMCR295	EDCh
IMCR296	EE0h
IMCR297	EE4h
IMCR298	EE8h
IMCR299	EECh
IMCR300	EF0h
IMCR301	EF4h
IMCR302	EF8h
IMCR303	EFCh
IMCR304	F00h
IMCR305	F04h
IMCR306	F08h
IMCR307	F0Ch
IMCR308	F10h
IMCR309	F14h
IMCR315	F2Ch
IMCR316	F30h
IMCR317	F34h
IMCR318	F38h
IMCR319	F3Ch
IMCR320	F40h
IMCR321	F44h
IMCR322	F48h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
IMCR323	F4Ch
IMCR324	F50h
IMCR325	F54h
IMCR343	F9Ch
IMCR344	FA0h
IMCR345	FA4h
IMCR346	FA8h
IMCR347	FACh
IMCR348	FB0h
IMCR349	FB4h
IMCR350	FB8h
IMCR351	FBCh
IMCR352	FC0h
IMCR353	FC4h
IMCR354	FC8h
IMCR355	FCCh
IMCR356	FD0h
IMCR357	FD4h
IMCR358	FD8h
IMCR359	FDCh
IMCR360	FE0h
IMCR361	FE4h
IMCR362	FE8h
IMCR363	FECh
IMCR364	FF0h
IMCR365	FF4h
IMCR366	FF8h
IMCR367	FFCh
IMCR368	1000h
IMCR369	1004h
IMCR370	1008h
IMCR373	1014h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
IMCR374	1018h
IMCR375	101Ch
IMCR376	1020h
IMCR377	1024h
IMCR378	1028h
IMCR389	1054h
IMCR398	1078h
IMCR399	107Ch
IMCR409	10A4h
IMCR410	10A8h
IMCR411	10ACh
IMCR412	10B0h
IMCR413	10B4h
IMCR414	10B8h
IMCR415	10BCh
IMCR416	10C0h
IMCR417	10C4h
IMCR418	10C8h
IMCR440	1120h
IMCR448	1140h
IMCR449	1144h
IMCR450	1148h
IMCR451	114Ch
IMCR452	1150h
IMCR453	1154h
IMCR454	1158h
IMCR455	115Ch
IMCR456	1160h
IMCR457	1164h
IMCR458	1168h
IMCR459	116Ch
IMCR460	1170h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
IMCR461	1174h
IMCR462	1178h
IMCR463	117Ch
IMCR464	1180h
IMCR465	1184h
IMCR466	1188h
IMCR467	118Ch
IMCR468	1190h
IMCR469	1194h
IMCR470	1198h
IMCR471	119Ch
IMCR472	11A0h
IMCR473	11A4h

Function

Selects the source signal connected to the register's associated destination, which is an internal module port that is an input port or can be configured as an input.

This register supports only 32-bit accesses. Byte and half-word write accesses are not supported.

For IMCR assignments and field values, see the IOMUX file attached to this document.

NOTE

- Configure these registers only during application initialization; you must not modify them during application runtime.
- Accessing a reserved IMCR*n* register generates a transfer error.
- These registers are a part of the SIUL memory map but the physical implementation of these registers is a part of the IOMUX RTL.
- SIUL2 interprets accesses to MSCR*n* at the module level.

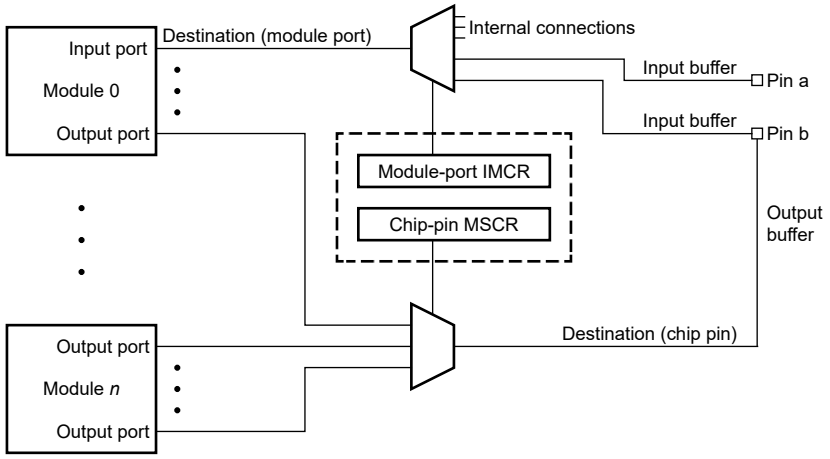


Figure 27. MSCR and IMCR port and pin connection

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SSS			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-4 —	Reserved
3-0 SSS	Source Signal Select Selects which source signal is connected to the associated destination (chip pin).

10.6.22 GPIO Pad Data Output (GPDO0 - GPDO323)

Offset

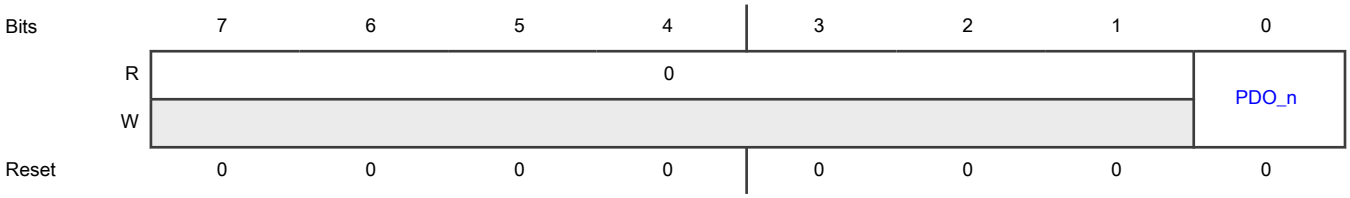
For n = 0 to 37; n = 40 to 140; n = 142 to 323:

Register	Offset
GPDOn	1300h + (n + 3 - 2 × (n mod 4))

Function

Writes 0 or 1 to a single GPIO pad with a byte access and supports 8-, 16-, and 32-bit accesses.

Diagram



Fields

Field	Function
7-1 —	Reserved
0 PDO_n	Pad Data Out Stores the data to be driven out on an external GPIO pad (controlled by this register) when you configure the pad as an output. It also represents PDO[*n*], where *n* is the instance of the register. 0b - Logic low value 1b - Logic high value

10.6.23 GPIO Pad Data Input (GPDI0 - GPDI323)

Offset

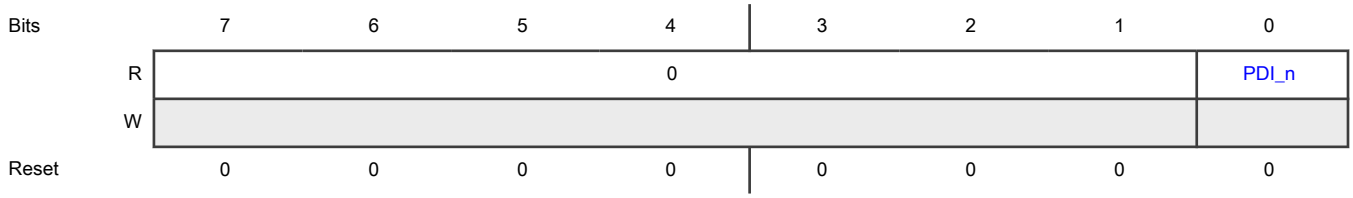
For n = 0 to 37; n = 40 to 140; n = 142 to 323:

Register	Offset
GPDIn	1500h + (n + 3 - 2 × (n mod 4))

Function

Reads the GPIO pad data with a byte access and supports 8-, 16-, and 32-bit accesses.

Diagram



Fields

Field	Function
7-1 —	Reserved
0 PDI_n	Pad Data In Stores the value of the external GPIO pad associated with this register. It represents PDI[*n*], where *n* is the instance of the register. 0b - Logic low 1b - Logic high

10.6.24 Parallel GPIO Pad Data Output (PGPDO0 - PGPDO3)

Offset

Register	Offset
PGPDO1	1700h
PGPDO0	1702h
PGPDO3	1704h

Function

Sets or clears the respective pads of the chip and supports 8-, 16-, and 32-bit accesses. This register also accesses the same physical resource as the PDO and MPGPDO address locations.

This register sets the values of all the output pins assigned to a chip port with a single 16-bit register write, while the GPDO*n* register sets the value on a specific pin with byte writes.

The access to this register's location is coherent with access to the bitwise GPDO*n*.

For a given PGPDO*x*[PPDO*y*], where *x* is the register instance index and *y* is the field index, the following equation shows the equivalent GPDO*n*[PDO_*n*] field:

PGPDO*x*[PPDO*y*] = GPDO(*x* × 16) + (15 - *y*)[PDO_(*x* × 16) + (15 - *y*)]

Following are some of the examples of mapping:

- PGPDO0[PPDO15] = GPDO0[PDO_0]
- PGPDO2[PPDO15] = GPDO32[PDO_32]
- PGPDO31[PPDO0] = GPDO511[PDO_511]

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PPDO 15	PPDO 14	PPDO 13	PPDO 12	PPDO 11	PPDO 10	PPDO 9	PPDO 8	PPDO 7	PPDO 6	PPDO 5	PPDO 4	PPDO 3	PPDO 2	PPDO 1	PPDO 0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15 PPDO15	Parallel Pad Data Out 15 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
14 PPDO14	Parallel Pad Data Out 14 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
13 PPDO13	Parallel Pad Data Out 13 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
12 PPDO12	Parallel Pad Data Out 12 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
11 PPDO11	Parallel Pad Data Out 11 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
10 PPDO10	Parallel Pad Data Out 10 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
9 PPDO9	Parallel Pad Data Out 9 Writes to or reads the data register that stores the value to be driven on the pad in Output mode.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Logic low 1b - Logic high
8 PPDO8	Parallel Pad Data Out 8 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
7 PPDO7	Parallel Pad Data Out 7 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
6 PPDO6	Parallel Pad Data Out 6 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
5 PPDO5	Parallel Pad Data Out 5 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
4 PPDO4	Parallel Pad Data Out 4 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
3 PPDO3	Parallel Pad Data Out 3 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
2 PPDO2	Parallel Pad Data Out 2 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
1	Parallel Pad Data Out 1

Table continues on the next page...

Table continued from the previous page...

Field	Function
PPDO1	Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
0 PPDO0	Parallel Pad Data Out 0 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high

10.6.25 Parallel GPIO Pad Data Output (PGPDO2)

Offset

Register	Offset
PGPDO2	1706h

Function

Sets or clears the respective pads of the chip and supports 8-, 16-, and 32-bit accesses. This register also accesses the same physical resource as the PDO and MPGPDO address locations.

This register sets the values of all the output pins assigned to a chip port with a single 16-bit register write, while the GPDO*n* register sets the value on a specific pin with byte writes.

The access to this register's location is coherent with access to the bitwise GPDO*n*.

For a given PGPDO*x*[PPDO*y*], where *x* is the register instance index and *y* is the field index, the following equation shows the equivalent GPDO*n*[PDO_*n*] field:

PGPDO*x*[PPDO*y*] = GPDO(*x* × 16) + (15 - *y*)[PDO_(*x* × 16) + (15 - *y*)]

Following are some of the examples of mapping:

- PGPDO0[PPDO15] = GPDO0[PDO_0]
- PGPDO2[PPDO15] = GPDO32[PDO_32]
- PGPDO31[PPDO0] = GPDO511[PDO_511]

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PPDO 15	PPDO 14	PPDO 13	PPDO 12	PPDO 11	PPDO 10	0	0	PPDO 7	PPDO 6	PPDO 5	PPDO 4	PPDO 3	PPDO 2	PPDO 1	PPDO 0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15 PPDO15	Parallel Pad Data Out 15 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
14 PPDO14	Parallel Pad Data Out 14 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
13 PPDO13	Parallel Pad Data Out 13 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
12 PPDO12	Parallel Pad Data Out 12 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
11 PPDO11	Parallel Pad Data Out 11 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
10 PPDO10	Parallel Pad Data Out 10 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
9 —	Reserved Always write zero to this field.

Table continues on the next page...

Table continued from the previous page...

Field	Function
8 —	Reserved Always write zero to this field.
7 PPDO7	Parallel Pad Data Out 7 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
6 PPDO6	Parallel Pad Data Out 6 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
5 PPDO5	Parallel Pad Data Out 5 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
4 PPDO4	Parallel Pad Data Out 4 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
3 PPDO3	Parallel Pad Data Out 3 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
2 PPDO2	Parallel Pad Data Out 2 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
1 PPDO1	Parallel Pad Data Out 1 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
0	Parallel Pad Data Out 0

Table continues on the next page...

Table continued from the previous page...

Field	Function
PPDO0	Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high

10.6.26 Parallel GPIO Pad Data Output (PGPDO4 - PGPDO9)

Offset

Register	Offset
PGPDO5	1708h
PGPDO4	170Ah
PGPDO7	170Ch
PGPDO6	170Eh
PGPDO9	1710h

Function

Sets or clears the respective pads of the chip and supports 8-, 16-, and 32-bit accesses. This register also accesses the same physical resource as the PDO and MPGPDO address locations.

This register sets the values of all the output pins assigned to a chip port with a single 16-bit register write, while the GPDO*n* register sets the value on a specific pin with byte writes.

The access to this register's location is coherent with access to the bitwise GPDO*n*.

For a given PGPDO*x*[PPDO*y*], where *x* is the register instance index and *y* is the field index, the following equation shows the equivalent GPDO*n*[PDO_*n*] field:

PGPDO*x*[PPDO*y*] = GPDO(*x* × 16) + (15 - *y*)[PDO_(*x* × 16) + (15 - *y*)]

Following are some of the examples of mapping:

- PGPDO0[PPDO15] = GPDO0[PDO_0]
- PGPDO2[PPDO15] = GPDO32[PDO_32]
- PGPDO31[PPDO0] = GPDO511[PDO_511]

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PPDO 15	PPDO 14	PPDO 13	PPDO 12	PPDO 11	PPDO 10	PPDO 9	PPDO 8	PPDO 7	PPDO 6	PPDO 5	PPDO 4	PPDO 3	PPDO 2	PPDO 1	PPDO 0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15 PPDO15	Parallel Pad Data Out 15 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
14 PPDO14	Parallel Pad Data Out 14 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
13 PPDO13	Parallel Pad Data Out 13 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
12 PPDO12	Parallel Pad Data Out 12 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
11 PPDO11	Parallel Pad Data Out 11 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
10 PPDO10	Parallel Pad Data Out 10 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
9 PPDO9	Parallel Pad Data Out 9 Writes to or reads the data register that stores the value to be driven on the pad in Output mode.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Logic low 1b - Logic high
8 PPDO8	Parallel Pad Data Out 8 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
7 PPDO7	Parallel Pad Data Out 7 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
6 PPDO6	Parallel Pad Data Out 6 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
5 PPDO5	Parallel Pad Data Out 5 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
4 PPDO4	Parallel Pad Data Out 4 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
3 PPDO3	Parallel Pad Data Out 3 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
2 PPDO2	Parallel Pad Data Out 2 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
1	Parallel Pad Data Out 1

Table continues on the next page...

Table continued from the previous page...

Field	Function
PPDO1	Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
0 PPDO0	Parallel Pad Data Out 0 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high

10.6.27 Parallel GPIO Pad Data Output (PGPDO8)

Offset

Register	Offset
PGPDO8	1712h

Function

Sets or clears the respective pads of the chip and supports 8-, 16-, and 32-bit accesses. This register also accesses the same physical resource as the PDO and MPGPDO address locations.

This register sets the values of all the output pins assigned to a chip port with a single 16-bit register write, while the GPDO*n* register sets the value on a specific pin with byte writes.

The access to this register's location is coherent with access to the bitwise GPDO*n*.

For a given PGPDO*x*[PPDO*y*], where *x* is the register instance index and *y* is the field index, the following equation shows the equivalent GPDO*n*[PDO_*n*] field:

PGPDO*x*[PPDO*y*] = GPDO(*x* × 16) + (15 - *y*)[PDO_(*x* × 16) + (15 - *y*)]

Following are some of the examples of mapping:

- PGPDO0[PPDO15] = GPDO0[PDO_0]
- PGPDO2[PPDO15] = GPDO32[PDO_32]
- PGPDO31[PPDO0] = GPDO511[PDO_511]

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PPDO 15	PPDO 14	PPDO 13	PPDO 12	PPDO 11	PPDO 10	PPDO 9	PPDO 8	PPDO 7	PPDO 6	PPDO 5	PPDO 4	PPDO 3	0	PPDO 1	PPDO 0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15 PPDO15	Parallel Pad Data Out 15 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
14 PPDO14	Parallel Pad Data Out 14 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
13 PPDO13	Parallel Pad Data Out 13 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
12 PPDO12	Parallel Pad Data Out 12 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
11 PPDO11	Parallel Pad Data Out 11 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
10 PPDO10	Parallel Pad Data Out 10 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
9 PPDO9	Parallel Pad Data Out 9 Writes to or reads the data register that stores the value to be driven on the pad in Output mode.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Logic low 1b - Logic high
8 PPDO8	Parallel Pad Data Out 8 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
7 PPDO7	Parallel Pad Data Out 7 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
6 PPDO6	Parallel Pad Data Out 6 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
5 PPDO5	Parallel Pad Data Out 5 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
4 PPDO4	Parallel Pad Data Out 4 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
3 PPDO3	Parallel Pad Data Out 3 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
2 —	Reserved Always write zero to this field.
1 PPDO1	Parallel Pad Data Out 1 Writes to or reads the data register that stores the value to be driven on the pad in Output mode.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Logic low 1b - Logic high
0 PPDO0	Parallel Pad Data Out 0 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high

10.6.28 Parallel GPIO Pad Data Output (PGPDO10 - PGPDO19)

Offset

For n = 10 to 19:

Register	Offset
PGPDOn	1714h + 2 × (n + 1 – 2 × (n mod 2))

Function

Sets or clears the respective pads of the chip and supports 8-, 16-, and 32-bit accesses. This register also accesses the same physical resource as the PDO and MPGPDO address locations.

This register sets the values of all the output pins assigned to a chip port with a single 16-bit register write, while the GPDO*n* register sets the value on a specific pin with byte writes.

The access to this register's location is coherent with access to the bitwise GPDO*n*.

For a given PGPDO*x*[PPDO*y*], where *x* is the register instance index and *y* is the field index, the following equation shows the equivalent GPDO*n*[PDO_*n*] field:

PGPDO*x*[PPDO*y*] = GPDO(*x* × 16) + (15 - *y*)[PDO_(*x* × 16) + (15 - *y*)]

Following are some of the examples of mapping:

- PGPDO0[PPDO15] = GPDO0[PDO_0]
- PGPDO2[PPDO15] = GPDO32[PDO_32]
- PGPDO31[PPDO0] = GPDO511[PDO_511]

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PPDO 15	PPDO 14	PPDO 13	PPDO 12	PPDO 11	PPDO 10	PPDO 9	PPDO 8	PPDO 7	PPDO 6	PPDO 5	PPDO 4	PPDO 3	PPDO 2	PPDO 1	PPDO 0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15 PPDO15	Parallel Pad Data Out 15 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
14 PPDO14	Parallel Pad Data Out 14 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
13 PPDO13	Parallel Pad Data Out 13 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
12 PPDO12	Parallel Pad Data Out 12 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
11 PPDO11	Parallel Pad Data Out 11 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
10 PPDO10	Parallel Pad Data Out 10 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
9 PPDO9	Parallel Pad Data Out 9 Writes to or reads the data register that stores the value to be driven on the pad in Output mode.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Logic low 1b - Logic high
8 PPDO8	Parallel Pad Data Out 8 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
7 PPDO7	Parallel Pad Data Out 7 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
6 PPDO6	Parallel Pad Data Out 6 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
5 PPDO5	Parallel Pad Data Out 5 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
4 PPDO4	Parallel Pad Data Out 4 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
3 PPDO3	Parallel Pad Data Out 3 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
2 PPDO2	Parallel Pad Data Out 2 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
1	Parallel Pad Data Out 1

Table continues on the next page...

Table continued from the previous page...

Field	Function
PPDO1	Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high
0 PPDO0	Parallel Pad Data Out 0 Writes to or reads the data register that stores the value to be driven on the pad in Output mode. 0b - Logic low 1b - Logic high

10.6.29 Parallel GPIO Pad Data Input (PGPDI0 - PGPDI3)

Offset

Register	Offset
PGPDI1	1740h
PGPDI0	1742h
PGPDI3	1744h

Function

Holds the synchronized input value from the pads and supports 8-, 16-, and 32-bit accesses.

This register reads the values of all input pins assigned to a chip port with a single 16-bit register read, while GPDI*n* registers read the value on a specific pin with a byte read.

The access to this register's location is coherent with the access to the bitwise GPDI*n*.

For a given PGPDI*x*[PPDI*y*], where *x* is the register instance index and *y* is the field index, the following equation shows the equivalent GPDI*n*[PDI_*n*] field:

PGPDI*x*[PPDI*y*] = GPDI(*x* × 16) + (15 - *y*)[PDI_(*x* × 16) + (15 - *y*)]

Following are some of the examples of mapping:

- PGPDI0[PPDI15] = GPDI0[PDI_0]
- PGPDI2[PPDI15] = GPDI32[PDI_32]
- PGPDI31[PPDI0] = GPDI511[PDI_511]

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PPDI15	PPDI14	PPDI13	PPDI12	PPDI11	PPDI10	PPDI9	PPDI8	PPDI7	PPDI6	PPDI5	PPDI4	PPDI3	PPDI2	PPDI1	PPDI0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15 PPDI15	Parallel Pad Data Input 15 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
14 PPDI14	Parallel Pad Data Input 14 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
13 PPDI13	Parallel Pad Data Input 13 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
12 PPDI12	Parallel Pad Data Input 12 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
11 PPDI11	Parallel Pad Data Input 11 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
10 PPDI10	Parallel Pad Data Input 10 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
9	Parallel Pad Data Input 9

Table continues on the next page...

Table continued from the previous page...

Field	Function
PPDI9	Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
8 PPDI8	Parallel Pad Data Input 8 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
7 PPDI7	Parallel Pad Data Input 7 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
6 PPDI6	Parallel Pad Data Input 6 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
5 PPDI5	Parallel Pad Data Input 5 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
4 PPDI4	Parallel Pad Data Input 4 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
3 PPDI3	Parallel Pad Data Input 3 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
2 PPDI2	Parallel Pad Data Input 2 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 PPDI1	Parallel Pad Data Input 1 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
0 PPDI0	Parallel Pad Data Input 0 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high

10.6.30 Parallel GPIO Pad Data Input (PGPDI2)

Offset

Register	Offset
PGPDI2	1746h

Function

Holds the synchronized input value from the pads and supports 8-, 16-, and 32-bit accesses.

This register reads the values of all input pins assigned to a chip port with a single 16-bit register read, while GPDI*n* registers read the value on a specific pin with a byte read.

The access to this register's location is coherent with the access to the bitwise GPDI*n*.

For a given PGPDI*x*[PPDI*y*], where *x* is the register instance index and *y* is the field index, the following equation shows the equivalent GPDI*n*[PDI_*n*] field:

PGPDI*x*[PPDI*y*] = GPDI(*x* × 16) + (15 - *y*)[PDI_(*x* × 16) + (15 - *y*)]

Following are some of the examples of mapping:

- PGPDI0[PPDI15] = GPDI0[PDI_0]
- PGPDI2[PPDI15] = GPDI32[PDI_32]
- PGPDI31[PPDI0] = GPDI511[PDI_511]

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PPDI15	PPDI14	PPDI13	PPDI12	PPDI11	PPDI10	Reserved	Reserved	PPDI7	PPDI6	PPDI5	PPDI4	PPDI3	PPDI2	PPDI1	PPDI0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15 PPDI15	Parallel Pad Data Input 15 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
14 PPDI14	Parallel Pad Data Input 14 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
13 PPDI13	Parallel Pad Data Input 13 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
12 PPDI12	Parallel Pad Data Input 12 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
11 PPDI11	Parallel Pad Data Input 11 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
10 PPDI10	Parallel Pad Data Input 10 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
9	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
8 —	Reserved
7 PPDI7	Parallel Pad Data Input 7 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
6 PPDI6	Parallel Pad Data Input 6 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
5 PPDI5	Parallel Pad Data Input 5 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
4 PPDI4	Parallel Pad Data Input 4 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
3 PPDI3	Parallel Pad Data Input 3 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
2 PPDI2	Parallel Pad Data Input 2 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
1 PPDI1	Parallel Pad Data Input 1 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 PPDI0	Parallel Pad Data Input 0 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high

10.6.31 Parallel GPIO Pad Data Input (PGPDI4 - PGPDI9)

Offset

Register	Offset
PGPDI5	1748h
PGPDI4	174Ah
PGPDI7	174Ch
PGPDI6	174Eh
PGPDI9	1750h

Function

Holds the synchronized input value from the pads and supports 8-, 16-, and 32-bit accesses.

This register reads the values of all input pins assigned to a chip port with a single 16-bit register read, while GPDI*n* registers read the value on a specific pin with a byte read.

The access to this register's location is coherent with the access to the bitwise GPDI*n*.

For a given PGPDI*x*[PPDI*y*], where *x* is the register instance index and *y* is the field index, the following equation shows the equivalent GPDI*n*[PDI_*n*] field:

PGPDI*x*[PPDI*y*] = GPDI(*x* × 16) + (15 - *y*)[PDI_(*x* × 16) + (15 - *y*)]

Following are some of the examples of mapping:

- PGPDI0[PPDI15] = GPDI0[PDI_0]
- PGPDI2[PPDI15] = GPDI32[PDI_32]
- PGPDI31[PPDI0] = GPDI511[PDI_511]

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PPDI15	PPDI14	PPDI13	PPDI12	PPDI11	PPDI10	PPDI9	PPDI8	PPDI7	PPDI6	PPDI5	PPDI4	PPDI3	PPDI2	PPDI1	PPDI0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15 PPDI15	Parallel Pad Data Input 15 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
14 PPDI14	Parallel Pad Data Input 14 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
13 PPDI13	Parallel Pad Data Input 13 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
12 PPDI12	Parallel Pad Data Input 12 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
11 PPDI11	Parallel Pad Data Input 11 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
10 PPDI10	Parallel Pad Data Input 10 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
9	Parallel Pad Data Input 9

Table continues on the next page...

Table continued from the previous page...

Field	Function
PPDI9	Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
8 PPDI8	Parallel Pad Data Input 8 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
7 PPDI7	Parallel Pad Data Input 7 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
6 PPDI6	Parallel Pad Data Input 6 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
5 PPDI5	Parallel Pad Data Input 5 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
4 PPDI4	Parallel Pad Data Input 4 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
3 PPDI3	Parallel Pad Data Input 3 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
2 PPDI2	Parallel Pad Data Input 2 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 PPDI1	Parallel Pad Data Input 1 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
0 PPDI0	Parallel Pad Data Input 0 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high

10.6.32 Parallel GPIO Pad Data Input (PGPDI8)

Offset

Register	Offset
PGPDI8	1752h

Function

Holds the synchronized input value from the pads and supports 8-, 16-, and 32-bit accesses.

This register reads the values of all input pins assigned to a chip port with a single 16-bit register read, while GPDI*n* registers read the value on a specific pin with a byte read.

The access to this register's location is coherent with the access to the bitwise GPDI*n*.

For a given PGPDI*x*[PPDI*y*], where *x* is the register instance index and *y* is the field index, the following equation shows the equivalent GPDI*n*[PDI_*n*] field:

PGPDI*x*[PPDI*y*] = GPDI(*x* × 16) + (15 - *y*)[PDI_(*x* × 16) + (15 - *y*)]

Following are some of the examples of mapping:

- PGPDI0[PPDI15] = GPDI0[PDI_0]
- PGPDI2[PPDI15] = GPDI32[PDI_32]
- PGPDI31[PPDI0] = GPDI511[PDI_511]

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PPDI15	PPDI14	PPDI13	PPDI12	PPDI11	PPDI10	PPDI9	PPDI8	PPDI7	PPDI6	PPDI5	PPDI4	PPDI3	Reserved	PPDI1	PPDI0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15 PPDI15	Parallel Pad Data Input 15 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
14 PPDI14	Parallel Pad Data Input 14 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
13 PPDI13	Parallel Pad Data Input 13 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
12 PPDI12	Parallel Pad Data Input 12 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
11 PPDI11	Parallel Pad Data Input 11 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
10 PPDI10	Parallel Pad Data Input 10 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
9	Parallel Pad Data Input 9

Table continues on the next page...

Table continued from the previous page...

Field	Function
PPDI9	Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
8 PPDI8	Parallel Pad Data Input 8 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
7 PPDI7	Parallel Pad Data Input 7 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
6 PPDI6	Parallel Pad Data Input 6 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
5 PPDI5	Parallel Pad Data Input 5 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
4 PPDI4	Parallel Pad Data Input 4 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
3 PPDI3	Parallel Pad Data Input 3 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
2 —	Reserved
1 PPDI1	Parallel Pad Data Input 1 Reads the current pad value of the corresponding pad.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Logic low 1b - Logic high
0 PPDI0	Parallel Pad Data Input 0 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high

10.6.33 Parallel GPIO Pad Data Input (PGPDI10 - PGPDI19)

Offset

For n = 10 to 19:

Register	Offset
PGPDIn	1754h + 2 × (n + 1 – 2 × (n mod 2))

Function

Holds the synchronized input value from the pads and supports 8-, 16-, and 32-bit accesses.

This register reads the values of all input pins assigned to a chip port with a single 16-bit register read, while GPDI*n* registers read the value on a specific pin with a byte read.

The access to this register's location is coherent with the access to the bitwise GPDI*n*.

For a given PGPDI*x*[PPDI*y*], where *x* is the register instance index and *y* is the field index, the following equation shows the equivalent GPDI*n*[PDI_*n*] field:

PGPDI*x*[PPDI*y*] = GPDI(*x* × 16) + (15 - *y*)[PDI_(*x* × 16) + (15 - *y*)]

Following are some of the examples of mapping:

- PGPDI0[PPDI15] = GPDI0[PDI_0]
- PGPDI2[PPDI15] = GPDI32[PDI_32]
- PGPDI31[PPDI0] = GPDI511[PDI_511]

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PPDI15	PPDI14	PPDI13	PPDI12	PPDI11	PPDI10	PPDI9	PPDI8	PPDI7	PPDI6	PPDI5	PPDI4	PPDI3	PPDI2	PPDI1	PPDI0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15 PPDI15	Parallel Pad Data Input 15 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
14 PPDI14	Parallel Pad Data Input 14 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
13 PPDI13	Parallel Pad Data Input 13 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
12 PPDI12	Parallel Pad Data Input 12 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
11 PPDI11	Parallel Pad Data Input 11 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
10 PPDI10	Parallel Pad Data Input 10 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
9	Parallel Pad Data Input 9

Table continues on the next page...

Table continued from the previous page...

Field	Function
PPDI9	Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
8 PPDI8	Parallel Pad Data Input 8 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
7 PPDI7	Parallel Pad Data Input 7 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
6 PPDI6	Parallel Pad Data Input 6 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
5 PPDI5	Parallel Pad Data Input 5 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
4 PPDI4	Parallel Pad Data Input 4 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
3 PPDI3	Parallel Pad Data Input 3 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
2 PPDI2	Parallel Pad Data Input 2 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 PPDI1	Parallel Pad Data Input 1 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high
0 PPDI0	Parallel Pad Data Input 0 Reads the current pad value of the corresponding pad. 0b - Logic low 1b - Logic high

10.6.34 Masked Parallel GPIO Pad Data Output (MPGPDO0 - MPGPDO1)

Offset

Register	Offset
MPGPDO0	1780h
MPGPDO1	1784h

Function

Modifies the pad values associated with PGPDO*n* selectively and supports only 32-bit accesses. It does not support byte and half-word accesses.

NOTE

Access this register only with 32-bit writes. 8-bit or 16-bit writes do not modify any bits in the register resulting in a transfer error. Read access returns 0.

The accesses to each of this register location is coherent with access to the bitwise GPDO*n*.

For a given MPGPDO*x*[MPPDO*y*], where *x* is the register instance index and *y* is the field index, the following equation shows the equivalent GPDO*n*[PDO_*n*]:

MPGPDO*x*[MPPDO*y*] = GPDO(*x* × 16) + (15 - *y*)[PDO_(*x* × 16) + (15 - *y*)]

Following are some of the examples of mapping:

- MPGPDO0[MPPDO15] = GPDO0[PDO_0]
- MPGPDO2[MPPDO15] = GPDO32[PDO_32]
- MPGPDO31[MPPDO0] = GPDO511[PDO_511]

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	MASK 15	MASK 14	MASK 13	MASK 12	MASK 11	MASK 10	MASK 9	MASK 8	MASK 7	MASK 6	MASK 5	MASK 4	MASK 3	MASK 2	MASK 1	MASK 0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	MPPDO15	MPPDO14	MPPDO13	MPPDO12	MPPDO11	MPPDO10	MPPDO9	MPPDO8	MPPDO7	MPPDO6	MPPDO5	MPPDO4	MPPDO3	MPPDO2	MPPDO1	MPPDO0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 MASK15	Mask Field 15 Masks MPPDO15 in the corresponding MPGPDO*n* register instance. 0b - MPPDO15 is ignored 1b - MPPDO15 is written
30 MASK14	Mask Field 14 Masks MPPDO14 in the corresponding MPGPDO*n* register instance. 0b - MPPDO14 is ignored 1b - MPPDO14 is written
29 MASK13	Mask Field 13 Masks MPPDO13 in the corresponding MPGPDO*n* register instance. 0b - MPPDO13 is ignored 1b - MPPDO13 is written
28 MASK12	Mask Field 12 Masks MPPDO12 in the corresponding MPGPDO*n* register instance. 0b - MPPDO12 is ignored 1b - MPPDO12 is written
27 MASK11	Mask Field 11 Masks MPPDO11 in the corresponding MPGPDO*n* register instance. 0b - MPPDO11 is ignored 1b - MPPDO11 is written
26	Mask Field 10

Table continues on the next page...

Table continued from the previous page...

Field	Function
MASK10	Masks MPPDO10 in the corresponding MPGPDO*n* register instance. 0b - MPPDO10 is ignored 1b - MPPDO10 is written
25 MASK9	Mask Field 9 Masks MPPDO9 in the corresponding MPGPDO*n* register instance. 0b - MPPDO9 is ignored 1b - MPPDO9 is written
24 MASK8	Mask Field 8 Masks MPPDO8 in the corresponding MPGPDO*n* register instance. 0b - MPPDO8 is ignored 1b - MPPDO8 is written
23 MASK7	Mask Field 7 Masks MPPDO7 in the corresponding MPGPDO*n* register instance. 0b - MPPDO7 is ignored 1b - MPPDO7 is written
22 MASK6	Mask Field 6 Masks MPPDO6 in the corresponding MPGPDO*n* register instance. 0b - MPPDO6 is ignored 1b - MPPDO6 is written
21 MASK5	Mask Field 5 Masks MPPDO5 in the corresponding MPGPDO*n* register instance. 0b - MPPDO5 is ignored 1b - MPPDO5 is written
20 MASK4	Mask Field 4 Masks MPPDO4 in the corresponding MPGPDO*n* register instance. 0b - MPPDO4 is ignored 1b - MPPDO4 is written
19 MASK3	Mask Field 3 Masks MPPDO3 in the corresponding MPGPDO*n* register instance. 0b - MPPDO3 is ignored 1b - MPPDO3 is written

Table continues on the next page...

Table continued from the previous page...

Field	Function
18 MASK2	Mask Field 2 Masks MPPDO2 in the corresponding MPGPDO*n* register instance. 0b - MPPDO2 is ignored 1b - MPPDO2 is written
17 MASK1	Mask Field 1 Masks MPPDO1 in the corresponding MPGPDO*n* register instance. 0b - MPPDO1 is ignored 1b - MPPDO1 is written
16 MASK0	Mask Field 0 Masks MPPDO0 in the corresponding MPGPDO*n* register instance. 0b - MPPDO0 is ignored 1b - MPPDO0 is written
15 MPPDO15	Masked Parallel Pad Data Out 15 Writes to the data register that stores the value to be driven on the pad in Output mode.
14 MPPDO14	Masked Parallel Pad Data Out 14 Writes to the data register that stores the value to be driven on the pad in Output mode.
13 MPPDO13	Masked Parallel Pad Data Out 13 Writes to the data register that stores the value to be driven on the pad in Output mode.
12 MPPDO12	Masked Parallel Pad Data Out 12 Writes to the data register that stores the value to be driven on the pad in Output mode.
11 MPPDO11	Masked Parallel Pad Data Out 11 Writes to the data register that stores the value to be driven on the pad in Output mode.
10 MPPDO10	Masked Parallel Pad Data Out 10 Writes to the data register that stores the value to be driven on the pad in Output mode.
9 MPPDO9	Masked Parallel Pad Data Out 9 Writes to the data register that stores the value to be driven on the pad in Output mode.
8 MPPDO8	Masked Parallel Pad Data Out 8 Writes to the data register that stores the value to be driven on the pad in Output mode.
7 MPPDO7	Masked Parallel Pad Data Out 7 Writes to the data register that stores the value to be driven on the pad in Output mode.

Table continues on the next page...

Table continued from the previous page...

Field	Function
6 MPPDO6	Masked Parallel Pad Data Out 6 Writes to the data register that stores the value to be driven on the pad in Output mode.
5 MPPDO5	Masked Parallel Pad Data Out 5 Writes to the data register that stores the value to be driven on the pad in Output mode.
4 MPPDO4	Masked Parallel Pad Data Out 4 Writes to the data register that stores the value to be driven on the pad in Output mode.
3 MPPDO3	Masked Parallel Pad Data Out 3 Writes to the data register that stores the value to be driven on the pad in Output mode.
2 MPPDO2	Masked Parallel Pad Data Out 2 Writes to the data register that stores the value to be driven on the pad in Output mode.
1 MPPDO1	Masked Parallel Pad Data Out 1 Writes to the data register that stores the value to be driven on the pad in Output mode.
0 MPPDO0	Masked Parallel Pad Data Out 0 Writes to the data register that stores the value to be driven on the pad in Output mode.

10.6.35 Masked Parallel GPIO Pad Data Output (MPGPDO2)

Offset

Register	Offset
MPGPDO2	1788h

Function

Modifies the pad values associated with PGPDO*n* selectively and supports only 32-bit accesses. It does not support byte and half-word accesses.

NOTE

Access this register only with 32-bit writes. 8-bit or 16-bit writes do not modify any bits in the register resulting in a transfer error. Read access returns 0.

The accesses to each of this register location is coherent with access to the bitwise GPDO*n*.

For a given MPGPDO*x*[MPPDO*y*], where *x* is the register instance index and *y* is the field index, the following equation shows the equivalent GPDO*n*[PDO_*n*]:

MPGPDO*x*[MPPDO*y*] = GPDO(*x* × 16) + (15 - *y*)[PDO_(*x* × 16) + (15 - *y*)]

Following are some of the examples of mapping:

- MPGPDO0[MPPDO15] = GPDO0[PDO_0]

- MPGPDO2[MPPDO15] = GPDO32[PDO_32]
- MPGPDO31[MPPDO0] = GPDO511[PDO_511]

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	MASK 15	MASK 14	MASK 13	MASK 12	MASK 11	MASK 10	Reserved	Reserved	MASK 7	MASK 6	MASK 5	MASK 4	MASK 3	MASK 2	MASK 1	MASK 0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	MPPDO15	MPPDO14	MPPDO13	MPPDO12	MPPDO11	MPPDO10	Reserved	Reserved	MPPDO7	MPPDO6	MPPDO5	MPPDO4	MPPDO3	MPPDO2	MPPDO1	MPPDO0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 MASK15	Mask Field 15 Masks MPPDO15 in the corresponding MPGPDO*n* register instance. 0b - MPPDO15 is ignored 1b - MPPDO15 is written
30 MASK14	Mask Field 14 Masks MPPDO14 in the corresponding MPGPDO*n* register instance. 0b - MPPDO14 is ignored 1b - MPPDO14 is written
29 MASK13	Mask Field 13 Masks MPPDO13 in the corresponding MPGPDO*n* register instance. 0b - MPPDO13 is ignored 1b - MPPDO13 is written
28 MASK12	Mask Field 12 Masks MPPDO12 in the corresponding MPGPDO*n* register instance. 0b - MPPDO12 is ignored 1b - MPPDO12 is written
27 MASK11	Mask Field 11 Masks MPPDO11 in the corresponding MPGPDO*n* register instance.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - MPPDO11 is ignored 1b - MPPDO11 is written
26 MASK10	Mask Field 10 Masks MPPDO10 in the corresponding MPGPDO*n* register instance. 0b - MPPDO10 is ignored 1b - MPPDO10 is written
25 —	Reserved Always write zero to this field.
24 —	Reserved Always write zero to this field.
23 MASK7	Mask Field 7 Masks MPPDO7 in the corresponding MPGPDO*n* register instance. 0b - MPPDO7 is ignored 1b - MPPDO7 is written
22 MASK6	Mask Field 6 Masks MPPDO6 in the corresponding MPGPDO*n* register instance. 0b - MPPDO6 is ignored 1b - MPPDO6 is written
21 MASK5	Mask Field 5 Masks MPPDO5 in the corresponding MPGPDO*n* register instance. 0b - MPPDO5 is ignored 1b - MPPDO5 is written
20 MASK4	Mask Field 4 Masks MPPDO4 in the corresponding MPGPDO*n* register instance. 0b - MPPDO4 is ignored 1b - MPPDO4 is written
19 MASK3	Mask Field 3 Masks MPPDO3 in the corresponding MPGPDO*n* register instance. 0b - MPPDO3 is ignored 1b - MPPDO3 is written
18	Mask Field 2

Table continues on the next page...

Table continued from the previous page...

Field	Function
MASK2	Masks MPPDO2 in the corresponding MPGPDO*n* register instance. 0b - MPPDO2 is ignored 1b - MPPDO2 is written
17 MASK1	Mask Field 1 Masks MPPDO1 in the corresponding MPGPDO*n* register instance. 0b - MPPDO1 is ignored 1b - MPPDO1 is written
16 MASK0	Mask Field 0 Masks MPPDO0 in the corresponding MPGPDO*n* register instance. 0b - MPPDO0 is ignored 1b - MPPDO0 is written
15 MPPDO15	Masked Parallel Pad Data Out 15 Writes to the data register that stores the value to be driven on the pad in Output mode.
14 MPPDO14	Masked Parallel Pad Data Out 14 Writes to the data register that stores the value to be driven on the pad in Output mode.
13 MPPDO13	Masked Parallel Pad Data Out 13 Writes to the data register that stores the value to be driven on the pad in Output mode.
12 MPPDO12	Masked Parallel Pad Data Out 12 Writes to the data register that stores the value to be driven on the pad in Output mode.
11 MPPDO11	Masked Parallel Pad Data Out 11 Writes to the data register that stores the value to be driven on the pad in Output mode.
10 MPPDO10	Masked Parallel Pad Data Out 10 Writes to the data register that stores the value to be driven on the pad in Output mode.
9 —	Reserved Always write zero to this field.
8 —	Reserved Always write zero to this field.
7 MPPDO7	Masked Parallel Pad Data Out 7 Writes to the data register that stores the value to be driven on the pad in Output mode.
6	Masked Parallel Pad Data Out 6

Table continues on the next page...

Table continued from the previous page...

Field	Function
MPPDO6	Writes to the data register that stores the value to be driven on the pad in Output mode.
5 MPPDO5	Masked Parallel Pad Data Out 5 Writes to the data register that stores the value to be driven on the pad in Output mode.
4 MPPDO4	Masked Parallel Pad Data Out 4 Writes to the data register that stores the value to be driven on the pad in Output mode.
3 MPPDO3	Masked Parallel Pad Data Out 3 Writes to the data register that stores the value to be driven on the pad in Output mode.
2 MPPDO2	Masked Parallel Pad Data Out 2 Writes to the data register that stores the value to be driven on the pad in Output mode.
1 MPPDO1	Masked Parallel Pad Data Out 1 Writes to the data register that stores the value to be driven on the pad in Output mode.
0 MPPDO0	Masked Parallel Pad Data Out 0 Writes to the data register that stores the value to be driven on the pad in Output mode.

10.6.36 Masked Parallel GPIO Pad Data Output (MPGPDO3 - MPGPDO7)

Offset

Register	Offset
MPGPDO3	178Ch
MPGPDO4	1790h
MPGPDO5	1794h
MPGPDO6	1798h
MPGPDO7	179Ch

Function

Modifies the pad values associated with PGPDO*n* selectively and supports only 32-bit accesses. It does not support byte and half-word accesses.

NOTE

Access this register only with 32-bit writes. 8-bit or 16-bit writes do not modify any bits in the register resulting in a transfer error. Read access returns 0.

The accesses to each of this register location is coherent with access to the bitwise GPDO*n*.

For a given MPGPDO*x*[MPPDO*y*], where *x* is the register instance index and *y* is the field index, the following equation shows the equivalent GPDO*n*[PDO_*n*]:

MPGPDO*x*[MPPDO*y*] = GPDO(*x* × 16) + (15 - *y*)[PDO_(*x* × 16) + (15 - *y*)]

Following are some of the examples of mapping:

- MPGPDO0[MPPDO15] = GPDO0[PDO_0]
- MPGPDO2[MPPDO15] = GPDO32[PDO_32]
- MPGPDO31[MPPDO0] = GPDO511[PDO_511]

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	MASK 15	MASK 14	MASK 13	MASK 12	MASK 11	MASK 10	MASK 9	MASK 8	MASK 7	MASK 6	MASK 5	MASK 4	MASK 3	MASK 2	MASK 1	MASK 0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	MPPDO15	MPPDO14	MPPDO13	MPPDO12	MPPDO11	MPPDO10	MPPDO9	MPPDO8	MPPDO7	MPPDO6	MPPDO5	MPPDO4	MPPDO3	MPPDO2	MPPDO1	MPPDO0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 MASK15	Mask Field 15 Masks MPPDO15 in the corresponding MPGPDO*n* register instance. 0b - MPPDO15 is ignored 1b - MPPDO15 is written
30 MASK14	Mask Field 14 Masks MPPDO14 in the corresponding MPGPDO*n* register instance. 0b - MPPDO14 is ignored 1b - MPPDO14 is written
29 MASK13	Mask Field 13 Masks MPPDO13 in the corresponding MPGPDO*n* register instance. 0b - MPPDO13 is ignored 1b - MPPDO13 is written
28 MASK12	Mask Field 12 Masks MPPDO12 in the corresponding MPGPDO*n* register instance. 0b - MPPDO12 is ignored 1b - MPPDO12 is written

Table continues on the next page...

Table continued from the previous page...

Field	Function
27 MASK11	Mask Field 11 Masks MPPDO11 in the corresponding MPGPDO*n* register instance. 0b - MPPDO11 is ignored 1b - MPPDO11 is written
26 MASK10	Mask Field 10 Masks MPPDO10 in the corresponding MPGPDO*n* register instance. 0b - MPPDO10 is ignored 1b - MPPDO10 is written
25 MASK9	Mask Field 9 Masks MPPDO9 in the corresponding MPGPDO*n* register instance. 0b - MPPDO9 is ignored 1b - MPPDO9 is written
24 MASK8	Mask Field 8 Masks MPPDO8 in the corresponding MPGPDO*n* register instance. 0b - MPPDO8 is ignored 1b - MPPDO8 is written
23 MASK7	Mask Field 7 Masks MPPDO7 in the corresponding MPGPDO*n* register instance. 0b - MPPDO7 is ignored 1b - MPPDO7 is written
22 MASK6	Mask Field 6 Masks MPPDO6 in the corresponding MPGPDO*n* register instance. 0b - MPPDO6 is ignored 1b - MPPDO6 is written
21 MASK5	Mask Field 5 Masks MPPDO5 in the corresponding MPGPDO*n* register instance. 0b - MPPDO5 is ignored 1b - MPPDO5 is written
20 MASK4	Mask Field 4 Masks MPPDO4 in the corresponding MPGPDO*n* register instance. 0b - MPPDO4 is ignored

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - MPPDO4 is written
19 MASK3	Mask Field 3 Masks MPPDO3 in the corresponding MPGPDO*n* register instance. 0b - MPPDO3 is ignored 1b - MPPDO3 is written
18 MASK2	Mask Field 2 Masks MPPDO2 in the corresponding MPGPDO*n* register instance. 0b - MPPDO2 is ignored 1b - MPPDO2 is written
17 MASK1	Mask Field 1 Masks MPPDO1 in the corresponding MPGPDO*n* register instance. 0b - MPPDO1 is ignored 1b - MPPDO1 is written
16 MASK0	Mask Field 0 Masks MPPDO0 in the corresponding MPGPDO*n* register instance. 0b - MPPDO0 is ignored 1b - MPPDO0 is written
15 MPPDO15	Masked Parallel Pad Data Out 15 Writes to the data register that stores the value to be driven on the pad in Output mode.
14 MPPDO14	Masked Parallel Pad Data Out 14 Writes to the data register that stores the value to be driven on the pad in Output mode.
13 MPPDO13	Masked Parallel Pad Data Out 13 Writes to the data register that stores the value to be driven on the pad in Output mode.
12 MPPDO12	Masked Parallel Pad Data Out 12 Writes to the data register that stores the value to be driven on the pad in Output mode.
11 MPPDO11	Masked Parallel Pad Data Out 11 Writes to the data register that stores the value to be driven on the pad in Output mode.
10 MPPDO10	Masked Parallel Pad Data Out 10 Writes to the data register that stores the value to be driven on the pad in Output mode.
9	Masked Parallel Pad Data Out 9

Table continues on the next page...

Table continued from the previous page...

Field	Function
MPPDO9	Writes to the data register that stores the value to be driven on the pad in Output mode.
8 MPPDO8	Masked Parallel Pad Data Out 8 Writes to the data register that stores the value to be driven on the pad in Output mode.
7 MPPDO7	Masked Parallel Pad Data Out 7 Writes to the data register that stores the value to be driven on the pad in Output mode.
6 MPPDO6	Masked Parallel Pad Data Out 6 Writes to the data register that stores the value to be driven on the pad in Output mode.
5 MPPDO5	Masked Parallel Pad Data Out 5 Writes to the data register that stores the value to be driven on the pad in Output mode.
4 MPPDO4	Masked Parallel Pad Data Out 4 Writes to the data register that stores the value to be driven on the pad in Output mode.
3 MPPDO3	Masked Parallel Pad Data Out 3 Writes to the data register that stores the value to be driven on the pad in Output mode.
2 MPPDO2	Masked Parallel Pad Data Out 2 Writes to the data register that stores the value to be driven on the pad in Output mode.
1 MPPDO1	Masked Parallel Pad Data Out 1 Writes to the data register that stores the value to be driven on the pad in Output mode.
0 MPPDO0	Masked Parallel Pad Data Out 0 Writes to the data register that stores the value to be driven on the pad in Output mode.

10.6.37 Masked Parallel GPIO Pad Data Output (MPGPDO8)

Offset

Register	Offset
MPGPDO8	17A0h

Function

Modifies the pad values associated with PGPDO*n* selectively and supports only 32-bit accesses. It does not support byte and half-word accesses.

NOTE

Access this register only with 32-bit writes. 8-bit or 16-bit writes do not modify any bits in the register resulting in a transfer error. Read access returns 0.

The accesses to each of this register location is coherent with access to the bitwise GPDO*n*.

For a given MPGPDO*x*[MPPDO*y*], where *x* is the register instance index and *y* is the field index, the following equation shows the equivalent GPDO*n*[PDO_*n*]:

MPGPDO*x*[MPPDO*y*] = GPDO(*x* × 16) + (15 - *y*)[PDO_(*x* × 16) + (15 - *y*)]

Following are some of the examples of mapping:

- MPGPDO0[MPPDO15] = GPDO0[PDO_0]
- MPGPDO2[MPPDO15] = GPDO32[PDO_32]
- MPGPDO31[MPPDO0] = GPDO511[PDO_511]

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	MASK 15	MASK 14	MASK 13	MASK 12	MASK 11	MASK 10	MASK 9	MASK 8	MASK 7	MASK 6	MASK 5	MASK 4	MASK 3	Reserved	MASK 1	MASK 0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	MPPDO15	MPPDO14	MPPDO13	MPPDO12	MPPDO11	MPPDO10	MPPDO9	MPPDO8	MPPDO7	MPPDO6	MPPDO5	MPPDO4	MPPDO3	Reserved	MPPDO1	MPPDO0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 MASK15	Mask Field 15 Masks MPPDO15 in the corresponding MPGPDO*n* register instance. 0b - MPPDO15 is ignored 1b - MPPDO15 is written
30 MASK14	Mask Field 14 Masks MPPDO14 in the corresponding MPGPDO*n* register instance. 0b - MPPDO14 is ignored 1b - MPPDO14 is written
29 MASK13	Mask Field 13 Masks MPPDO13 in the corresponding MPGPDO*n* register instance. 0b - MPPDO13 is ignored 1b - MPPDO13 is written
28	Mask Field 12

Table continues on the next page...

Table continued from the previous page...

Field	Function
MASK12	Masks MPPDO12 in the corresponding MPGPDO*n* register instance. 0b - MPPDO12 is ignored 1b - MPPDO12 is written
27 MASK11	Mask Field 11 Masks MPPDO11 in the corresponding MPGPDO*n* register instance. 0b - MPPDO11 is ignored 1b - MPPDO11 is written
26 MASK10	Mask Field 10 Masks MPPDO10 in the corresponding MPGPDO*n* register instance. 0b - MPPDO10 is ignored 1b - MPPDO10 is written
25 MASK9	Mask Field 9 Masks MPPDO9 in the corresponding MPGPDO*n* register instance. 0b - MPPDO9 is ignored 1b - MPPDO9 is written
24 MASK8	Mask Field 8 Masks MPPDO8 in the corresponding MPGPDO*n* register instance. 0b - MPPDO8 is ignored 1b - MPPDO8 is written
23 MASK7	Mask Field 7 Masks MPPDO7 in the corresponding MPGPDO*n* register instance. 0b - MPPDO7 is ignored 1b - MPPDO7 is written
22 MASK6	Mask Field 6 Masks MPPDO6 in the corresponding MPGPDO*n* register instance. 0b - MPPDO6 is ignored 1b - MPPDO6 is written
21 MASK5	Mask Field 5 Masks MPPDO5 in the corresponding MPGPDO*n* register instance. 0b - MPPDO5 is ignored 1b - MPPDO5 is written

Table continues on the next page...

Table continued from the previous page...

Field	Function
20 MASK4	Mask Field 4 Masks MPPDO4 in the corresponding MPGPDO*n* register instance. 0b - MPPDO4 is ignored 1b - MPPDO4 is written
19 MASK3	Mask Field 3 Masks MPPDO3 in the corresponding MPGPDO*n* register instance. 0b - MPPDO3 is ignored 1b - MPPDO3 is written
18 —	Reserved Always write zero to this field.
17 MASK1	Mask Field 1 Masks MPPDO1 in the corresponding MPGPDO*n* register instance. 0b - MPPDO1 is ignored 1b - MPPDO1 is written
16 MASK0	Mask Field 0 Masks MPPDO0 in the corresponding MPGPDO*n* register instance. 0b - MPPDO0 is ignored 1b - MPPDO0 is written
15 MPPDO15	Masked Parallel Pad Data Out 15 Writes to the data register that stores the value to be driven on the pad in Output mode.
14 MPPDO14	Masked Parallel Pad Data Out 14 Writes to the data register that stores the value to be driven on the pad in Output mode.
13 MPPDO13	Masked Parallel Pad Data Out 13 Writes to the data register that stores the value to be driven on the pad in Output mode.
12 MPPDO12	Masked Parallel Pad Data Out 12 Writes to the data register that stores the value to be driven on the pad in Output mode.
11 MPPDO11	Masked Parallel Pad Data Out 11 Writes to the data register that stores the value to be driven on the pad in Output mode.
10 MPPDO10	Masked Parallel Pad Data Out 10 Writes to the data register that stores the value to be driven on the pad in Output mode.

Table continues on the next page...

Table continued from the previous page...

Field	Function
9 MPPDO9	Masked Parallel Pad Data Out 9 Writes to the data register that stores the value to be driven on the pad in Output mode.
8 MPPDO8	Masked Parallel Pad Data Out 8 Writes to the data register that stores the value to be driven on the pad in Output mode.
7 MPPDO7	Masked Parallel Pad Data Out 7 Writes to the data register that stores the value to be driven on the pad in Output mode.
6 MPPDO6	Masked Parallel Pad Data Out 6 Writes to the data register that stores the value to be driven on the pad in Output mode.
5 MPPDO5	Masked Parallel Pad Data Out 5 Writes to the data register that stores the value to be driven on the pad in Output mode.
4 MPPDO4	Masked Parallel Pad Data Out 4 Writes to the data register that stores the value to be driven on the pad in Output mode.
3 MPPDO3	Masked Parallel Pad Data Out 3 Writes to the data register that stores the value to be driven on the pad in Output mode.
2 —	Reserved Always write zero to this field.
1 MPPDO1	Masked Parallel Pad Data Out 1 Writes to the data register that stores the value to be driven on the pad in Output mode.
0 MPPDO0	Masked Parallel Pad Data Out 0 Writes to the data register that stores the value to be driven on the pad in Output mode.

10.6.38 Masked Parallel GPIO Pad Data Output (MPGPDO9 - MPGPDO19)

Offset

For a = 9 to 19:

Register	Offset
MPGPDOa	1780h + (a × 4h)

Function

Modifies the pad values associated with PGPDO*n* selectively and supports only 32-bit accesses. It does not support byte and half-word accesses.

NOTE

Access this register only with 32-bit writes. 8-bit or 16-bit writes do not modify any bits in the register resulting in a transfer error. Read access returns 0.

The accesses to each of this register location is coherent with access to the bitwise GPDO*n*.

For a given MPGPDO*x*[MPPDO*y*], where *x* is the register instance index and *y* is the field index, the following equation shows the equivalent GPDO*n*[PDO_*n*]:

MPGPDO*x*[MPPDO*y*] = GPDO(*x* × 16) + (15 - *y*)[PDO_(*x* × 16) + (15 - *y*)]

Following are some of the examples of mapping:

- MPGPDO0[MPPDO15] = GPDO0[PDO_0]
- MPGPDO2[MPPDO15] = GPDO32[PDO_32]
- MPGPDO31[MPPDO0] = GPDO511[PDO_511]

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	MASK 15	MASK 14	MASK 13	MASK 12	MASK 11	MASK 10	MASK 9	MASK 8	MASK 7	MASK 6	MASK 5	MASK 4	MASK 3	MASK 2	MASK 1	MASK 0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	MPPDO15	MPPDO14	MPPDO13	MPPDO12	MPPDO11	MPPDO10	MPPDO9	MPPDO8	MPPDO7	MPPDO6	MPPDO5	MPPDO4	MPPDO3	MPPDO2	MPPDO1	MPPDO0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 MASK15	Mask Field 15 Masks MPPDO15 in the corresponding MPGPDO*n* register instance. 0b - MPPDO15 is ignored 1b - MPPDO15 is written
30 MASK14	Mask Field 14 Masks MPPDO14 in the corresponding MPGPDO*n* register instance. 0b - MPPDO14 is ignored 1b - MPPDO14 is written
29 MASK13	Mask Field 13 Masks MPPDO13 in the corresponding MPGPDO*n* register instance.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - MPPDO13 is ignored 1b - MPPDO13 is written
28 MASK12	Mask Field 12 Masks MPPDO12 in the corresponding MPGPDO*n* register instance. 0b - MPPDO12 is ignored 1b - MPPDO12 is written
27 MASK11	Mask Field 11 Masks MPPDO11 in the corresponding MPGPDO*n* register instance. 0b - MPPDO11 is ignored 1b - MPPDO11 is written
26 MASK10	Mask Field 10 Masks MPPDO10 in the corresponding MPGPDO*n* register instance. 0b - MPPDO10 is ignored 1b - MPPDO10 is written
25 MASK9	Mask Field 9 Masks MPPDO9 in the corresponding MPGPDO*n* register instance. 0b - MPPDO9 is ignored 1b - MPPDO9 is written
24 MASK8	Mask Field 8 Masks MPPDO8 in the corresponding MPGPDO*n* register instance. 0b - MPPDO8 is ignored 1b - MPPDO8 is written
23 MASK7	Mask Field 7 Masks MPPDO7 in the corresponding MPGPDO*n* register instance. 0b - MPPDO7 is ignored 1b - MPPDO7 is written
22 MASK6	Mask Field 6 Masks MPPDO6 in the corresponding MPGPDO*n* register instance. 0b - MPPDO6 is ignored 1b - MPPDO6 is written
21	Mask Field 5

Table continues on the next page...

Table continued from the previous page...

Field	Function
MASK5	Masks MPPDO5 in the corresponding MPGPDO*n* register instance. 0b - MPPDO5 is ignored 1b - MPPDO5 is written
20 MASK4	Mask Field 4 Masks MPPDO4 in the corresponding MPGPDO*n* register instance. 0b - MPPDO4 is ignored 1b - MPPDO4 is written
19 MASK3	Mask Field 3 Masks MPPDO3 in the corresponding MPGPDO*n* register instance. 0b - MPPDO3 is ignored 1b - MPPDO3 is written
18 MASK2	Mask Field 2 Masks MPPDO2 in the corresponding MPGPDO*n* register instance. 0b - MPPDO2 is ignored 1b - MPPDO2 is written
17 MASK1	Mask Field 1 Masks MPPDO1 in the corresponding MPGPDO*n* register instance. 0b - MPPDO1 is ignored 1b - MPPDO1 is written
16 MASK0	Mask Field 0 Masks MPPDO0 in the corresponding MPGPDO*n* register instance. 0b - MPPDO0 is ignored 1b - MPPDO0 is written
15 MPPDO15	Masked Parallel Pad Data Out 15 Writes to the data register that stores the value to be driven on the pad in Output mode.
14 MPPDO14	Masked Parallel Pad Data Out 14 Writes to the data register that stores the value to be driven on the pad in Output mode.
13 MPPDO13	Masked Parallel Pad Data Out 13 Writes to the data register that stores the value to be driven on the pad in Output mode.
12	Masked Parallel Pad Data Out 12 Writes to the data register that stores the value to be driven on the pad in Output mode.

Table continues on the next page...

Table continued from the previous page...

Field	Function
MPPDO12	
11 MPPDO11	Masked Parallel Pad Data Out 11 Writes to the data register that stores the value to be driven on the pad in Output mode.
10 MPPDO10	Masked Parallel Pad Data Out 10 Writes to the data register that stores the value to be driven on the pad in Output mode.
9 MPPDO9	Masked Parallel Pad Data Out 9 Writes to the data register that stores the value to be driven on the pad in Output mode.
8 MPPDO8	Masked Parallel Pad Data Out 8 Writes to the data register that stores the value to be driven on the pad in Output mode.
7 MPPDO7	Masked Parallel Pad Data Out 7 Writes to the data register that stores the value to be driven on the pad in Output mode.
6 MPPDO6	Masked Parallel Pad Data Out 6 Writes to the data register that stores the value to be driven on the pad in Output mode.
5 MPPDO5	Masked Parallel Pad Data Out 5 Writes to the data register that stores the value to be driven on the pad in Output mode.
4 MPPDO4	Masked Parallel Pad Data Out 4 Writes to the data register that stores the value to be driven on the pad in Output mode.
3 MPPDO3	Masked Parallel Pad Data Out 3 Writes to the data register that stores the value to be driven on the pad in Output mode.
2 MPPDO2	Masked Parallel Pad Data Out 2 Writes to the data register that stores the value to be driven on the pad in Output mode.
1 MPPDO1	Masked Parallel Pad Data Out 1 Writes to the data register that stores the value to be driven on the pad in Output mode.
0 MPPDO0	Masked Parallel Pad Data Out 0 Writes to the data register that stores the value to be driven on the pad in Output mode.

Chapter 11
Touch Sensing Pin Coupling (TSPC)

11.1 Chip-specific TSPC information

11.1.1 TSPC instances and configuration

This chip has one instance of TSPC module. The GPIO pads being used for TSPC should be configured with Source Signal Select (SSS) = 0. TSPC module works in conjunction with SIUL2.

NOTE

TSPC clock shall be enabled to control the I/Os assigned to TSPC.

For the GPIOs and ADC pads supporting TSPC, see the IOMUX file attached to this document. There are two groups of pads which can be configured via TSPC:

1. Group 1 consists of 32 ADC channels and 14 GPIOs
2. Group 2 consists of 32 ADC channels and 6 GPIOs

NOTE

Use preferably the same type of I/O pins for driving each electrode. Such I/O pins matching will improve resolution of the capacitance sensing. For more information about types of I/O pins, see the IOMUX file attached to this document.

11.2 Overview

TSPC provides simultaneous transitioning of a group of pads into high impedance to support a more robust recognition of touch sense.

11.2.1 Block diagram

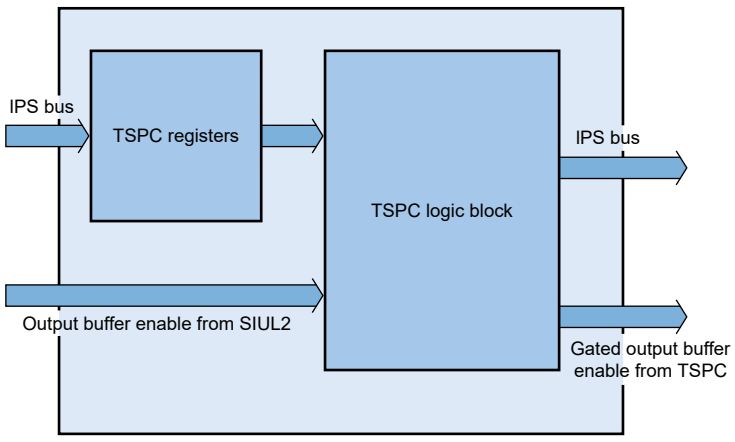


Figure 28. TSPC block diagram

11.2.2 Features

TSPC provides a set of control registers for pairing up GPIO pads for the touch sensing pairing operation. These control registers select the set of GPIO pads to pair with the set of ADC channels of an ADC instance. See the chip-specific TSPC configuration for group of pads information.

TSPC supports the following features:

- Up to 512 GPIO pads allow grouping
- OBE of all the grouped pads transitions from high to low simultaneously if SIUL2 transitions a pad's OBE from high to low
- Programming of group registers defines grouping

11.3 Functional description

TSPC contains these types of registers:

- Group enable: Each field of this register enables the respective group's OBE register—that is, if any OBE within a group transitions from high to low, it enables that group of OBEs for simultaneous transition from high to low. For example, GRP1_EN is the enable field for group 1 OBE registers. The number of fields in this register is equal to the number of groups.
- Group *n* OBE register *n*: The number of registers for a particular group *n* depends on the number of grouped OBEs in the respective group. For example, if the number of OBEs in group 1 is 36, then two registers of 32 fields each exist. Each field of this register represents an OBE pad. If you write 1 to the field, it indicates the OBE's participation in the grouping. If one of the pad's OBEs transitions from high to low, all the participating OBEs transition from high to low. The remaining OBEs that are not participating in the grouping are not affected.

11.3.1 Submodules

11.3.1.1 Logic block description

This block takes the information from register fields to produce gated OBEs.

The simultaneous transition of grouped OBEs occurs when the GRP*n*_EN field in Group Enable (GRP_EN) for the respective group becomes 1.

The logic diagram below shows that if the GRP_EN is high for a particular group, and one of the output buffer enable (ipp_obe_in_b) from SIUL2 belonging to that group transitions from high to low, then all the other output buffer enables belonging to that group also transitions from high to low. If the GRP_EN is set low, the transitions in OBE do not affect the other OBE's belonging to that group.

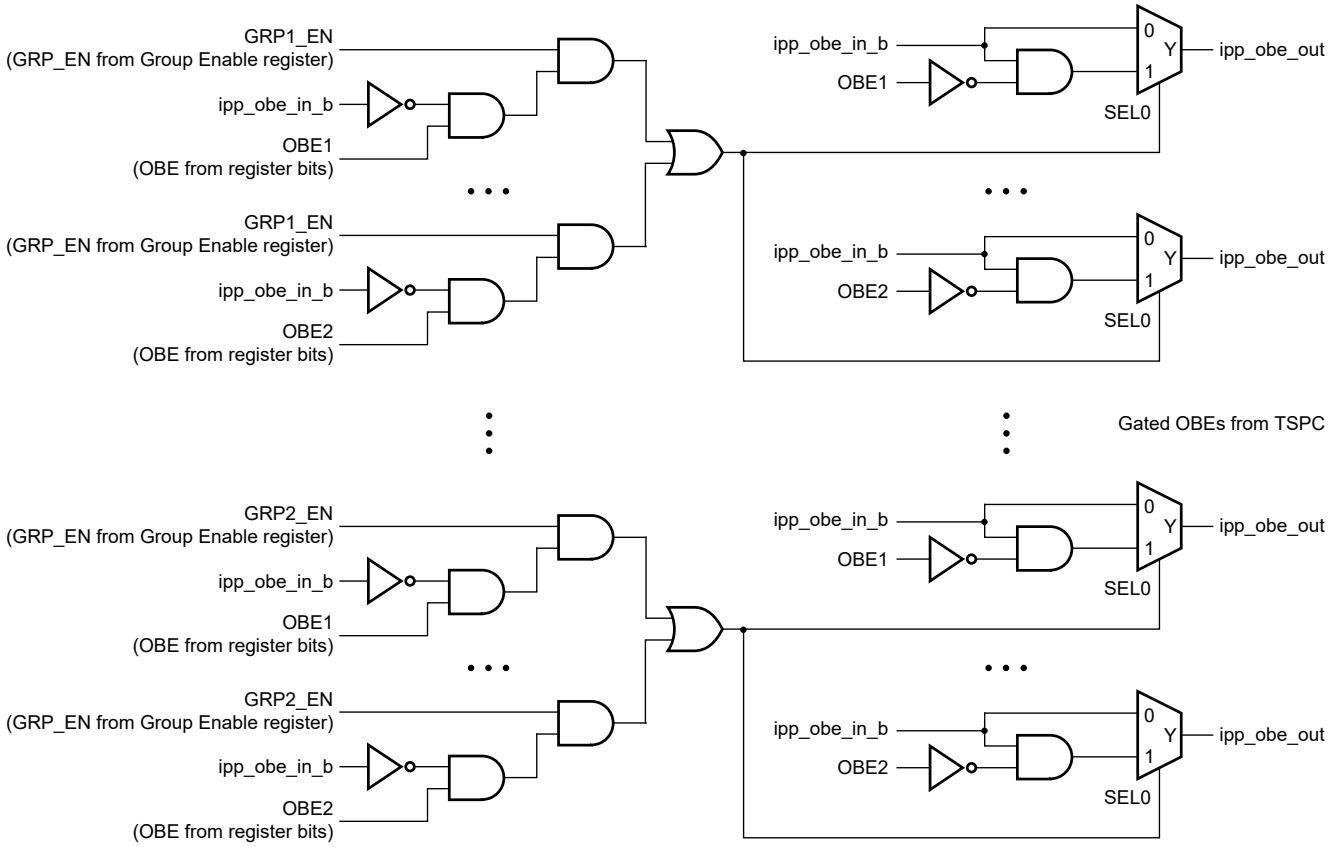


Figure 29. Logic block diagram

11.3.2 Operation

This section describes the TSPC_SPEC module's operations.

1. Reset all the registers.
2. Write 1 to the GRP*n*_OBE*x*[OBE*n*] of the OBEs that participate in grouping.
3. Program the corresponding group enable field participating in grouping in Group Enable (GRP_EN).

You must write 0 to the corresponding GRP_EN field when you configure the group *n* OBE fields. You must write 1 to the OBE*n* field in Group OBE (GRP1_OBE1 - GRP2_OBE1) and then write 1 to the corresponding GRP_EN field in Group Enable (GRP_EN).

11.3.3 Clocking

The TSPC module has two clocks.

Clock
ipg_clk
ipg_clk_s

11.3.4 Interrupts

This module has no interrupts.

11.4 External signals

The TSPC module has two external signals.

Signal	IO	Description
ipp_obe_out	Output	Gated output buffer enable from TSPC
ipp_obe_in_b	Input	Output buffer enable from SIUL and input for TSPC

11.5 Initialization

This module does not require initialization.

11.6 Application information

TSPC IP block provides the functionality of simultaneously transitioning two pads into high impedance, in order to experience a more robust recognition of touch sense.

11.7 TSPC register descriptions

11.7.1 TSPC memory map

TSPC base address: 402C_4000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Group Enable (GRP_EN)	32	RW	0000_0000h
50h	Group OBE (GRP1_OBE1)	32	RW	0000_0000h
54h	Group OBE (GRP1_OBE2)	32	RW	0000_0000h
A0h	Group OBE (GRP2_OBE1)	32	RW	0000_0000h
A4h	Group OBE (GRP2_OBE2)	32	RW	0000_0000h

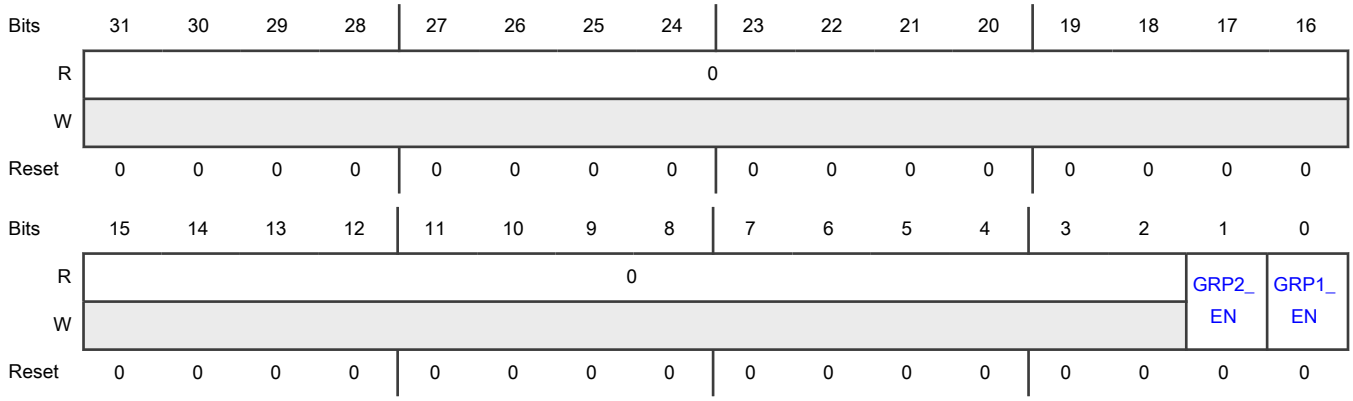
11.7.2 Group Enable (GRP_EN)

Offset

Register	Offset
GRP_EN	0h

Function
Allows enabling of the group, the pads of which participate in simultaneous transition.

Diagram



Fields

Field	Function
31-2 —	Reserved
1 GRP2_EN	Enable for GRP2_OBEn Register Enables or disables the GRP2_OBE*n* register that participates in grouping. 0b - Disable 1b - Enable
0 GRP1_EN	Enable for GRP1_OBEn Register Enables or disables the GRP1_OBE*n* register that participates in grouping. 0b - Disable 1b - Enable

11.7.3 Group OBE (GRP1_OBE1 - GRP2_OBE1)

Offset

Register	Offset
GRP1_OBE1	50h
GRP2_OBE1	A0h

Function

Allows grouping of OBEs in the respective group. Each field corresponds to an OBE pad.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	OBE31	OBE30	OBE29	OBE28	OBE27	OBE26	OBE25	OBE24	OBE23	OBE22	OBE21	OBE20	OBE19	OBE18	OBE17	OBE16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	OBE15	OBE14	OBE13	OBE12	OBE11	OBE10	OBE9	OBE8	OBE7	OBE6	OBE5	OBE4	OBE3	OBE2	OBE1	OBE0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 OBEn	Output Buffer Enable Indicates if a particular OBE participates in grouping. 0b - OBE pad does not transition if any OBE in group transitions from high to low. The OBE pad value is unaltered at the output. 1b - OBE pad transitions from high to low if any OBE in group transitions from high to low.

11.7.4 Group OBE (GRP1_OBE2)

Offset

Register	Offset
GRP1_OBE2	54h

Function
Allows grouping of OBEs in the respective group. Each field corresponds to an OBE pad.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		OBE45	OBE44	OBE43	OBE42	OBE41	OBE40	OBE39	OBE38	OBE37	OBE36	OBE35	OBE34	OBE33	OBE32
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-14 —	Reserved
13-0 OBEn	Output Buffer Enable Indicates if a particular OBE participates in grouping. 0b - OBE pad does not transition if any OBE in group transitions from high to low. The OBE pad value is unaltered at the output. 1b - OBE pad transitions from high to low if any OBE in group transitions from high to low.

11.7.5 Group OBE (GRP2_OBE2)

Offset

Register	Offset
GRP2_OBE2	A4h

Function
Allows grouping of OBEs in the respective group. Each field corresponds to an OBE pad.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0										OBE37	OBE36	OBE35	OBE34	OBE33	OBE32
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-6 —	Reserved
5-0	Output Buffer Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
OBEn	Indicates if a particular OBE participates in grouping. 0b - OBE pad does not transition if any OBE in group transitions from high to low. The OBE pad value is unaltered at the output. 1b - OBE pad transitions from high to low if any OBE in group transitions from high to low.

11.8 Glossary

GPIO	General purpose input and output
OBE	Output buffer enable

Chapter 12
Crossbar Switch (AXBS)

12.1 Chip-specific AXBS information

12.1.1 AXBS instances and connectivity matrix

This chip has up to seven instances of AXBS.

NOTE

AXBS support concurrent transaction requests from initiator to same target and do time multiplex internally. Also, it supports concurrent transactions between mutually exclusive initiator-target pairs.

Table 45. AXBS instances

Instance	S32K388/ S32K389	S32K358/ S32K356/ S32K348/ S32K338/ S32K328	S32K344/ S32K324/ S32K314	S32K342/ S32K322/ S32K341/ S32K322	S32K310/ S32K311/ S32K312
AXBS_0	Yes	Yes	Yes	Yes	Yes[1]
AXBS_1	Yes	Yes	Yes	Yes	No
AXBS_2	Yes	Yes	Yes	Yes	No
AXBS_3	Yes	Yes	Yes	Yes	No
AXBS_4	Yes	Yes	Yes	Yes	No
AXBS_5	Yes	No	No	No	No
AXBS_6	Yes	No	No	No	No

1. AXBS for S32K311 does not have programming model.

Table 46. AXBS connectivity matrix for S32K388/S32K389

Instance	Initiator and target assignments			
AXBS_0 (main)				
	Initiator port	**Initiator module**	**Target port**	**Target module**
	M0	Cortex-M7_0 AHBM	S0	S32K388: Flash memory port 0 S32K389: Flash memory 0 port 0
	M1	DMA	S1	S32K388: Flash memory port 1 S32K389: Flash memory 1 port 0
	M2	HSE_B/AES_ACCEL	S2	PRAM_0
	M3	GMAC_0	S3	S32K388: Cortex-M7 TCM/ PRAM_2 S32K389: Cortex-M7 TCM/ PRAM_2 / PRAM_3

Instance	Initiator and target assignments

M4	Cortex-M7_1 AHBM	S4	S32K388: Flash memory Port 2 S32K389: Flash memory 0 port 1
M5	Cortex-M7_2 AHBM	S5	QuadSPI
M6	GMAC_1	S6	PRAM_1
M7	Cortex-M7_3 AHBM	S7	S32K388: Flash memory port 3 S32K389: Flash memory 1 port 1

AXBS_1 (peripheral)

Initiator port	Initiator module	Target port	Target module
M0	Cortex-M7_0 AHBP	S0	AIPS_0
M1	DMA	S1	AIPS_1
M2	HSE_B/ ACE	S2	AIPS_2
M3	Cortex-M7_1 AHBP	S3	ACE target
M4	Cortex-M7_2 AHBP		
M5	Cortex-M7_3 AHBP		

AXBS_2 (eDMA)

Initiator port	Initiator module	Target port	Target module
M0	eDMA	S0	System AXBS
		S1	Peripheral AXBS

AXBS_3 (Cortex-M7 TCM)

Initiator port	Initiator module	Target port	Target module
M0	TCM PRAM AXBS	S0	Cortex-M7_0 TCM
		S1	Cortex-M7_1 TCM
		S2	Cortex-M7_2 TCM
		S3	Cortex-M7_3 TCM

AXBS_4 (TCM PRAM)

Initiator port	Initiator module	Target port	Target module
M0	System AXBS	S0	PRAM_2

Table 46. AXBS connectivity matrix for S32K388/S32K389 (continued)

Instance	Initiator and target assignments				
				S1	TCM AXBS
				S2	PRAM_3
AXBS_5 (ACE HSE_B AXBS)	**Initiator port**	**Initiator module**		**Target port**	**Target module**
	M0	HSE_B		S0	Main AXBS
	M1	ACE		S1	Peripheral AXBS
AXBS_6 (ACE AXBS)	**Initiator port**	**Initiator module**		**Target port**	**Target module**
	M0	ACE M0		S0	ACE HSE AXBS
	M1	ACE M1			

Table 47. AXBS connectivity matrix for S32K358/S32K356/S32K348/S32K338/S32K328

Instance	Initiator and target assignments				
AXBS_0 (main)	**Initiator port**	**Initiator module**		**Target port**	**Target module**
	M0	Cortex-M7_0 AHBM		S0	Flash memory port 0
	M1	AXBS_2 S0		S1	Flash memory port 1
	M2	AXBS_4 S0		S2	PRAM_0
	M3	GMAC		S3	PRAM2_TCM splitter
	M4	Cortex-M7_1 AHBM		S4	Flash memory port 2
	M5	Cortex-M7_2 AHBM		S5	QuadSPI
	M6	uSDHC		S6	PRAM_1
				S7	Flash memory port 3
AXBS_1 (peripheral)	**Initiator port**	**Initiator module**		**Target port**	**Target module**
	M0	Cortex-M7_0 AHBP		S0	AIPS_0
	M1	AXBS_2 S1		S1	AIPS_1
	M2	AXBS_4 S1		S2	AIPS_2

Instance	Initiator and target assignments
	M3 — Cortex-M7_1 AHBP M4 — Cortex-M7_2 AHBP
AXBS_2 (eDMA)	Initiator port / Initiator module / Target port / Target module M0 — eDMA — S0 — AXBS_0 M1 M1 — Reserved — S1 — AXBS_1 M1
AXBS_3 (Cortex-M7 TCM)	Not applicable
AXBS_4 (HSE)	Initiator port / Initiator module / Target port / Target module M0 — HSE_B — S0 — AXBS_0 M2 S1 — AXBS_1 M2
AXBS_5 (TCM_PRAM)	Initiator port / Initiator module / Target port / Target module M0 — AXBS_0 S3 — S0 — PRAM_2 S1 — Cortex-M7_0 TCM S2 — Cortex-M7_1 TCM S3 — Cortex-M7_2 TCM

Table 48. AXBS connectivity matrix for S32K344, S32K324, S32K314, S32K342, S32K322, S32K341, and S32K322

Instance	Initiator and target assignments
AXBS_0 (main)	Initiator port / Initiator module / Target port / Target module M0 — Cortex-M7_0 AHBM — S0 — Flash memory port 0 M1 — AXBS_2 S0 — S1 — Flash memory port 1 M2 — HSE_B — S2 — PRAM_0 M3 — EMAC — S3 — Cortex-M7 TCM M4[1] — Cortex-M7_1 AHBM — S4 — Flash memory port 2 S5 — QuadSPI

Instance	Initiator and target assignments

			S6	PRAM_1[2]

1. These ports are reserved for S32K314.
2. These ports are reserved for S32K342/S32K322/S32K341.

AXBS_1 (peripheral)

Initiator port	Initiator module		Target port	Target module
M0	Cortex-M7_0 AHBP		S0	AIPS_0
M1	AXBS_2 S1		S1	AIPS_1
M2	HSE_B		S2	AIPS_2
M3[1]	Cortex-M7_1 AHBP			

AXBS_2 (eDMA)

Initiator port	Initiator module		Target port	Target module
M0	eDMA		S0	AXBS_0 M1
			S1	AXBS_1 M1

AXBS_3 (Cortex-M7 TCM)

Initiator port	Initiator module		Target port	Target module
M0	AXBS_0 S3		S0	Cortex-M7_0 TCM
			S1[1]	Cortex-M7_1 TCM

AXBS_4 (HSE)

Initiator port	Initiator module		Target port	Target module
M0	HSE_B		S0	AXBS_0 M2
			S1[1]	AXBS_1 M2

Table 49. AXBS connectivity matrix for S32K311 and S32K312

Instance	Initiator and target assignments
AXBS_0 (main)	

Initiator port	Initiator module		Target port	Target module
M0	Cortex-M7 AHBM		S0	Flash memory port 0

Instance	Initiator and target assignments			
	M1	eDMA	S1	Flash memory port 1
	M2	HSE_B	S2	PRAM_0
	M3	Cortex-M7 AHBP	S3	Cortex-M7 TCM
	M4	Reserved	S4	AIPS_0
			S5	AIPS_1

12.2 Overview

This section provides information on the layout, configuration, and programming of the crossbar switch.

The crossbar switch connects bus initiators and bus targets using a crossbar switch structure. This structure allows all bus initiators to access different bus targets simultaneously, while providing arbitration among the bus initiators when they access the same target. A variety of bus arbitration methods and attributes may be programmed on a target-by-target basis.

12.2.1 Features

- Symmetric crossbar bus switch implementation
  - Allows concurrent access from different Initiators to different Targets
  - Target arbitration attributes configured on a Target-by-Target basis
- Single-clock 64-bit transfer
- Support for burst transfers of 64 bits of data
- Support for low-power park mode
- Initiator high-priority elevation
- 64-bit AHB crossbar bus switch compatible with ARM's AMBA Specification v2.0

12.3 Functional description

Information about general operation and arbitration are provided in this section.

12.3.1 General operation

When a Initiator accesses the crossbar switch, the access is immediately taken. If the targeted target port of the access is available, then the access is immediately presented on the target port. Single-clock or zero-wait-state accesses are possible through the crossbar. If the targeted target port of the access is busy or parked on a different Initiator port, the requesting Initiator sees wait states inserted until the targeted target port can service the Initiator's request. The latency in servicing the request depends on each Initiator's priority level and the responding target's access time.

Because the crossbar switch appears to be just another target to the Initiator device, the Initiator device does not know whether it owns the target port it is targeting. The Initiator waits while it does not have control of the target port it is targeting.

After the Initiator acquires control of the target port, it controls the port until it relinquishes the port by running an IDLE cycle or by targeting a different target port for its next access.

The Initiator can also lose control of the target port if another higher-priority Initiator makes a request to the target port. However, if the Initiator is running a fixed-length burst transfer, it retains control of the target port until that transfer completes.

The crossbar terminates all Initiator IDLE transfers, as opposed to allowing the termination to come from one of the target buses. Additionally, when no Initiator is requesting access to a target port, the crossbar drives IDLE transfers onto the target bus, even though a default Initiator may be granted access to the target port.

When a target bus is being idled by the crossbar, it can park the target port on the Initiator port indicated by CRS*n*[PARK]. This is done to save the initial clock of arbitration delay that otherwise would be seen if the same Initiator had to arbitrate to gain control of the target port. The target port can also be put into low-power park mode to save power, by using CRS*n*[PCTL].

12.3.2 Register coherency

The operation of the crossbar is affected as soon as a register is written. The values of the registers do not track with target-port-related Initiator accesses, but instead track only with target accesses.

12.3.3 Arbitration

The crossbar switch supports the following arbitration algorithms:

- Fixed priority
- Round-robin

The arbitration scheme is independently programmable for each target port.

12.3.3.1 Fixed-priority operation

When operating in fixed-priority mode, each initiator is assigned a unique priority level in the priority registers (PRS*n*). If two initiators request access to the same target port, the initiator with the highest priority in the selected priority register gains control over the target port.

NOTE

In this arbitration mode, a higher-priority initiator can monopolize a target port, preventing access from any lower-priority initiator to the port.

When a initiator makes a request to a target port, the target port checks whether the new requesting initiator's priority level is higher than that of the initiator that currently has control over the target port, unless the target port is in a parked state. The target port performs an arbitration check at every clock edge to ensure that the initiator, if any, has control of the target port.

The following table describes possible scenarios based on the requesting initiator port.

Table 50. Methods of how the crossbar switch grants control of a target port to a initiator

When	Then the crossbar switch grants control to the requesting initiator
Both of the following are true: • The current initiator is not running a transfer. • The new requesting initiator's priority level is higher than that of the current initiator.	At the next clock edge
Both of the following are true: • The current initiator is running a fixed-length burst transfer or a locked transfer. • The requesting initiator's priority level is higher than that of the current initiator.	At the end of the burst transfer or locked transfer
The requesting initiator's priority level is lower than the current initiator.	At the conclusion of one of the following cycles:

Table continues on the next page...

Table 50. Methods of how the crossbar switch grants control of a target port to a initiator (continued)

When	Then the crossbar switch grants control to the requesting initiator
	• An IDLE cycle • A non-IDLE cycle to a location other than the current target port

12.3.3.2 Round-robin priority operation

When operating in round-robin mode, each initiator is assigned a relative priority based on the initiator port number. This relative priority is compared to the initiator port number (ID) of the last initiator to perform a transfer on the target bus. The highest priority requests the initiator owns the target bus at the next transfer boundary, accounting for locked and fixed-length burst transfers. Priority is based on how far ahead the ID of the requesting initiator is of the ID of the last initiator.

After a initiator is granted access to a target port, a initiator may perform as many transfers as desired to that port until another initiator requests the same target port. The next initiator in line is granted access to the target port at the next transfer boundary, or possibly on the next clock cycle, if the current initiator has no pending access request.

As an example of arbitration in round-robin mode, assume that the crossbar is implemented with initiator ports 0, 1, 4, and 5. If the last initiator of the target port was initiator 1, and initiators 0, 4, and 5 make simultaneous requests, they are serviced in this order: 4,5, and then 0.

Parking may continue to be used in a round-robin mode, but does not affect the round-robin pointer unless the parked initiator performs a transfer. Handoff occurs to the next initiator in line after one cycle of arbitration. If the target port is put into low-power park mode, the round-robin pointer is reset to point at initiator port 0, giving it the highest priority.

12.3.3.3 Clocking

This module has no clocking considerations.

12.3.3.4 Interrupts

This module has no interrupts.

12.3.3.5 Priority assignment

Each initiator port must be assigned a unique 3-bit priority level. If an attempt is made to program multiple initiator ports with the same priority level within the priority registers (PRS*n*), the crossbar switch responds with a bus error and the registers are not updated.

12.4 External signals

This module has no external signals.

12.5 Initialization/application information

No initialization is required for the crossbar switch.

Hardware reset ensures that all the register bits used by the crossbar switch are properly initialized to a valid state. However, the following settings and priorities may be programmed to achieve the maximum system performance:

- During the configuration of the crossbar switch, all other initiators must be IDLE.
- To prevent reconfiguration of the crossbar switch, write 1 to CRS*n*[RO].

12.6 Memory map and register definition

Each target port of the crossbar switch contains configuration registers. Read- and write transfers require two bus clock cycles. The registers can be read from and written to only in supervisor mode. Additionally, these registers can be read from or written to only by 32-bit accesses.

A bus error response is returned if an unimplemented location is accessed within the crossbar switch.

The CRS*n* and PRS*n* registers can be programmed as read-only to prevent changes to their configuration. After being read-only protected, future writes to them terminate with a data storage error.

NOTE

This section shows the registers for all eight initiator and target ports. If a initiator or target is not used on this particular chip, then unexpected results occur when writing to its registers. See the chip configuration details for the exact initiator and target assignments for your chip.

All references to the crossbar switch registers are based on the physical port connections.

12.6.1 AXBS register descriptions

12.6.1.1 AXBS memory map

AXBS_LITE base address: 4020_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Priority Target Registers (PRS0)	32	RW	7654_3210h
10h	Control Register (CRS0)	32	RW	0002_0110h
100h	Priority Target Registers (PRS1)	32	RW	7654_3210h
110h	Control Register (CRS1)	32	RW	0002_0110h
200h	Priority Target Registers (PRS2)	32	RW	7654_3210h
210h	Control Register (CRS2)	32	RW	0002_0110h
300h	Priority Target Registers (PRS3)	32	RW	7654_3210h
310h	Control Register (CRS3)	32	RW	0002_0110h
400h	Priority Target Registers (PRS4)	32	RW	7654_3210h
410h	Control Register (CRS4)	32	RW	0002_0110h
500h	Priority Target Registers (PRS5)	32	RW	7654_3210h
510h	Control Register (CRS5)	32	RW	0002_0110h
600h	Priority Target Registers (PRS6)	32	RW	7654_3210h
610h	Control Register (CRS6)	32	RW	0002_0110h
700h	Priority Target Registers (PRS7)	32	RW	7654_3210h
710h	Control Register (CRS7)	32	RW	0002_0110h

12.6.1.2 Priority Target Registers (PRS0 - PRS7)

Offset

Register	Offset
PRS0	0h
PRS1	100h
PRS2	200h
PRS3	300h
PRS4	400h
PRS5	500h
PRS6	600h
PRS7	700h

Function

The priority target registers(PRS*n*) set the priority of each initiator port on a per target port basis and reside in each target port. The priority register can be accessed only with 32-bit access. After the CRS*n*[RO] bit is set, the PRS*n* register can only be read; attempts to write to it have no effect on PRS*n* and result in a bus-error response to the initiator initiating the write.

Two available initiators must not be programmed with the same priority level. Attempts to program two or more initiators with the same priority level result in a bus-error response and the PRS*n* is not updated.

NOTE

Valid values for the M*n* priority fields depend on which initiators are available on the chip. This information can be found in the chip-specific information for the crossbar.

- If the chip contains fewer than three initiators, only one bit is valid.
- If the chip contains fewer than five initiators, only two bits are valid.
- If five or more initiators are present, all three bits of the priority field are used.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	M7			0	M6			0	M5			0	M4		
W																
Reset	0	1	1	1	0	1	1	0	0	1	0	1	0	1	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	M3			0	M2			0	M1			0	M0		
W																
Reset	0	0	1	1	0	0	1	0	0	0	0	1	0	0	0	0

Fields

Field	Function
31 —	Reserved
30-28 M7	Initiator 7 Priority This field sets the arbitration priority for this port on the associated target port. 000b - This initiator has level 1 or highest priority when accessing the target port. 001b - This initiator has level 2 priority when accessing the target port. 010b - This initiator has level 3 priority when accessing the target port. 011b - This initiator has level 4 priority when accessing the target port. 100b - This initiator has level 5 priority when accessing the target port. 101b - This initiator has level 6 priority when accessing the target port. 110b - This initiator has level 7 priority when accessing the target port. 111b - This initiator has level 8 or lowest priority when accessing the target port.
27 —	Reserved
26-24 M6	Initiator 6 Priority This field sets the arbitration priority for this port on the associated target port. 000b - This initiator has level 1 or highest priority when accessing the target port. 001b - This initiator has level 2 priority when accessing the target port. 010b - This initiator has level 3 priority when accessing the target port. 011b - This initiator has level 4 priority when accessing the target port. 100b - This initiator has level 5 priority when accessing the target port. 101b - This initiator has level 6 priority when accessing the target port. 110b - This initiator has level 7 priority when accessing the target port. 111b - This initiator has level 8the or lowest priority when accessing the target port.
23 —	Reserved
22-20 M5	Initiator 5 Priority This field sets the arbitration priority for this port on the associated target port. 000b - This initiator has level 1 or highest priority when accessing the target port. 001b - This initiator has level 2 priority when accessing the target port. 010b - This initiator has level 3 priority when accessing the target port.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	011b - This initiator has level 4 priority when accessing the target port. 100b - This initiator has level 5 priority when accessing the target port. 101b - This initiator has level 6 priority when accessing the target port. 110b - This initiator has level 7 priority when accessing the target port. 111b - This initiator has level 8 or lowest priority when accessing the target port.
19 —	Reserved
18-16 M4	Initiator 4 Priority This field sets the arbitration priority for this port on the associated target port. 000b - This initiator has level 1 or highest priority when accessing the target port. 001b - This initiator has level 2 priority when accessing the target port. 010b - This initiator has level 3 priority when accessing the target port. 011b - This initiator has level 4 priority when accessing the target port. 100b - This initiator has level 5 priority when accessing the target port. 101b - This initiator has level 6 priority when accessing the target port. 110b - This initiator has level 7 priority when accessing the target port. 111b - This initiator has level 8 or lowest priority when accessing the target port.
15 —	Reserved
14-12 M3	Initiator 3 Priority This field sets the arbitration priority for this port on the associated target port. 000b - This initiator has level 1 or highest priority when accessing the target port. 001b - This initiator has level 2 priority when accessing the target port. 010b - This initiator has level 3 priority when accessing the target port. 011b - This initiator has level 4 priority when accessing the target port. 100b - This initiator has level 5 priority when accessing the target port. 101b - This initiator has level 6 priority when accessing the target port. 110b - This initiator has level 7 priority when accessing the target port. 111b - This initiator has level 8the or lowest priority when accessing the target port.
11 —	Reserved
10-8	Initiator 2 Priority

Table continues on the next page...

Table continued from the previous page...

Field	Function
M2	This field sets the arbitration priority for this port on the associated target port. 000b - This initiator has level 1 or highest priority when accessing the target port. 001b - This initiator has level 2 priority when accessing the target port. 010b - This initiator has level 3 priority when accessing the target port. 011b - This initiator has level 4 priority when accessing the target port. 100b - This initiator has level 5 priority when accessing the target port. 101b - This initiator has level 6 priority when accessing the target port. 110b - This initiator has level 7 priority when accessing the target port. 111b - This initiator has level 8the or lowest priority when accessing the target port.
7 —	Reserved
6-4 M1	Initiator 1 Priority This field sets the arbitration priority for this port on the associated target port. 000b - This initiator has level 1 or highest priority when accessing the target port. 001b - This initiator has level 2 priority when accessing the target port. 010b - This initiator has level 3 priority when accessing the target port. 011b - This initiator has level 4 priority when accessing the target port. 100b - This initiator has level 5 priority when accessing the target port. 101b - This initiator has level 6 priority when accessing the target port. 110b - This initiator has level 7 priority when accessing the target port. 111b - This initiator has level 8 or lowest priority when accessing the target port.
3 —	Reserved
2-0 M0	Initiator 0 Priority This field sets the arbitration priority for this port on the associated target port. 000b - This initiator has level 1 or highest priority when accessing the target port. 001b - This initiator has level 2 priority when accessing the target port. 010b - This initiator has level 3 priority when accessing the target port. 011b - This initiator has level 4 priority when accessing the target port. 100b - This initiator has level 5 priority when accessing the target port. 101b - This initiator has level 6 priority when accessing the target port. 110b - This initiator has level 7 priority when accessing the target port.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	111b - This initiator has level 8 or the lowest priority when accessing the target port.

12.6.1.3 Control Register (CRS0 - CRS7)

Offset

Register	Offset
CRS0	10h
CRS1	110h
CRS2	210h
CRS3	310h
CRS4	410h
CRS5	510h
CRS6	610h
CRS7	710h

Function

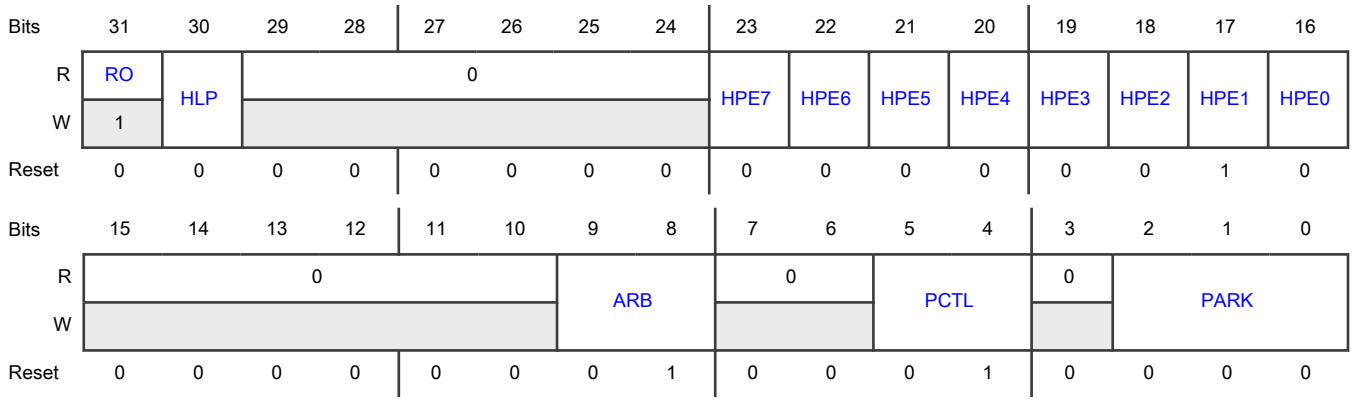
These registers control several features of each target port and must be accessed using 32-bit accesses. After CRS*n*[RO] is set, the PRS*n* can only be read; attempts to write to it have no effect and results in an error response.

NOTE

See the chip-specific crossbar information for the reset value of this register.

Not all HPE *n* fields may be active. See the chip-specific crossbar information for which initiators support initiator high-priority elevation. Setting a field corresponding to a initiator that does not support initiator, high-priority elevation has no effect.

Diagram



Fields

Field	Function
31 RO	Read Only Forces the PRS*n* and CRS*n* registers to be read-only. After being set, only a hardware reset clears this field. 0b - The CRS*n* and PRS*n* registers are writeable 1b - The CRS*n* and PRS*n* registers are read-only and cannot be written (attempted writes have no effect on the registers and result in a bus error response).
30 HLP	Halt Low Priority This field sets the initial arbitration priority for low power-mode requests . Setting this bit will not affect the ,request for low power-mode from attaining the highest priority after it has control of the target ports. 0b - The low-power mode request has the highest priority for arbitration on this target port. 1b - The low-power mode request has the lowest initial priority for arbitration on this target port.
29-24 —	Reserved
23 HPE7	High Priority Elevation 7 This field enables initiator high-priority elevation for initiator 7 on this target port. If enabled, the initiator is able to elevate its priority to the highest. 0b - Initiator high-priority elevation for initiator 7 is disabled on this target port. 1b - Initiator high-priority elevation for initiator 7 is enabled on this target port.
22 HPE6	High Priority Elevation 6 This field enables initiator high-priority elevation for initiator 6 on this target port. If enabled, the initiator is able to elevate its priority to the highest. 0b - Initiator high-priority elevation for initiator 6 is disabled on this target port. 1b - Initiator high-priority elevation for initiator 6 is enabled on this target port.
21 HPE5	High Priority Elevation 5 This field enables initiator high-priority elevation for initiator 5 on this target port. If enabled, the initiator is able to elevate its priority to the highest. 0b - Initiator high-priority elevation for initiator 5 is disabled on this target port. 1b - Initiator high-priority elevation for initiator 5 is enabled on this target port.
20 HPE4	High Priority Elevation 4 This field enables initiator high-priority elevation for initiator 4 on this target port. If enabled, the initiator can elevate its priority to the highest. 0b - Initiator high-priority elevation for initiator 4 is disabled on this target port. 1b - Initiator high-priority elevation for initiator 4 is enabled on this target port.
19	High Priority Elevation 3

Table continues on the next page...

Table continued from the previous page...

Field	Function
HPE3	This field enables initiator high-priority elevation for initiator 3 on this target port. If enabled, the initiator can elevate its priority to the highest. 0b - Initiator high-priority elevation for initiator 3 is disabled on this target port. 1b - Initiator high-priority elevation for initiator 3 is enabled on this target port.
18 HPE2	High Priority Elevation 2 This field enables initiator high-priority elevation for initiator 2 on this target port. If enabled, the initiator can elevate its priority to the highest. 0b - Initiator high-priority elevation for initiator 2 is disabled on this target port. 1b - Initiator high-priority elevation for initiator 2 is enabled on this target port.
17 HPE1	High Priority Elevation 1 This field enables initiator high-priority elevation for initiator 1 on this target port. If enabled, the initiator can elevate its priority to the highest. 0b - Initiator high-priority elevation for initiator 1 is disabled on this target port. 1b - Initiator high-priority elevation for initiator 1 is enabled on this target port.
16 HPE0	High Priority Elevation 0 This field enables initiator high-priority elevation for initiator 0 on this target port. If enabled, the initiator can elevate its priority to the highest. 0b - Initiator high-priority elevation for initiator 0 is disabled on this target port. 1b - Initiator high-priority elevation for initiator 0 is enabled on this target port.
15-10 —	Reserved
9-8 ARB	Arbitration Mode This field selects the arbitration policy for the target port. 00b - Fixed priority 01b - Round-robin (rotating) priority 10b - Reserved 11b - Reserved
7-6 —	Reserved
5-4 PCTL	Parking Control This field determines the target port's parking control. The low-power park feature results in an overall power savings if the target port is not saturated; however, this forces an extra latency clock when any initiator tries to access the target port when not in use because it is not parked on any initiator.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00b - When no initiator makes a request, the arbiter parks the target port on the initiator port defined by the PARK bit field. 01b - When no initiator makes a request, the arbiter parks the target port on the last initiator to be in control of the target port. 10b - Low-power park. When no initiator makes a request, the target port is not parked on a initiator and the arbiter drives all outputs to a constant safe state. 11b - Reserved
3 —	Reserved
2-0 PARK	Park This field determines which initiator port the current target port parks on when no initiators are actively making requests and the PCTL bits are cleared. **NOTE** Select only initiator ports that are present on the chip. Otherwise, undefined behavior might occur. 000b - Park on initiator port M0 001b - Park on initiator port M1 010b - Park on initiator port M2 011b - Park on initiator port M3 100b - Park on initiator port M4 101b - Park on initiator port M5 110b - Park on initiator port M6 111b - Park on initiator port M7

12.7 Glossary

AMBA Advanced Microcontroller Bus Architecture

IDLE A type of transfer that a initiator uses when it does not want to perform a data transfer

ID Initiator port number

Chapter 13
Peripheral Bridge (AIPS_Lite)

13.1 Chip-specific AIPS_Lite information

13.1.1 AIPS_Lite instances

The following table summarizes AIPS_Lite instances on S32K3xx product series to support simultaneous data transfers to different peripherals. See the memory map file attached to this document to find information related to these peripherals that are associated with AIPS_Lite instances.

Table 51. AIPS_Lite instances

Instance	S32K388, S32K389, S32K358, S32K356, S32K348, S32K338, S32K328, S32K341, S32K342, S32K344, S32K324, S32K322, S32K314	S32K312, S32K311, S32K310
AIPS_0	Yes	Yes
AIPS_1	Yes	Yes
AIPS_2	Yes	No

13.2 Overview

AIPS_Lite converts the crossbar switch interface to an interface that can access most of the slave peripherals on this chip.

This peripheral bridge occupies 64 MB of the address space, which is divided into peripheral slots of 16 KB each. All the peripherals may not be used. See the memory map chapter for details on slot assignments. The bridge includes separate clock enable inputs for each of the slots to accommodate slower peripherals.

13.2.1 Features

Following are the key features of the peripheral bridge:

- Supports peripheral slots with 8-, 16-, and 32-bit datapath width
- Supports a pair of 32-bit transactions for selected 64-bit memory accesses

13.2.2 General operation

The slave devices connected to the peripheral bridge are modules that contain a programming model of control and status registers. The system masters read and write these registers through the peripheral bridge.

The register maps of the peripherals are located on 16 KB boundaries. Each peripheral is allocated one or more 16-KB block(s) of the memory map.

13.3 Functional description

The peripheral bridge functions as a bus protocol translator between the crossbar switch and the slave peripheral bus. Support is provided for generating a pair of 32-bit slave accesses when performing certain 64-bit peripheral accesses.

The peripheral bridge manages all transactions for the attached slave devices and generates select signals for modules on the peripheral bus by decoding accesses within the attached address space.

13.3.1 Access support

All combinations of access size and peripheral data port width are supported. An access that is larger than the target peripheral's data width is decomposed to multiple, smaller accesses. Bus decomposition is terminated by a transfer error caused by an access to an empty register area.

13.3.2 Clocking

This module has no clocking considerations.

13.3.3 Interrupts

This module has no interrupts.

13.4 External signals

This module has no external signals.

13.5 Memory map and register definition

The AIPS module(s) on this chip do(es) not contain any user-programmable registers.

Chapter 14
Direct Memory Access Multiplexer (DMAMUX)

14.1 Chip-specific DMAMUX information

14.1.1 DMAMUX instances

This chip has two instances of DMAMUX: DMAMUX_0 and DMAMUX_1. See the DMAMUX map file attached to this document to find information related to the slot number for each DMA trigger source.

NOTE

- For S32K310, S32K311, and S32K312: DMAMUX_0 channels 0-5 and DMAMUX_1 channels 0-5 are mapped to eDMA Transfer Control Descriptor(TCD) 0-5 and eDMA Transfer Control Descriptor(TCD) 6-11, respectively. Programming of DMAMUX_0 channels 6-15 and DMAMUX_1 channels 6-15 is not therefore expected however if programmed then any access will terminate with either error response for channels 8-15 or without error response for channels 6-7.
- For remaining S32K3xx devices: DMAMUX_0 channel 0-15 and DMAMUX_1 channel 0-15 are mapped to eDMA Transfer Control Descriptor(TCD) 0-15 and eDMA Transfer Control Descriptor(TCD) 16-31, respectively.

14.1.2 Periodic DMA triggering

PIT generates periodic trigger events to DMAMUX as shown in this table.

Table 52. PIT channel assignment for periodic DMA triggering through DMAMUX

DMAMUX0/1 ports	PIT channel
pit_dma_trigger[0]	PIT0 channel 0
pit_dma_trigger[1]	PIT0 channel 1
pit_dma_trigger[2]	PIT0 channel 2
pit_dma_trigger[3]	PIT0 channel 3

14.2 Introduction

14.2.1 Overview

DMAMUX routes a DMA source, called slot, to any of the 16 DMA channels. See the chip-specific information for details.

14.2.2 Block diagram

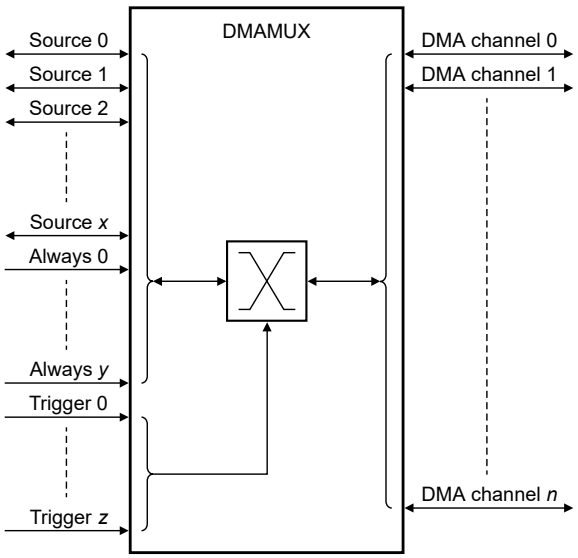


Figure 30. Block diagram

14.2.3 Features

- Ability to route up to 61 peripheral slots and up to 2 always-on slots to 16 channels
- Supports 16 independently selectable DMA channel routers (the first 4 channels provide an additional trigger functionality)
- Allows assignment of each channel router to one of the possible peripheral DMA slots or always-on slots

14.2.4 Modes of operation

The following table describes the functional operating modes of DMAMUX.

Table 53. Modes of operation

Mode	Description
Disabled	In this mode, the DMA channel is disabled. Because disabling and enabling of DMA channels is done primarily via the DMA configuration registers, this mode is used mainly as the reset state for a DMA channel in the DMA channel MUX. You can also use this mode to temporarily suspend a DMA channel while system reconfiguration takes place, for example, by changing the period of a DMA trigger.
Normal	In this mode, a DMA source is routed directly to the specified DMA channel. The operation of DMAMUX in this mode is completely transparent to the system.
Periodic trigger	In this mode, a DMA source can only request a DMA transfer, such as, when a transmit buffer becomes empty or a receive buffer becomes full, periodically. You configure the period using an external periodic interrupt timer (PIT). This mode is available only for channels 0–3. See the chip-specific section for more details.

14.3 Functional description

The primary purpose of DMAMUX is to provide flexibility in the system's use of the available DMA channels. Configuration of DMAMUX is intended to be a static procedure performed during the execution of the system boot code. However, if you follow the procedure outlined in Enabling and configuring sources, you can change the configuration of DMAMUX during the normal operation of the system. All DMA channels must be inactive before and during a configuration change.

Functionally, DMAMUX channels can be divided into two classes:

- Channels that implement the normal routing functionality plus the periodic triggering capability
- Channels that implement only the normal routing functionality

14.3.1 DMA channels with periodic triggering capability

Besides the normal routing functionality, the first four channels of DMAMUX provide a special periodic triggering capability that you can use to provide an automatic mechanism to transmit bytes, frames, or packets at fixed intervals without the need for processor intervention (see Figure 31).

An external periodic interrupt timer (PIT) generates the trigger; for example, by configuring the external periodic timer, you configure the periodic triggering interval.

NOTE

Because of the dynamic nature of the system (owing to DMA channel priorities, bus arbitration, interrupt service routine lengths, and so on), the number of clock cycles between a trigger and the actual DMA transfer cannot be guaranteed.

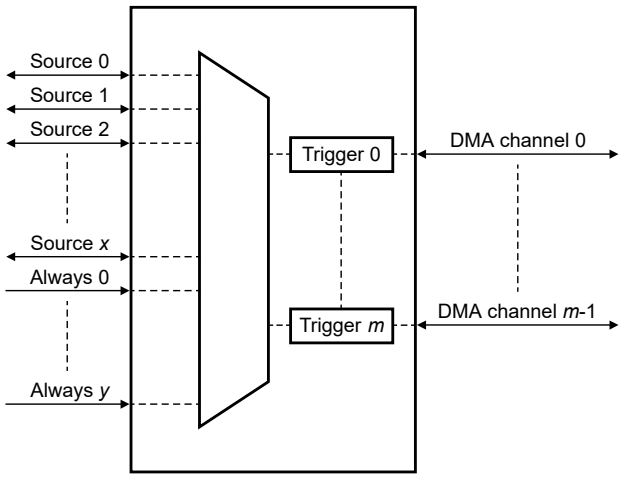


Figure 31. DMAMUX triggered channels

The DMA channel triggering capability allows the system to schedule regular DMA transfers, usually on the transmit side of certain peripherals, without the intervention of the processor. This trigger works by gating the request from the peripheral to DMA until a trigger event is observed (see Figure 32).

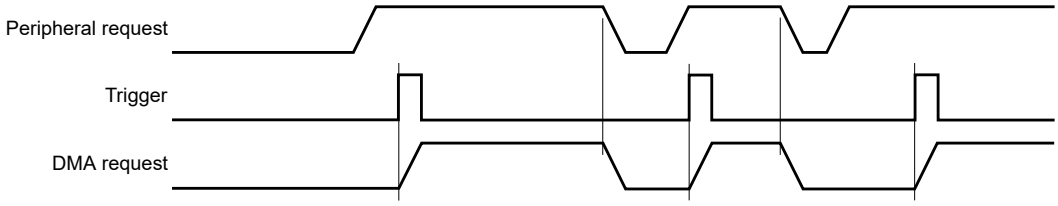


Figure 32. DMAMUX channel triggering: normal operation

After the DMA request is serviced, the peripheral negates its request, effectively resetting the gating mechanism until the peripheral reasserts its request and the next trigger event is seen. This means that if a trigger is seen, but the peripheral is not requesting a transfer, then that trigger is ignored (see Figure 33).

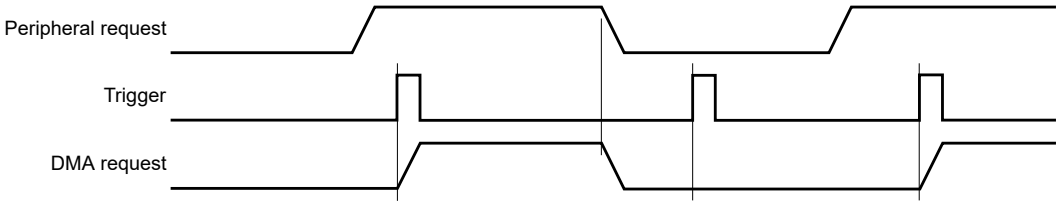


Figure 33. DMAMUX channel triggering: ignored trigger

You can use this triggering capability with any peripheral that supports DMA transfers, and is most useful in the following situations:

- Periodic polling of external devices on a particular bus: as an example, the transmit side of an SPI is assigned to a DMA channel with a trigger, as described in the aforementioned text. After periodic polling is set up, SPI requests DMA transfers, presumably from memory, as long as its transmit buffer is empty. By using a trigger on this channel, the SPI transfers can be automatically performed every 5 µs (as an example). On the receive side of SPI, you can configure SPI and DMA to transfer the received data into memory, effectively implementing a method to periodically read data from external devices and transferring the results into memory without processor intervention.
- Using the GPIO ports to drive or sample waveforms: by configuring DMA to transfer data to one or more GPIO ports, it is possible to create complex waveforms using tabular data stored in on-chip memory. Conversely, using DMA to periodically transfer data from one or more GPIO ports, it is possible to sample complex waveforms and store the results in a tabular form in on-chip memory.

14.3.2 DMA channels with no triggering capability

The other channels of DMAMUX provide the normal routing functionality as described in Modes of operation.

14.3.3 Always-enabled DMA sources

In addition to the peripherals that you can use as DMA sources, 2 additional DMA sources are always enabled. Unlike the peripheral DMA sources, where the peripheral controls the flow of data during DMA transfers, the sources that are always enabled provide no such "throttling" of data transfers. These sources are most useful in the following cases:

- Performing DMA transfers to and from GPIO: moving data to and from one or more GPIO pins, either unthrottled (that is, as fast as possible), or periodically (using the DMA triggering capability)
- Performing DMA transfers from memory to memory: moving data from memory to memory, typically as fast as possible, sometimes with software activation
- Performing DMA transfers from memory to the external bus, or vice-versa: similar to memory-to-memory transfers, this is typically done as quickly as possible
- Any DMA transfer that software must explicitly start or activate

In cases where software must initiate the start of a DMA transfer, an always-enabled DMA source can be used to provide maximum flexibility. When activating a DMA channel via software, subsequent executions of the minor loop require that a new start event be sent. This can either be a new software activation or a transfer request from the DMA channel MUX. The options for doing this are as follows.

- Transfer all data in a single minor loop: by configuring DMA to transfer all of the data in a single minor loop (that is, major loop counter = 1), no reactivation of the channel is necessary. The disadvantage of this option is reduced granularity in determining the load that the DMA transfer imposes on the system. For this option, the DMA channel must be disabled in the DMA channel MUX.
- Use explicit software reactivation: using this option, DMA is configured to transfer the data using both minor and major loops, but the processor is required to reactivate the channel by writing to the DMA registers after every minor loop. For this option, the DMA channel must be disabled in the DMA channel MUX.
- Use an always-enabled DMA source: using this option, DMA is configured to transfer the data using both minor and major loops, and the DMA channel MUX does the channel reactivation. For this option, you must enable the DMA channel and point it to an "always-enabled" source. Channel reactivation can be continuous (DMA triggering is disabled) or can use the DMA triggering capability. In this manner, it is possible to execute periodic transfers of data packets from one source to another, without processor intervention.

14.4 Initialization and application information

This section provides instructions for initializing the DMA channel MUX.

14.4.1 Reset

The reset state of each field is shown in Memory map and register definition. In summary, after reset, all channels are disabled and you must enable them explicitly before use.

14.4.2 Enabling and configuring sources

14.4.2.1 Enable a source with periodic triggering

Perform the following procedure to enable a source with periodic triggering:

1. Determine the DMA channel with which the source is associated. Only the first 4 DMA channels have periodic triggering capability.
2. Write 0 to CHCFG*n*[ENBL] and CHCFG*n*[TRIG] of the DMA channel.
3. Ensure that the DMA channel is properly configured in DMA. You can enable the DMA channel at this point.
4. Configure the corresponding timer.
5. Select the source to be routed to the DMA channel. Write to the corresponding CHCFG register, ensuring that CHCFG*n*[ENBL] and CHCFG*n*[TRIG] are 1.

The following is an example. See the chip-specific information for the number of DMA channels on this chip that have triggering capability.

Example: To configure source 5 transmit for use with DMA channel 1, with periodic triggering capability:

1. Write 0h to CHCFG1.
2. Configure channel 1 in DMA, including enabling the channel.
3. Configure a timer for the desired trigger interval.
4. Write C5h to CHCFG1.

The following code example illustrates steps 1 and 4 above:

```
In File registers.h:
#define DMAMUX_BASE_ADDR     0x40021000/* Example only ! */
/* Following example assumes char is 8-bits */
volatile unsigned char *CHCFG0 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0003);
volatile unsigned char *CHCFG1 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0002);
volatile unsigned char *CHCFG2 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0001);
volatile unsigned char *CHCFG3 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0000);
volatile unsigned char *CHCFG4 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0007);
volatile unsigned char *CHCFG5 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0006);
volatile unsigned char *CHCFG6 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0005);
volatile unsigned char *CHCFG7 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0004);
volatile unsigned char *CHCFG8 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000B);
volatile unsigned char *CHCFG9 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000A);
volatile unsigned char *CHCFG10= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0009);
volatile unsigned char *CHCFG11= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0008);
volatile unsigned char *CHCFG12= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000F);
volatile unsigned char *CHCFG13= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000E);
volatile unsigned char *CHCFG14= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000D);
volatile unsigned char *CHCFG15= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000C);

In File main.c:
#include "registers.h"
:
:
*CHCFG1 = 0x00;                              /* Clear all the fields of CHCFG1 register */
*CHCFG1 = 0xC5;                              /* ENBL = 1  DMA Channel is enabled */
                                             /* TRIG = 1  Triggering is enabled */
                                             /* SOURCE = 5  DMA Source 5 is selected */
```

Code Listing 1. Configuring source 5 with DMA channel 1 (with periodic triggering)

14.4.2.2 Enable a source without periodic triggering

Perform the following procedure to enable a source without periodic triggering:

1. Determine the DMA channel with which the source is associated. Only the first 4 DMA channels have periodic triggering capability.
2. Write 0 to CHCFG*n*[ENBL] and CHCFG*n*[TRIG] of the DMA channel.
3. Ensure that the DMA channel is properly configured in DMA. You can enable the DMA channel at this point.
4. Select the source to be routed to the DMA channel. Write to the corresponding CHCFG register, ensuring that CHCFG*n*[ENBL] is 1 while CHCFG*n*[TRIG] is 0.

The following is an example. See the chip configuration details for the number of DMA channels on this chip that have triggering capability.

Example: To configure source 5 transmit for use with DMA channel 1, with no periodic triggering capability:

1. Write 0h to CHCFG1.
2. Configure channel 1 in DMA, including enabling the channel.
3. Write 85h to CHCFG1.

The following code example illustrates steps 1 and 3 above:

```
In File registers.h:
#define DMAMUX_BASE_ADDR     0x40021000/* Example only ! */
/* Following example assumes char is 8-bits */
volatile unsigned char *CHCFG0 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0003);
volatile unsigned char *CHCFG1 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0002);
volatile unsigned char *CHCFG2 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0001);
volatile unsigned char *CHCFG3 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0000);
volatile unsigned char *CHCFG4 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0007);
volatile unsigned char *CHCFG5 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0006);
volatile unsigned char *CHCFG6 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0005);
volatile unsigned char *CHCFG7 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0004);
volatile unsigned char *CHCFG8 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000B);
volatile unsigned char *CHCFG9 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000A);
volatile unsigned char *CHCFG10= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0009);
volatile unsigned char *CHCFG11= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0008);
volatile unsigned char *CHCFG12= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000F);
volatile unsigned char *CHCFG13= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000E);
volatile unsigned char *CHCFG14= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000D);
volatile unsigned char *CHCFG15= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000C);

In File main.c:
#include "registers.h"
:
:
*CHCFG1 = 0x00;                              /* Clear all the fields of CHCFG1 register */
*CHCFG1 = 0x85;                              /* ENBL = 1  DMA Channel is enabled */
                                             /* TRIG = 0  Triggering is disabled */
                                             /* SOURCE = 5  DMA Source 5 is selected */
```

Code Listing 2. Configuring source 5 with DMA channel 1 (without periodic triggering)

14.4.2.3 Disable a source

You can disable a particular DMA source by not writing the corresponding source value into any of the CHCFG registers. Additionally, some module-specific configurations may be necessary. See the "Enhanced Direct Memory Access (eDMA)" chapter for details.

To switch the source of a DMA channel:

1. Disable the DMA channel in the DMA and reconfigure the channel for the new source.
2. Write 0 to CHCFG*n*[ENBL] and CHCFG*n*[TRIG] of the DMA channel.
3. Select the source to be routed to the DMA channel. Write to the corresponding CHCFG register, ensuring that CHCFG*n*[ENBL] and CHCFG*n*[TRIG] are 1.

Example: To switch DMA channel 8 from source 5 transmit to source 7 transmit:

1. In the DMA configuration registers, disable DMA channel 8 and reconfigure it to handle the transfers to peripheral slot 7. This example assumes channel 8 does not have the triggering capability.
2. Write 0h to CHCFG8.
3. Write 87h to CHCFG8. (In this example, writing 1 to CHCFG[TRIG] has no effect because of the assumption that channel 8 does not support the periodic triggering functionality.)

The following code example illustrates steps 2 and 3 above:

```
In File registers.h:
#define DMAMUX_BASE_ADDR     0x40021000/* Example only ! */
/* Following example assumes char is 8-bits */
volatile unsigned char *CHCFG0 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0003);
volatile unsigned char *CHCFG1 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0002);
volatile unsigned char *CHCFG2 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0001);
volatile unsigned char *CHCFG3 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0000);
volatile unsigned char *CHCFG4 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0007);
volatile unsigned char *CHCFG5 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0006);
volatile unsigned char *CHCFG6 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0005);
volatile unsigned char *CHCFG7 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0004);
volatile unsigned char *CHCFG8 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000B);
volatile unsigned char *CHCFG9 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000A);
volatile unsigned char *CHCFG10= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0009);
volatile unsigned char *CHCFG11= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0008);
volatile unsigned char *CHCFG12= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000F);
volatile unsigned char *CHCFG13= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000E);
volatile unsigned char *CHCFG14= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000D);
volatile unsigned char *CHCFG15= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000C);

In File main.c:
#include "registers.h"
:
:
*CHCFG8 = 0x00;                              /* Clear all the fields of CHCFG1 register */
*CHCFG8 = 0x87;                              /* ENBL = 1  DMA Channel is enabled */
                                             /* TRIG = 0  Triggering is disabled */
                                             /* SOURCE = 7  DMA Source 7 is selected */
```

Code Listing 3. Switching DMA channel 8 from source 5 transmit to source 7 transmit

14.5 Memory map and register definition

This section provides a detailed description of all memory-mapped registers in DMAMUX.

14.5.1 DMAMUX register descriptions

14.5.1.1 DMAMUX memory map

DMAMUX_0 base address: 4028_0000h

DMAMUX_1 base address: 4028_4000h

Offset	Register	Width (In bits)	Access	Reset value
0h - Fh	Channel Configuration (CHCFG0 - CHCFG15) [1]	8	RW	00h

1. In this array, the index and offset values of the registers do not increment in direct alignment. For details, see the register description.

14.5.1.2 Channel Configuration (CHCFG0 - CHCFG15)

Offset

For n = 0 to 15:

Register	Offset
CHCFGn	0h + (n + 3 - 2 × (n mod 4))

Function

Provides you with the configuration to enable or disable, and to associate each of the DMA channels with one of the DMA slots (peripheral slots or always-on slots) in the system.

NOTE

Writing to multiple CHCFG registers with the same source value results in unpredictable behavior. This is true, even if a channel is disabled (CHCFG*n*[ENBL] == 0).

Before changing the trigger or source settings, you must disable a DMA channel using CHCFG*n*[ENBL].

Diagram

Bits	7	6	5	4	3	2	1	0
R W	ENBL	TRIG	SOURCE					
Reset	0	0	0	0	0	0	0	0

Fields

Field	Function
7 ENBL	DMA Channel Enable Enables the DMA channel. The condition when the DMA channel is disabled (ENBL = 0) is called Disabled mode. This mode is primarily used during the configuration of DMAMUX. DMA has separate channel enables or disables, which you must use to disable or reconfigure a DMA channel. 0b - Disable 1b - Enable
6 TRIG	DMA Channel Trigger Enable Enables the periodic triggering capability for the triggered DMA channel. If triggering is disabled and ENBL = 1, the DMA channel routes the specified source to the DMA channel (Normal mode). If triggering is enabled and ENBL = 1, then DMAMUX is in Periodic Trigger mode.

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>DMAMUX_0</td><td>CHCFG0–CHCFG3</td><td>CHCFG4–CHCFG7
CHCFG8–CHCFG11
CHCFG12–CHCFG15</td></tr>
<tr><td>DMAMUX_1</td><td>CHCFG0–CHCFG3</td><td>CHCFG4–CHCFG7
CHCFG8–CHCFG11
CHCFG12–CHCFG15</td></tr>
</table>
0b - Disable
1b - Enable
</td></tr>
<tr><td>5-0
SOURCE</td><td>DMA Channel Source (Slot)
Specifies which DMA source, if any, is routed to a particular DMA channel. See the chip-specific DMAMUX information for details about peripherals and their slot numbers.</td></tr>
</table>

14.6 Glossary

Slot DMA source that either reflects a peripheral source or an always-enabled source

Periodic trigger Trigger that an external source generates at periodic intervals for DMA transfer

Chapter 15
Enhanced Direct Memory Access (eDMA)

15.1 Chip-specific eDMA information

15.1.1 eDMA instances

This chip has one instance of eDMA without any lockstep.

15.1.2 DMA channels

See the following table for number of DMA channels across S32K3xx product series.

Table 54. DMA channels

Chip	No. of DMA channels
S32K388/S32K389/S32K358/S32K356/S32K348/ S32K338/S32K328/S32K344/S32K324/S32K314/S32K342/ S32K322/S32K341	32
S32K311/S32K312/S32K310	12

NOTE

TCD_CH[12-31] are not available in S32K310, S32K11, and S32K312, so the registers corresponding to these channels are not available. See the memory map file attached to this document for details.

15.1.3 eDMA ID replication

The eDMA replicates the IDs for the access based on the configurations of XRDC and eDMA itself. The below figure describes the details of the hmaster and DID corresponding the the eDMA.

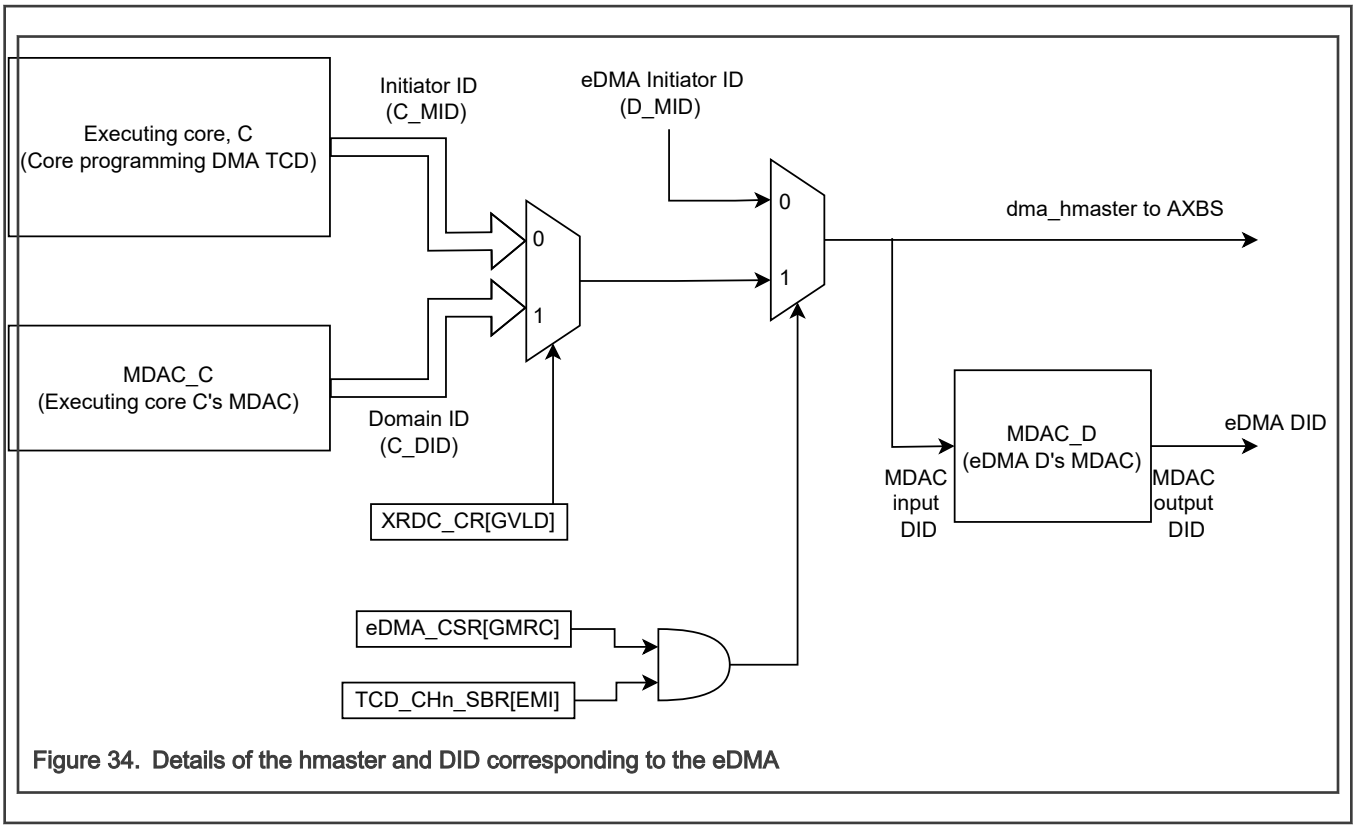


Figure 34. Details of the hmaster and DID corresponding to the eDMA

15.2 Overview

The enhanced direct memory access (eDMA) controller is capable of performing complex data transfers with minimal intervention from a host processor. The hardware microarchitecture includes:

- A DMA engine that performs:
  - Source address and destination address calculations
  - Data-movement operations
- Local memory containing transfer control descriptors for each of the 32 channels

NOTE

The terminology in this chapter has been updated to align with NXP's inclusive language standards, as shown in the table below.

Table 55. Updated terms

Updated term	Deprecated term
Initiator	Master
Target	Slave

15.2.1 Block diagram

Figure 35 illustrates the components of the eDMA system, including the eDMA module (engine).

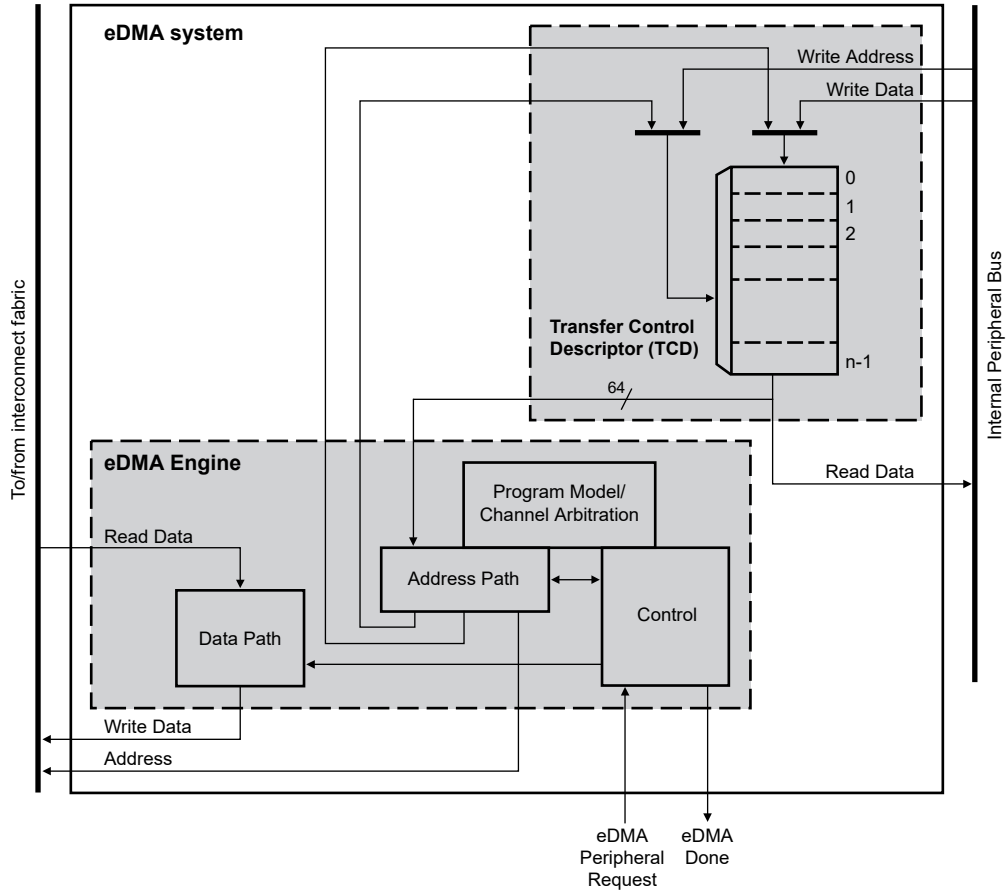


Figure 35. Block diagram

15.2.2 Block parts

The eDMA module is partitioned into two major modules: the eDMA engine and the transfer control descriptor local memory.

The eDMA engine is further partitioned into four submodules:

Table 56. eDMA engine submodules

Submodule	Function
Address path	This block: • Implements a primary channel and secondary (preempt) channel • Manages all initiator bus-address calculations All the channels provide the same functionality. This structure allows data transfers associated with one channel to be preempted after the completion of a read/write sequence if a higher priority channel activation is asserted while the primary channel is active. After a channel is activated, it runs until the minor loop is completed, unless preempted by a higher priority channel. This provides a mechanism (enabled by CH*n*_PRI[ECP]) where a large data transfer can be preempted to minimize the time another channel is blocked from execution.

Table continues on the next page...

Table 56. eDMA engine submodules (continued)

Submodule	Function
	When any channel is selected to execute, the contents of its TCD are read from local memory and loaded into the address path channel x registers for a normal start and into channel y registers for a preemption start. After the minor loop completes execution, the address path hardware writes the new values for the TCD*n*_{SADDR, DADDR, CITER} back to local memory. If the major iteration count is exhausted, additional processing is performed, including the final address pointer updates, reloading the TCD*n*_CITER field, and a possible fetch of a new TCD*n* from memory as part of a scatter/gather operation. See Dynamic scatter/gather for more details.
Data path	This block implements the bus initiator read/write data path. It includes a data buffer and the necessary multiplex logic to support any required data alignment. The internal read data bus is the primary input, and the internal write data bus is the primary output. The address and data path modules directly support the 2-stage pipelined internal bus. The address path module represents the first stage of the bus pipeline (address phase), and the data path module implements the second stage of the pipeline (data phase).
Program model/channel arbitration	This block implements the first section of the eDMA programming model as well as the channel arbitration logic. The programming model registers are connected to the internal peripheral bus. The eDMA peripheral request inputs and interrupt request outputs are also connected to this block (via control logic).
Control	This block provides all the control functions for the eDMA engine. For data transfers where the source and destination sizes are equal, the eDMA engine performs a series of source read/destination write operations until the number of bytes specified in the minor loop byte count has been moved from the source to the destination. For descriptors where the sizes are not equal, multiple accesses of the smaller size data are required for each reference of the larger size. As an example, if the source size references 16-bit data and the destination is 32-bit data, the eDMA performs two reads, then one 32-bit write.

The transfer control descriptor local memory is further partitioned into:

Table 57. Transfer control descriptor memory

Submodule	Description
Memory controller	This logic implements the required dual-ported controller, and manages accesses from the eDMA engine as well as references from the internal peripheral bus. In simultaneous accesses, the eDMA engine is given priority and the peripheral transaction is stalled.
Memory array	TCD storage for each channel's transfer profile.

15.2.3 Features

The eDMA is a highly programmable data-transfer engine optimized to minimize any required intervention from the host processor. It is intended for use in applications where the data size to be transferred is statically known and not defined within the transferred data itself. The eDMA module features:

- All data movement via dual-address transfers: read from source, write to destination
  - Programmable source and destination addresses and transfer size
  - Support for complex address calculations
- Implementation that performs complex data transfers with minimal intervention from a host processor
  - Internal data buffer, used as temporary storage for all transfers

  - Connections to the crossbar switch for bus driving the data movement
- TCD organized to support two-deep, nested transfer operations
  - 32-byte TCD stored in local memory for each channel
  - An inner data transfer loop defined by a minor byte transfer count
  - An outer data transfer loop defined by a major iteration count
- Channel activation via one of three methods:
  - Explicit software initiation
  - Initiation via a channel-to-channel linking mechanism for continuous transfers
  - Peripheral-paced hardware requests, one per channel
- Fixed-priority and round-robin channel arbitration
- Channel completion reported via programmable interrupt requests
  - One interrupt per channel, which can be asserted at completion of major iteration count
  - Programmable error terminations per channel that are logically summed together to form one error interrupt to the interrupt controller
- Programmable support for scatter/gather DMA processing
- Support for complex data structures

In the discussion of this module, *n* is used to reference the channel number.

15.3 Functional description

The operation of eDMA is described in the following subsections.

15.3.1 Modes of operation

The eDMA operates in the following modes:

Table 58. Modes of operation

Mode	Description
Normal	In Normal mode, eDMA transfers data between a source and a destination. The source and destination can be a memory block or an I/O block capable of operation with eDMA. A service request initiates a transfer of a specific number of bytes (NBYTES) as specified in the TCD. The minor loop is the sequence of read-write operations that transfers these NBYTES per service request. Each service request executes one iteration of the major loop, which transfers NBYTES of data.
Debug	eDMA operation is configurable in Debug mode via the control register: • If CSR[EDBG] is cleared to 0, eDMA continues to operate. • If CSR[EDBG] is set to 1, eDMA stops transferring data. If Debug mode is entered when a channel is active, eDMA continues operation until the channel retires.

15.3.2 eDMA basic data flow

The basic flow of a data transfer can be partitioned into three segments.

As shown in the following diagram, the first segment involves the channel activation:

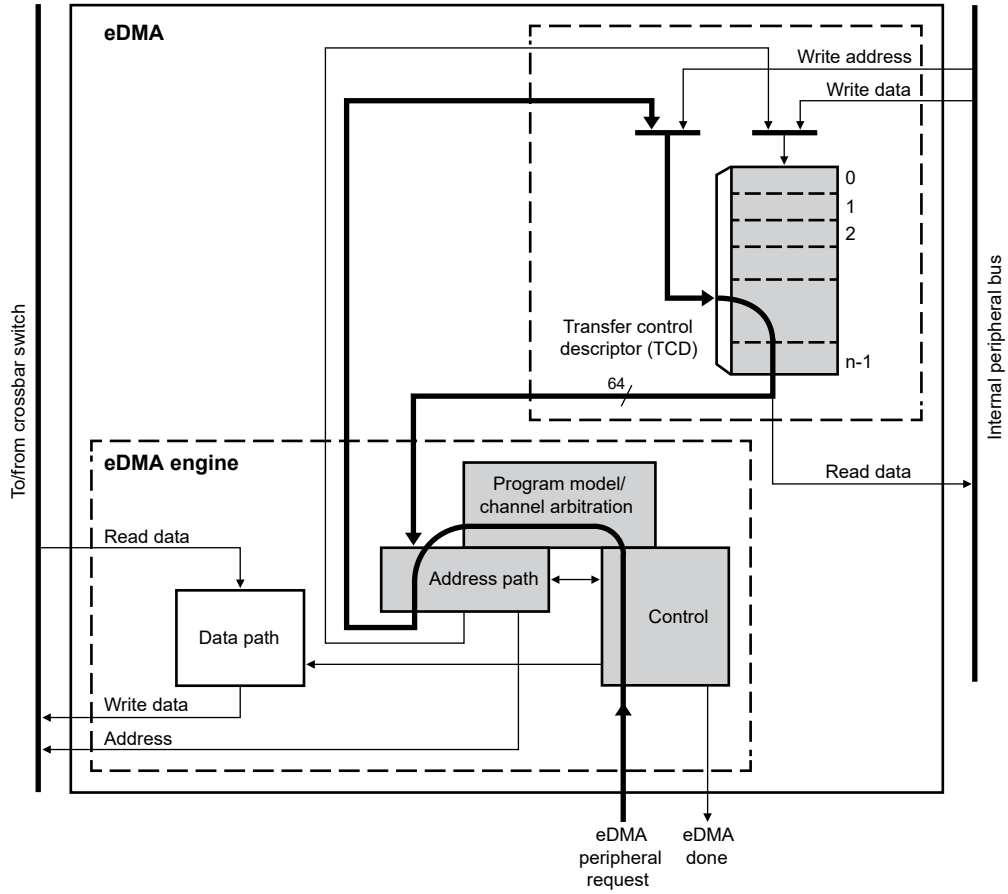


Figure 36. eDMA operation, part 1

This example uses the assertion of the eDMA peripheral request signal to request service for channel *n*. Channel activation via software and the TCD*n*_CSR[START] field follows the same basic flow as peripheral requests. The eDMA request input signal is registered internally and then routed through the eDMA engine: first through the control module, then into the program model and channel arbitration.

In the next cycle, the channel arbitration begins using fixed-priority plus the optional round-robin algorithm. After arbitration is complete, the activated channel number is sent through the address path and converted into the required address to access the local memory for TCD*n*. Next, the TCD memory is accessed and the required descriptor is read from the local memory and then loaded into the eDMA engine address path's primary or secondary channel execution registers. The TCD memory is 64 bits wide to minimize the time needed to fetch the activated channel descriptor and load it into the address path registers.

The following diagram illustrates the second part of the basic data flow:

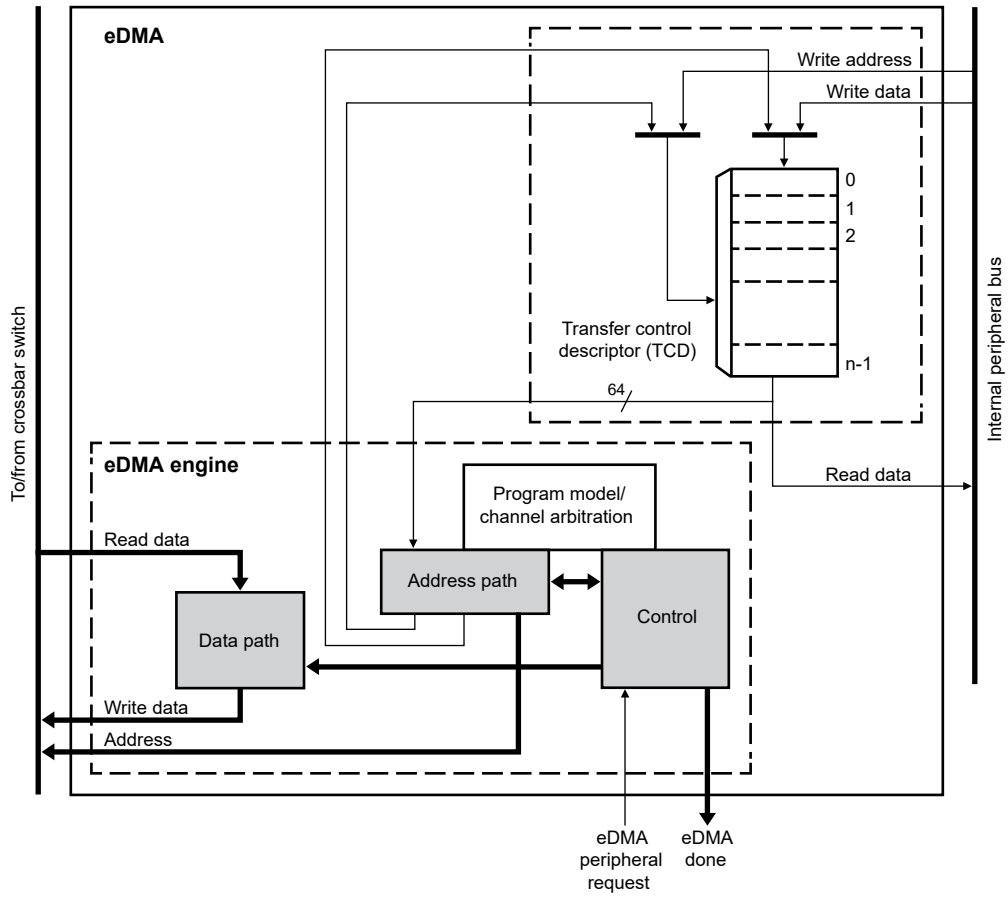


Figure 37. eDMA operation, part 2

The modules associated with the data transfer (address path, data path, and control) go through the required sequence of source reads and destination writes to perform the actual data movement. The source reads are initiated, and the fetched data is temporarily stored in the data path block until it is gated onto the internal bus during the destination write. This source read/destination write processing continues until the byte count, NBYTES, has been transferred.

After NBYTES of data has been moved, the final phase of the basic data flow is performed. In this segment, the address path logic performs the required updates to certain fields in the appropriate TCD (for example, SADDR, DADDR, CITER). If the major iteration count is exhausted, additional operations are performed. These include the final address adjustments and reloading of the BITER field into the CITER field. Assertion of an optional interrupt request also occurs at this time, as does a possible fetch of a new TCD from memory using the scatter/gather address pointer included in the descriptor (if scatter/gather is enabled). The updates to the TCD memory and the assertion of an interrupt request are shown in the following diagram.

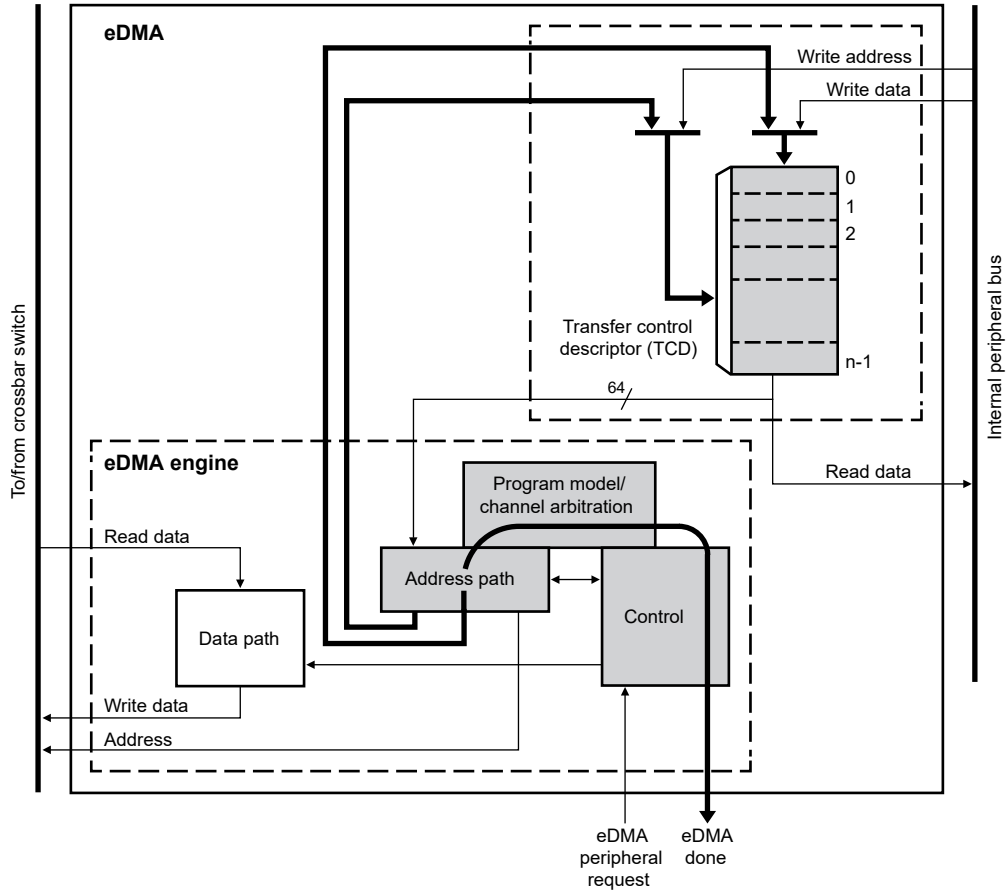


Figure 38. eDMA operation, part 3

15.3.3 Fault reporting and handling

Channel errors are reported in the Error Status register (CHn_ES) and can be caused by any of the following:

- A configuration error, which is an illegal setting in the transfer control descriptor
- An active channel canceled via a "cancel transfer with error" hardware or software request
- A TCD memory error
- An error termination to a bus initiator read or write cycle

A configuration error is reported when an inconsistent state is represented by one of these factors:

- Starting source or destination address
- Source or destination offsets
- Minor loop byte count
- Transfer size

Each of these possible causes is detailed below:

- The addresses and offsets must be aligned on zero-modulo-transfer-sized boundaries.
- The minor loop byte count must be a multiple of the source and destination transfer sizes.
- All source reads and destination writes must be configured to the natural boundary of the programmed transfer size.

NOTE

To aid in debugging, set the Halt After Error field in the DMA's Control Status register, CSR[HAE]. Upon any error condition, the DMA is halted after the error is recorded. The DMA remains halted and does not process any channel service requests. After the error is fixed, the DMA may be enabled again by clearing the Halt field, CSR[HALT].

- If a scatter/gather operation is enabled upon channel completion, a configuration error is reported if the scatter/gather address (TCDn_DLAST_SGA) is not aligned on a 32-byte boundary.
- If minor loop channel linking is enabled upon channel completion, a configuration error is reported when the link is attempted if the TCDn_CITER[ELINK] field does not equal the TCDn_BITER[ELINK] field.

If enabled, all configuration error conditions, except the scatter/gather and minor-loop link errors, are reported as the channel activates and asserts an error interrupt request. A scatter/gather configuration error is reported when the scatter/gather operation begins at major loop completion if properly enabled. A minor loop channel link configuration error is reported when the link operation is serviced at minor loop completion.

If a system bus read or write is terminated with an error, the data transfer is stopped and the appropriate bus error flag set. In this case, the state of the channel's transfer control descriptor is updated by the eDMA engine with the current source address, destination address, and current iteration count at the point of the fault. When a system bus error occurs, the channel terminates after the next transfer. Due to pipeline effect, the next transfer is already in progress when the bus error is received by the eDMA. If a bus error occurs on the last read prior to beginning the write sequence, the write executes using the data captured during the bus error. If a bus error occurs on the last write prior to switching to the next read sequence, the read sequence executes before the channel terminates due to the destination bus error.

The occurrence of any error causes the eDMA engine to stop normal processing of the active channel immediately (it goes to its error processing states and the transaction to the system bus still has pipeline effect), and the appropriate channel field in the eDMA error register is set to 1. At the same time, the details of the error condition are loaded into the Error Status register (CHn_ES). The major loop complete indicators, setting the transfer control descriptor DONE flag, and the possible assertion of an interrupt request are not affected when an error is detected.

After the error status has been updated, the eDMA engine continues operating by servicing the next appropriate channel. A channel that experiences an error condition is not automatically disabled. If a channel is terminated by an error and then issues another service request before the error is fixed, that channel executes and terminates with the same error condition.

The error status fields are read-only. These error indicators are sticky and cannot be cleared. They show the last recorded error until the DMA is reset. CHn_ES[ERR] is used to determine if a new error condition exists. This field is the logical OR of each channel's error interrupt field (ERR).

After the software has resolved all errors and cleared all of the error interrupt fields, the ES[VLD] is cleared to 0 but the cause of the last error is still indicated.

15.3.4 Channel preemption

The eDMA uses a priority vector value to determine the highest priority channel requesting service.

The priority vector is a combination of:

1. the channel's group priority, CHn_GRPRI
2. the channel's priority level, CHn_PRI[APL]
3. the channel number

Priority vector = ((CHn_GRPRI << 8) + (CHn_PRI[APL] << 5) + CHn_*)

A channel requesting service with the highest priority vector value will receive the next execution slot.

An execution slot is available:

1. immediately if the eDMA is idle
2. when an active channel retires
3. when valid preemption conditions exist

NOTE

Preemption is strictly priority based. Preemption is not bound by a specific group number as defined by CHn_GRPRI.

Channel preemption is enabled on a per-channel basis by setting the CHn_PRI[ECP] field. Channel preemption allows the executing channel's data transfers to temporarily suspend in favor of starting a higher-priority channel. After the preempting channel has completed all of its minor loop data transfers, the preempted channel is restored and resumes execution.

After the restored channel completes one read/write sequence, it is again eligible for preemption. If any higher priority channel is requesting service, the restored channel is suspended, and the higher-priority channel is serviced. Nested preemption, that is, attempting to preempt a preempting channel, is not supported. After a preempting channel begins execution, it cannot be preempted.

A channel's ability to preempt another channel can be disabled by setting CHn_PRI[DPA] to 1. When a channel's preempt ability is disabled, that channel cannot suspend a lower-priority channel's data transfer, regardless of the lower-priority channel's ECP setting. This allows for a pool of low-priority, large-data-moving channels to be defined.

You can configure these low-priority channels to not preempt each other, thus preventing a low-priority channel from consuming the preempt slot normally available to a true high-priority channel. When you enable round-robin channel arbitration mode (CSR[ERCA] is set to 1), any channel with a priority level equal to 0 (CHn_PRI[APL] = 0) has preemption disabled and cannot preempt another channel.

15.3.5 Clocking

This module has no clocking considerations.

15.3.6 Interrupts

Software can enable the interrupt for each channel for the following events:

1. The major loop is half complete (INTHALF)
2. The major loop is complete (INTMAJOR)
3. A configuration error occurs (EEI)

15.4 External signals

This module has no external signals.

15.5 Initialization

The following sections discuss initialization of the eDMA and programming considerations.

15.5.1 eDMA initialization

To initialize the eDMA:

1. Write to the CSR if a configuration other than the default is wanted.
2. Write the channel priority levels to the CHn_PRI registers and group priority levels to the CHn_GRPRI registers if a configuration other than the default is wanted.
3. Enable error interrupts in the CHn_CSR[EEI] registers if they are wanted.
4. Write the 32-byte TCD for each channel that may request service.
5. Enable any hardware service requests via the CHn_CSR[ERQ] registers.
6. Request channel service via either:
   - Software: setting TCDn_CSR[START]
   - Hardware: target device asserting its eDMA peripheral request signal

After any channel requests service, a channel is selected for execution based on the arbitration and priority levels written into the programmer's model. The eDMA engine reads the entire TCD, including the TCD control and status fields, as shown in Table 59, for the selected channel into its internal address path module.

As the TCD is read, the first transfer is initiated on the internal bus, unless a configuration error is detected. Transfers from the source, defined by TCDn_SADDR, to the destination, defined by TCD_DADDR, continue until the number of bytes specified by TCDn_NBYTES are transferred.

When the transfer is complete, the eDMA engine's local TCDn_SADDR, TCDn_DADDR, and TCDn_CITER are written back to the main TCD memory and any minor loop channel linking is performed, if enabled. If the major loop is exhausted, then eDMA executes further post-processing, such as interrupts, major loop channel linking, and scatter/gather operations, if enabled.

Table 59. TCD control and status (TCDn_CSR) fields

TCDn_CSR field name	Description
START	Control field to start the channel explicitly when using a software-initiated DMA service (automatically cleared by hardware)
ESDA	Control field to enable storing of the destination address to system memory after the major loop completes
DREQ	Control field to disable hardware-initiated DMA service requests after major loop completion
BWC	Control field for throttling the bandwidth control of a channel
ESG	Control field to enable the scatter-gather feature
INTHALF	Control field to enable interrupt when major loop is half-complete
INTMAJOR	Control field to enable interrupt when major loop completes

Table 60. Channel control and status (CHn_CSR) fields

CHn_CSR field name	Description
ACTIVE	Status field indicating the channel is currently in execution
DONE	Status field indicating major loop completion (cleared by software when a channel begins execution)
EEI	Control field to enable error interrupts
EARQ	Control field to enable external, asynchronous wakeup event in conjunction with the ERQ field
ERQ	Control field to enable hardware service requests

The following figure shows how each DMA request initiates one minor-loop transfer, or iteration, without CPU intervention. DMA arbitration can occur after each minor loop, and one level of minor loop DMA preemption is allowed. The number of minor loops in a major loop is specified by the beginning iteration count (BITER).

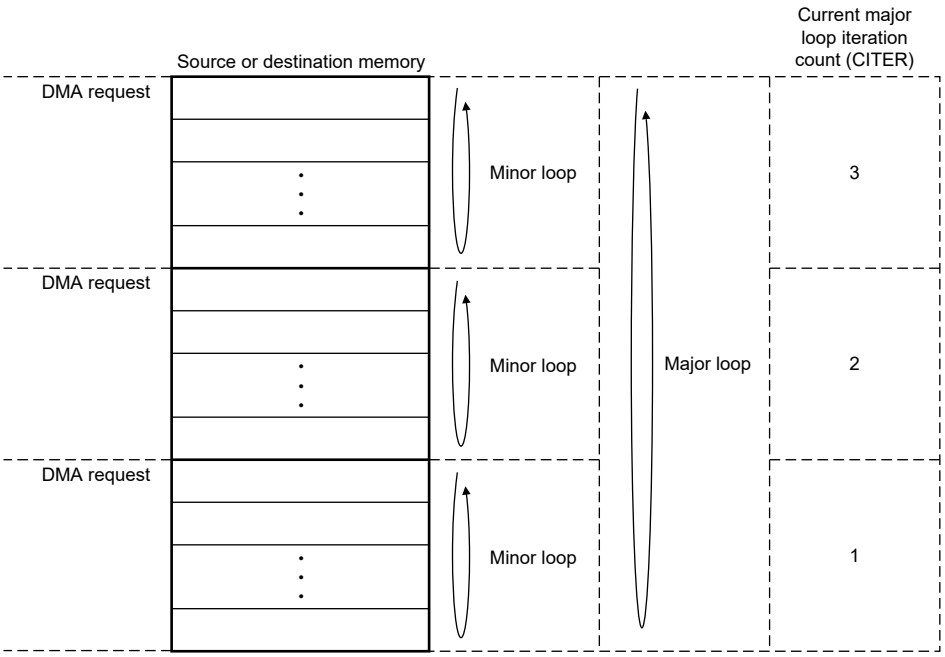


Figure 39. Example of multiple loop iterations

The following figure lists the memory array terms and how the TCD settings are related.

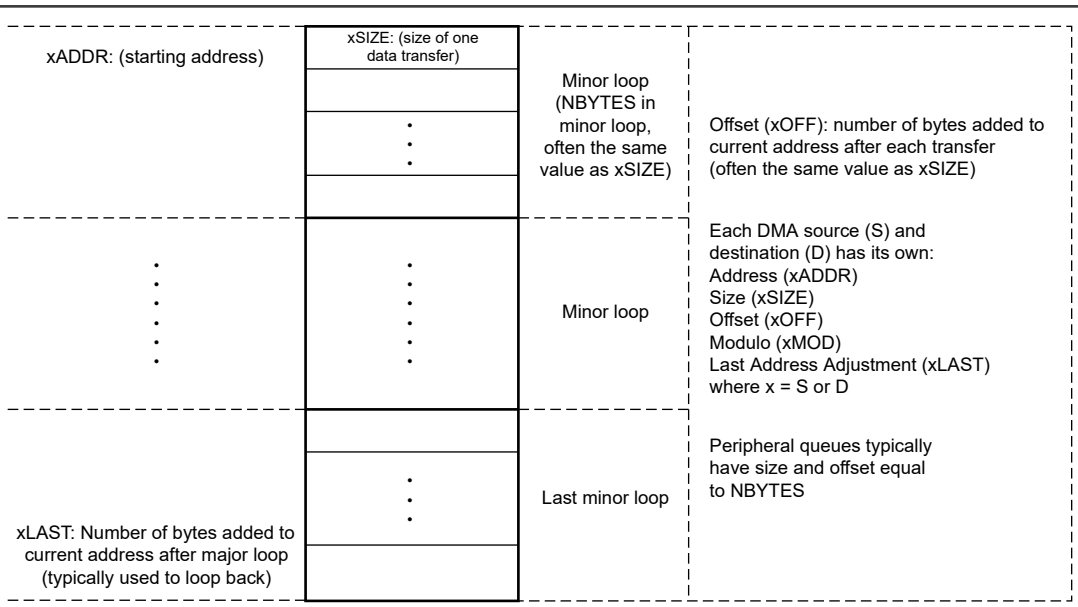


Figure 40. Memory array terms

15.5.2 eDMA arbitration

The eDMA uses a layered arbitration scheme composed of multiple priority levels. The eDMA uses a fixed-priority arbitration scheme with optional round-robin arbitration under specific conditions. The priorities are evaluated in the following order:

Table 61. eDMA arbitration priorities

Priority	Scheme	Description
1 (Highest)	Arbitration group priority	Each channel is assigned an arbitration group via the CHn_GRPRI registers. Priority is given to the highest value (31 being the highest possible value) down to the lowest value (zero, the default).
2	Channel priority	Each channel is assigned a channel priority level via the CHn_PRI registers. The channel priority is a relative priority level within an arbitration group. Priority is given to the highest value (seven being the highest possible value) down to the lowest value (zero, the default). Channel priorities within each arbitration group need not be unique. If multiple channels have the same channel priority level, the channel number will be used to determine priority as defined in row three.
3	Channel number	When two or more channels have the same arbitration group priority and channel priority, the channel number (CHn_NUM) is used to determine the highest priority. Priority is giver to the highest channel number. Lowest priority is channel 0. The channel numbers are static and cannot be changed in the programmer's model.
4 (Lowest)	Round-robin	When round-robin is enabled, any channel configured for round-robin operation has lowest priority within an arbitration group. Round-robin is enabled by setting the CSR[ERCA] field to 1. After being enabled, channels with a channel priority of zero (CHn_PRI=0) will use round-robin arbitration. Round-robin arbitration will rotate the channel selection among the channels requesting service with CHn_PRI=0 within the arbitration group. Any non-zero channel within the arbitration group will continue to use fixed-priority arbitration, and if requesting service will be selected over any round-robin channels.

For fixed arbitration, the overall priority can be considered a number composed of three concatenated priority levels: CHn_GRPRI:CHn_PRI:CH_NUM. The largest number has the highest priority and the lowest number has the lowest priority.

For round-robin arbitration, the priority number is CHn_GRPRI:0:X. The module rotates through the CHn_PRI=0 channels requesting service without regard to priority among these channels. Any channel within the arbitration group for which CHn_PRI is greater than 0 will be serviced before the round-robin channels.

15.5.3 Programming errors

The eDMA performs various tests on the transfer control descriptor to verify consistency in the descriptor data.

The channel number causing the error is recorded in the Error Status register (CHn_ES). If the error source is not removed before the next activation of the problematic channel, the error is detected and recorded again. Setting the halt after error field, CSR[HAE], will halt the DMA and prevent recurrence of the error.

15.5.4 Arbitration mode considerations

This section discusses arbitration considerations for eDMA.

15.5.4.1 Fixed group arbitration, fixed channel arbitration

In this mode, eDMA selects for execution the channel service request from the highest-priority channel in the highest-priority group. If eDMA is programmed so that the channels within a high-priority group have a high number of requests or large data transfers, that group may consume all the bandwidth of the eDMA controller. That is, no lower-priority groups are serviced if there is always at least one DMA request pending on a channel in the highest-priority group when the controller arbitrates the next DMA request. The advantage of this scenario is that latency can be small for channels that need to be serviced quickly.

15.5.4.2 Fixed group arbitration, round-robin channel arbitration

The highest-priority group with a request is serviced. Lower-priority groups are serviced if no pending requests exist in the higher-priority groups.

Within each group, channels are serviced starting with the highest non-zero channel priority. For all channels with a channel priority programmed to 0, selection begins with the highest channel number requesting service and then rotates through to the lowest channel number requesting service. The round-robin channel arbitration can provide a fairness mechanism to lower-priority channels.

This scenario could cause the same bandwidth consumption problem as indicated in Fixed group arbitration, fixed channel arbitration, but all the channels in the highest-priority group will be serviced. Service latency is short on the highest-priority group, but could potentially be very much longer as the group priority decreases.

15.5.5 Performing DMA transfers

This section presents examples on how to perform DMA transfers with the eDMA.

15.5.5.1 Single request

To perform a simple transfer of n bytes of data with one activation, set the major loop to one (TCD*n*_CITER = TCD*n*_BITER = 1). The data transfer begins after the channel service request is acknowledged and the channel is selected to execute. After the transfer is complete, the CH*n*_CSR[DONE] field is set to 1 and an interrupt is generated if properly enabled.

For example, the following TCD entry is configured to transfer 16 bytes of data. The eDMA is programmed for one iteration of the major loop transferring 16 bytes per iteration. The source memory has a byte-wide memory port located at 0x1000. The destination memory has a 32-bit port located at 0x2000. The address offsets are programmed in increments to match the transfer size: one byte for the source, and four bytes for the destination. The final source and destination addresses are adjusted to return to their beginning values.

```
TCDn_CITER = TCDn_BITER = 1
TCDn_NBYTES = 16
TCDn_SADDR = 0x1000
TCDn_SOFF = 1
TCDn_ATTR[SSIZE] = 0
TCDn_SLAST = -16
TCDn_DADDR = 0x2000
TCDn_DOFF = 4
TCDn_ATTR[DSIZE] = 2
```

```
TCDn_DLAST_SGA= -16
TCDn_CSR[INTMAJ] = 1
TCDn_CSR[START] = 1 (should be written last after all other fields have been initialized)
All other TCDn fields = 0
```

This generates the following event sequence:

1. User write to the TCD*n*_CSR[START] field requests channel service.
2. The channel is selected by arbitration for servicing.
3. eDMA engine writes:
   - CH*n*_CSR[DONE] = 0
   - TCD*n*_CSR[START] = 0
   - CH*n*_CSR[ACTIVE] = 1
4. eDMA engine reads: channel TCD data from local memory to internal register file.
5. The source-to-destination transfers are executed as follows:
   a. Read byte from location 0x1000, read byte from location 0x1001, read byte from 0x1002, read byte from 0x1003.
   b. Write 32 bits to location 0x2000 → first iteration of the minor loop.
   c. Read byte from location 0x1004, read byte from location 0x1005, read byte from 0x1006, read byte from 0x1007.
   d. Write 32 bits to location 0x2004 → second iteration of the minor loop.
   e. Read byte from location 0x1008, read byte from location 0x1009, read byte from 0x100A, read byte from 0x100B.
   f. Write 32 bits to location 0x2008 → third iteration of the minor loop.
   g. Read byte from location 0x100C, read byte from location 0x100D, read byte from 0x100E, read byte from 0x100F.
   h. Write 32 bits to location 0x200C → last iteration of the minor loop → major loop complete.
6. The eDMA engine writes: TCD*n*_SADDR = 0x1000, TCD*n*_DADDR = 0x2000, TCD*n*_CITER = 1 (TCD*n*_BITER).
7. The eDMA engine writes: CH*n*_CSR[ACTIVE] = 0, CH*n*_CSR[DONE] = 1, CH*n*_INT[INT] = 1.
8. The channel retires and the eDMA goes idle or services the next channel.

15.5.5.2 Multiple requests

The following example transfers 32 bytes via two hardware requests, but is otherwise the same as the previous example. The only fields that change are the major loop iteration count and the final address offsets. The eDMA is programmed for two iterations of the major loop, transferring 16 bytes per iteration. After the channel's hardware requests are enabled via the CHn_CSR[ERQ] register field, the target device initiates channel service requests.

```
TCDn_CITER = TCDn_BITER = 2
TCDn_SLAST = -32
TCDn_DLAST_SGA = -32
```

This would generate the following sequence of events:

1. First hardware (eDMA peripheral) requests channel service.
2. The channel is selected by arbitration for servicing.
3. eDMA engine writes: CH*n*_CSR[DONE] = 0, TCD*n*_CSR[START] = 0, CH*n*_CSR[ACTIVE] = 1.
4. eDMA engine reads: channel TCD*n* data from local memory to internal register file.
5. The source-to-destination transfers are executed as follows:

a. Read byte from location 0x1000, read byte from location 0x1001, read byte from 0x1002, read byte from 0x1003.
b. Write 32 bits to location 0x2000 → first iteration of the minor loop.
c. Read byte from location 0x1004, read byte from location 0x1005, read byte from 0x1006, read byte from 0x1007.
d. Write 32 bits to location 0x2004 → second iteration of the minor loop.
e. Read byte from location 0x1008, read byte from location 0x1009, read byte from 0x100A, read byte from 0x100B.
f. Write 32 bits to location 0x2008 → third iteration of the minor loop.
g. Read byte from location 0x100C, read byte from location 0x100D, read byte from 0x100E, read byte from 0x100F.
h. Write 32 bits to location 0x200C → last iteration of the minor loop.

6. eDMA engine writes: TCD*n*_SADDR = 0x1010, TCD*n*_DADDR = 0x2010, TCD*n*_CITER = 1.
7. eDMA engine writes: CH*n*_CSR[ACTIVE] = 0.
8. The channel retires, which concludes one iteration of the major loop. The eDMA goes idle or services the next channel.
9. Second hardware (eDMA peripheral) requests channel service.
10. The channel is selected by arbitration for servicing.
11. eDMA engine writes: CH*n*_CSR[DONE] = 0, TCD*n*_CSR[START] = 0, CH*n*_CSR[ACTIVE] = 1.
12. eDMA engine reads: Channel TCD data from local memory to internal register file.
13. The source-to-destination transfers are executed as follows:
    a. Read byte from location 0x1010, read byte from location 0x1011, read byte from 0x1012, read byte from 0x1013.
    b. Write 32 bits to location 0x2010 → first iteration of the minor loop.
    c. Read byte from location 0x1014, read byte from location 0x1015, read byte from 0x1016, read byte from 0x1017.
    d. Write 32 bits to location 0x2014 → second iteration of the minor loop.
    e. Read byte from location 0x1018, read byte from location 0x1019, read byte from 0x101A, read byte from 0x101B.
    f. Write 32 bits to location 0x2018 → third iteration of the minor loop.
    g. Read byte from location 0x101C, read byte from location 0x101D, read byte from 0x101E, read byte from 0x101F.
    h. Write 32 bits to location 0x201C → last iteration of the minor loop → major loop complete.
14. eDMA engine writes: TCD*n*_SADDR = 0x1000, TCD*n*_DADDR = 0x2000, TCD*n*_CITER = 2 (TCD*n*_BITER).
15. eDMA engine writes: CH*n*_CSR[ACTIVE] = 0, CH*n*_CSR[DONE] = 1, CH*n*_INT[INT] = 1.
16. The channel retires, which concludes with the major loop complete. The eDMA goes idle or services the next channel.

15.5.5.3 Using the modulo feature

The modulo feature of the eDMA allows implementation of a circular data queue in which the size of the queue is a power of 2. xMOD is a 5-bit field for the source and destination in the TCD, and it specifies which lower address bits increment from their original value after the address+offset calculation. All upper address bits remain the same as in the original value. A setting of 0 for this field disables the modulo feature. Modulo addressing applies to cases where the minor loop offset is enabled; that is, the upper address bits remain the same after the minor loop offset is added to the source or destination address.

The following table shows how the transfer addresses are specified based on the setting of the MOD field. Here a circular buffer is created where the address wraps to the original value but the 28 upper address bits (0x1234567*x*) retain their original value. In this example, the source address is set to 0x12345670, the offset is set to four bytes, and the MOD field is set to four, which allows for a 2^4 byte (16 byte) queue size.

Table 62. Modulo example

Transfer number	Address
1	0x12345670
2	0x12345674
3	0x12345678
4	0x1234567C
5	0x12345670
6	0x12345674

15.5.6 Monitoring transfer descriptor status

This section discusses how to monitor eDMA status.

15.5.6.1 Testing for minor loop completion

There are two methods to test for minor loop completion when using software-initiated service requests.

1. The first method is to read the TCDn_CITER field and test for a change.
2. The second method, extracted from the sequence shown below, is to test the TCDn_CSR[START] field and the CHn_CSR[ACTIVE] field. The minor-loop-complete condition is indicated by both fields reading 0 after TCDn_CSR[START] is set to 1. Polling the CHn_CSR[ACTIVE] field only may be inconclusive because the active status may be missed if the channel execution is short in duration.

The CHn_CSR and TCDn_CSR status fields execute the following sequence for a software-activated channel:

Stage	TCDn_CSR field	CHn_CSR fields		State
	START	ACTIVE	DONE	
1	1	0	0	Initiate channel service request via software.
2	0	1	0	Channel is executing.
3a	0	0	0	Channel has completed the minor loop and is idle.
3b	0	0	1	Channel has completed the major loop and is idle.

The best method to test for minor-loop completion when using hardware-initiated (that is, peripheral-initiated) service requests is to read the TCDn_CITER field and test for a change. The hardware request and acknowledge handshake signals are not visible in the programmer's model.

The TCD status fields execute the following sequence for a hardware-activated channel:

Stage	TCDn_CSR field	CHn_CSR fields		State
	START	ACTIVE	DONE	
1	0	0	0	Initiate channel service request via hardware (peripheral request asserted).
2	0	1	0	Channel is executing.
3a	0	0	0	Channel has completed the minor loop and is idle.
3b	0	0	1	Channel has completed the major loop and is idle.

For both activation types, the major-loop-complete status is explicitly indicated via the CHn_CSR[DONE] field.

The TCDn_CSR[START] field is cleared to 0 automatically when the channel begins execution, regardless of how the channel activates.

15.5.6.2 Reading the transfer descriptors of active channels

The eDMA reads back the true TCDn_SADDR, TCDn_DADDR, and TCDn_NBYTES values if they are read when a channel executes. The true values of SADDR, DADDR, and NBYTES are the values the eDMA engine currently uses in its internal register file, and not the values in the TCD local memory for that channel. The addresses, SADDR and DADDR, and NBYTES (which decrements to zero as the transfer progresses), can give an indication of the progress of the transfer. All other values are read back from the TCD local memory.

15.5.6.3 Checking channel preemption status

A preemptive situation is one in which a preempt-enabled channel is executing and a higher-priority request becomes active. When round-robin channel arbitration mode is enabled, all channels with their channel priority set to 0 lose their preempt ability. Channel priorities of 0 are treated as equal, that is, they are constantly rotating, when round-robin arbitration mode is enabled.

The CHn_CSR[ACTIVE] field for the preempted channel remains asserted throughout the preemption. The preempted channel is temporarily suspended when the preempting channel executes one major loop iteration. If two CHn_CSR[ACTIVE] fields are set simultaneously in the global TCD map, a higher-priority channel is actively preempting a lower-priority channel.

15.5.7 Channel linking

Channel linking (or chaining) is a mechanism in which one channel sets the TCDn_CSR[START] field of another channel (or itself), thus initiating a service request for that channel. When properly enabled, the eDMA engine automatically performs this operation at the major or minor loop completion.

The minor loop channel linking occurs at the completion of the minor loop (or one iteration of the major loop). The TCDn_CITER[ELINK] field determines whether a minor loop link is requested. When enabled, the channel link is made after each iteration of the major loop except for the last. When the major loop is exhausted, only the major loop channel link fields are used to determine if a channel link should be made. For example, using an initial field setting of:

```
TCDn_CITER[ELINK] = 1
TCDn_CITER[LINKCH] = 0xC
TCDn_CITER[CITER] value = 0x4
TCDn_CSR[MAJORELINK] = 1
TCDn_CSR[MAJORLINKCH] = 0x7
```

executes as:

1. Minor loop done → set TCD12_CSR[START] field

2. Minor loop done → set TCD12_CSR[START] field
3. Minor loop done → set TCD12_CSR[START] field
4. Minor loop done, major loop done→ set TCD7_CSR[START] field

When minor loop linking is enabled (TCDn_CITER[ELINK] = 1), the TCDn_CITER[CITER] field uses a nine-bit vector to form the current iteration count. When minor loop linking is disabled (TCDn_CITER[ELINK] = 0), the TCDn_CITER[CITER] field uses a 15-bit vector to form the current iteration count. The bits associated with the TCDn_CITER[LINKCH] field are concatenated onto the CITER value to increase the range of the CITER.

NOTE

The TCDn_CITER[ELINK] field and the TCDn_BITER[ELINK] field must be equal — if they are not, a configuration error is reported. The CITER and BITER vector widths must be equal to calculate the major loop halfway done interrupt point.

The following table summarizes how a DMA channel can link to another DMA channel, that is, use another channel's TCD, at the end of a loop.

Table 63. Channel linking parameters

Wanted link behavior	TCD control field name	Description
Link at end of minor loop	TCDn_CITER[ELINK]	Enable channel-to-channel linking on minor loop completion (current iteration)
	TCDn_CITER[LINKCH]	Link channel number when linking at end of minor loop (current iteration)
Link at end of major loop	TCDn_CSR[MAJORELINK]	Enable channel-to-channel linking on major loop completion
	TCDn_CSR[MAJORLINKCH]	Link channel number when linking at end of major loop

15.5.8 Dynamic programming

This section provides recommended methods to change the programming model during channel execution.

15.5.8.1 Dynamically changing the channel priority

To change group or channel priority levels:

1. Halt the DMA by writing 1 to the CSR[HALT] field.
2. Change the group or channel priorities as wanted.
3. Enable normal DMA operations by writing 0 to the CSR[HALT] field.

15.5.8.2 Dynamic channel linking

Dynamic channel linking is the process of setting the TCDn_CSR[MAJORELINK] field during channel execution (see the diagram in TCD structure). This field is read from the TCD local memory at the end of channel execution, thus allowing you to enable the feature during channel execution.

Because you are allowed to change the configuration during execution, you need a coherency model. Consider the scenario where you attempt to execute a dynamic channel link by enabling the TCDn_CSR[MAJORELINK] field at the same time the eDMA engine is retiring the channel. TCDn_CSR[MAJORELINK] would be set in the programmer's model, but it would be unclear whether the actual link was made before the channel retired.

We recommend that you use the following coherency model when executing a dynamic channel link request.

1. Write 1 to the TCDn_CSR[MAJORELINK] field.

2. Read back the TCDn_CSR[MAJORELINK] field.
3. Test the TCDn_CSR[MAJORELINK] request status:
   - If TCDn_CSR[MAJORELINK] = 1, the dynamic link attempt was successful.
   - If TCDn_CSR[MAJORELINK] = 0, the attempted dynamic link did not succeed (the channel was already retiring).

For this request, the TCD local memory controller forces the TCDn_CSR[MAJORELINK] field to 0 on any writes to a channel's TCDn_CSR[7:0] after that channel's CHn_CSR[DONE] field is set to 1, indicating the major loop is complete.

NOTE

You must clear the CHn_CSR[DONE] field to 0 before writing to the TCDn_CSR[MAJORELINK] field. The CHn_CSR[DONE] field is cleared to 0 automatically by the eDMA engine after a channel begins execution.

15.5.8.3 Dynamic scatter/gather

Scatter/gather is the process of automatically loading a new TCD into a channel. It allows a DMA channel to use multiple TCDs; this enables a DMA channel to scatter the DMA data to multiple destinations or gather it from multiple sources. When scatter/gather is enabled and the channel has finished its major loop, a new TCD is fetched from system memory and loaded into that channel's descriptor location in the eDMA programmer's model, thus replacing the current descriptor.

Because you are allowed to change the configuration during execution, you need a coherency model. Consider the scenario where you attempt to execute a dynamic scatter/gather operation by enabling the TCDn_CSR[ESG] field at the same time the eDMA engine is retiring the channel. The TCDn_CSR[ESG] field would be set in the programmer's model, but it would be unclear whether the actual scatter/gather request was honored before the channel retired.

Two methods are recommended for executing a dynamic scatter/gather request. Whenever the TCDn_CSR is written, the TCD local memory controller forces the TCDn_CSR[ESG] field to 0 on any writes to a channel's TCDn_CSR[7:0] after that channel's CHn_CSR[DONE] field has been set to 1, indicating the major loop is complete. If attempting to set the ESG, ensure the DONE field is cleared to 0.

NOTE

You must clear the CHn_CSR[DONE] field to 0 before writing the TCDn_CSR[MAJORELINK] or TCDn_CSR[ESG] fields. The CHn_CSR[DONE] field is cleared to 0 automatically by the eDMA engine after a channel begins execution and is set to 1 upon major loop completion.

15.5.8.3.1 Method 1 (channel not using major loop channel linking)

For a channel not using major loop channel linking, the coherency model described here may be used for a dynamic scatter/gather request.

When the TCDn_CSR[MAJORELINK] field is 0, the TCDn_CSR[MAJORLINKCH] field is not used by the eDMA. In this case, the TCDn_CSR[MAJORLINKCH] bits may be used for other purposes. This method uses the TCDn_CSR[MAJORLINKCH] field as a TCDn_CSR identification (ID).

When the descriptors are built, write a unique TCDn_CSR ID in the TCDn_CSR[MAJORLINKCH] field for each TCDn_CSR associated with a channel using dynamic scatter/gather.

1. Write a 1 to the TCDn_CSR[DREQ] field. Should a dynamic scatter/gather attempt fail, setting the TCDn_CSR[DREQ] field to 1 will prevent future hardware activation of this channel. This stops the channel from executing with a destination address (daddr) that was calculated using a scatter/gather address (written in the next step) instead of a DLAST final offset value.
2. Write the TCDn_DLAST_SGA field with the scatter/gather address.
3. Write a 1 to the TCDn_CSR[ESG] field.
4. Read back the 16-bit TCDn_CSR control/status field.
5. Test the TCDn_CSR[ESG] request status and TCDn_CSR[MAJORLINKCH] value:
   - If ESG = 1, the dynamic scatter/gather attempt was successful.

- If ESG = 0 and the MAJORLINKCH (ID) did not change, the dynamic scatter/gather attempt was not successful (the channel was already retiring).
- If ESG = 0 and the MAJORLINKCH (ID) changed, the dynamic scatter/gather attempt was successful (the new TCDn_CSR's ESG value cleared the ESG field to 0).

15.5.8.3.2 Method 2 (channel using major loop channel linking)

For a channel using major loop channel linking, the coherency model described here may be used for a dynamic scatter/gather request. This method uses the TCDn_DLAST_SGA field as a TCD identification (ID).

1. Write a 1 to the TCDn_CSR[DREQ] field. Should a dynamic scatter/gather attempt fail, setting the DREQ field to 1 will prevent a future hardware activation of this channel. This stops the channel from executing with a destination address (DADDR) that was calculated using a scatter/gather address (written in the next step) instead of a DLAST final offset value.
2. Write the TCDn_DLAST_SGA field with the scatter/gather address.
3. Write a 1 to the TCDn_CSR[ESG] field.
4. Read back the TCDn_CSR[ESG] field.
5. Test the TCDn_CSR[ESG] request status:
   - If ESG = 1, the dynamic scatter/gather attempt was successful.
   - If ESG = 0, read the 32-bit TCDn_DLAST_SGA field.
   - If ESG = 0 and the TCDn_DLAST_SGA did not change, the dynamic scatter/gather attempt was not successful (the channel was already retiring).
   - If ESG = 0 and the TCDn_DLAST_SGA changed, the dynamic scatter/gather attempt was successful (the new TCDn_CSR's ESG value cleared the ESG field to 0).

15.5.9 Suspend/resume a DMA channel with active hardware service requests

The DMA allows you to move data from memory or peripheral registers to another location in memory or to peripheral registers without CPU interaction. After the DMA and peripherals are configured and active, it is rare but supported to suspend a peripheral's service request dynamically. In this scenario, there are certain restrictions to disabling a DMA hardware service request. For coherency, you must follow a specific procedure. This section provides guidance on how to coherently suspend and resume a Direct Memory Access (DMA) channel when the DMA is triggered by a target module such as the Serial Peripheral Interface (SPI), Sigma Delta Analog to Digital Convertor (SDADC), or other module.

15.5.9.1 Suspend an active DMA channel

To suspend an active DMA channel:

1. Stop the DMA service request at the peripheral first. Confirm it has been disabled by reading back the appropriate register in the peripheral.
2. Check the DMA's Hardware Request Status (HRS) to ensure there is no service request to the DMA channel being suspended. Then disable the hardware service request by clearing the ERQ field to 0 on the appropriate DMA channel.

For example, assume the SPI is set as a initiator for transmitting data via a DMA service request when the TXFIFO has an empty slot. The DMA will transfer the next command and data to the TXFIFO upon the request. If you need to suspend the DMA/SPI transfer loop, perform the following steps:

1. Disable the DMA service request at the source by writing 0 to DSPI_RSER[TFFF_RE]. Confirm that DSPI_RSER[TFFF_RE] is 0.
2. Ensure there is no DMA service request from the SPI by verifying that HRS[HRS] is 0 for the appropriate channel. If no service request is present, disable the DMA channel by clearing the channel's ERQ field to 0. If a service request is present, wait until the request has been processed and the HRS field reads 0.

15.5.9.2 Resume a DMA channel

To resume a DMA channel:

1. Enable the DMA service request on the appropriate channel by setting its ERQ field to 1.
2. Enable the DMA service request at the peripheral.

15.6 Memory map/register definition

The eDMA programming model is partitioned into three parts:

1. The first part defines a number of registers providing overall control functions and is known as the management page.
2. The second part corresponds to the channel (CH) control, status, and configuration.
3. The third part corresponds to the local TCD memory.

TCD memory

Each channel requires a 32-byte transfer control descriptor for defining the data movement operation. Each TCDn definition is presented as 11 registers of 16 or 32 bits.

TCD initialization

The TCD memory is in an unknown state after reset. Only the TCD START bit is initialized to 0. Prior to activating a channel, you must initialize its TCD with the appropriate transfer profile.

TCD structure

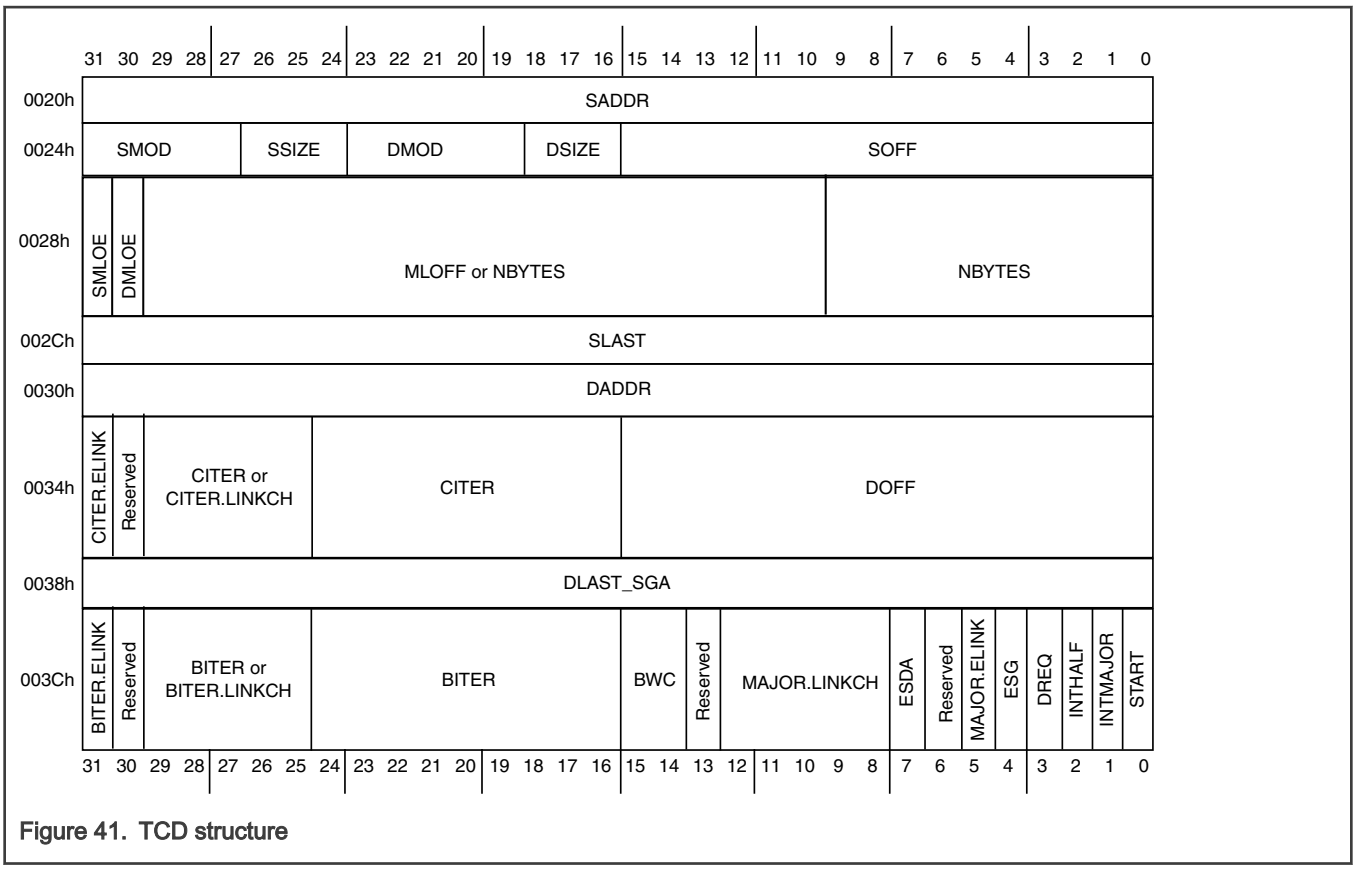


Figure 41. TCD structure

Accesses to reserved memory and fields

- Reading reserved fields in a register returns the value of zero.
- Writes to reserved fields in a register are ignored.
- Reading or writing a reserved memory location generates a bus error.

15.6.1 eDMA register descriptions

15.6.1.1 eDMA memory map

eDMA base address: 4020_C000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Management Page Control (CSR)	32	RW	0030_0000h
4h	Management Page Error Status (ES)	32	R	0000_0000h
8h	Management Page Interrupt Request Status (INT)	32	R	0000_0000h
Ch	Management Page Hardware Request Status (HRS)	32	R	0000_0000h
100h - 17Ch	Channel Arbitration Group (CH0_GRPRI - CH31_GRPRI)	32	RW	0000_0000h

15.6.1.2 Management Page Control (CSR)

Offset

Register	Offset
CSR	0h

Function

The Management Page Control register defines the basic operating configuration of the DMA.

Arbitration uses a two-tier priority system; group and channel priority. The eDMA assigns each channel to a priority group. Group arbitration is fixed-priority and cannot be changed. Channel arbitration uses fixed priority and may be configured to use a selective round-robin scheme for specified channels within each priority group. For fixed-priority arbitration, eDMA selects for execution the highest priority channel requesting service in the highest priority arbitration group.

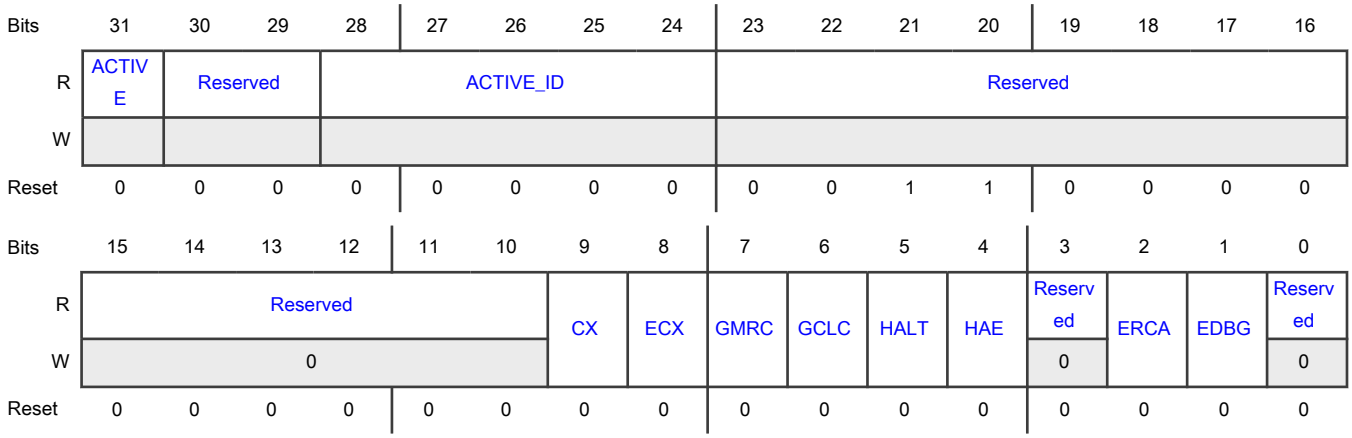
The channel priority registers assign the relative priorities within each arbitration group; see CHn_PRI. All channels with a non-zero CHn_PRI value use fixed-priority arbitration.

When you enable round-robin arbitration, all channels with channel priority set to zero do not have a priority and, of those channels requesting service, are cycled through (from high to low channel number) without regard to priority relative to each other within the same priority group. Any channel with a non-zero CHn_PRI value automatically has a higher priority over the round-robin channels. A channel's priority group is assigned in Channel Arbitration Group (CH0_GRPRI - CH31_GRPRI).

NOTE

For correct operation, changes to the CSR[ERCA, GCLC, GMRC] fields must be performed when the DMA channels are inactive; that is, when the CSR[ACTIVE] field is 0.

Diagram



Fields

Field	Function
31 ACTIVE	DMA Active Status 0b - eDMA is idle 1b - eDMA is executing a channel
30-29 —	Reserved
28-24 ACTIVE_ID	Active Channel ID This field identifies the channel number that is executing when the ACTIVE bit is 1.
23-16 —	Reserved
15-10 —	Reserved
9 CX	Cancel Transfer When set to 1, this field cancels the remaining data transfer, stops the executing channel, and forces the minor loop to finish. The cancel takes effect after the last write of the current read/write sequence. CX clears itself to 0 after the cancel has been honored. This cancel retires the channel normally as if the minor loop had been completed. 0b - Normal operation 1b - Cancel the remaining data transfer
8 ECX	Cancel Transfer With Error Cancellation of the remaining data transfer is similar to that of the CX field. Execution of the channel is stopped and the minor loop is forced to finish. The cancellation takes effect after the last write of the current read/write sequence. The ECX field clears itself to 0 after the cancel is honored. In addition to

Table continues on the next page...

Table continued from the previous page...

Field	Function
	cancelling the transfer, ECX treats the cancel as an error condition, thus updating Management Page Error Status (ES) and generating an optional error interrupt. 0b - Normal operation 1b - Cancel the remaining data transfer
7 GMRC	Global Initiator ID Replication Control **NOTE** If initiator ID replication is disabled, the privileged protection level (Supervisor mode) for DMA transfers is used. See chip-specific eDMA information for details on ID replication. 0b - Initiator ID replication disabled for all channels 1b - Initiator ID replication available and controlled by each channel's CHn_SBR[EMI] setting
6 GCLC	Global Channel Linking Control 0b - Channel linking disabled for all channels 1b - Channel linking available and controlled by each channel's link settings
5 HALT	Halt DMA Operations This field stalls the start of any new channels. Executing channels are allowed to complete. Channel execution resumes when this field is cleared to 0. 0b - Normal operation 1b - Stall the start of any new channels
4 HAE	Halt After Error When this field is set to 1, any error causes the HALT field to be set to 1. Then all service requests are ignored until the HALT field is cleared to 0. 0b - Normal operation 1b - Any error causes the HALT field to be set to 1
3 —	Reserved
2 ERCA	Enable Round Robin Channel Arbitration 0b - Round-robin channel arbitration disabled. Fixed priority arbitration used for channel selection within each group 1b - Round-robin channel arbitration enabled. Round-robin arbitration used for channel selection within each group
1 EDBG	Enable Debug When in debug mode, the DMA stalls the start of a new channel. Executing channels are allowed to complete. DMA resumes channel execution when the system exits debug mode or clears the EDBG field to 0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Debug mode disabled. When in debug mode, the DMA continues to operate 1b - Debug mode is enabled. When in debug mode, the DMA stalls the start of a new channel
0 —	Reserved

15.6.1.3 Management Page Error Status (ES)

Offset

Register	Offset
ES	4h

Function

The ES provides information concerning the last recorded channel error. Channel errors can be caused by:

- An illegal setting in the transfer control descriptor
- An error termination to a bus initiator read or write cycle
- An uncorrectable error that occurred when the device was accessing the TCD SRAM
- A "cancel transfer with error" request was made via the corresponding cancel transfer field or input signal

Upon any error condition, the software must initialize the TCD of the channel that contains the error, as it is in an incomplete state after an error. See Fault reporting and handling for more details.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	VLD	Reserved			ERRCHN				0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0						UCE	ECX	SAE	SOE	DAE	DOE	NCE	SGE	SBE	DBE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31	Valid

Table continues on the next page...

Table continued from the previous page...

Field	Function
VLD	Logical OR of all ERR status fields. 0b - No ERR fields are set to 1 1b - At least one ERR field is set to 1, indicating a valid error exists that software has not cleared
30-29 —	Reserved
28-24 ERRCHN	Error Channel Number or Canceled Channel Number The channel number of the last recorded error or last recorded error-canceled transfer.
23-10 —	Reserved
9 UCE	Uncorrectable TCD Error During Channel Execution UCE is set to 1 only when an uncorrectable ECC error occurs on an access generated by the DMA. If a CPU access to the TCD causes an uncorrectable ECC error, then that access receives a bus error response. **NOTE** When the eDMA sees a RAM error on an IPS access (when you are accessing a TCD), it reports the error as a bus abort. When the DMA engine receives a RAM error (the execution engine is accessing a TCD) it is recorded in the Error Status register, ES[UCE], along with the channel number. 0b - No uncorrectable ECC error 1b - Last recorded error was an uncorrectable TCD RAM error
8 ECX	Transfer Canceled The ECX operation is a management page function. When employed, the targeted channel's CHn_ES register reports an unspecified error; that is, only the ERR field is set to 1. The management page has full view of the error condition. 0b - No canceled transfers 1b - Last recorded entry was a canceled transfer by the error cancel transfer input
7 SAE	Source Address Error When this field is 1, it indicates that TCDn_SADDR is inconsistent with TCDn_ATTR[SSIZE]. 0b - No source address configuration error 1b - Last recorded error was a configuration error detected in the TCDn_SADDR field
6 SOE	Source Offset Error When this field is 1, it indicates that TCDn_SOFF is inconsistent with TCDn_ATTR[SSIZE]. 0b - No source offset configuration error 1b - Last recorded error was a configuration error detected in the TCDn_SOFF field

Table continues on the next page...

Table continued from the previous page...

Field	Function
5 DAE	Destination Address Error When this field is 1, it indicates that TCDn_DADDR is inconsistent with TCDn_ATTR[DSIZE]. 0b - No destination address configuration error 1b - Last recorded error was a configuration error detected in the TCDn_DADDR field
4 DOE	Destination Offset Error When this field is 1, it indicates that TCDn_DOFF is inconsistent with TCDn_ATTR[DSIZE]. 0b - No destination offset configuration error 1b - Last recorded error was a configuration error detected in the TCDn_DOFF field
3 NCE	NBYTES/CITER Configuration Error This error indicates that one of the following has occurred: • TCDn_NBYTES is not a multiple of TCDn_ATTR[SSIZE] and TCDn_ATTR[DSIZE] • TCDn_CITER[CITER] is equal to zero • TCDn_CITER[ELINK] is not equal to TCDn_BITER[ELINK] 0b - No NBYTES/CITER configuration error 1b - The last recorded error was NBYTES equal to zero or a CITER not equal to BITER error. Last recorded error was a configuration error detected in the TCDn_NBYTES or TCDn_CITER fields
2 SGE	Scatter/Gather Configuration Error When this field is 1, it indicates that TCDn_DLAST_SGA is not on a 32-byte boundary. This field is checked at the beginning of a scatter/gather operation after major loop completion if TCDn_CSR[ESG] is enabled. 0b - No scatter/gather configuration error 1b - Last recorded error was a configuration error detected in the TCDn_DLAST_SGA field
1 SBE	Source Bus Error 0b - No source bus error 1b - Last recorded error was a bus error on a source read
0 DBE	Destination Bus Error 0b - No destination bus error 1b - Last recorded error was a bus error on a destination write

15.6.1.4 Management Page Interrupt Request Status (INT)

Offset

Register	Offset
INT	8h

Function

This register shows the current state of the interrupt service requests for all eDMA channels.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	INT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	INT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 INT	Interrupt Request Status The INT register presents the interrupt request status for each eDMA channel. Depending on the appropriate field setting in the transfer control descriptors, the eDMA engine generates an interrupt on data transfer completion or an error condition. The eDMA routes channel interrupt requests to the interrupt controller. During the interrupt service routine associated with any given channel, it is the software's responsibility to clear the appropriate field in the channel's interrupt request register, CHn_INT, thus negating the interrupt request. 0b - Interrupt request for corresponding channel not present 1b - Interrupt request for corresponding channel present

15.6.1.5 Management Page Hardware Request Status (HRS)

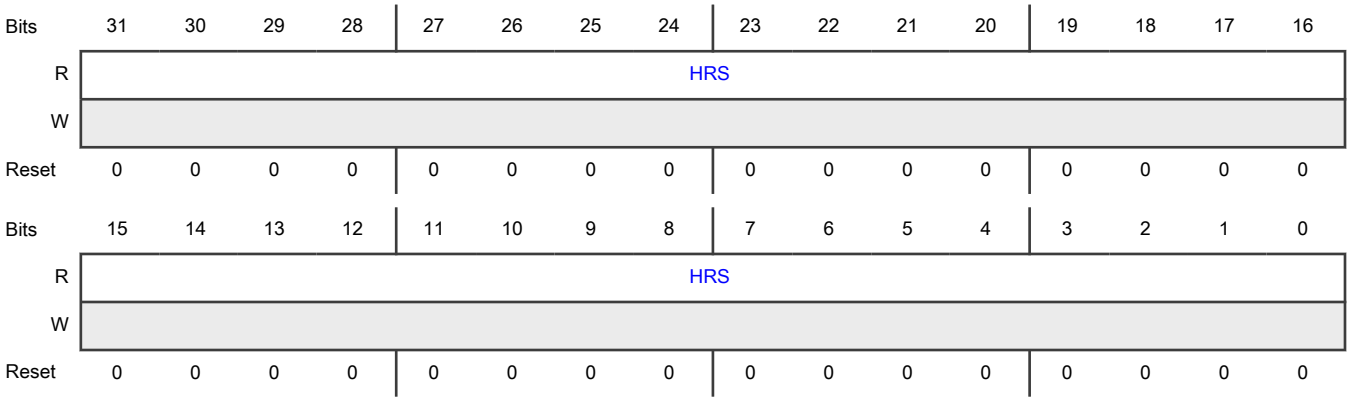
Offset

Register	Offset
HRS	Ch

Function

The hardware request status register (HRS) shows the current state of the hardware service request signaling as seen by eDMA's arbitration logic.

Diagram



Fields

Field	Function
31-0 HRS	Hardware Request Status The HRS bit for its respective channel remains asserted for the period when a hardware request is present on the channel. 0b - Hardware service request for corresponding channel is not present 1b - Hardware service request for corresponding channel is present

15.6.1.6 Channel Arbitration Group (CH0_GRPRI - CH31_GRPRI)

Offset

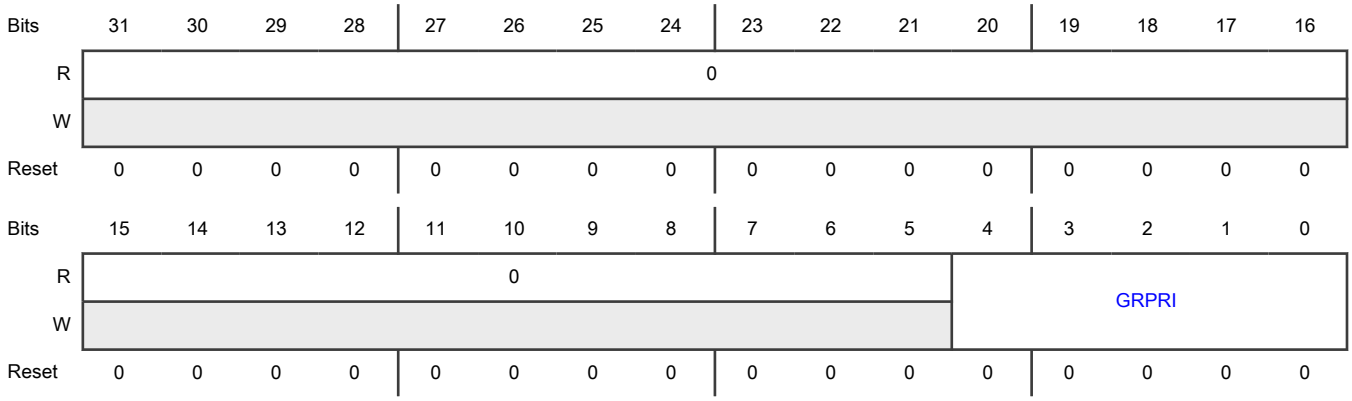
For n = 0 to 31:

Register	Offset
CHn_GRPRI	100h + (n × 4h)

Function

The contents of this register define the arbitration group associated with each channel. Using a fixed-priority group arbitration scheme, eDMA evaluates the arbitration group priorities by numeric value from highest group number to lowest; for example, 0 is the lowest priority, 1 is the next higher priority, then 2, 3, and so on. The range of the group priority values is limited to the values of 0 through 31. Within each arbitration group, the channel priority assignment CH*n*_PRI determines the highest-priority channel.

Diagram



Fields

Field	Function
31-5 —	Reserved
4-0 GRPRI	Arbitration Group For Channel n Fixed-priority arbitration group number.

15.6.2 TCD register descriptions

15.6.2.1 TCD memory map

TCD base address: 4021_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Channel Control and Status (CH0_CSR)	32	RW	0000_0000h
4h	Channel Error Status (CH0_ES)	32	RW	0000_0000h
8h	Channel Interrupt Status (CH0_INT)	32	RW	0000_0000h
Ch	Channel System Bus (CH0_SBR)	32	RW	0000_8002h
10h	Channel Priority (CH0_PRI)	32	RW	0000_0000h
20h	TCD Source Address (TCD0_SADDR)	32	RW	0000_0000h
24h	TCD Signed Source Address Offset (TCD0_SOFF)	16	RW	0000h
26h	TCD Transfer Attributes (TCD0_ATTR)	16	RW	0000h
28h	TCD Transfer Size Without Minor Loop Offsets (TCD0_NBYTES_MLOFFNO)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
28h	TCD Transfer Size with Minor Loop Offsets (TCD0_NBYTES_MLOFFYES)	32	RW	0000_0000h
2Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD0_SLAST_SDA)	32	RW	0000_0000h
30h	TCD Destination Address (TCD0_DADDR)	32	RW	0000_0000h
34h	TCD Signed Destination Address Offset (TCD0_DOFF)	16	RW	0000h
36h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD0_CITER_ELINKNO)	16	RW	0000h
36h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD0_CITER_ELINKYES)	16	RW	0000h
38h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD0_DLAST_SGA)	32	RW	0000_0000h
3Ch	TCD Control and Status (TCD0_CSR)	16	RW	0000h
3Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD0_BITER_ELINKNO)	16	RW	0000h
3Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD0_BITER_ELINKYES)	16	RW	0000h
4000h	Channel Control and Status (CH1_CSR)	32	RW	0000_0000h
4004h	Channel Error Status (CH1_ES)	32	RW	0000_0000h
4008h	Channel Interrupt Status (CH1_INT)	32	RW	0000_0000h
400Ch	Channel System Bus (CH1_SBR)	32	RW	0000_8002h
4010h	Channel Priority (CH1_PRI)	32	RW	0000_0000h
4020h	TCD Source Address (TCD1_SADDR)	32	RW	0000_0000h
4024h	TCD Signed Source Address Offset (TCD1_SOFF)	16	RW	0000h
4026h	TCD Transfer Attributes (TCD1_ATTR)	16	RW	0000h
4028h	TCD Transfer Size Without Minor Loop Offsets (TCD1_NBYTES_MLOFFNO)	32	RW	0000_0000h
4028h	TCD Transfer Size with Minor Loop Offsets (TCD1_NBYTES_MLOFFYES)	32	RW	0000_0000h
402Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD1_SLAST_SDA)	32	RW	0000_0000h
4030h	TCD Destination Address (TCD1_DADDR)	32	RW	0000_0000h
4034h	TCD Signed Destination Address Offset (TCD1_DOFF)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
4036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD1_CITER_ELINKNO)	16	RW	0000h
4036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD1_CITER_ELINKYES)	16	RW	0000h
4038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD1_DLAST_SGA)	32	RW	0000_0000h
403Ch	TCD Control and Status (TCD1_CSR)	16	RW	0000h
403Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD1_BITER_ELINKNO)	16	RW	0000h
403Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD1_BITER_ELINKYES)	16	RW	0000h
8000h	Channel Control and Status (CH2_CSR)	32	RW	0000_0000h
8004h	Channel Error Status (CH2_ES)	32	RW	0000_0000h
8008h	Channel Interrupt Status (CH2_INT)	32	RW	0000_0000h
800Ch	Channel System Bus (CH2_SBR)	32	RW	0000_8002h
8010h	Channel Priority (CH2_PRI)	32	RW	0000_0000h
8020h	TCD Source Address (TCD2_SADDR)	32	RW	0000_0000h
8024h	TCD Signed Source Address Offset (TCD2_SOFF)	16	RW	0000h
8026h	TCD Transfer Attributes (TCD2_ATTR)	16	RW	0000h
8028h	TCD Transfer Size Without Minor Loop Offsets (TCD2_NBYTES_MLOFFNO)	32	RW	0000_0000h
8028h	TCD Transfer Size with Minor Loop Offsets (TCD2_NBYTES_MLOFFYES)	32	RW	0000_0000h
802Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD2_SLAST_SDA)	32	RW	0000_0000h
8030h	TCD Destination Address (TCD2_DADDR)	32	RW	0000_0000h
8034h	TCD Signed Destination Address Offset (TCD2_DOFF)	16	RW	0000h
8036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD2_CITER_ELINKNO)	16	RW	0000h
8036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD2_CITER_ELINKYES)	16	RW	0000h
8038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD2_DLAST_SGA)	32	RW	0000_0000h
803Ch	TCD Control and Status (TCD2_CSR)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
803Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD2_BITER_ELINKNO)	16	RW	0000h
803Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD2_BITER_ELINKYES)	16	RW	0000h
C000h	Channel Control and Status (CH3_CSR)	32	RW	0000_0000h
C004h	Channel Error Status (CH3_ES)	32	RW	0000_0000h
C008h	Channel Interrupt Status (CH3_INT)	32	RW	0000_0000h
C00Ch	Channel System Bus (CH3_SBR)	32	RW	0000_8002h
C010h	Channel Priority (CH3_PRI)	32	RW	0000_0000h
C020h	TCD Source Address (TCD3_SADDR)	32	RW	0000_0000h
C024h	TCD Signed Source Address Offset (TCD3_SOFF)	16	RW	0000h
C026h	TCD Transfer Attributes (TCD3_ATTR)	16	RW	0000h
C028h	TCD Transfer Size Without Minor Loop Offsets (TCD3_NBYTES_MLOFFNO)	32	RW	0000_0000h
C028h	TCD Transfer Size with Minor Loop Offsets (TCD3_NBYTES_MLOFFYES)	32	RW	0000_0000h
C02Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD3_SLAST_SDA)	32	RW	0000_0000h
C030h	TCD Destination Address (TCD3_DADDR)	32	RW	0000_0000h
C034h	TCD Signed Destination Address Offset (TCD3_DOFF)	16	RW	0000h
C036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD3_CITER_ELINKNO)	16	RW	0000h
C036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD3_CITER_ELINKYES)	16	RW	0000h
C038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD3_DLAST_SGA)	32	RW	0000_0000h
C03Ch	TCD Control and Status (TCD3_CSR)	16	RW	0000h
C03Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD3_BITER_ELINKNO)	16	RW	0000h
C03Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD3_BITER_ELINKYES)	16	RW	0000h
1_0000h	Channel Control and Status (CH4_CSR)	32	RW	0000_0000h
1_0004h	Channel Error Status (CH4_ES)	32	RW	0000_0000h
1_0008h	Channel Interrupt Status (CH4_INT)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_000Ch	Channel System Bus (CH4_SBR)	32	RW	0000_8002h
1_0010h	Channel Priority (CH4_PRI)	32	RW	0000_0000h
1_0020h	TCD Source Address (TCD4_SADDR)	32	RW	0000_0000h
1_0024h	TCD Signed Source Address Offset (TCD4_SOFF)	16	RW	0000h
1_0026h	TCD Transfer Attributes (TCD4_ATTR)	16	RW	0000h
1_0028h	TCD Transfer Size Without Minor Loop Offsets (TCD4_NBYTES_MLOFFNO)	32	RW	0000_0000h
1_0028h	TCD Transfer Size with Minor Loop Offsets (TCD4_NBYTES_MLOFFYES)	32	RW	0000_0000h
1_002Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD4_SLAST_SDA)	32	RW	0000_0000h
1_0030h	TCD Destination Address (TCD4_DADDR)	32	RW	0000_0000h
1_0034h	TCD Signed Destination Address Offset (TCD4_DOFF)	16	RW	0000h
1_0036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD4_CITER_ELINKNO)	16	RW	0000h
1_0036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD4_CITER_ELINKYES)	16	RW	0000h
1_0038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD4_DLAST_SGA)	32	RW	0000_0000h
1_003Ch	TCD Control and Status (TCD4_CSR)	16	RW	0000h
1_003Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD4_BITER_ELINKNO)	16	RW	0000h
1_003Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD4_BITER_ELINKYES)	16	RW	0000h
1_4000h	Channel Control and Status (CH5_CSR)	32	RW	0000_0000h
1_4004h	Channel Error Status (CH5_ES)	32	RW	0000_0000h
1_4008h	Channel Interrupt Status (CH5_INT)	32	RW	0000_0000h
1_400Ch	Channel System Bus (CH5_SBR)	32	RW	0000_8002h
1_4010h	Channel Priority (CH5_PRI)	32	RW	0000_0000h
1_4020h	TCD Source Address (TCD5_SADDR)	32	RW	0000_0000h
1_4024h	TCD Signed Source Address Offset (TCD5_SOFF)	16	RW	0000h
1_4026h	TCD Transfer Attributes (TCD5_ATTR)	16	RW	0000h
1_4028h	TCD Transfer Size Without Minor Loop Offsets (TCD5_NBYTES_MLOFFNO)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_4028h	TCD Transfer Size with Minor Loop Offsets (TCD5_NBYTES_MLOFFYES)	32	RW	0000_0000h
1_402Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD5_SLAST_SDA)	32	RW	0000_0000h
1_4030h	TCD Destination Address (TCD5_DADDR)	32	RW	0000_0000h
1_4034h	TCD Signed Destination Address Offset (TCD5_DOFF)	16	RW	0000h
1_4036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD5_CITER_ELINKNO)	16	RW	0000h
1_4036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD5_CITER_ELINKYES)	16	RW	0000h
1_4038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD5_DLAST_SGA)	32	RW	0000_0000h
1_403Ch	TCD Control and Status (TCD5_CSR)	16	RW	0000h
1_403Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD5_BITER_ELINKNO)	16	RW	0000h
1_403Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD5_BITER_ELINKYES)	16	RW	0000h
1_8000h	Channel Control and Status (CH6_CSR)	32	RW	0000_0000h
1_8004h	Channel Error Status (CH6_ES)	32	RW	0000_0000h
1_8008h	Channel Interrupt Status (CH6_INT)	32	RW	0000_0000h
1_800Ch	Channel System Bus (CH6_SBR)	32	RW	0000_8002h
1_8010h	Channel Priority (CH6_PRI)	32	RW	0000_0000h
1_8020h	TCD Source Address (TCD6_SADDR)	32	RW	0000_0000h
1_8024h	TCD Signed Source Address Offset (TCD6_SOFF)	16	RW	0000h
1_8026h	TCD Transfer Attributes (TCD6_ATTR)	16	RW	0000h
1_8028h	TCD Transfer Size Without Minor Loop Offsets (TCD6_NBYTES_MLOFFNO)	32	RW	0000_0000h
1_8028h	TCD Transfer Size with Minor Loop Offsets (TCD6_NBYTES_MLOFFYES)	32	RW	0000_0000h
1_802Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD6_SLAST_SDA)	32	RW	0000_0000h
1_8030h	TCD Destination Address (TCD6_DADDR)	32	RW	0000_0000h
1_8034h	TCD Signed Destination Address Offset (TCD6_DOFF)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_8036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD6_CITER_ELINKNO)	16	RW	0000h
1_8036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD6_CITER_ELINKYES)	16	RW	0000h
1_8038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD6_DLAST_SGA)	32	RW	0000_0000h
1_803Ch	TCD Control and Status (TCD6_CSR)	16	RW	0000h
1_803Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD6_BITER_ELINKNO)	16	RW	0000h
1_803Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD6_BITER_ELINKYES)	16	RW	0000h
1_C000h	Channel Control and Status (CH7_CSR)	32	RW	0000_0000h
1_C004h	Channel Error Status (CH7_ES)	32	RW	0000_0000h
1_C008h	Channel Interrupt Status (CH7_INT)	32	RW	0000_0000h
1_C00Ch	Channel System Bus (CH7_SBR)	32	RW	0000_8002h
1_C010h	Channel Priority (CH7_PRI)	32	RW	0000_0000h
1_C020h	TCD Source Address (TCD7_SADDR)	32	RW	0000_0000h
1_C024h	TCD Signed Source Address Offset (TCD7_SOFF)	16	RW	0000h
1_C026h	TCD Transfer Attributes (TCD7_ATTR)	16	RW	0000h
1_C028h	TCD Transfer Size Without Minor Loop Offsets (TCD7_NBYTES_MLOFFNO)	32	RW	0000_0000h
1_C028h	TCD Transfer Size with Minor Loop Offsets (TCD7_NBYTES_MLOFFYES)	32	RW	0000_0000h
1_C02Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD7_SLAST_SDA)	32	RW	0000_0000h
1_C030h	TCD Destination Address (TCD7_DADDR)	32	RW	0000_0000h
1_C034h	TCD Signed Destination Address Offset (TCD7_DOFF)	16	RW	0000h
1_C036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD7_CITER_ELINKNO)	16	RW	0000h
1_C036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD7_CITER_ELINKYES)	16	RW	0000h
1_C038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD7_DLAST_SGA)	32	RW	0000_0000h
1_C03Ch	TCD Control and Status (TCD7_CSR)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_C03Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD7_BITER_ELINKNO)	16	RW	0000h
1_C03Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD7_BITER_ELINKYES)	16	RW	0000h
2_0000h	Channel Control and Status (CH8_CSR)	32	RW	0000_0000h
2_0004h	Channel Error Status (CH8_ES)	32	RW	0000_0000h
2_0008h	Channel Interrupt Status (CH8_INT)	32	RW	0000_0000h
2_000Ch	Channel System Bus (CH8_SBR)	32	RW	0000_8002h
2_0010h	Channel Priority (CH8_PRI)	32	RW	0000_0000h
2_0020h	TCD Source Address (TCD8_SADDR)	32	RW	0000_0000h
2_0024h	TCD Signed Source Address Offset (TCD8_SOFF)	16	RW	0000h
2_0026h	TCD Transfer Attributes (TCD8_ATTR)	16	RW	0000h
2_0028h	TCD Transfer Size Without Minor Loop Offsets (TCD8_NBYTES_MLOFFNO)	32	RW	0000_0000h
2_0028h	TCD Transfer Size with Minor Loop Offsets (TCD8_NBYTES_MLOFFYES)	32	RW	0000_0000h
2_002Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD8_SLAST_SDA)	32	RW	0000_0000h
2_0030h	TCD Destination Address (TCD8_DADDR)	32	RW	0000_0000h
2_0034h	TCD Signed Destination Address Offset (TCD8_DOFF)	16	RW	0000h
2_0036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD8_CITER_ELINKNO)	16	RW	0000h
2_0036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD8_CITER_ELINKYES)	16	RW	0000h
2_0038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD8_DLAST_SGA)	32	RW	0000_0000h
2_003Ch	TCD Control and Status (TCD8_CSR)	16	RW	0000h
2_003Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD8_BITER_ELINKNO)	16	RW	0000h
2_003Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD8_BITER_ELINKYES)	16	RW	0000h
2_4000h	Channel Control and Status (CH9_CSR)	32	RW	0000_0000h
2_4004h	Channel Error Status (CH9_ES)	32	RW	0000_0000h
2_4008h	Channel Interrupt Status (CH9_INT)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
2_400Ch	Channel System Bus (CH9_SBR)	32	RW	0000_8002h
2_4010h	Channel Priority (CH9_PRI)	32	RW	0000_0000h
2_4020h	TCD Source Address (TCD9_SADDR)	32	RW	0000_0000h
2_4024h	TCD Signed Source Address Offset (TCD9_SOFF)	16	RW	0000h
2_4026h	TCD Transfer Attributes (TCD9_ATTR)	16	RW	0000h
2_4028h	TCD Transfer Size Without Minor Loop Offsets (TCD9_NBYTES_MLOFFNO)	32	RW	0000_0000h
2_4028h	TCD Transfer Size with Minor Loop Offsets (TCD9_NBYTES_MLOFFYES)	32	RW	0000_0000h
2_402Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD9_SLAST_SDA)	32	RW	0000_0000h
2_4030h	TCD Destination Address (TCD9_DADDR)	32	RW	0000_0000h
2_4034h	TCD Signed Destination Address Offset (TCD9_DOFF)	16	RW	0000h
2_4036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD9_CITER_ELINKNO)	16	RW	0000h
2_4036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD9_CITER_ELINKYES)	16	RW	0000h
2_4038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD9_DLAST_SGA)	32	RW	0000_0000h
2_403Ch	TCD Control and Status (TCD9_CSR)	16	RW	0000h
2_403Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD9_BITER_ELINKNO)	16	RW	0000h
2_403Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD9_BITER_ELINKYES)	16	RW	0000h
2_8000h	Channel Control and Status (CH10_CSR)	32	RW	0000_0000h
2_8004h	Channel Error Status (CH10_ES)	32	RW	0000_0000h
2_8008h	Channel Interrupt Status (CH10_INT)	32	RW	0000_0000h
2_800Ch	Channel System Bus (CH10_SBR)	32	RW	0000_8002h
2_8010h	Channel Priority (CH10_PRI)	32	RW	0000_0000h
2_8020h	TCD Source Address (TCD10_SADDR)	32	RW	0000_0000h
2_8024h	TCD Signed Source Address Offset (TCD10_SOFF)	16	RW	0000h
2_8026h	TCD Transfer Attributes (TCD10_ATTR)	16	RW	0000h
2_8028h	TCD Transfer Size Without Minor Loop Offsets (TCD10_NBYTES_MLOFFNO)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
2_8028h	TCD Transfer Size with Minor Loop Offsets (TCD10_NBYTES_MLOFFYES)	32	RW	0000_0000h
2_802Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD10_SLAST_SDA)	32	RW	0000_0000h
2_8030h	TCD Destination Address (TCD10_DADDR)	32	RW	0000_0000h
2_8034h	TCD Signed Destination Address Offset (TCD10_DOFF)	16	RW	0000h
2_8036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD10_CITER_ELINKNO)	16	RW	0000h
2_8036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD10_CITER_ELINKYES)	16	RW	0000h
2_8038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD10_DLAST_SGA)	32	RW	0000_0000h
2_803Ch	TCD Control and Status (TCD10_CSR)	16	RW	0000h
2_803Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD10_BITER_ELINKNO)	16	RW	0000h
2_803Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD10_BITER_ELINKYES)	16	RW	0000h
2_C000h	Channel Control and Status (CH11_CSR)	32	RW	0000_0000h
2_C004h	Channel Error Status (CH11_ES)	32	RW	0000_0000h
2_C008h	Channel Interrupt Status (CH11_INT)	32	RW	0000_0000h
2_C00Ch	Channel System Bus (CH11_SBR)	32	RW	0000_8002h
2_C010h	Channel Priority (CH11_PRI)	32	RW	0000_0000h
2_C020h	TCD Source Address (TCD11_SADDR)	32	RW	0000_0000h
2_C024h	TCD Signed Source Address Offset (TCD11_SOFF)	16	RW	0000h
2_C026h	TCD Transfer Attributes (TCD11_ATTR)	16	RW	0000h
2_C028h	TCD Transfer Size Without Minor Loop Offsets (TCD11_NBYTES_MLOFFNO)	32	RW	0000_0000h
2_C028h	TCD Transfer Size with Minor Loop Offsets (TCD11_NBYTES_MLOFFYES)	32	RW	0000_0000h
2_C02Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD11_SLAST_SDA)	32	RW	0000_0000h
2_C030h	TCD Destination Address (TCD11_DADDR)	32	RW	0000_0000h
2_C034h	TCD Signed Destination Address Offset (TCD11_DOFF)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
2_C036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD11_CITER_ELINKNO)	16	RW	0000h
2_C036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD11_CITER_ELINKYES)	16	RW	0000h
2_C038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD11_DLAST_SGA)	32	RW	0000_0000h
2_C03Ch	TCD Control and Status (TCD11_CSR)	16	RW	0000h
2_C03Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD11_BITER_ELINKNO)	16	RW	0000h
2_C03Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD11_BITER_ELINKYES)	16	RW	0000h
20_0000h	Channel Control and Status (CH12_CSR)	32	RW	0000_0000h
20_0004h	Channel Error Status (CH12_ES)	32	RW	0000_0000h
20_0008h	Channel Interrupt Status (CH12_INT)	32	RW	0000_0000h
20_000Ch	Channel System Bus (CH12_SBR)	32	RW	0000_8002h
20_0010h	Channel Priority (CH12_PRI)	32	RW	0000_0000h
20_0020h	TCD Source Address (TCD12_SADDR)	32	RW	0000_0000h
20_0024h	TCD Signed Source Address Offset (TCD12_SOFF)	16	RW	0000h
20_0026h	TCD Transfer Attributes (TCD12_ATTR)	16	RW	0000h
20_0028h	TCD Transfer Size Without Minor Loop Offsets (TCD12_NBYTES_MLOFFNO)	32	RW	0000_0000h
20_0028h	TCD Transfer Size with Minor Loop Offsets (TCD12_NBYTES_MLOFFYES)	32	RW	0000_0000h
20_002Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD12_SLAST_SDA)	32	RW	0000_0000h
20_0030h	TCD Destination Address (TCD12_DADDR)	32	RW	0000_0000h
20_0034h	TCD Signed Destination Address Offset (TCD12_DOFF)	16	RW	0000h
20_0036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD12_CITER_ELINKNO)	16	RW	0000h
20_0036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD12_CITER_ELINKYES)	16	RW	0000h
20_0038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD12_DLAST_SGA)	32	RW	0000_0000h
20_003Ch	TCD Control and Status (TCD12_CSR)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
20_003Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD12_BITER_ELINKNO)	16	RW	0000h
20_003Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD12_BITER_ELINKYES)	16	RW	0000h
20_4000h	Channel Control and Status (CH13_CSR)	32	RW	0000_0000h
20_4004h	Channel Error Status (CH13_ES)	32	RW	0000_0000h
20_4008h	Channel Interrupt Status (CH13_INT)	32	RW	0000_0000h
20_400Ch	Channel System Bus (CH13_SBR)	32	RW	0000_8002h
20_4010h	Channel Priority (CH13_PRI)	32	RW	0000_0000h
20_4020h	TCD Source Address (TCD13_SADDR)	32	RW	0000_0000h
20_4024h	TCD Signed Source Address Offset (TCD13_SOFF)	16	RW	0000h
20_4026h	TCD Transfer Attributes (TCD13_ATTR)	16	RW	0000h
20_4028h	TCD Transfer Size Without Minor Loop Offsets (TCD13_NBYTES_MLOFFNO)	32	RW	0000_0000h
20_4028h	TCD Transfer Size with Minor Loop Offsets (TCD13_NBYTES_MLOFFYES)	32	RW	0000_0000h
20_402Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD13_SLAST_SDA)	32	RW	0000_0000h
20_4030h	TCD Destination Address (TCD13_DADDR)	32	RW	0000_0000h
20_4034h	TCD Signed Destination Address Offset (TCD13_DOFF)	16	RW	0000h
20_4036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD13_CITER_ELINKNO)	16	RW	0000h
20_4036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD13_CITER_ELINKYES)	16	RW	0000h
20_4038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD13_DLAST_SGA)	32	RW	0000_0000h
20_403Ch	TCD Control and Status (TCD13_CSR)	16	RW	0000h
20_403Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD13_BITER_ELINKNO)	16	RW	0000h
20_403Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD13_BITER_ELINKYES)	16	RW	0000h
20_8000h	Channel Control and Status (CH14_CSR)	32	RW	0000_0000h
20_8004h	Channel Error Status (CH14_ES)	32	RW	0000_0000h
20_8008h	Channel Interrupt Status (CH14_INT)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
20_800Ch	Channel System Bus (CH14_SBR)	32	RW	0000_8002h
20_8010h	Channel Priority (CH14_PRI)	32	RW	0000_0000h
20_8020h	TCD Source Address (TCD14_SADDR)	32	RW	0000_0000h
20_8024h	TCD Signed Source Address Offset (TCD14_SOFF)	16	RW	0000h
20_8026h	TCD Transfer Attributes (TCD14_ATTR)	16	RW	0000h
20_8028h	TCD Transfer Size Without Minor Loop Offsets (TCD14_NBYTES_MLOFFNO)	32	RW	0000_0000h
20_8028h	TCD Transfer Size with Minor Loop Offsets (TCD14_NBYTES_MLOFFYES)	32	RW	0000_0000h
20_802Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD14_SLAST_SDA)	32	RW	0000_0000h
20_8030h	TCD Destination Address (TCD14_DADDR)	32	RW	0000_0000h
20_8034h	TCD Signed Destination Address Offset (TCD14_DOFF)	16	RW	0000h
20_8036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD14_CITER_ELINKNO)	16	RW	0000h
20_8036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD14_CITER_ELINKYES)	16	RW	0000h
20_8038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD14_DLAST_SGA)	32	RW	0000_0000h
20_803Ch	TCD Control and Status (TCD14_CSR)	16	RW	0000h
20_803Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD14_BITER_ELINKNO)	16	RW	0000h
20_803Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD14_BITER_ELINKYES)	16	RW	0000h
20_C000h	Channel Control and Status (CH15_CSR)	32	RW	0000_0000h
20_C004h	Channel Error Status (CH15_ES)	32	RW	0000_0000h
20_C008h	Channel Interrupt Status (CH15_INT)	32	RW	0000_0000h
20_C00Ch	Channel System Bus (CH15_SBR)	32	RW	0000_8002h
20_C010h	Channel Priority (CH15_PRI)	32	RW	0000_0000h
20_C020h	TCD Source Address (TCD15_SADDR)	32	RW	0000_0000h
20_C024h	TCD Signed Source Address Offset (TCD15_SOFF)	16	RW	0000h
20_C026h	TCD Transfer Attributes (TCD15_ATTR)	16	RW	0000h
20_C028h	TCD Transfer Size Without Minor Loop Offsets (TCD15_NBYTES_MLOFFNO)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
20_C028h	TCD Transfer Size with Minor Loop Offsets (TCD15_NBYTES_MLOFFYES)	32	RW	0000_0000h
20_C02Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD15_SLAST_SDA)	32	RW	0000_0000h
20_C030h	TCD Destination Address (TCD15_DADDR)	32	RW	0000_0000h
20_C034h	TCD Signed Destination Address Offset (TCD15_DOFF)	16	RW	0000h
20_C036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD15_CITER_ELINKNO)	16	RW	0000h
20_C036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD15_CITER_ELINKYES)	16	RW	0000h
20_C038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD15_DLAST_SGA)	32	RW	0000_0000h
20_C03Ch	TCD Control and Status (TCD15_CSR)	16	RW	0000h
20_C03Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD15_BITER_ELINKNO)	16	RW	0000h
20_C03Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD15_BITER_ELINKYES)	16	RW	0000h
21_0000h	Channel Control and Status (CH16_CSR)	32	RW	0000_0000h
21_0004h	Channel Error Status (CH16_ES)	32	RW	0000_0000h
21_0008h	Channel Interrupt Status (CH16_INT)	32	RW	0000_0000h
21_000Ch	Channel System Bus (CH16_SBR)	32	RW	0000_8002h
21_0010h	Channel Priority (CH16_PRI)	32	RW	0000_0000h
21_0020h	TCD Source Address (TCD16_SADDR)	32	RW	0000_0000h
21_0024h	TCD Signed Source Address Offset (TCD16_SOFF)	16	RW	0000h
21_0026h	TCD Transfer Attributes (TCD16_ATTR)	16	RW	0000h
21_0028h	TCD Transfer Size Without Minor Loop Offsets (TCD16_NBYTES_MLOFFNO)	32	RW	0000_0000h
21_0028h	TCD Transfer Size with Minor Loop Offsets (TCD16_NBYTES_MLOFFYES)	32	RW	0000_0000h
21_002Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD16_SLAST_SDA)	32	RW	0000_0000h
21_0030h	TCD Destination Address (TCD16_DADDR)	32	RW	0000_0000h
21_0034h	TCD Signed Destination Address Offset (TCD16_DOFF)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
21_0036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD16_CITER_ELINKNO)	16	RW	0000h
21_0036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD16_CITER_ELINKYES)	16	RW	0000h
21_0038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD16_DLAST_SGA)	32	RW	0000_0000h
21_003Ch	TCD Control and Status (TCD16_CSR)	16	RW	0000h
21_003Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD16_BITER_ELINKNO)	16	RW	0000h
21_003Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD16_BITER_ELINKYES)	16	RW	0000h
21_4000h	Channel Control and Status (CH17_CSR)	32	RW	0000_0000h
21_4004h	Channel Error Status (CH17_ES)	32	RW	0000_0000h
21_4008h	Channel Interrupt Status (CH17_INT)	32	RW	0000_0000h
21_400Ch	Channel System Bus (CH17_SBR)	32	RW	0000_8002h
21_4010h	Channel Priority (CH17_PRI)	32	RW	0000_0000h
21_4020h	TCD Source Address (TCD17_SADDR)	32	RW	0000_0000h
21_4024h	TCD Signed Source Address Offset (TCD17_SOFF)	16	RW	0000h
21_4026h	TCD Transfer Attributes (TCD17_ATTR)	16	RW	0000h
21_4028h	TCD Transfer Size Without Minor Loop Offsets (TCD17_NBYTES_MLOFFNO)	32	RW	0000_0000h
21_4028h	TCD Transfer Size with Minor Loop Offsets (TCD17_NBYTES_MLOFFYES)	32	RW	0000_0000h
21_402Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD17_SLAST_SDA)	32	RW	0000_0000h
21_4030h	TCD Destination Address (TCD17_DADDR)	32	RW	0000_0000h
21_4034h	TCD Signed Destination Address Offset (TCD17_DOFF)	16	RW	0000h
21_4036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD17_CITER_ELINKNO)	16	RW	0000h
21_4036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD17_CITER_ELINKYES)	16	RW	0000h
21_4038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD17_DLAST_SGA)	32	RW	0000_0000h
21_403Ch	TCD Control and Status (TCD17_CSR)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
21_403Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD17_BITER_ELINKNO)	16	RW	0000h
21_403Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD17_BITER_ELINKYES)	16	RW	0000h
21_8000h	Channel Control and Status (CH18_CSR)	32	RW	0000_0000h
21_8004h	Channel Error Status (CH18_ES)	32	RW	0000_0000h
21_8008h	Channel Interrupt Status (CH18_INT)	32	RW	0000_0000h
21_800Ch	Channel System Bus (CH18_SBR)	32	RW	0000_8002h
21_8010h	Channel Priority (CH18_PRI)	32	RW	0000_0000h
21_8020h	TCD Source Address (TCD18_SADDR)	32	RW	0000_0000h
21_8024h	TCD Signed Source Address Offset (TCD18_SOFF)	16	RW	0000h
21_8026h	TCD Transfer Attributes (TCD18_ATTR)	16	RW	0000h
21_8028h	TCD Transfer Size Without Minor Loop Offsets (TCD18_NBYTES_MLOFFNO)	32	RW	0000_0000h
21_8028h	TCD Transfer Size with Minor Loop Offsets (TCD18_NBYTES_MLOFFYES)	32	RW	0000_0000h
21_802Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD18_SLAST_SDA)	32	RW	0000_0000h
21_8030h	TCD Destination Address (TCD18_DADDR)	32	RW	0000_0000h
21_8034h	TCD Signed Destination Address Offset (TCD18_DOFF)	16	RW	0000h
21_8036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD18_CITER_ELINKNO)	16	RW	0000h
21_8036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD18_CITER_ELINKYES)	16	RW	0000h
21_8038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD18_DLAST_SGA)	32	RW	0000_0000h
21_803Ch	TCD Control and Status (TCD18_CSR)	16	RW	0000h
21_803Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD18_BITER_ELINKNO)	16	RW	0000h
21_803Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD18_BITER_ELINKYES)	16	RW	0000h
21_C000h	Channel Control and Status (CH19_CSR)	32	RW	0000_0000h
21_C004h	Channel Error Status (CH19_ES)	32	RW	0000_0000h
21_C008h	Channel Interrupt Status (CH19_INT)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
21_C00Ch	Channel System Bus (CH19_SBR)	32	RW	0000_8002h
21_C010h	Channel Priority (CH19_PRI)	32	RW	0000_0000h
21_C020h	TCD Source Address (TCD19_SADDR)	32	RW	0000_0000h
21_C024h	TCD Signed Source Address Offset (TCD19_SOFF)	16	RW	0000h
21_C026h	TCD Transfer Attributes (TCD19_ATTR)	16	RW	0000h
21_C028h	TCD Transfer Size Without Minor Loop Offsets (TCD19_NBYTES_MLOFFNO)	32	RW	0000_0000h
21_C028h	TCD Transfer Size with Minor Loop Offsets (TCD19_NBYTES_MLOFFYES)	32	RW	0000_0000h
21_C02Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD19_SLAST_SDA)	32	RW	0000_0000h
21_C030h	TCD Destination Address (TCD19_DADDR)	32	RW	0000_0000h
21_C034h	TCD Signed Destination Address Offset (TCD19_DOFF)	16	RW	0000h
21_C036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD19_CITER_ELINKNO)	16	RW	0000h
21_C036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD19_CITER_ELINKYES)	16	RW	0000h
21_C038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD19_DLAST_SGA)	32	RW	0000_0000h
21_C03Ch	TCD Control and Status (TCD19_CSR)	16	RW	0000h
21_C03Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD19_BITER_ELINKNO)	16	RW	0000h
21_C03Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD19_BITER_ELINKYES)	16	RW	0000h
22_0000h	Channel Control and Status (CH20_CSR)	32	RW	0000_0000h
22_0004h	Channel Error Status (CH20_ES)	32	RW	0000_0000h
22_0008h	Channel Interrupt Status (CH20_INT)	32	RW	0000_0000h
22_000Ch	Channel System Bus (CH20_SBR)	32	RW	0000_8002h
22_0010h	Channel Priority (CH20_PRI)	32	RW	0000_0000h
22_0020h	TCD Source Address (TCD20_SADDR)	32	RW	0000_0000h
22_0024h	TCD Signed Source Address Offset (TCD20_SOFF)	16	RW	0000h
22_0026h	TCD Transfer Attributes (TCD20_ATTR)	16	RW	0000h
22_0028h	TCD Transfer Size Without Minor Loop Offsets (TCD20_NBYTES_MLOFFNO)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
22_0028h	TCD Transfer Size with Minor Loop Offsets (TCD20_NBYTES_MLOFFYES)	32	RW	0000_0000h
22_002Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD20_SLAST_SDA)	32	RW	0000_0000h
22_0030h	TCD Destination Address (TCD20_DADDR)	32	RW	0000_0000h
22_0034h	TCD Signed Destination Address Offset (TCD20_DOFF)	16	RW	0000h
22_0036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD20_CITER_ELINKNO)	16	RW	0000h
22_0036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD20_CITER_ELINKYES)	16	RW	0000h
22_0038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD20_DLAST_SGA)	32	RW	0000_0000h
22_003Ch	TCD Control and Status (TCD20_CSR)	16	RW	0000h
22_003Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD20_BITER_ELINKNO)	16	RW	0000h
22_003Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD20_BITER_ELINKYES)	16	RW	0000h
22_4000h	Channel Control and Status (CH21_CSR)	32	RW	0000_0000h
22_4004h	Channel Error Status (CH21_ES)	32	RW	0000_0000h
22_4008h	Channel Interrupt Status (CH21_INT)	32	RW	0000_0000h
22_400Ch	Channel System Bus (CH21_SBR)	32	RW	0000_8002h
22_4010h	Channel Priority (CH21_PRI)	32	RW	0000_0000h
22_4020h	TCD Source Address (TCD21_SADDR)	32	RW	0000_0000h
22_4024h	TCD Signed Source Address Offset (TCD21_SOFF)	16	RW	0000h
22_4026h	TCD Transfer Attributes (TCD21_ATTR)	16	RW	0000h
22_4028h	TCD Transfer Size Without Minor Loop Offsets (TCD21_NBYTES_MLOFFNO)	32	RW	0000_0000h
22_4028h	TCD Transfer Size with Minor Loop Offsets (TCD21_NBYTES_MLOFFYES)	32	RW	0000_0000h
22_402Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD21_SLAST_SDA)	32	RW	0000_0000h
22_4030h	TCD Destination Address (TCD21_DADDR)	32	RW	0000_0000h
22_4034h	TCD Signed Destination Address Offset (TCD21_DOFF)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
22_4036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD21_CITER_ELINKNO)	16	RW	0000h
22_4036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD21_CITER_ELINKYES)	16	RW	0000h
22_4038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD21_DLAST_SGA)	32	RW	0000_0000h
22_403Ch	TCD Control and Status (TCD21_CSR)	16	RW	0000h
22_403Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD21_BITER_ELINKNO)	16	RW	0000h
22_403Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD21_BITER_ELINKYES)	16	RW	0000h
22_8000h	Channel Control and Status (CH22_CSR)	32	RW	0000_0000h
22_8004h	Channel Error Status (CH22_ES)	32	RW	0000_0000h
22_8008h	Channel Interrupt Status (CH22_INT)	32	RW	0000_0000h
22_800Ch	Channel System Bus (CH22_SBR)	32	RW	0000_8002h
22_8010h	Channel Priority (CH22_PRI)	32	RW	0000_0000h
22_8020h	TCD Source Address (TCD22_SADDR)	32	RW	0000_0000h
22_8024h	TCD Signed Source Address Offset (TCD22_SOFF)	16	RW	0000h
22_8026h	TCD Transfer Attributes (TCD22_ATTR)	16	RW	0000h
22_8028h	TCD Transfer Size Without Minor Loop Offsets (TCD22_NBYTES_MLOFFNO)	32	RW	0000_0000h
22_8028h	TCD Transfer Size with Minor Loop Offsets (TCD22_NBYTES_MLOFFYES)	32	RW	0000_0000h
22_802Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD22_SLAST_SDA)	32	RW	0000_0000h
22_8030h	TCD Destination Address (TCD22_DADDR)	32	RW	0000_0000h
22_8034h	TCD Signed Destination Address Offset (TCD22_DOFF)	16	RW	0000h
22_8036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD22_CITER_ELINKNO)	16	RW	0000h
22_8036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD22_CITER_ELINKYES)	16	RW	0000h
22_8038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD22_DLAST_SGA)	32	RW	0000_0000h
22_803Ch	TCD Control and Status (TCD22_CSR)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
22_803Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD22_BITER_ELINKNO)	16	RW	0000h
22_803Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD22_BITER_ELINKYES)	16	RW	0000h
22_C000h	Channel Control and Status (CH23_CSR)	32	RW	0000_0000h
22_C004h	Channel Error Status (CH23_ES)	32	RW	0000_0000h
22_C008h	Channel Interrupt Status (CH23_INT)	32	RW	0000_0000h
22_C00Ch	Channel System Bus (CH23_SBR)	32	RW	0000_8002h
22_C010h	Channel Priority (CH23_PRI)	32	RW	0000_0000h
22_C020h	TCD Source Address (TCD23_SADDR)	32	RW	0000_0000h
22_C024h	TCD Signed Source Address Offset (TCD23_SOFF)	16	RW	0000h
22_C026h	TCD Transfer Attributes (TCD23_ATTR)	16	RW	0000h
22_C028h	TCD Transfer Size Without Minor Loop Offsets (TCD23_NBYTES_MLOFFNO)	32	RW	0000_0000h
22_C028h	TCD Transfer Size with Minor Loop Offsets (TCD23_NBYTES_MLOFFYES)	32	RW	0000_0000h
22_C02Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD23_SLAST_SDA)	32	RW	0000_0000h
22_C030h	TCD Destination Address (TCD23_DADDR)	32	RW	0000_0000h
22_C034h	TCD Signed Destination Address Offset (TCD23_DOFF)	16	RW	0000h
22_C036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD23_CITER_ELINKNO)	16	RW	0000h
22_C036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD23_CITER_ELINKYES)	16	RW	0000h
22_C038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD23_DLAST_SGA)	32	RW	0000_0000h
22_C03Ch	TCD Control and Status (TCD23_CSR)	16	RW	0000h
22_C03Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD23_BITER_ELINKNO)	16	RW	0000h
22_C03Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD23_BITER_ELINKYES)	16	RW	0000h
23_0000h	Channel Control and Status (CH24_CSR)	32	RW	0000_0000h
23_0004h	Channel Error Status (CH24_ES)	32	RW	0000_0000h
23_0008h	Channel Interrupt Status (CH24_INT)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
23_000Ch	Channel System Bus (CH24_SBR)	32	RW	0000_8002h
23_0010h	Channel Priority (CH24_PRI)	32	RW	0000_0000h
23_0020h	TCD Source Address (TCD24_SADDR)	32	RW	0000_0000h
23_0024h	TCD Signed Source Address Offset (TCD24_SOFF)	16	RW	0000h
23_0026h	TCD Transfer Attributes (TCD24_ATTR)	16	RW	0000h
23_0028h	TCD Transfer Size Without Minor Loop Offsets (TCD24_NBYTES_MLOFFNO)	32	RW	0000_0000h
23_0028h	TCD Transfer Size with Minor Loop Offsets (TCD24_NBYTES_MLOFFYES)	32	RW	0000_0000h
23_002Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD24_SLAST_SDA)	32	RW	0000_0000h
23_0030h	TCD Destination Address (TCD24_DADDR)	32	RW	0000_0000h
23_0034h	TCD Signed Destination Address Offset (TCD24_DOFF)	16	RW	0000h
23_0036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD24_CITER_ELINKNO)	16	RW	0000h
23_0036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD24_CITER_ELINKYES)	16	RW	0000h
23_0038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD24_DLAST_SGA)	32	RW	0000_0000h
23_003Ch	TCD Control and Status (TCD24_CSR)	16	RW	0000h
23_003Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD24_BITER_ELINKNO)	16	RW	0000h
23_003Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD24_BITER_ELINKYES)	16	RW	0000h
23_4000h	Channel Control and Status (CH25_CSR)	32	RW	0000_0000h
23_4004h	Channel Error Status (CH25_ES)	32	RW	0000_0000h
23_4008h	Channel Interrupt Status (CH25_INT)	32	RW	0000_0000h
23_400Ch	Channel System Bus (CH25_SBR)	32	RW	0000_8002h
23_4010h	Channel Priority (CH25_PRI)	32	RW	0000_0000h
23_4020h	TCD Source Address (TCD25_SADDR)	32	RW	0000_0000h
23_4024h	TCD Signed Source Address Offset (TCD25_SOFF)	16	RW	0000h
23_4026h	TCD Transfer Attributes (TCD25_ATTR)	16	RW	0000h
23_4028h	TCD Transfer Size Without Minor Loop Offsets (TCD25_NBYTES_MLOFFNO)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
23_4028h	TCD Transfer Size with Minor Loop Offsets (TCD25_NBYTES_MLOFFYES)	32	RW	0000_0000h
23_402Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD25_SLAST_SDA)	32	RW	0000_0000h
23_4030h	TCD Destination Address (TCD25_DADDR)	32	RW	0000_0000h
23_4034h	TCD Signed Destination Address Offset (TCD25_DOFF)	16	RW	0000h
23_4036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD25_CITER_ELINKNO)	16	RW	0000h
23_4036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD25_CITER_ELINKYES)	16	RW	0000h
23_4038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD25_DLAST_SGA)	32	RW	0000_0000h
23_403Ch	TCD Control and Status (TCD25_CSR)	16	RW	0000h
23_403Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD25_BITER_ELINKNO)	16	RW	0000h
23_403Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD25_BITER_ELINKYES)	16	RW	0000h
23_8000h	Channel Control and Status (CH26_CSR)	32	RW	0000_0000h
23_8004h	Channel Error Status (CH26_ES)	32	RW	0000_0000h
23_8008h	Channel Interrupt Status (CH26_INT)	32	RW	0000_0000h
23_800Ch	Channel System Bus (CH26_SBR)	32	RW	0000_8002h
23_8010h	Channel Priority (CH26_PRI)	32	RW	0000_0000h
23_8020h	TCD Source Address (TCD26_SADDR)	32	RW	0000_0000h
23_8024h	TCD Signed Source Address Offset (TCD26_SOFF)	16	RW	0000h
23_8026h	TCD Transfer Attributes (TCD26_ATTR)	16	RW	0000h
23_8028h	TCD Transfer Size Without Minor Loop Offsets (TCD26_NBYTES_MLOFFNO)	32	RW	0000_0000h
23_8028h	TCD Transfer Size with Minor Loop Offsets (TCD26_NBYTES_MLOFFYES)	32	RW	0000_0000h
23_802Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD26_SLAST_SDA)	32	RW	0000_0000h
23_8030h	TCD Destination Address (TCD26_DADDR)	32	RW	0000_0000h
23_8034h	TCD Signed Destination Address Offset (TCD26_DOFF)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
23_8036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD26_CITER_ELINKNO)	16	RW	0000h
23_8036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD26_CITER_ELINKYES)	16	RW	0000h
23_8038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD26_DLAST_SGA)	32	RW	0000_0000h
23_803Ch	TCD Control and Status (TCD26_CSR)	16	RW	0000h
23_803Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD26_BITER_ELINKNO)	16	RW	0000h
23_803Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD26_BITER_ELINKYES)	16	RW	0000h
23_C000h	Channel Control and Status (CH27_CSR)	32	RW	0000_0000h
23_C004h	Channel Error Status (CH27_ES)	32	RW	0000_0000h
23_C008h	Channel Interrupt Status (CH27_INT)	32	RW	0000_0000h
23_C00Ch	Channel System Bus (CH27_SBR)	32	RW	0000_8002h
23_C010h	Channel Priority (CH27_PRI)	32	RW	0000_0000h
23_C020h	TCD Source Address (TCD27_SADDR)	32	RW	0000_0000h
23_C024h	TCD Signed Source Address Offset (TCD27_SOFF)	16	RW	0000h
23_C026h	TCD Transfer Attributes (TCD27_ATTR)	16	RW	0000h
23_C028h	TCD Transfer Size Without Minor Loop Offsets (TCD27_NBYTES_MLOFFNO)	32	RW	0000_0000h
23_C028h	TCD Transfer Size with Minor Loop Offsets (TCD27_NBYTES_MLOFFYES)	32	RW	0000_0000h
23_C02Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD27_SLAST_SDA)	32	RW	0000_0000h
23_C030h	TCD Destination Address (TCD27_DADDR)	32	RW	0000_0000h
23_C034h	TCD Signed Destination Address Offset (TCD27_DOFF)	16	RW	0000h
23_C036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD27_CITER_ELINKNO)	16	RW	0000h
23_C036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD27_CITER_ELINKYES)	16	RW	0000h
23_C038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD27_DLAST_SGA)	32	RW	0000_0000h
23_C03Ch	TCD Control and Status (TCD27_CSR)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
23_C03Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD27_BITER_ELINKNO)	16	RW	0000h
23_C03Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD27_BITER_ELINKYES)	16	RW	0000h
24_0000h	Channel Control and Status (CH28_CSR)	32	RW	0000_0000h
24_0004h	Channel Error Status (CH28_ES)	32	RW	0000_0000h
24_0008h	Channel Interrupt Status (CH28_INT)	32	RW	0000_0000h
24_000Ch	Channel System Bus (CH28_SBR)	32	RW	0000_8002h
24_0010h	Channel Priority (CH28_PRI)	32	RW	0000_0000h
24_0020h	TCD Source Address (TCD28_SADDR)	32	RW	0000_0000h
24_0024h	TCD Signed Source Address Offset (TCD28_SOFF)	16	RW	0000h
24_0026h	TCD Transfer Attributes (TCD28_ATTR)	16	RW	0000h
24_0028h	TCD Transfer Size Without Minor Loop Offsets (TCD28_NBYTES_MLOFFNO)	32	RW	0000_0000h
24_0028h	TCD Transfer Size with Minor Loop Offsets (TCD28_NBYTES_MLOFFYES)	32	RW	0000_0000h
24_002Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD28_SLAST_SDA)	32	RW	0000_0000h
24_0030h	TCD Destination Address (TCD28_DADDR)	32	RW	0000_0000h
24_0034h	TCD Signed Destination Address Offset (TCD28_DOFF)	16	RW	0000h
24_0036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD28_CITER_ELINKNO)	16	RW	0000h
24_0036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD28_CITER_ELINKYES)	16	RW	0000h
24_0038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD28_DLAST_SGA)	32	RW	0000_0000h
24_003Ch	TCD Control and Status (TCD28_CSR)	16	RW	0000h
24_003Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD28_BITER_ELINKNO)	16	RW	0000h
24_003Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD28_BITER_ELINKYES)	16	RW	0000h
24_4000h	Channel Control and Status (CH29_CSR)	32	RW	0000_0000h
24_4004h	Channel Error Status (CH29_ES)	32	RW	0000_0000h
24_4008h	Channel Interrupt Status (CH29_INT)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
24_400Ch	Channel System Bus (CH29_SBR)	32	RW	0000_8002h
24_4010h	Channel Priority (CH29_PRI)	32	RW	0000_0000h
24_4020h	TCD Source Address (TCD29_SADDR)	32	RW	0000_0000h
24_4024h	TCD Signed Source Address Offset (TCD29_SOFF)	16	RW	0000h
24_4026h	TCD Transfer Attributes (TCD29_ATTR)	16	RW	0000h
24_4028h	TCD Transfer Size Without Minor Loop Offsets (TCD29_NBYTES_MLOFFNO)	32	RW	0000_0000h
24_4028h	TCD Transfer Size with Minor Loop Offsets (TCD29_NBYTES_MLOFFYES)	32	RW	0000_0000h
24_402Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD29_SLAST_SDA)	32	RW	0000_0000h
24_4030h	TCD Destination Address (TCD29_DADDR)	32	RW	0000_0000h
24_4034h	TCD Signed Destination Address Offset (TCD29_DOFF)	16	RW	0000h
24_4036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD29_CITER_ELINKNO)	16	RW	0000h
24_4036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD29_CITER_ELINKYES)	16	RW	0000h
24_4038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD29_DLAST_SGA)	32	RW	0000_0000h
24_403Ch	TCD Control and Status (TCD29_CSR)	16	RW	0000h
24_403Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD29_BITER_ELINKNO)	16	RW	0000h
24_403Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD29_BITER_ELINKYES)	16	RW	0000h
24_8000h	Channel Control and Status (CH30_CSR)	32	RW	0000_0000h
24_8004h	Channel Error Status (CH30_ES)	32	RW	0000_0000h
24_8008h	Channel Interrupt Status (CH30_INT)	32	RW	0000_0000h
24_800Ch	Channel System Bus (CH30_SBR)	32	RW	0000_8002h
24_8010h	Channel Priority (CH30_PRI)	32	RW	0000_0000h
24_8020h	TCD Source Address (TCD30_SADDR)	32	RW	0000_0000h
24_8024h	TCD Signed Source Address Offset (TCD30_SOFF)	16	RW	0000h
24_8026h	TCD Transfer Attributes (TCD30_ATTR)	16	RW	0000h
24_8028h	TCD Transfer Size Without Minor Loop Offsets (TCD30_NBYTES_MLOFFNO)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
24_8028h	TCD Transfer Size with Minor Loop Offsets (TCD30_NBYTES_MLOFFYES)	32	RW	0000_0000h
24_802Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD30_SLAST_SDA)	32	RW	0000_0000h
24_8030h	TCD Destination Address (TCD30_DADDR)	32	RW	0000_0000h
24_8034h	TCD Signed Destination Address Offset (TCD30_DOFF)	16	RW	0000h
24_8036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD30_CITER_ELINKNO)	16	RW	0000h
24_8036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD30_CITER_ELINKYES)	16	RW	0000h
24_8038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD30_DLAST_SGA)	32	RW	0000_0000h
24_803Ch	TCD Control and Status (TCD30_CSR)	16	RW	0000h
24_803Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD30_BITER_ELINKNO)	16	RW	0000h
24_803Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD30_BITER_ELINKYES)	16	RW	0000h
24_C000h	Channel Control and Status (CH31_CSR)	32	RW	0000_0000h
24_C004h	Channel Error Status (CH31_ES)	32	RW	0000_0000h
24_C008h	Channel Interrupt Status (CH31_INT)	32	RW	0000_0000h
24_C00Ch	Channel System Bus (CH31_SBR)	32	RW	0000_8002h
24_C010h	Channel Priority (CH31_PRI)	32	RW	0000_0000h
24_C020h	TCD Source Address (TCD31_SADDR)	32	RW	0000_0000h
24_C024h	TCD Signed Source Address Offset (TCD31_SOFF)	16	RW	0000h
24_C026h	TCD Transfer Attributes (TCD31_ATTR)	16	RW	0000h
24_C028h	TCD Transfer Size Without Minor Loop Offsets (TCD31_NBYTES_MLOFFNO)	32	RW	0000_0000h
24_C028h	TCD Transfer Size with Minor Loop Offsets (TCD31_NBYTES_MLOFFYES)	32	RW	0000_0000h
24_C02Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD31_SLAST_SDA)	32	RW	0000_0000h
24_C030h	TCD Destination Address (TCD31_DADDR)	32	RW	0000_0000h
24_C034h	TCD Signed Destination Address Offset (TCD31_DOFF)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
24_C036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD31_CITER_ELINKNO)	16	RW	0000h
24_C036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD31_CITER_ELINKYES)	16	RW	0000h
24_C038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD31_DLAST_SGA)	32	RW	0000_0000h
24_C03Ch	TCD Control and Status (TCD31_CSR)	16	RW	0000h
24_C03Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD31_BITER_ELINKNO)	16	RW	0000h
24_C03Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD31_BITER_ELINKYES)	16	RW	0000h

15.6.2.2 Channel Control and Status (CH0_CSR - CH31_CSR)

Offset

Register	Offset
CH0_CSR	0h
CH1_CSR	4000h
CH2_CSR	8000h
CH3_CSR	C000h
CH4_CSR	1_0000h
CH5_CSR	1_4000h
CH6_CSR	1_8000h
CH7_CSR	1_C000h
CH8_CSR	2_0000h
CH9_CSR	2_4000h
CH10_CSR	2_8000h
CH11_CSR	2_C000h
CH12_CSR	20_0000h
CH13_CSR	20_4000h
CH14_CSR	20_8000h
CH15_CSR	20_C000h

Table continues on the next page...

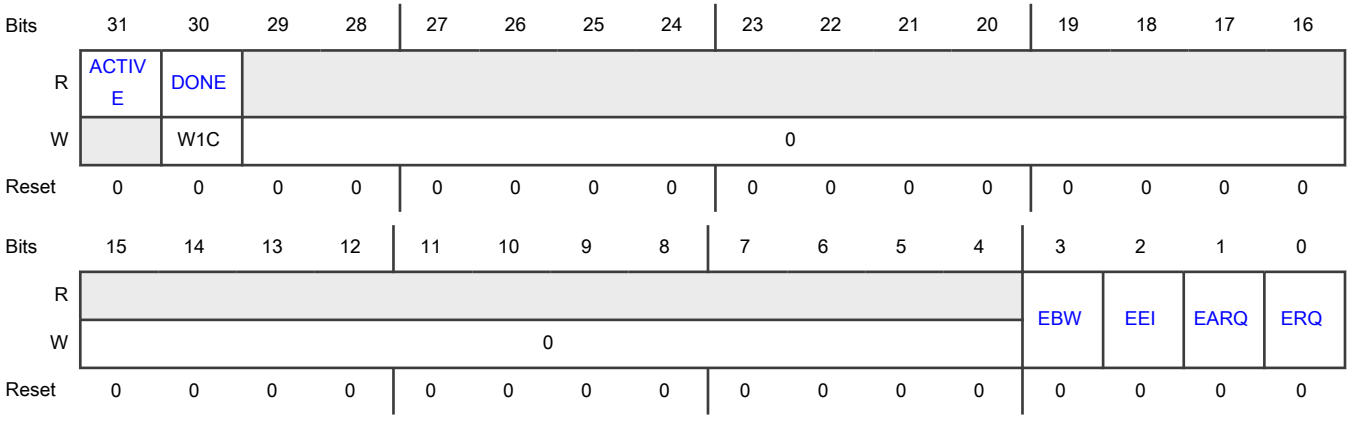
Table continued from the previous page...

Register	Offset
CH16_CSR	21_0000h
CH17_CSR	21_4000h
CH18_CSR	21_8000h
CH19_CSR	21_C000h
CH20_CSR	22_0000h
CH21_CSR	22_4000h
CH22_CSR	22_8000h
CH23_CSR	22_C000h
CH24_CSR	23_0000h
CH25_CSR	23_4000h
CH26_CSR	23_8000h
CH27_CSR	23_C000h
CH28_CSR	24_0000h
CH29_CSR	24_4000h
CH30_CSR	24_8000h
CH31_CSR	24_C000h

Function

This register contains several fields related to hardware and interrupt requests, configuration, and status for the given channel.

Diagram



Fields

Field	Function
31 ACTIVE	Channel Active The ACTIVE field indicates the channel was selected by arbitration and is executing the prescribed transfers. The eDMA sets it to 1 when channel service begins, and clears it to 0 as the minor loop completes or when any error condition is detected. Except for dynamic scatter/gather or dynamic channel linking, you must not modify the transfer control descriptor when a channel is active.
30 DONE	Channel Done The DONE field indicates the eDMA has completed the major loop. The eDMA engine sets this field as the CITER count reaches zero. If enabled, the eDMA generates an interrupt request corresponding to this completed channel. The software clears it, or the hardware clears it when the channel is activated. **NOTE** This field must be cleared to 0 before writing the MAJORELINK or ESG fields.
29-4 —	Reserved
3 EBW	Enable Buffered Writes When buffered writes are enabled, all writes except for the last write sequence of the minor loop are signaled by the eDMA as bufferable. 0b - Buffered writes on system bus disabled. Buffered writes on system bus disabled 1b - Buffered writes on system bus enabled. Bufferable write signal asserted on all system bus writes except during last write sequence
2 EEI	Enable Error Interrupt The EEI field enables the error interrupt signal for the channel. The DMA error indicator and the error interrupt enable flag must be asserted before an error interrupt request for a given channel is asserted to the interrupt controller. 0b - Error signal for corresponding channel does not generate error interrupt 1b - Assertion of error signal for corresponding channel generates error interrupt request
1 EARQ	Enable Asynchronous DMA Request In Stop Mode For Channel The enable asynchronous DMA request field (EARQ) does not affect DMA operations. When set to 1, this field allows the hardware service request enable field (ERQ) to propagate out of the DMA to the power controller. When cleared to 0, this field masks the hardware service request enable field to the power controller. 0b - Disable asynchronous DMA request for the channel 1b - Enable asynchronous DMA request for the channel
0 ERQ	Enable DMA Request Disable a channel's hardware service request at the source before clearing the channel's ERQ field. The DMA hardware request input signal and the enable request field (ERQ) must be asserted before a channel's hardware service request is accepted. The state of the eDMA enable request field does not

Table continues on the next page...

Table continued from the previous page...

Field	Function
	affect a channel service request made explicitly through software or channel linking. The state of the ERQ field does not affect the channel's START field. 0b - DMA hardware request signal for corresponding channel disabled 1b - DMA hardware request signal for corresponding channel enabled

15.6.2.3 Channel Error Status (CH0_ES - CH31_ES)

Offset

Register	Offset
CH0_ES	4h
CH1_ES	4004h
CH2_ES	8004h
CH3_ES	C004h
CH4_ES	1_0004h
CH5_ES	1_4004h
CH6_ES	1_8004h
CH7_ES	1_C004h
CH8_ES	2_0004h
CH9_ES	2_4004h
CH10_ES	2_8004h
CH11_ES	2_C004h
CH12_ES	20_0004h
CH13_ES	20_4004h
CH14_ES	20_8004h
CH15_ES	20_C004h
CH16_ES	21_0004h
CH17_ES	21_4004h
CH18_ES	21_8004h
CH19_ES	21_C004h
CH20_ES	22_0004h
CH21_ES	22_4004h
CH22_ES	22_8004h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
CH23_ES	22_C004h
CH24_ES	23_0004h
CH25_ES	23_4004h
CH26_ES	23_8004h
CH27_ES	23_C004h
CH28_ES	24_0004h
CH29_ES	24_4004h
CH30_ES	24_8004h
CH31_ES	24_C004h

Function

The ES provides information concerning the last recorded channel error. Channel errors can be caused by:

- An illegal setting in the transfer control descriptor
- An error termination to a bus initiator read or write cycle

The ERR field signals the presence of an error for the channel. The eDMA engine signals the occurrence of an error condition by setting the appropriate field in this register. The outputs of this register are enabled by the contents of the CHn_CSR[EEI] field, then logically summed across all channels to form an error interrupt request, which may be routed to the interrupt controller. In addition, this enabled error status is logically OR'd onto the channel done interrupt, CHn_INT[INT], thus forming a done or error interrupt on a per channel basis.

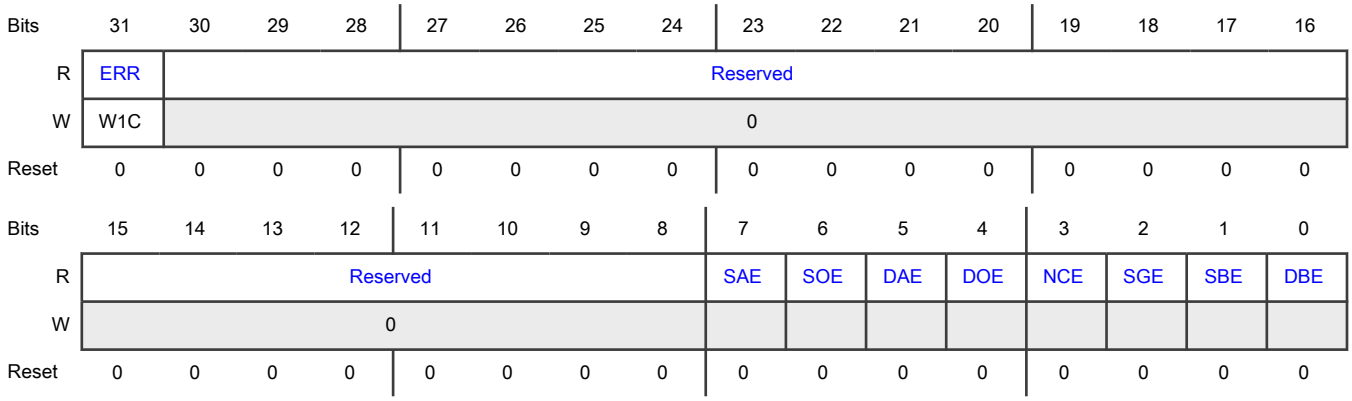
During the execution of the interrupt service routine associated with any DMA errors, it is software's responsibility to clear the appropriate bit, negating the error-interrupt request. The normal DMA channel completion indicators (setting the transfer control descriptor DONE flag and the possible assertion of an interrupt request) are not affected when eDMA detects an error. The contents of this ERR register field can also be polled because a non-zero value indicates the presence of a channel error, regardless of the state of the EEI mask.

The state of any given channel's error indicators is affected by writes to this register. Writing a 1 to the ERR field clears the channel's error status, and writing a 0 has no effect.

An unspecified error, where only the ERR field is set to 1, indicates that either a transfer was cancelled with an error or else an uncorrectable TCD error occurred. The Management Page Error Status register has full view of the error condition.

See Fault reporting and handling for more details.

Diagram



Fields

Field	Function
31 ERR	Error In Channel 0b - An error in this channel has not occurred 1b - An error in this channel has occurred
30-8 —	Reserved
7 SAE	Source Address Error TCDn_SADDR is inconsistent with TCDn_ATTR[SSIZE]. 0b - No source address configuration error 1b - Last recorded error was a configuration error detected in the TCDn_SADDR field
6 SOE	Source Offset Error TCDn_SOFF is inconsistent with TCDn_ATTR[SSIZE]. 0b - No source offset configuration error 1b - Last recorded error was a configuration error detected in the TCDn_SOFF field
5 DAE	Destination Address Error TCDn_DADDR is inconsistent with TCDn_ATTR[DSIZE]. 0b - No destination address configuration error 1b - Last recorded error was a configuration error detected in the TCDn_DADDR field
4 DOE	Destination Offset Error TCDn_DOFF is inconsistent with TCDn_ATTR[DSIZE]. 0b - No destination offset configuration error 1b - Last recorded error was a configuration error detected in the TCDn_DOFF field

Table continues on the next page...

Table continued from the previous page...

Field	Function
3 NCE	NBYTES/CITER Configuration Error This error indicates that one of the following has occurred: • TCDn_NBYTES is not a multiple of TCDn_ATTR[SSIZE] and TCDn_ATTR[DSIZE] • TCDn_CITER[CITER] is equal to zero • TCDn_CITER[ELINK] is not equal to TCDn_BITER[ELINK] 0b - No NBYTES/CITER configuration error 1b - Last recorded error was a configuration error detected in the TCDn_NBYTES or TCDn_CITER fields
2 SGE	Scatter/Gather Configuration Error When this field is 1, it indicates that TCDn_DLAST_SGA is not on a 32-byte boundary. This field is checked at the beginning of a scatter/gather operation after major loop completion if TCDn_CSR[ESG]is enabled. 0b - No scatter/gather configuration error 1b - Last recorded error was a configuration error detected in the TCDn_DLAST_SGA field
1 SBE	Source Bus Error 0b - No source bus error 1b - Last recorded error was bus error on source read
0 DBE	Destination Bus Error 0b - No destination bus error 1b - Last recorded error was bus error on destination write

15.6.2.4 Channel Interrupt Status (CH0_INT - CH31_INT)

Offset

Register	Offset
CH0_INT	8h
CH1_INT	4008h
CH2_INT	8008h
CH3_INT	C008h
CH4_INT	1_0008h
CH5_INT	1_4008h
CH6_INT	1_8008h
CH7_INT	1_C008h

Table continues on the next page...

Table continued from the previous page...

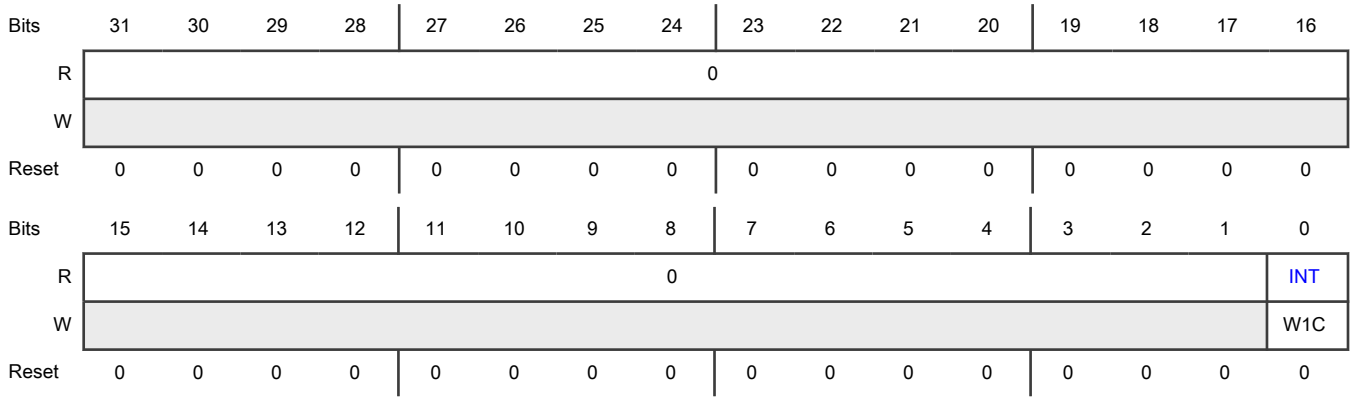
Register	Offset
CH8_INT	2_0008h
CH9_INT	2_4008h
CH10_INT	2_8008h
CH11_INT	2_C008h
CH12_INT	20_0008h
CH13_INT	20_4008h
CH14_INT	20_8008h
CH15_INT	20_C008h
CH16_INT	21_0008h
CH17_INT	21_4008h
CH18_INT	21_8008h
CH19_INT	21_C008h
CH20_INT	22_0008h
CH21_INT	22_4008h
CH22_INT	22_8008h
CH23_INT	22_C008h
CH24_INT	23_0008h
CH25_INT	23_4008h
CH26_INT	23_8008h
CH27_INT	23_C008h
CH28_INT	24_0008h
CH29_INT	24_4008h
CH30_INT	24_8008h
CH31_INT	24_C008h

Function

The INT field signals the presence of an interrupt request for the channel. Depending on the appropriate bit setting in the transfer control descriptors, the eDMA engine generates an interrupt on data transfer completion or an error condition.

The outputs of this register are directly routed to the interrupt controller. During the interrupt service routine associated with any given channel, it is the software's responsibility to clear the appropriate bit, negating the interrupt request. On writes to INT, a 1 clears the channel's interrupt request. A zero has no effect on the channel's current interrupt status.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 INT	Interrupt Request 0b - Interrupt request for corresponding channel cleared 1b - Interrupt request for corresponding channel active

15.6.2.5 Channel System Bus (CH0_SBR - CH31_SBR)

Offset

Register	Offset
CH0_SBR	Ch
CH1_SBR	400Ch
CH2_SBR	800Ch
CH3_SBR	C00Ch
CH4_SBR	1_000Ch
CH5_SBR	1_400Ch
CH6_SBR	1_800Ch
CH7_SBR	1_C00Ch
CH8_SBR	2_000Ch
CH9_SBR	2_400Ch
CH10_SBR	2_800Ch
CH11_SBR	2_C00Ch

Table continues on the next page...

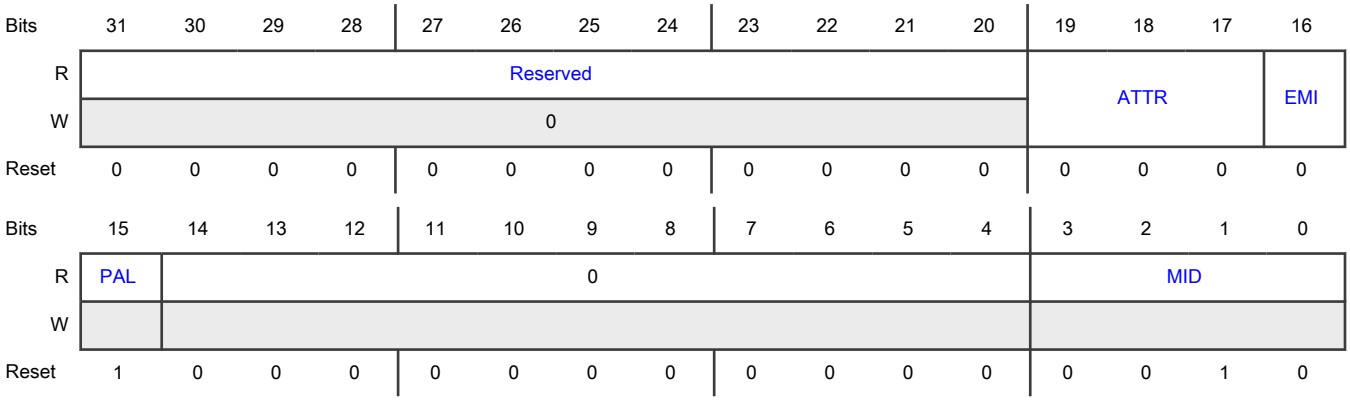
Table continued from the previous page...

Register	Offset
CH12_SBR	20_000Ch
CH13_SBR	20_400Ch
CH14_SBR	20_800Ch
CH15_SBR	20_C00Ch
CH16_SBR	21_000Ch
CH17_SBR	21_400Ch
CH18_SBR	21_800Ch
CH19_SBR	21_C00Ch
CH20_SBR	22_000Ch
CH21_SBR	22_400Ch
CH22_SBR	22_800Ch
CH23_SBR	22_C00Ch
CH24_SBR	23_000Ch
CH25_SBR	23_400Ch
CH26_SBR	23_800Ch
CH27_SBR	23_C00Ch
CH28_SBR	24_000Ch
CH29_SBR	24_400Ch
CH30_SBR	24_800Ch
CH31_SBR	24_C00Ch

Function

The Channel System Bus register places identification and attribute information on the system bus interface for the eDMA.

Diagram



Fields

Field	Function
31-20 —	Reserved
19-17 ATTR	Attribute Output DMA's system bus attribute output value.
16 EMI	Enable Initiator ID Replication The eDMA initiator ID replication field allows the eDMA to use the same protection level and system bus ID of the initiator programming the eDMA's TCD. When enabled, the eDMA uses the initiator ID and protection level stored in the CHn_SBR registers, instead of the eDMA's default values. When a initiator (for example a core) programs a TCD, its initiator ID and protection level are captured when the TCD*n*_CSR control attributes are written. A scatter/gather operation does not affect the CHn_SBR registers. You can write the EMI only if CSR[GMRC] = 1, which means Global Initiator ID Replication Control is enabled; otherwise, the EMI is forced to zero. See chip-specific eDMA information for details on ID replication. **NOTE** If initiator ID replication is disabled, the privileged protection level (Supervisor mode) for DMA transfers is used. 0b - Initiator ID replication is disabled 1b - Initiator ID replication is enabled
15 PAL	Privileged Access Level This field controls DMA's protection level on the system bus when the channel is active. **NOTE** When you enable initiator ID replication, the value captured in this register is the privilege level of the core or other initiator writing the channel's transfer control descriptor, which is the lower byte of TCD*n*_CSR. 0b - User protection level for DMA transfers 1b - Privileged protection level for DMA transfers
14-4 —	Reserved
3-0 MID	Initiator ID This field controls the DMA's initiator ID on the system bus when the channel is active. **NOTE** The ID captured in this register reflects the initiator ID of the core or other initiator writing the channel's control attributes, which are in the lower byte of TCD*n*_CSR.

15.6.2.6 Channel Priority (CH0_PRI - CH31_PRI)

Offset

Register	Offset
CH0_PRI	10h
CH1_PRI	4010h
CH2_PRI	8010h
CH3_PRI	C010h
CH4_PRI	1_0010h
CH5_PRI	1_4010h
CH6_PRI	1_8010h
CH7_PRI	1_C010h
CH8_PRI	2_0010h
CH9_PRI	2_4010h
CH10_PRI	2_8010h
CH11_PRI	2_C010h
CH12_PRI	20_0010h
CH13_PRI	20_4010h
CH14_PRI	20_8010h
CH15_PRI	20_C010h
CH16_PRI	21_0010h
CH17_PRI	21_4010h
CH18_PRI	21_8010h
CH19_PRI	21_C010h
CH20_PRI	22_0010h
CH21_PRI	22_4010h
CH22_PRI	22_8010h
CH23_PRI	22_C010h
CH24_PRI	23_0010h
CH25_PRI	23_4010h
CH26_PRI	23_8010h
CH27_PRI	23_C010h
CH28_PRI	24_0010h
CH29_PRI	24_4010h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
CH30_PRI	24_8010h
CH31_PRI	24_C010h

Function

The contents of these registers define unique priorities associated with each channel within the same channel group. Channel grouping is programmed via Channel Arbitration Group (CH0_GRPRI - CH31_GRPRI).

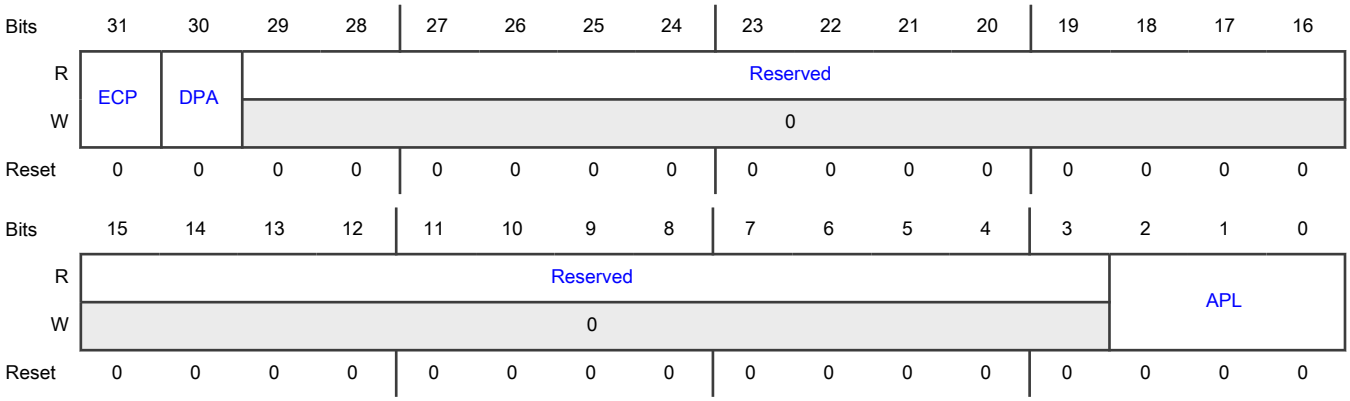
The channel priorities within a group are evaluated by numeric value; for example, 0 is the lowest priority, 1 is the next higher priority, then 2, 3, and so on. Software must program the channel priorities with unique values; otherwise, channel numbers with the same, non-zero value, will be selected based on channel number with the higher channel number having higher priority.

If more than one channel in a group has an arbitration priority level value of zero, then the arbitration mode field CSR[ERCA] is used to determine the arbitration scheme for all channels with APL=0 within a group.

When you enable round-robin channel arbitration (CSR[ERCA] = 1), all channels with APL=0 within a group will use a round-robin arbitration scheme, which rotates among these channels requesting service without regard to priority. Round-robin provides a fairness mechanism within an arbitration group.

When you enable fixed-priority channel arbitration (CSR[ERCA] = 0), eDMA selects channels with APL=0 based on channel number, with the higher channel number having higher priority.

Diagram



Fields

Field	Function
31 ECP	Enable Channel Preemption 0b - Channel cannot be suspended by a higher-priority channel's service request 1b - Channel can be temporarily suspended by a higher-priority channel's service request
30 DPA	Disable Preempt Ability 0b - Channel can suspend a lower-priority channel 1b - Channel cannot suspend any other channel, regardless of channel priority

Table continues on the next page...

Table continued from the previous page...

Field	Function
29-3 —	Reserved
2-0 APL	Arbitration Priority Level Channel priority level for arbitration within the assigned arbitration group.

15.6.2.7 TCD Source Address (TCD0_SADDR - TCD31_SADDR)

Offset

Register	Offset
TCD0_SADDR	20h
TCD1_SADDR	4020h
TCD2_SADDR	8020h
TCD3_SADDR	C020h
TCD4_SADDR	1_0020h
TCD5_SADDR	1_4020h
TCD6_SADDR	1_8020h
TCD7_SADDR	1_C020h
TCD8_SADDR	2_0020h
TCD9_SADDR	2_4020h
TCD10_SADDR	2_8020h
TCD11_SADDR	2_C020h
TCD12_SADDR	20_0020h
TCD13_SADDR	20_4020h
TCD14_SADDR	20_8020h
TCD15_SADDR	20_C020h
TCD16_SADDR	21_0020h
TCD17_SADDR	21_4020h
TCD18_SADDR	21_8020h
TCD19_SADDR	21_C020h
TCD20_SADDR	22_0020h
TCD21_SADDR	22_4020h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
TCD22_SADDR	22_8020h
TCD23_SADDR	22_C020h
TCD24_SADDR	23_0020h
TCD25_SADDR	23_4020h
TCD26_SADDR	23_8020h
TCD27_SADDR	23_C020h
TCD28_SADDR	24_0020h
TCD29_SADDR	24_4020h
TCD30_SADDR	24_8020h
TCD31_SADDR	24_C020h

Function

This register contains the address for the read transactions.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SADDR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SADDR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 SADDR	Source Address Memory address pointing to the source data.

15.6.2.8 TCD Signed Source Address Offset (TCD0_SOFF - TCD31_SOFF)

Offset

Register	Offset
TCD0_SOFF	24h
TCD1_SOFF	4024h
TCD2_SOFF	8024h
TCD3_SOFF	C024h
TCD4_SOFF	1_0024h
TCD5_SOFF	1_4024h
TCD6_SOFF	1_8024h
TCD7_SOFF	1_C024h
TCD8_SOFF	2_0024h
TCD9_SOFF	2_4024h
TCD10_SOFF	2_8024h
TCD11_SOFF	2_C024h
TCD12_SOFF	20_0024h
TCD13_SOFF	20_4024h
TCD14_SOFF	20_8024h
TCD15_SOFF	20_C024h
TCD16_SOFF	21_0024h
TCD17_SOFF	21_4024h
TCD18_SOFF	21_8024h
TCD19_SOFF	21_C024h
TCD20_SOFF	22_0024h
TCD21_SOFF	22_4024h
TCD22_SOFF	22_8024h
TCD23_SOFF	22_C024h
TCD24_SOFF	23_0024h
TCD25_SOFF	23_4024h
TCD26_SOFF	23_8024h
TCD27_SOFF	23_C024h
TCD28_SOFF	24_0024h
TCD29_SOFF	24_4024h

Table continues on the next page...

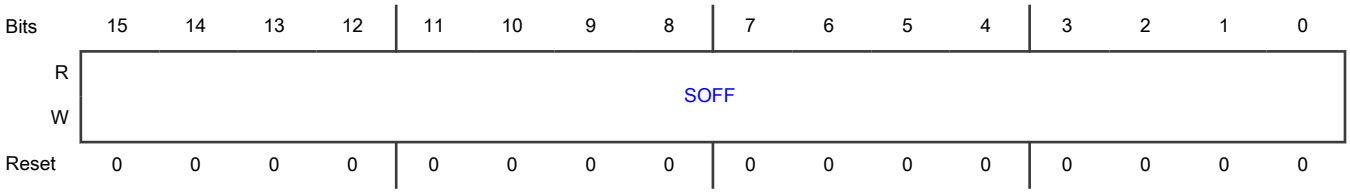
Table continued from the previous page...

Register	Offset
TCD30_SOFF	24_8024h
TCD31_SOFF	24_C024h

Function

This register contains the sign-extended value added to Source Address register after each read transaction.

Diagram



Fields

Field	Function
15-0 SOFF	Source Address Signed Offset Sign-extended offset applied to the current source address to form the next-state value as each source read is completed.

15.6.2.9 TCD Transfer Attributes (TCD0_ATTR - TCD31_ATTR)

Offset

Register	Offset
TCD0_ATTR	26h
TCD1_ATTR	4026h
TCD2_ATTR	8026h
TCD3_ATTR	C026h
TCD4_ATTR	1_0026h
TCD5_ATTR	1_4026h
TCD6_ATTR	1_8026h
TCD7_ATTR	1_C026h
TCD8_ATTR	2_0026h
TCD9_ATTR	2_4026h
TCD10_ATTR	2_8026h

Table continues on the next page...

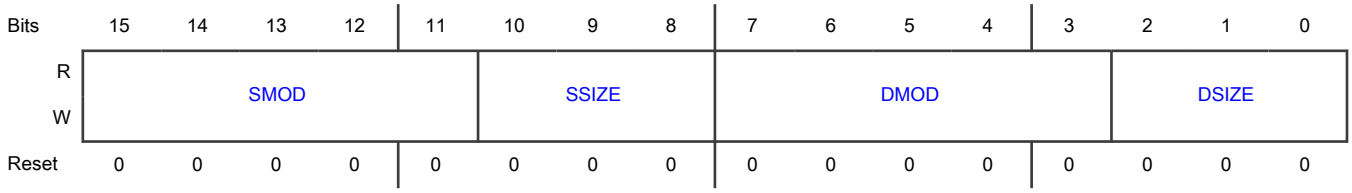
Table continued from the previous page...

Register	Offset
TCD11_ATTR	2_C026h
TCD12_ATTR	20_0026h
TCD13_ATTR	20_4026h
TCD14_ATTR	20_8026h
TCD15_ATTR	20_C026h
TCD16_ATTR	21_0026h
TCD17_ATTR	21_4026h
TCD18_ATTR	21_8026h
TCD19_ATTR	21_C026h
TCD20_ATTR	22_0026h
TCD21_ATTR	22_4026h
TCD22_ATTR	22_8026h
TCD23_ATTR	22_C026h
TCD24_ATTR	23_0026h
TCD25_ATTR	23_4026h
TCD26_ATTR	23_8026h
TCD27_ATTR	23_C026h
TCD28_ATTR	24_0026h
TCD29_ATTR	24_4026h
TCD30_ATTR	24_8026h
TCD31_ATTR	24_C026h

Function

This register contains size and option modulo addressing information for source and destination addresses.

Diagram



Fields

Field	Function
15-11 SMOD	Source Address Modulo This field defines a specific address range, which is the value after the SADDR + SOFF calculation is performed on the original register value. Setting this field makes it easy to implement a circular data queue. For data queues requiring power-of-2-sized bytes, the queue must start at a 0-modulo-size address and the SMOD field must be set to the appropriate value for the queue, freezing the required number of upper address bits. The value programmed into this field specifies the number of lower address bits that are allowed to change. For a circular queue application, you typically set TCDn_SOFF[SOFF] to the transfer size to implement post-increment addressing, with the SMOD function constraining the addresses to a 0-modulo-size range. 0_0000b - Source address modulo feature disabled 0_0001b - Source address modulo feature enabled for any non-zero value [1-31]
10-8 SSIZE	Source Data Transfer Size 000b - 8-bit 001b - 16-bit 010b - 32-bit 011b - 64-bit 100b - 16-byte 101b - 32-byte 110b - 64-byte 111b - Reserved
7-3 DMOD	Destination Address Modulo See the SMOD definition.
2-0 DSIZE	Destination Data Transfer Size See the SSIZE definition.

15.6.2.10 TCD Transfer Size Without Minor Loop Offsets (TCD0_NBYTES_MLOFFNO - TCD31_NBYTES_MLOFFNO)

Offset

Register	Offset
TCD0_NBYTES_MLOFFNO	28h
TCD1_NBYTES_MLOFFNO	4028h
TCD2_NBYTES_MLOFFNO	8028h
TCD3_NBYTES_MLOFFNO	C028h
TCD4_NBYTES_MLOFFNO	1_0028h
TCD5_NBYTES_MLOFFNO	1_4028h
TCD6_NBYTES_MLOFFNO	1_8028h
TCD7_NBYTES_MLOFFNO	1_C028h
TCD8_NBYTES_MLOFFNO	2_0028h
TCD9_NBYTES_MLOFFNO	2_4028h
TCD10_NBYTES_MLOFFNO	2_8028h
TCD11_NBYTES_MLOFFNO	2_C028h
TCD12_NBYTES_MLOFFNO	20_0028h
TCD13_NBYTES_MLOFFNO	20_4028h
TCD14_NBYTES_MLOFFNO	20_8028h
TCD15_NBYTES_MLOFFNO	20_C028h
TCD16_NBYTES_MLOFFNO	21_0028h
TCD17_NBYTES_MLOFFNO	21_4028h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
TCD18_NBYTES_MLOFFNO	21_8028h
TCD19_NBYTES_MLOFFNO	21_C028h
TCD20_NBYTES_MLOFFNO	22_0028h
TCD21_NBYTES_MLOFFNO	22_4028h
TCD22_NBYTES_MLOFFNO	22_8028h
TCD23_NBYTES_MLOFFNO	22_C028h
TCD24_NBYTES_MLOFFNO	23_0028h
TCD25_NBYTES_MLOFFNO	23_4028h
TCD26_NBYTES_MLOFFNO	23_8028h
TCD27_NBYTES_MLOFFNO	23_C028h
TCD28_NBYTES_MLOFFNO	24_0028h
TCD29_NBYTES_MLOFFNO	24_4028h
TCD30_NBYTES_MLOFFNO	24_8028h
TCD31_NBYTES_MLOFFNO	24_C028h

Function

The TCDn_NBYTES field defines the number of bytes to transfer per service request.

Minor loop offsets are address offset values added to the final source address (TCDn_SADDR), or destination address (TCDn_DADDR), upon minor loop completion. Minor loop completion is when the channel has finished the service request and has transferred NBYTES. When minor loop offsets are enabled, the minor loop offset value (TCDn_NBYTES_MLOFFYES[MLOFF]) is added to the final source address (TCDn_SADDR), to the final destination address (TCDn_DADDR), or to both, prior to the addresses being written back to the TCD. If the major loop is complete, the minor loop offset is ignored and the major loop address offsets (TCDn_SLAST_SDA and TCDn_DLAST_SGA) are used to compute the next TCDn_SADDR and TCDn_DADDR values.

When minor loop mapping is enabled (SMLOE or DMLOE is 1), TCDn_NBYTES_MLOFFNO/TCDn_NBYTES_MLOFFYES is redefined. A portion of TCDn_NBYTES_MLOFFNO/TCDn_NBYTES_MLOFFYES is used to specify multiple fields:

- A source enable bit (SMLOE) to specify the minor loop offset must be applied to the source address (TCDn_SADDR) upon minor loop completion
- A destination enable bit (DMLOE) to specify the minor loop offset must be applied to the destination address (TCDn_DADDR) upon minor loop completion
- The sign extended minor loop offset value (MLOFF)

The same offset value (MLOFF) is used for both source and destination minor loop offsets. When either minor loop offset is enabled (SMLOE set or DMLOE set), the NBYTES field is reduced to 10 bits. If both minor loop offsets are disabled (SMLOE cleared and DMLOE cleared), the NBYTES field is a 30-bit vector.

One of two register profiles (this register or TCDn_NBYTES_MLOFFYES), defines the number of bytes to transfer per request. Which register to use depends on whether source or destination minor loop mapping is enabled.

TCDn_NBYTES_MLOFFNO/TCDn_NBYTES_MLOFFYES is defined as follows:

- If SMLOE = 0 and DMLOE = 0, then see the TCDn_NBYTES_MLOFFNO register description.
- If either SMLOE or DMLOE is 1, then see the TCDn_NBYTES_MLOFFYES register description.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SMLOE	DMLOE	NBYTES													
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	NBYTES															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SMLOE	Source Minor Loop Offset Enable Selects whether the minor loop offset is applied to the source address upon minor loop completion. 0b - Minor loop offset not applied to SADDR 1b - Minor loop offset applied to SADDR
30 DMLOE	Destination Minor Loop Offset Enable Selects whether the minor loop offset is applied to the destination address upon minor loop completion. 0b - Minor loop offset not applied to DADDR 1b - Minor loop offset applied to DADDR
29-0 NBYTES	Number of Bytes To Transfer Per Service Request Number of bytes to be transferred for each service request of the channel.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When a channel activates, the module loads the appropriate TCD contents into the eDMA engine and performs the appropriate reads and writes until the byte transfer count has been reached. This process is normally an indivisible operation and cannot be halted. It can, however, be stalled by using the bandwidth control field, or via preemption. After the byte count is exhausted, the SADDR and DADDR values are written back into the TCD memory, and the major loop iteration count (CITER) is decremented by one and written back to the TCD memory. If the major iteration count is complete, additional processing is performed.

15.6.2.11 TCD Transfer Size with Minor Loop Offsets (TCD0_NBYTES_MLOFFYES - TCD31_NBYTES_MLOFFYES)

Offset

Register	Offset
TCD0_NBYTES_MLOFFYES	28h
TCD1_NBYTES_MLOFFYES	4028h
TCD2_NBYTES_MLOFFYES	8028h
TCD3_NBYTES_MLOFFYES	C028h
TCD4_NBYTES_MLOFFYES	1_0028h
TCD5_NBYTES_MLOFFYES	1_4028h
TCD6_NBYTES_MLOFFYES	1_8028h
TCD7_NBYTES_MLOFFYES	1_C028h
TCD8_NBYTES_MLOFFYES	2_0028h
TCD9_NBYTES_MLOFFYES	2_4028h
TCD10_NBYTES_MLOFFYES	2_8028h
TCD11_NBYTES_MLOFFYES	2_C028h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
TCD12_NBYTES_MLOFFYES	20_0028h
TCD13_NBYTES_MLOFFYES	20_4028h
TCD14_NBYTES_MLOFFYES	20_8028h
TCD15_NBYTES_MLOFFYES	20_C028h
TCD16_NBYTES_MLOFFYES	21_0028h
TCD17_NBYTES_MLOFFYES	21_4028h
TCD18_NBYTES_MLOFFYES	21_8028h
TCD19_NBYTES_MLOFFYES	21_C028h
TCD20_NBYTES_MLOFFYES	22_0028h
TCD21_NBYTES_MLOFFYES	22_4028h
TCD22_NBYTES_MLOFFYES	22_8028h
TCD23_NBYTES_MLOFFYES	22_C028h
TCD24_NBYTES_MLOFFYES	23_0028h
TCD25_NBYTES_MLOFFYES	23_4028h
TCD26_NBYTES_MLOFFYES	23_8028h
TCD27_NBYTES_MLOFFYES	23_C028h
TCD28_NBYTES_MLOFFYES	24_0028h
TCD29_NBYTES_MLOFFYES	24_4028h
TCD30_NBYTES_MLOFFYES	24_8028h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
TCD31_NBYTES_MLOFFYES	24_C028h

Function

The TCDn_NBYTES field defines the number of bytes to transfer per service request.

Minor loop offset is an address offset value added to the final source address (TCDn_SADDR) or destination address (TCDn_DADDR) upon minor loop completion. Minor loop completion occurs when the channel has finished the service request and has transferred NBYTES. Minor loop offsets are enabled by setting either the source enable bit (SMLOE) or the destination enable bit (DMLOE).

The source enable bit (SMLOE) specifies the minor loop offset value (MLOFF) that is to be applied to the source address (TCDn_SADDR) upon minor loop completion. The destination enable bit (DMLOE) specifies the minor loop offset (MLOFF) that is to be applied to the destination address (TCDn_DADDR) upon minor loop completion.

If the major loop is complete, the minor loop offsets are ignored and the major loop address offsets (TCDn_SLAST_SDA and TCDn_DLAST_SGA) are used to compute the next TCDn_SADDR and TCDn_DADDR values.

When you enable the minor loop offset overlay (either SMLOE or DMLOE is 1), eDMA redefines TCDn_NBYTES_MLOFFNO/TCDn_NBYTES_MLOFFYES. A portion of TCDn_NBYTES_MLOFFNO/TCDn_NBYTES_MLOFFYES specifies the sign-extended minor loop offset value (MLOFF). The same offset value (MLOFF) applies to both source and destination minor loop offsets. When the minor loop offset is enabled, you must align it to the transfer size of the source or destination it is associated with. When either minor loop offset is enabled (SMLOE set or DMLOE set), the NBYTES field is reduced to 10 bits. If both minor loop offsets are disabled (SMLOE cleared and DMLOE cleared), the NBYTES field is a 30-bit vector.

One of two register profiles (this register or TCDn_NBYTES_MLOFFNO) defines the number of bytes to transfer per request. Which register to use depends on whether source or destination minor loop mapping is enabled.

TCDn_NBYTES_MLOFFYES is defined as follows:

- If either minor loop offset is enabled (SMLOE or DMLOE = 1), then see the TCDn_NBYTES_MLOFFYES register description.
- If SMLOE and DMLOE are both 0, then see the TCDn_NBYTES_MLOFFNO register description.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SMLOE	DMLOE	MLOFF													
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MLOFF								NBYTES							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SMLOE	Source Minor Loop Offset Enable Selects whether the minor loop offset is applied to the source address upon minor loop completion. 0b - Minor loop offset not applied to SADDR 1b - Minor loop offset applied to SADDR
30 DMLOE	Destination Minor Loop Offset Enable Selects whether the minor loop offset is applied to the destination address upon minor loop completion. 0b - Minor loop offset not applied to DADDR 1b - Minor loop offset applied to DADDR
29-10 MLOFF	Minor Loop Offset If SMLOE or DMLOE is 1, this field represents a sign-extended offset applied to the source or destination address to form the next-state value after the minor loop completes.
9-0 NBYTES	Number of Bytes To Transfer Per Service Request The number of bytes to be transferred in each service request of the channel. As a channel activates, the module loads the appropriate TCD contents into the eDMA engine and performs the appropriate reads and writes until the minor byte transfer count has been reached. This is an indivisible operation and cannot be halted. It can, however, be stalled by using the bandwidth control field, or via preemption. After the minor count is exhausted, the SADDR and DADDR values are written back into the TCD memory, and the major iteration count is decremented and restored to the TCD memory. If the major iteration count is complete, additional processing is performed.

15.6.2.12 TCD Last Source Address Adjustment / Store DADDR Address (TCD0_SLAST_SDA - TCD31_SLAST_SDA)

Offset

Register	Offset
TCD0_SLAST_SDA	2Ch
TCD1_SLAST_SDA	402Ch
TCD2_SLAST_SDA	802Ch
TCD3_SLAST_SDA	C02Ch
TCD4_SLAST_SDA	1_002Ch
TCD5_SLAST_SDA	1_402Ch
TCD6_SLAST_SDA	1_802Ch
TCD7_SLAST_SDA	1_C02Ch

Table continues on the next page...

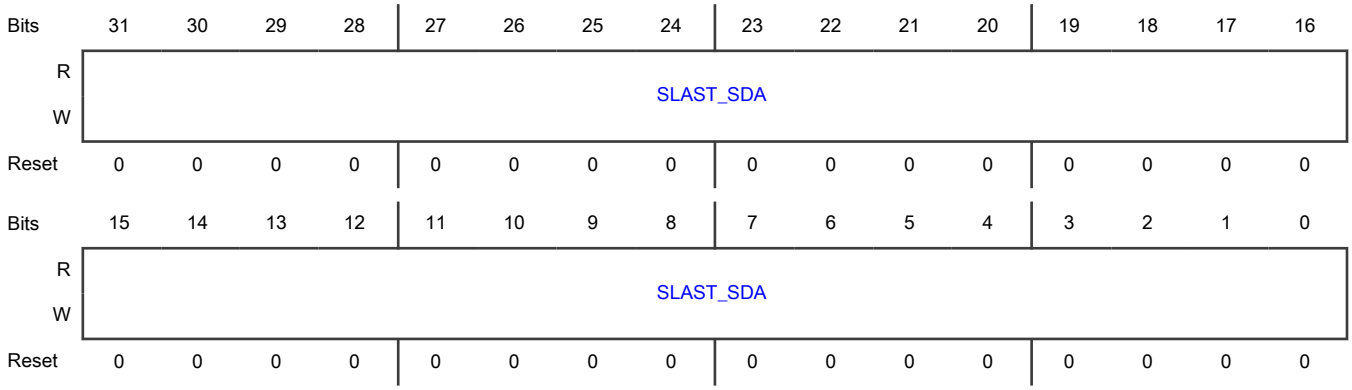
Table continued from the previous page...

Register	Offset
TCD8_SLAST_SDA	2_002Ch
TCD9_SLAST_SDA	2_402Ch
TCD10_SLAST_SDA	2_802Ch
TCD11_SLAST_SDA	2_C02Ch
TCD12_SLAST_SDA	20_002Ch
TCD13_SLAST_SDA	20_402Ch
TCD14_SLAST_SDA	20_802Ch
TCD15_SLAST_SDA	20_C02Ch
TCD16_SLAST_SDA	21_002Ch
TCD17_SLAST_SDA	21_402Ch
TCD18_SLAST_SDA	21_802Ch
TCD19_SLAST_SDA	21_C02Ch
TCD20_SLAST_SDA	22_002Ch
TCD21_SLAST_SDA	22_402Ch
TCD22_SLAST_SDA	22_802Ch
TCD23_SLAST_SDA	22_C02Ch
TCD24_SLAST_SDA	23_002Ch
TCD25_SLAST_SDA	23_402Ch
TCD26_SLAST_SDA	23_802Ch
TCD27_SLAST_SDA	23_C02Ch
TCD28_SLAST_SDA	24_002Ch
TCD29_SLAST_SDA	24_402Ch
TCD30_SLAST_SDA	24_802Ch
TCD31_SLAST_SDA	24_C02Ch

Function

This register contains the value added to the source address when the major loop is complete. When the store destination address option is enabled, this field provides a pointer to memory for storing the final destination address.

Diagram



Fields

Field	Function
31-0 SLAST_SDA	Last Source Address Adjustment / Store DADDR Address Source last address adjustment or the system memory address for destination address (DADDR) storage. If (TCDn_CSR[ESDA] = 0), then: • Adjustment value is added to the source address at the completion of the major iteration count. This value can be used to restore the source address to the initial value or adjust the address to reference the next data structure. • This field uses two's complement notation for the final source address adjustment. Otherwise: • This address points to the 32-bit-aligned memory location where the destination address (DADDR) is to be stored in system memory. By saving the final destination address in system memory via the ESDA feature, you are able to compute the size of a variable destination data buffer by simply subtracting the beginning DADDR from the final, saved DADDR. This feature is used together with the scatter/gather operation to prevent the loss of the final DADDR, which is overwritten during the scatter/gather operation. The "Store Destination Address" (SDA) value must be a 32-bit-aligned location because the eDMA forces the lower two address bits of the SLAST_SDA field to zero when ESDA is enabled. The module performs this write operation when the major loop is done; that is, when the major iteration count (CITER) decrements to zero.

15.6.2.13 TCD Destination Address (TCD0_DADDR - TCD31_DADDR)

Offset

Register	Offset
TCD0_DADDR	30h
TCD1_DADDR	4030h
TCD2_DADDR	8030h

Table continues on the next page...

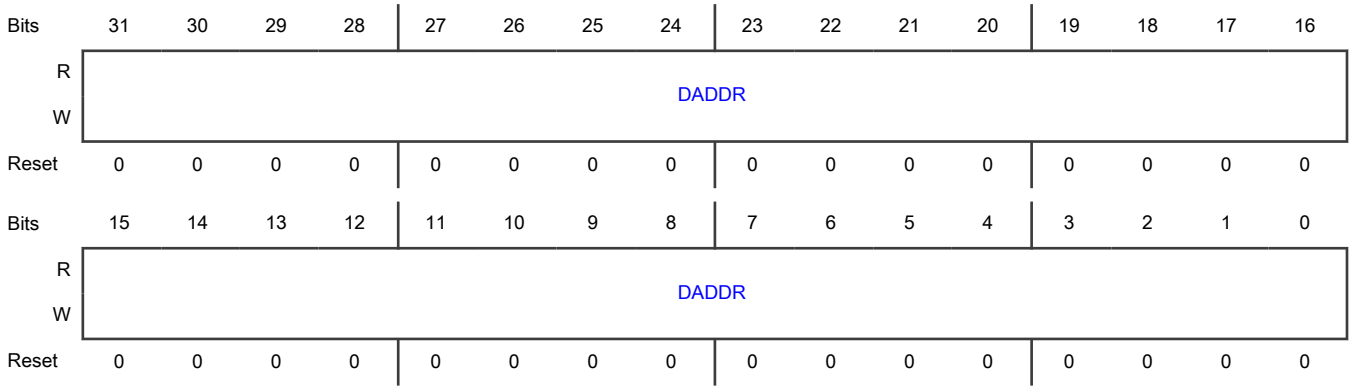
Table continued from the previous page...

Register	Offset
TCD3_DADDR	C030h
TCD4_DADDR	1_0030h
TCD5_DADDR	1_4030h
TCD6_DADDR	1_8030h
TCD7_DADDR	1_C030h
TCD8_DADDR	2_0030h
TCD9_DADDR	2_4030h
TCD10_DADDR	2_8030h
TCD11_DADDR	2_C030h
TCD12_DADDR	20_0030h
TCD13_DADDR	20_4030h
TCD14_DADDR	20_8030h
TCD15_DADDR	20_C030h
TCD16_DADDR	21_0030h
TCD17_DADDR	21_4030h
TCD18_DADDR	21_8030h
TCD19_DADDR	21_C030h
TCD20_DADDR	22_0030h
TCD21_DADDR	22_4030h
TCD22_DADDR	22_8030h
TCD23_DADDR	22_C030h
TCD24_DADDR	23_0030h
TCD25_DADDR	23_4030h
TCD26_DADDR	23_8030h
TCD27_DADDR	23_C030h
TCD28_DADDR	24_0030h
TCD29_DADDR	24_4030h
TCD30_DADDR	24_8030h
TCD31_DADDR	24_C030h

Function

This register contains the address for the write transactions.

Diagram



Fields

Field	Function
31-0 DADDR	Destination Address Memory address pointing to the destination data.

15.6.2.14 TCD Signed Destination Address Offset (TCD0_DOFF - TCD31_DOFF)

Offset

Register	Offset
TCD0_DOFF	34h
TCD1_DOFF	4034h
TCD2_DOFF	8034h
TCD3_DOFF	C034h
TCD4_DOFF	1_0034h
TCD5_DOFF	1_4034h
TCD6_DOFF	1_8034h
TCD7_DOFF	1_C034h
TCD8_DOFF	2_0034h
TCD9_DOFF	2_4034h
TCD10_DOFF	2_8034h
TCD11_DOFF	2_C034h
TCD12_DOFF	20_0034h
TCD13_DOFF	20_4034h
TCD14_DOFF	20_8034h

Table continues on the next page...

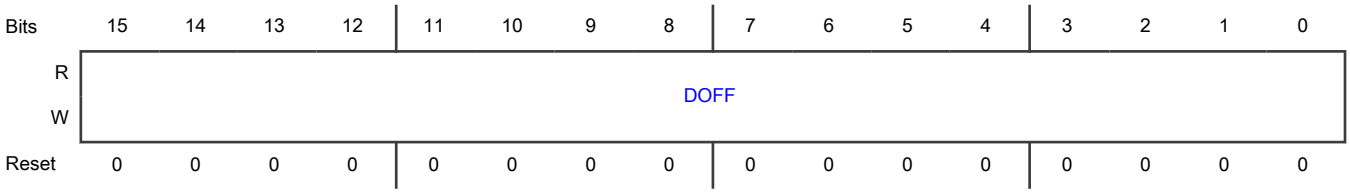
Table continued from the previous page...

Register	Offset
TCD15_DOFF	20_C034h
TCD16_DOFF	21_0034h
TCD17_DOFF	21_4034h
TCD18_DOFF	21_8034h
TCD19_DOFF	21_C034h
TCD20_DOFF	22_0034h
TCD21_DOFF	22_4034h
TCD22_DOFF	22_8034h
TCD23_DOFF	22_C034h
TCD24_DOFF	23_0034h
TCD25_DOFF	23_4034h
TCD26_DOFF	23_8034h
TCD27_DOFF	23_C034h
TCD28_DOFF	24_0034h
TCD29_DOFF	24_4034h
TCD30_DOFF	24_8034h
TCD31_DOFF	24_C034h

Function

This register contains the sign-extended value added to Destination Address register after each write transaction.

Diagram



Fields

Field	Function
15-0 DOFF	Destination Address Signed Offset Sign-extended offset that is applied to the current destination address to form the next-state value as each destination write is completed.

15.6.2.15 TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD0_CITER_ELINKNO - TCD31_CITER_ELINKNO)

Offset

Register	Offset
TCD0_CITER_ELINKNO	36h
TCD1_CITER_ELINKNO	4036h
TCD2_CITER_ELINKNO	8036h
TCD3_CITER_ELINKNO	C036h
TCD4_CITER_ELINKNO	1_0036h
TCD5_CITER_ELINKNO	1_4036h
TCD6_CITER_ELINKNO	1_8036h
TCD7_CITER_ELINKNO	1_C036h
TCD8_CITER_ELINKNO	2_0036h
TCD9_CITER_ELINKNO	2_4036h
TCD10_CITER_ELINKNO	2_8036h
TCD11_CITER_ELINKNO	2_C036h
TCD12_CITER_ELINKNO	20_0036h
TCD13_CITER_ELINKNO	20_4036h
TCD14_CITER_ELINKNO	20_8036h
TCD15_CITER_ELINKNO	20_C036h
TCD16_CITER_ELINKNO	21_0036h
TCD17_CITER_ELINKNO	21_4036h
TCD18_CITER_ELINKNO	21_8036h
TCD19_CITER_ELINKNO	21_C036h
TCD20_CITER_ELINKNO	22_0036h
TCD21_CITER_ELINKNO	22_4036h

Table continues on the next page...

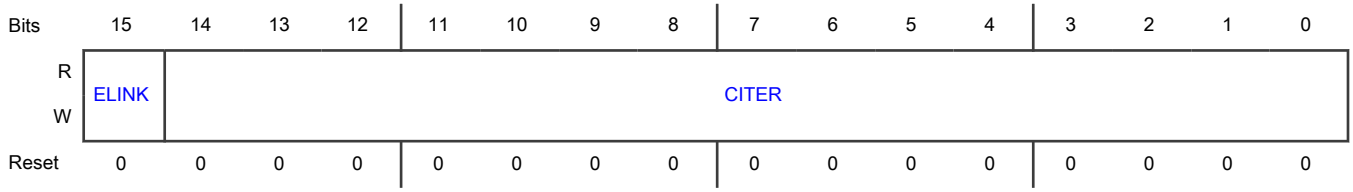
Table continued from the previous page...

Register	Offset
TCD22_CITER_ELINKNO	22_8036h
TCD23_CITER_ELINKNO	22_C036h
TCD24_CITER_ELINKNO	23_0036h
TCD25_CITER_ELINKNO	23_4036h
TCD26_CITER_ELINKNO	23_8036h
TCD27_CITER_ELINKNO	23_C036h
TCD28_CITER_ELINKNO	24_0036h
TCD29_CITER_ELINKNO	24_4036h
TCD30_CITER_ELINKNO	24_8036h
TCD31_CITER_ELINKNO	24_C036h

Function

If TCDn_CITER[ELINK] is 0, the TCDn_CITER register is defined as follows.

Diagram



Fields

Field	Function
15 ELINK	Enable Link As the channel completes the minor loop, this flag enables linking to another channel as defined by the relevant LINKCH field. The link target channel initiates a channel service request via an internal mechanism that sets the TCDn_CSR[START] bit of the specified channel to 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	If channel linking is disabled, the CITER value is extended to 15 bits in place of a link channel number. If the major loop is exhausted, this link mechanism is suppressed in favor of MAJORELINK channel linking. **NOTE** This field must be equal to the BITER[ELINK] field; otherwise, a configuration error is reported. 0b - Channel-to-channel linking disabled 1b - Channel-to-channel linking enabled
14-0 CITER	Current Major Iteration Count This 9-bit (ELINK = 1) or 15-bit (ELINK = 0) count represents the current major loop count for the channel. It is decremented each time the channel finishes a service request and is written back to TCD memory. After the major iteration count is exhausted, the channel performs a number of operations — for example, final source and destination address calculations — and optionally generates an interrupt to signal channel completion before reloading the CITER field from the Beginning Iteration Count (BITER) field. **NOTE** When the CITER field is initially loaded by software, it must be set to the same value as that contained in the BITER field. **NOTE** If the channel is configured to execute a single service request, the initial values of BITER and CITER should be 0x0001.

15.6.2.16 TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD0_CITER_ELINKYES - TCD31_CITER_ELINKYES)

Offset

Register	Offset
TCD0_CITER_ELINKYES	36h
TCD1_CITER_ELINKYES	4036h
TCD2_CITER_ELINKYES	8036h
TCD3_CITER_ELINKYES	C036h
TCD4_CITER_ELINKYES	1_0036h
TCD5_CITER_ELINKYES	1_4036h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
TCD6_CITER_ELINKYES	1_8036h
TCD7_CITER_ELINKYES	1_C036h
TCD8_CITER_ELINKYES	2_0036h
TCD9_CITER_ELINKYES	2_4036h
TCD10_CITER_ELINKYES	2_8036h
TCD11_CITER_ELINKYES	2_C036h
TCD12_CITER_ELINKYES	20_0036h
TCD13_CITER_ELINKYES	20_4036h
TCD14_CITER_ELINKYES	20_8036h
TCD15_CITER_ELINKYES	20_C036h
TCD16_CITER_ELINKYES	21_0036h
TCD17_CITER_ELINKYES	21_4036h
TCD18_CITER_ELINKYES	21_8036h
TCD19_CITER_ELINKYES	21_C036h
TCD20_CITER_ELINKYES	22_0036h
TCD21_CITER_ELINKYES	22_4036h
TCD22_CITER_ELINKYES	22_8036h
TCD23_CITER_ELINKYES	22_C036h
TCD24_CITER_ELINKYES	23_0036h

Table continues on the next page...

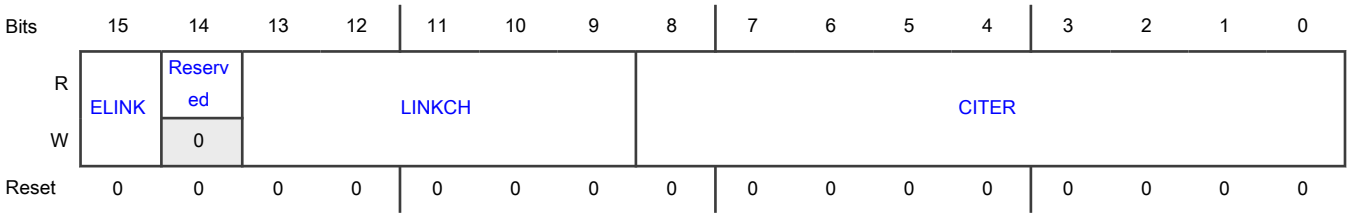
Table continued from the previous page...

Register	Offset
TCD25_CITER_ELINKYES	23_4036h
TCD26_CITER_ELINKYES	23_8036h
TCD27_CITER_ELINKYES	23_C036h
TCD28_CITER_ELINKYES	24_0036h
TCD29_CITER_ELINKYES	24_4036h
TCD30_CITER_ELINKYES	24_8036h
TCD31_CITER_ELINKYES	24_C036h

Function

If TCDn_CITER[ELINK] is 1, the TCDn_CITER register is defined as follows.

Diagram



Fields

Field	Function
15 ELINK	Enable Link As the channel completes the minor loop, this flag enables linking to another channel as defined by the relevant LINKCH field. When enabled, an internal mechanism sets the TCDn_CSR[START] field of the specified channel (LINKCH) upon minor loop completion. If channel linking is disabled, the CITER value is extended to 15 bits in place of a link channel number. If the major loop is exhausted, this link mechanism is suppressed in favor of MAJORELINK channel linking. **NOTE** This field must be equal to the BITER[ELINK] field; otherwise, a configuration error is reported.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Channel-to-channel linking disabled 1b - Channel-to-channel linking enabled
14 —	Reserved
13-9 LINKCH	Minor Loop Link Channel Number If channel-to-channel linking is enabled (ELINK = 1), then after the minor loop is exhausted the eDMA engine initiates a channel service request to the channel defined by this field by writing that channel's TCDn_CSR[START] field to 1.
8-0 CITER	Current Major Iteration Count This 9-bit (ELINK = 1) or 15-bit (ELINK = 0) count represents the current major loop count for the channel. It is decremented each time the channel finishes a service request and is written back to the TCD memory. After the major iteration count is exhausted, the channel performs a number of operations — for example, final source and destination address calculations — and optionally generates an interrupt to signal channel completion before reloading the CITER field from the Beginning Iteration Count (BITER) field. **NOTE** When the CITER field is initially loaded by software, it must be set to the same value as that contained in the BITER field. **NOTE** If the channel is configured to execute a single service request, the initial values of BITER and CITER should be 0x0001.

15.6.2.17 TCD Last Destination Address Adjustment / Scatter Gather Address (TCD0_DLAST_SGA - TCD31_DLAST_SGA)

Offset

Register	Offset
TCD0_DLAST_SGA	38h
TCD1_DLAST_SGA	4038h
TCD2_DLAST_SGA	8038h
TCD3_DLAST_SGA	C038h
TCD4_DLAST_SGA	1_0038h
TCD5_DLAST_SGA	1_4038h
TCD6_DLAST_SGA	1_8038h
TCD7_DLAST_SGA	1_C038h

Table continues on the next page...

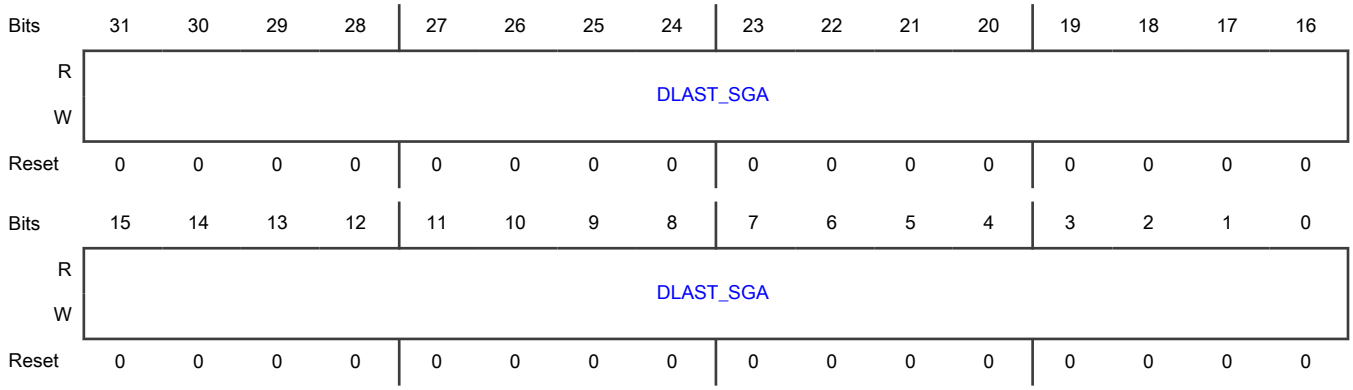
Table continued from the previous page...

Register	Offset
TCD8_DLAST_SGA	2_0038h
TCD9_DLAST_SGA	2_4038h
TCD10_DLAST_SGA	2_8038h
TCD11_DLAST_SGA	2_C038h
TCD12_DLAST_SGA	20_0038h
TCD13_DLAST_SGA	20_4038h
TCD14_DLAST_SGA	20_8038h
TCD15_DLAST_SGA	20_C038h
TCD16_DLAST_SGA	21_0038h
TCD17_DLAST_SGA	21_4038h
TCD18_DLAST_SGA	21_8038h
TCD19_DLAST_SGA	21_C038h
TCD20_DLAST_SGA	22_0038h
TCD21_DLAST_SGA	22_4038h
TCD22_DLAST_SGA	22_8038h
TCD23_DLAST_SGA	22_C038h
TCD24_DLAST_SGA	23_0038h
TCD25_DLAST_SGA	23_4038h
TCD26_DLAST_SGA	23_8038h
TCD27_DLAST_SGA	23_C038h
TCD28_DLAST_SGA	24_0038h
TCD29_DLAST_SGA	24_4038h
TCD30_DLAST_SGA	24_8038h
TCD31_DLAST_SGA	24_C038h

Function

This register contains the value added to the destination address when the major loop is complete. When the Scatter/Gather option is enabled, this field provides a pointer to memory for fetching a transfer control descriptor to reprogram the channel.

Diagram



Fields

Field	Function
31-0 DLAST_SGA	Last Destination Address Adjustment / Scatter Gather Address Adjustment of the last destination address or the memory address for the next transfer control descriptor to be loaded into this channel (scatter/gather). If (TCDn_CSR[ESG] = 0) then: • Adjustment value is added to the destination address at the completion of the major iteration count. This value can apply to restore the destination address to the initial value or adjust the address to reference the next data structure. • This field uses two's complement notation for the final destination address adjustment. Otherwise: • This address points to the beginning of a 0-modulo 32-byte region containing the next transfer control descriptor to be loaded into this channel. This channel reload is performed as the major iteration count completes. The scatter/gather address must be 0-modulo 32-byte, or else a configuration error is reported.

15.6.2.18 TCD Control and Status (TCD0_CSR - TCD31_CSR)

Offset

Register	Offset
TCD0_CSR	3Ch
TCD1_CSR	403Ch
TCD2_CSR	803Ch
TCD3_CSR	C03Ch
TCD4_CSR	1_003Ch
TCD5_CSR	1_403Ch
TCD6_CSR	1_803Ch

Table continues on the next page...

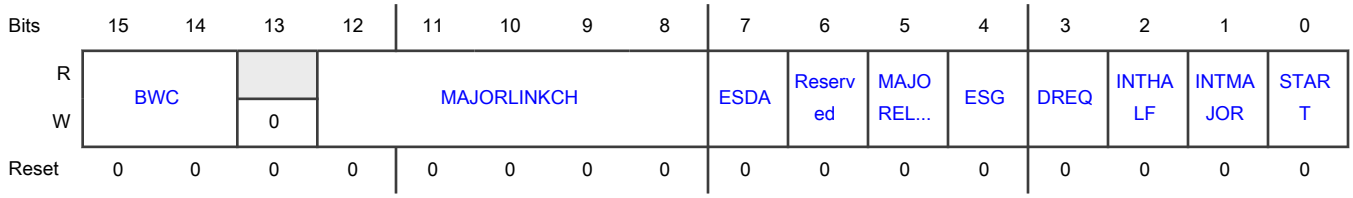
Table continued from the previous page...

Register	Offset
TCD7_CSR	1_C03Ch
TCD8_CSR	2_003Ch
TCD9_CSR	2_403Ch
TCD10_CSR	2_803Ch
TCD11_CSR	2_C03Ch
TCD12_CSR	20_003Ch
TCD13_CSR	20_403Ch
TCD14_CSR	20_803Ch
TCD15_CSR	20_C03Ch
TCD16_CSR	21_003Ch
TCD17_CSR	21_403Ch
TCD18_CSR	21_803Ch
TCD19_CSR	21_C03Ch
TCD20_CSR	22_003Ch
TCD21_CSR	22_403Ch
TCD22_CSR	22_803Ch
TCD23_CSR	22_C03Ch
TCD24_CSR	23_003Ch
TCD25_CSR	23_403Ch
TCD26_CSR	23_803Ch
TCD27_CSR	23_C03Ch
TCD28_CSR	24_003Ch
TCD29_CSR	24_403Ch
TCD30_CSR	24_803Ch
TCD31_CSR	24_C03Ch

Function

This register is used to enable optional features.

Diagram



Fields

Field	Function
15-14 BWC	Bandwidth Control Throttles the amount of bus bandwidth consumed by the eDMA. Generally, as the eDMA processes the minor loop, it continuously generates read/write sequences until the minor count is exhausted. This field forces eDMA to stall after the completion of each read/write access, to control the bus request bandwidth seen by the system bus interconnect. **NOTE** If the source and destination sizes are equal, this field is ignored between the first and second transfers and after the last write of each minor loop. This behavior is a side effect of reducing start-up latency. 00b - No eDMA engine stalls 01b - Enable eDMA initiator high-priority elevation (HPE) mode. No eDMA engine stalls. 10b - eDMA engine stalls for 4 cycles after each R/W 11b - eDMA engine stalls for 8 cycles after each R/W
13 —	Reserved
12-8 MAJORLINKCH	Major Loop Link Channel Number If (MAJORELINK = 0) then: • No channel-to-channel linking, or chaining, is performed after the major loop counter is exhausted. Otherwise: • After the major loop counter is exhausted, the eDMA engine initiates a channel service request at the channel defined by this field by setting that channel's TCDn_CSR[START] field to 1.
7 ESDA	Enable Store Destination Address As the channel completes the major loop by either the current iteration counter (CITER) decrementing to 0, this field enables writing the destination address (DADDR) to the address stored in the SLAST_SDA field. The value written to system memory is the last DADDR value prior to the DLAST_SGA offset being applied, or overwritten by an enabled scatter/gather operation. When the SDA bit is 1, SLAST_SDA contains the write pointer instead of the final source address offset. Because this is a pointer and not a final offset, a last source address offset of zero is applied to SADDR instead of the SLAST_SGA value. 0b - Ability to store destination address to system memory disabled 1b - Ability to store destination address to system memory enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
6 —	Reserved
5 MAJORELINK	Enable Link When Major Loop Complete As the channel completes the major loop, this flag enables linking to another channel defined by MAJORLINKCH. The link target channel initiates a channel service request via an internal mechanism that sets the TCDn_CSR[START] field of the specified channel. **NOTE** To support the dynamic linking coherency model, this field is forced to 0 if written when TCDn_CSR[DONE] is 1. 0b - Channel-to-channel linking disabled 1b - Channel-to-channel linking enabled
4 ESG	Enable Scatter/Gather Processing As the channel completes the major loop, this flag enables scatter/gather processing in the current channel. If enabled, the eDMA engine uses TCDn_DLAST_SGA as a memory pointer to a 0-modulo 32-bit address containing a 32-byte data structure, which is loaded as the transfer control descriptor into local memory. **NOTE** To support the dynamic scatter/gather coherency model, this field is forced to 0 if written when TCDn_CSR[DONE] is 1. 0b - Current channel's TCD is normal format 1b - Current channel's TCD specifies scatter/gather format.
3 DREQ	Disable Request If this flag is 1, the eDMA hardware automatically clears the corresponding ERQ bit when the current major iteration count reaches 0. 0b - No operation. Channel's ERQ field not affected 1b - Clear the ERQ field to 0 upon major loop completion, thus disabling hardware service requests. Channel's ERQ field cleared to 0 when major loop complete
2 INTHALF	Enable Interrupt If Major Counter Half-complete If this flag is 1, the channel generates an interrupt request by setting the appropriate field in the INT register to 1 when the current major iteration count reaches the halfway point. Specifically, the comparison performed by the eDMA engine is (CITER = (BITER/2)). This halfway point interrupt request is provided to support double-buffered, also known as ping-pong, schemes, or other types of data movement where the processor needs an early indication of the transfer's progress. **NOTE** If BITER = 1, do not use INTHALF; use INTMAJOR instead. 0b - Halfway point interrupt disabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Halfway point interrupt enabled
1 INTMAJOR	Enable Interrupt If Major count complete If this flag is 1, the channel generates an interrupt request by setting the appropriate field in the INT register to 1 when the current major iteration count (CITER) reaches 0. 0b - End-of-major loop interrupt disabled 1b - End-of-major loop interrupt enabled
0 START	Channel Start If this flag is 1, the channel is requesting service. The eDMA hardware automatically clears this flag to 0 after the channel begins execution. 0b - Channel not explicitly started 1b - Channel explicitly started via a software-initiated service request

15.6.2.19 TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD0_BITER_ELINKNO - TCD31_BITER_ELINKNO)

Offset

Register	Offset
TCD0_BITER_ELINKNO	3Eh
TCD1_BITER_ELINKNO	403Eh
TCD2_BITER_ELINKNO	803Eh
TCD3_BITER_ELINKNO	C03Eh
TCD4_BITER_ELINKNO	1_003Eh
TCD5_BITER_ELINKNO	1_403Eh
TCD6_BITER_ELINKNO	1_803Eh
TCD7_BITER_ELINKNO	1_C03Eh
TCD8_BITER_ELINKNO	2_003Eh
TCD9_BITER_ELINKNO	2_403Eh
TCD10_BITER_ELINKNO	2_803Eh
TCD11_BITER_ELINKNO	2_C03Eh
TCD12_BITER_ELINKNO	20_003Eh

Table continues on the next page...

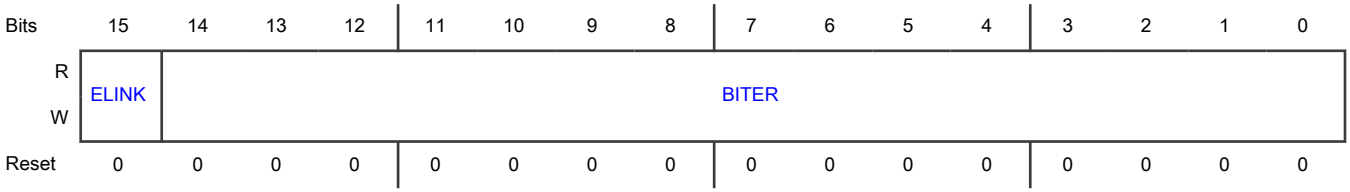
Table continued from the previous page...

Register	Offset
TCD13_BITER_ELINKNO	20_403Eh
TCD14_BITER_ELINKNO	20_803Eh
TCD15_BITER_ELINKNO	20_C03Eh
TCD16_BITER_ELINKNO	21_003Eh
TCD17_BITER_ELINKNO	21_403Eh
TCD18_BITER_ELINKNO	21_803Eh
TCD19_BITER_ELINKNO	21_C03Eh
TCD20_BITER_ELINKNO	22_003Eh
TCD21_BITER_ELINKNO	22_403Eh
TCD22_BITER_ELINKNO	22_803Eh
TCD23_BITER_ELINKNO	22_C03Eh
TCD24_BITER_ELINKNO	23_003Eh
TCD25_BITER_ELINKNO	23_403Eh
TCD26_BITER_ELINKNO	23_803Eh
TCD27_BITER_ELINKNO	23_C03Eh
TCD28_BITER_ELINKNO	24_003Eh
TCD29_BITER_ELINKNO	24_403Eh
TCD30_BITER_ELINKNO	24_803Eh
TCD31_BITER_ELINKNO	24_C03Eh

Function

If the TCDn_BITER[ELINK] field is 0, the TCDn_BITER register is defined as follows.

Diagram



Fields

Field	Function
15 ELINK	Enables Link As the channel completes the minor loop, this flag enables linking to another channel as defined by BITER[LINKCH]. The link target channel initiates a channel service request via an internal mechanism that sets the TCDn_CSR[START] field of the specified channel. If channel linking is disabled, the BITER value extends to 15 bits in place of a link channel number. If the major loop is exhausted, this link mechanism is suppressed in favor of the MAJORELINK channel linking. **NOTE** When the software loads the TCD, this field must be set equal to the corresponding CITER field; otherwise, a configuration error is reported. As the major iteration count is exhausted, eDMA reloads the contents of this field into the CITER field. 0b - Channel-to-channel linking disabled 1b - Channel-to-channel linking enabled
14-0 BITER	Starting Major Iteration Count As the transfer control descriptor is first loaded by software, this 9-bit (ELINK = 1) or 15-bit (ELINK = 0) field must be set equal to the value in the CITER field. As the major iteration count is exhausted, eDMA reloads the contents of this field into the CITER field. If the channel is configured to execute a single service request, the initial values of BITER and CITER must be 0x0001.

15.6.2.20 TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD0_BITER_ELINKYES - TCD31_BITER_ELINKYES)

Offset

Register	Offset
TCD0_BITER_ELINKYES	3Eh
TCD1_BITER_ELINKYES	403Eh

Table continues on the next page...

Table continued from the previous page...

Register	Offset
TCD2_BITER_ELINKYES	803Eh
TCD3_BITER_ELINKYES	C03Eh
TCD4_BITER_ELINKYES	1_003Eh
TCD5_BITER_ELINKYES	1_403Eh
TCD6_BITER_ELINKYES	1_803Eh
TCD7_BITER_ELINKYES	1_C03Eh
TCD8_BITER_ELINKYES	2_003Eh
TCD9_BITER_ELINKYES	2_403Eh
TCD10_BITER_ELINKYES	2_803Eh
TCD11_BITER_ELINKYES	2_C03Eh
TCD12_BITER_ELINKYES	20_003Eh
TCD13_BITER_ELINKYES	20_403Eh
TCD14_BITER_ELINKYES	20_803Eh
TCD15_BITER_ELINKYES	20_C03Eh
TCD16_BITER_ELINKYES	21_003Eh
TCD17_BITER_ELINKYES	21_403Eh
TCD18_BITER_ELINKYES	21_803Eh
TCD19_BITER_ELINKYES	21_C03Eh
TCD20_BITER_ELINKYES	22_003Eh

Table continues on the next page...

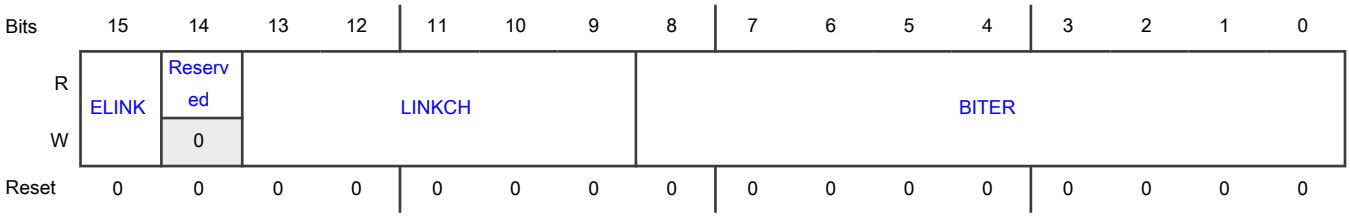
Table continued from the previous page...

Register	Offset
TCD21_BITER_ELINKYES	22_403Eh
TCD22_BITER_ELINKYES	22_803Eh
TCD23_BITER_ELINKYES	22_C03Eh
TCD24_BITER_ELINKYES	23_003Eh
TCD25_BITER_ELINKYES	23_403Eh
TCD26_BITER_ELINKYES	23_803Eh
TCD27_BITER_ELINKYES	23_C03Eh
TCD28_BITER_ELINKYES	24_003Eh
TCD29_BITER_ELINKYES	24_403Eh
TCD30_BITER_ELINKYES	24_803Eh
TCD31_BITER_ELINKYES	24_C03Eh

Function

If the TCDn_BITER[ELINK] field is set, the TCDn_BITER register is defined as follows.

Diagram



Fields

Field	Function
15 ELINK	Enable Link

Table continues on the next page...

Table continued from the previous page...

Field	Function
	As the channel completes the minor loop, this flag enables linking to another channel as defined by BITER[LINKCH]. The link target channel initiates a channel service request via an internal mechanism that sets the TCDn_CSR[START] field of the specified channel. If channel linking disables, the BITER value extends to 15 bits in place of a link channel number. If the major loop is exhausted, this link mechanism is suppressed in favor of the MAJORELINK channel linking. **NOTE** When the software loads the TCD, this field must be set equal to the corresponding CITER field; otherwise, a configuration error is reported. As the major iteration count is exhausted, eDMA reloads the contents of this field into the CITER field. 0b - Channel-to-channel linking disabled 1b - Channel-to-channel linking enabled
14 —	Reserved
13-9 LINKCH	Link Channel Number If channel-to-channel linking is enabled (ELINK = 1), then after the minor loop is exhausted, the eDMA engine initiates a channel service request at the channel defined by this field by setting that channel's TCDn_CSR[START] field. **NOTE** When the software loads the TCD, this field must be set equal to the corresponding CITER field; otherwise, a configuration error is reported. As the major iteration count is exhausted, eDMA reloads the contents of this field into the CITER field.
8-0 BITER	Starting Major Iteration Count As the transfer control descriptor is first loaded by software, this 9-bit (ELINK = 1) or 15-bit (ELINK = 0) field must be set equal to the value in the CITER field. As the major iteration count is exhausted, eDMA reloads the contents of this field into the CITER field. If the channel is configured to execute a single service request, the initial values of BITER and CITER must be 0x0001.

15.7 Glossary

TCD Transfer control descriptor

Chapter 16
Interrupt Monitor (INTM)

16.1 Overview

INTM provides a mechanism to monitor the latency of the responses on interrupt requests to ensure that the processing of these critical interrupts executes within the expected time frame, increasing the reliability of the device.

16.1.1 Block diagram

The following block diagram shows the major registers involved in monitoring the interrupt sources.

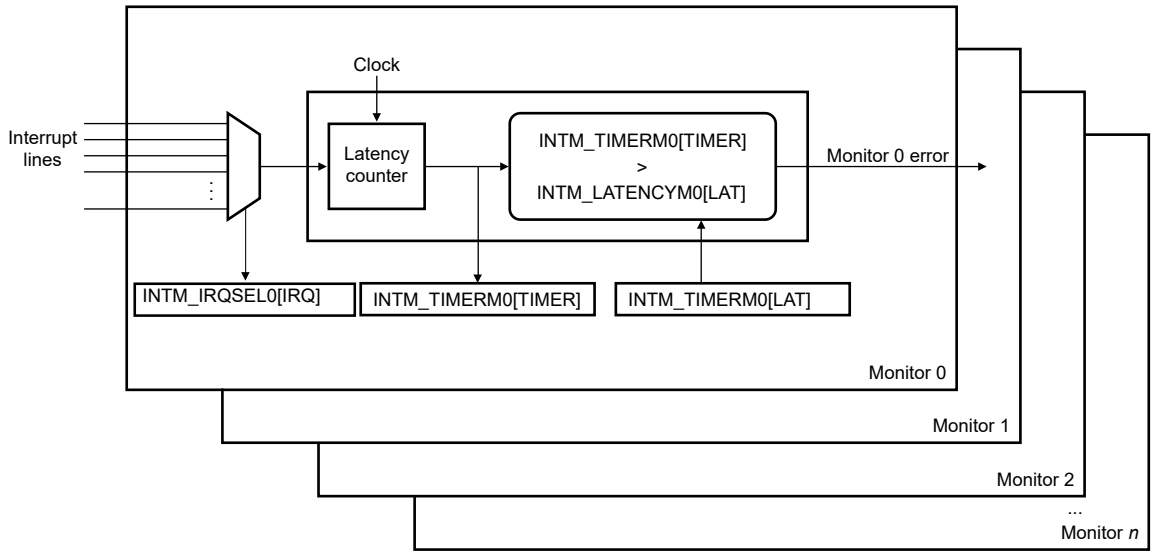


Figure 42. INTM block diagram

16.1.2 Features

INTM supports the following features:

- 4 programmable interrupt monitors
- Programmable interrupt source per monitor
- Programmable 24-bit maximum latency counter per monitor
- Timer expired status bit per monitor
- One interrupt acknowledge for all monitors

16.2 Functional description

Interrupt Request Select for Monitor a (INTM_IRQSEL0 - INTM_IRQSEL3) provides selection of the source, Monitor Mode (INTM_MM) enables the monitor, Interrupt Acknowledge (INTM_IACK) captures the event when the interrupt is acknowledged, Interrupt Latency for Monitor a (INTM_LATENCY0 - INTM_LATENCY3) can be programmed to trigger a monitor error when a threshold value is exceeded, Status for Monitor a (INTM_STATUS0 - INTM_STATUS3) is available to read the monitor error over peripheral bus.

NOTE

INTM supports only level-type interrupts. Hence, the pulse-type interrupts are converted into level-type interrupts for INTM.

16.2.1 Clocking

This module has no clocking considerations.

16.2.2 Interrupts

This module has no interrupts.

16.3 External signals

This module has no external signals.

16.4 Initialization

To monitor a subset of interrupt sources:

1. Program Interrupt Request Select for Monitor a (INTM_IRQSEL0 - INTM_IRQSEL3) with a value corresponding to the interrupt source number to be monitored.
   - You can monitor only defined interrupt sources.
2. Program Interrupt Latency for Monitor a (INTM_LATENCY0 - INTM_LATENCY3) to the maximum expected latency time for each monitored interrupt source.
3. Write 1 to INTM_MM[MM] to monitor the registers.
4. During the interrupt service routine, write the IRQ of the ISR to Interrupt Acknowledge (INTM_IACK).
5. If the maximum expected latency time exceeds the limit defined in Interrupt Latency for Monitor a (INTM_LATENCY0 - INTM_LATENCY3), then an error indication is sent to Fault Collection Control Unit(FCCU). You can also read Status for Monitor a (INTM_STATUS0 - INTM_STATUS3) to see this information.

To clear an error condition, write 0 to the corresponding INTM_TIMERa[TIMER] field for which the interrupt source exceeded the programmed latency value. In case the source is unknown, NXP recommends you to write 0 to all INTM_TIMERa[TIMER] fields.

16.5 INTM register descriptions

INTM allows you to set a maximum timer count for the interrupt latency from interrupt request to interrupt acknowledge. This mechanism monitors the latency of interrupt sources to ensure that these critical interrupts execute within the expected time frame, increasing the reliability of the chip. The hardware for this function is isolated in its own hierarchy to decrease the risk of fault interference.

An error indication is sent to FCCU if the interrupt processing latency exceeds the programmed threshold of the expected latency. The error indication can be read from INTM_STATUSa[STATUS].

16.5.1 INTM memory map

INTM base address: 4027_C000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Monitor Mode (INTM_MM)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
4h	Interrupt Acknowledge (INTM_IACK)	32	W	0000_0000h
8h	Interrupt Request Select for Monitor 0 (INTM_IRQSEL0)	32	RW	0000_0000h
Ch	Interrupt Latency for Monitor 0 (INTM_LATENCY0)	32	RW	0000_0000h
10h	Timer for Monitor 0 (INTM_TIMER0)	32	RW	0000_0000h
14h	Status for Monitor 0 (INTM_STATUS0)	32	R	0000_0000h
18h	Interrupt Request Select for Monitor 1 (INTM_IRQSEL1)	32	RW	0000_0000h
1Ch	Interrupt Latency for Monitor 1 (INTM_LATENCY1)	32	RW	0000_0000h
20h	Timer for Monitor 1 (INTM_TIMER1)	32	RW	0000_0000h
24h	Status for Monitor 1 (INTM_STATUS1)	32	R	0000_0000h
28h	Interrupt Request Select for Monitor 2 (INTM_IRQSEL2)	32	RW	0000_0000h
2Ch	Interrupt Latency for Monitor 2 (INTM_LATENCY2)	32	RW	0000_0000h
30h	Timer for Monitor 2 (INTM_TIMER2)	32	RW	0000_0000h
34h	Status for Monitor 2 (INTM_STATUS2)	32	R	0000_0000h
38h	Interrupt Request Select for Monitor 3 (INTM_IRQSEL3)	32	RW	0000_0000h
3Ch	Interrupt Latency for Monitor 3 (INTM_LATENCY3)	32	RW	0000_0000h
40h	Timer for Monitor 3 (INTM_TIMER3)	32	RW	0000_0000h
44h	Status for Monitor 3 (INTM_STATUS3)	32	R	0000_0000h

16.5.2 Monitor Mode (INTM_MM)

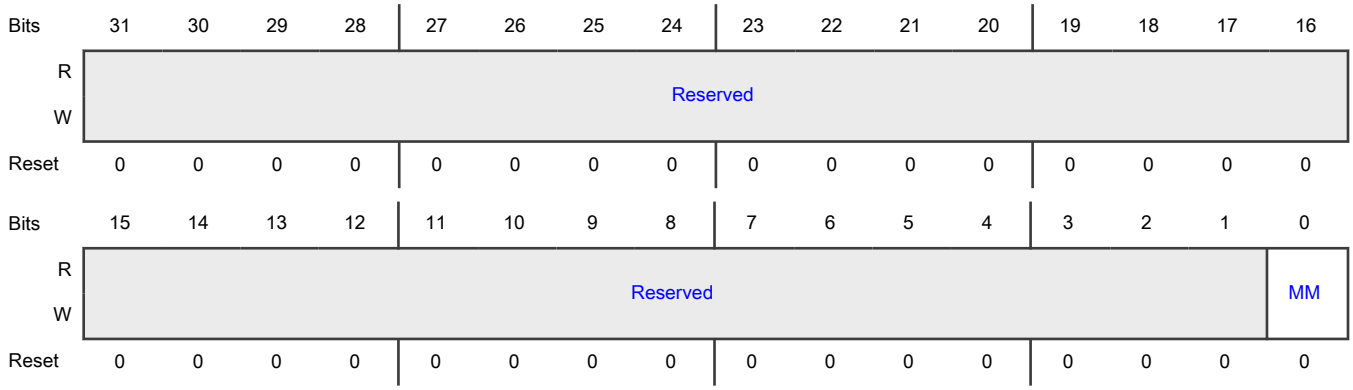
Offset

Register	Offset
INTM_MM	0h

Function

Enables the cycle count timer on a monitored interrupt request for comparison to the latency register.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 MM	Monitor Mode Controls whether the INTM monitors the latency of an interrupt response. 0b - Disable 1b - Enable

16.5.3 Interrupt Acknowledge (INTM_IACK)

Offset

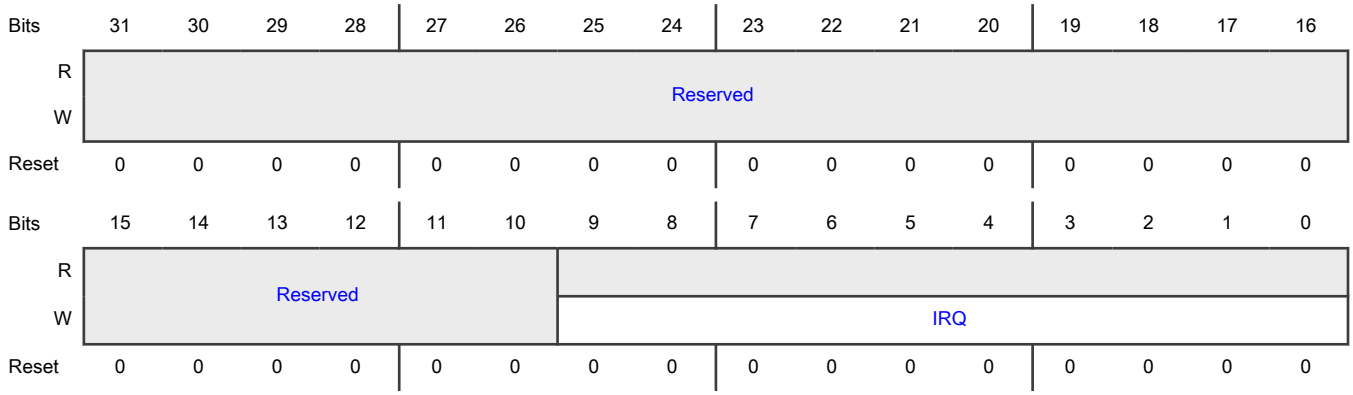
Register	Offset
INTM_IACK	4h

Function

Acknowledges the interrupt processing.

The Interrupt service routine writes to this register after the register acknowledges interrupt processing. This write operation must provide the number of the processed interrupt, which is compared with the interrupt request number in each Interrupt Request Select for Monitor a (INTM_IRQSEL0 - INTM_IRQSEL3) and stops the corresponding Timer for Monitor a (INTM_TIMER0 - INTM_TIMER3) when it found a match.

Diagram



Fields

Field	Function
31-10 —	Reserved
9-0 IRQ	Interrupt Request Specifies the interrupt request number to stop Timer for Monitor a (INTM_TIMER0 - INTM_TIMER3).

16.5.4 Interrupt Request Select for Monitor a (INTM_IRQSEL0 - INTM_IRQSEL3)

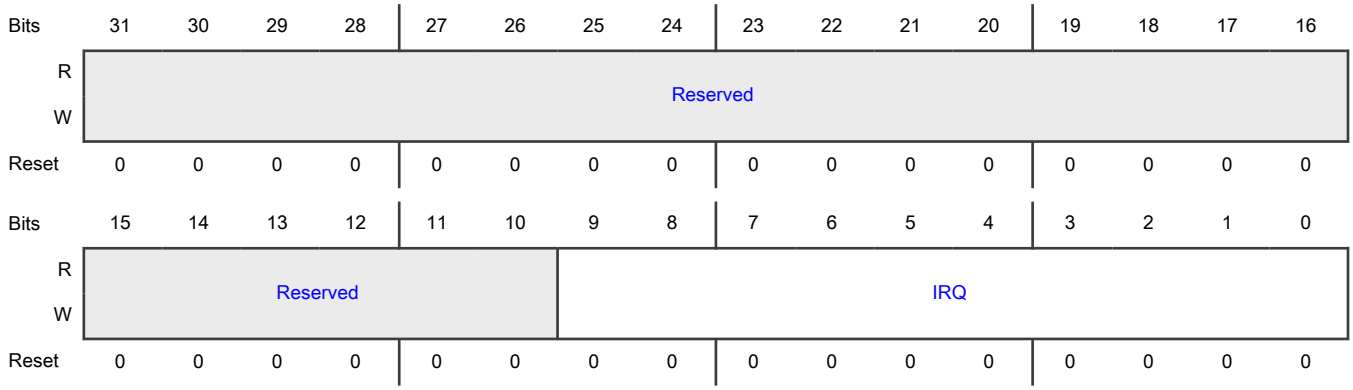
Offset

Register	Offset
INTM_IRQSEL0	8h
INTM_IRQSEL1	18h
INTM_IRQSEL2	28h
INTM_IRQSEL3	38h

Function

Indicates which interrupt request must be monitored or checked.

Diagram



Fields

Field	Function
31-10 —	Reserved
9-0 IRQ	Interrupt Request Selects the interrupt request number to monitor.

16.5.5 Interrupt Latency for Monitor a (INTM_LATENCY0 - INTM_LATENCY3)

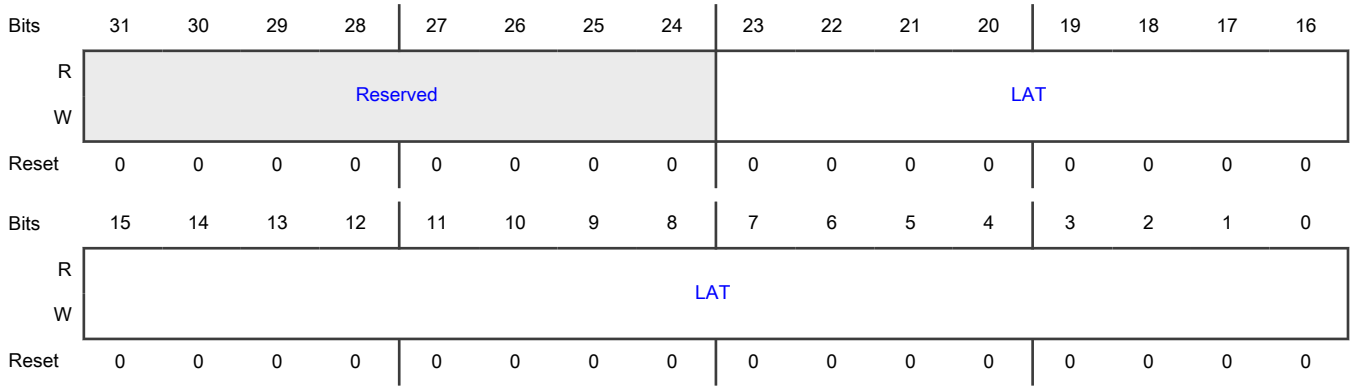
Offset

Register	Offset
INTM_LATENCY0	Ch
INTM_LATENCY1	1Ch
INTM_LATENCY2	2Ch
INTM_LATENCY3	3Ch

Function

Indicates the maximum time before an error signal is asserted for a detected interrupt request to interrupt acknowledge.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-0 LAT	Latency Specifies the maximum number of INTM clock cycles allowed for the monitored interrupt request. Maximum programmed value must not exceed FF_FFFDh to allow for proper timer error capture.

16.5.6 Timer for Monitor a (INTM_TIMER0 - INTM_TIMER3)

Offset

Register	Offset
INTM_TIMER0	10h
INTM_TIMER1	20h
INTM_TIMER2	30h
INTM_TIMER3	40h

Function

Counts the number of INTM clock cycles for a detected interrupt request to an acknowledged interrupt processing.

Initializes to 1 in following conditions:

- Interrupt Latency for Monitor a (INTM_LATENCY0 - INTM_LATENCY3) is nonzero
- Monitor error is not asserted
- Zero to one transition on the monitored interrupt request

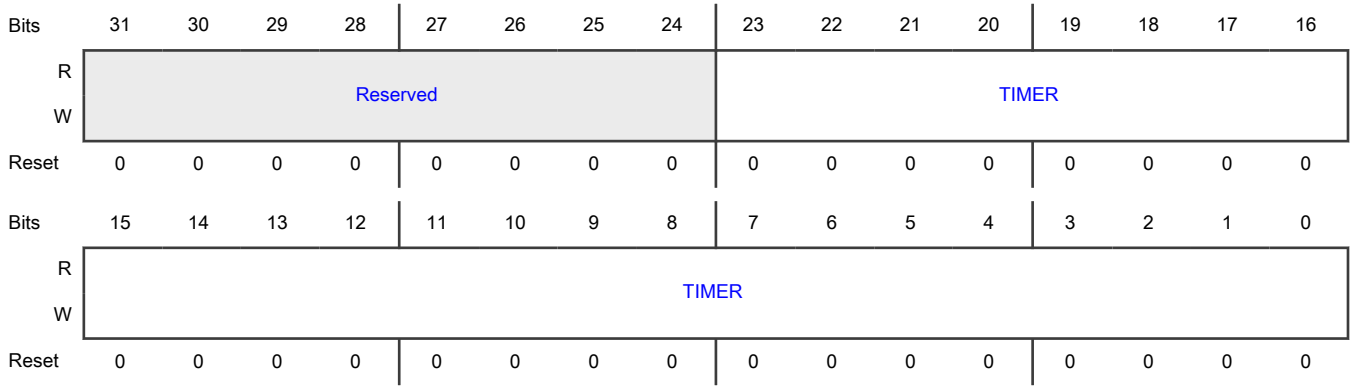
Following are the conditions to enable Timer for Monitor a (INTM_TIMER0 - INTM_TIMER3):

- INTM_MM = 1 and zero-to-one transition on the monitored interrupt source

Following are the conditions to disable Timer for Monitor a (INTM_TIMER0 - INTM_TIMER3):

- Reset
- INTM_MM = 1 and monitor error
- INTM_MM = 1 and monitored interrupt request is not asserted
- Interrupt acknowledge for corresponding interrupt request
- After the timer is enabled, writing 0 to INTM_MM[MM] has no effect.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-0 TIMER	Timer Counts the number of INTM clock cycles up to 24 bits of resolution.

16.5.7 Status for Monitor a (INTM_STATUS0 - INTM_STATUS3)

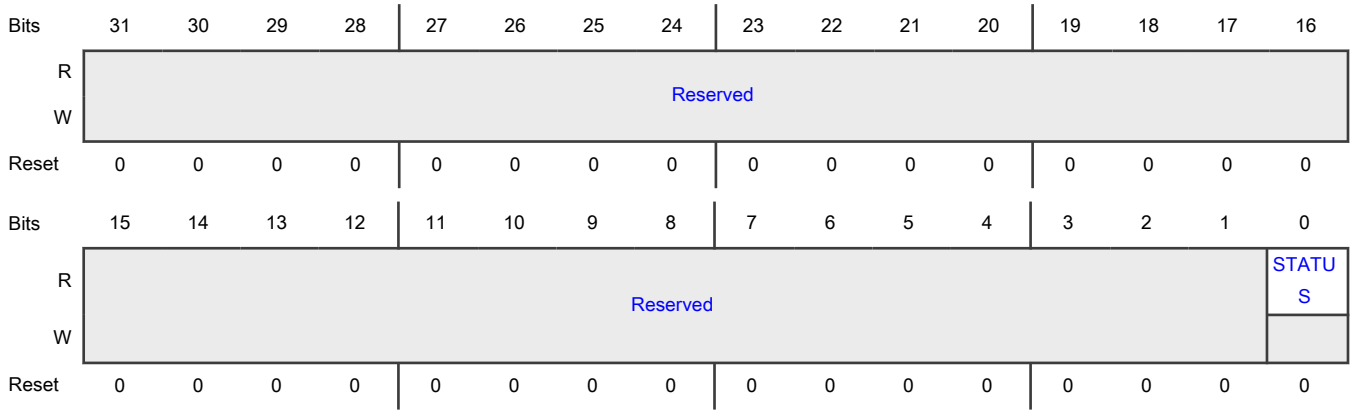
Offset

Register	Offset
INTM_STATUS0	14h
INTM_STATUS1	24h
INTM_STATUS2	34h
INTM_STATUS3	44h

Function

Defines the monitor status. Indicates whether Timer for Monitor a (INTM_TIMER0 - INTM_TIMER3) value has exceeded Interrupt Latency for Monitor a (INTM_LATENCY0 - INTM_LATENCY3) value. You can clear the monitor status by writing 0 to the corresponding INTM_TIMERa[TIMER].

Diagram



Fields

Field	Function
31-1 —	Reserved
0 STATUS	Monitor status Indicates whether Timer for Monitor a (INTM_TIMER0 - INTM_TIMER3) value has exceeded the Interrupt Latency for Monitor a (INTM_LATENCY0 - INTM_LATENCY3) value. 0b - Did not exceed 1b - Exceeded

Chapter 17
Semaphores2 (SEMA42)

17.1 Chip-specific SEMA42 information

17.1.1 SEMA42 instance and configuration

S32K3 does not support the ARM load- and store- exclusive instructions. This function is facilitated by a semaphore unit (SEMA42). This chip has one instance of SEMA42. See the following table for availability of SEMA42 instance across S32K3xx product series.

Table 64. SEMA42 instance

Instance	S32K388/S32K389/S32K358/S32K356/S32K348/S32K338/S32K328/ S32K344/S32K324/S32K314/S32K342/S32K322/S32K341	S32K310/S32K311/S32K312
SEMA42	Yes	No

See "Chip-specific XRDC information" for the domain ID of each master on the S32K3xx product series.

17.2 Overview

SEMA42 is a memory-mapped module that provides robust hardware support needed in multi-core systems for implementing semaphores and provides a simple mechanism to achieve "lock and unlock" operations via a single - write access. The hardware semaphore module provides hardware-enforced gates as well as other useful system functions related to the gating mechanisms.

NOTE

Terminology in this chapter has been updated as follows:

Table 65. Updated terms

Updated term	Deprecated term
Initiator	Master
Target	Slave
Stop	Abort

17.2.1 Block diagram

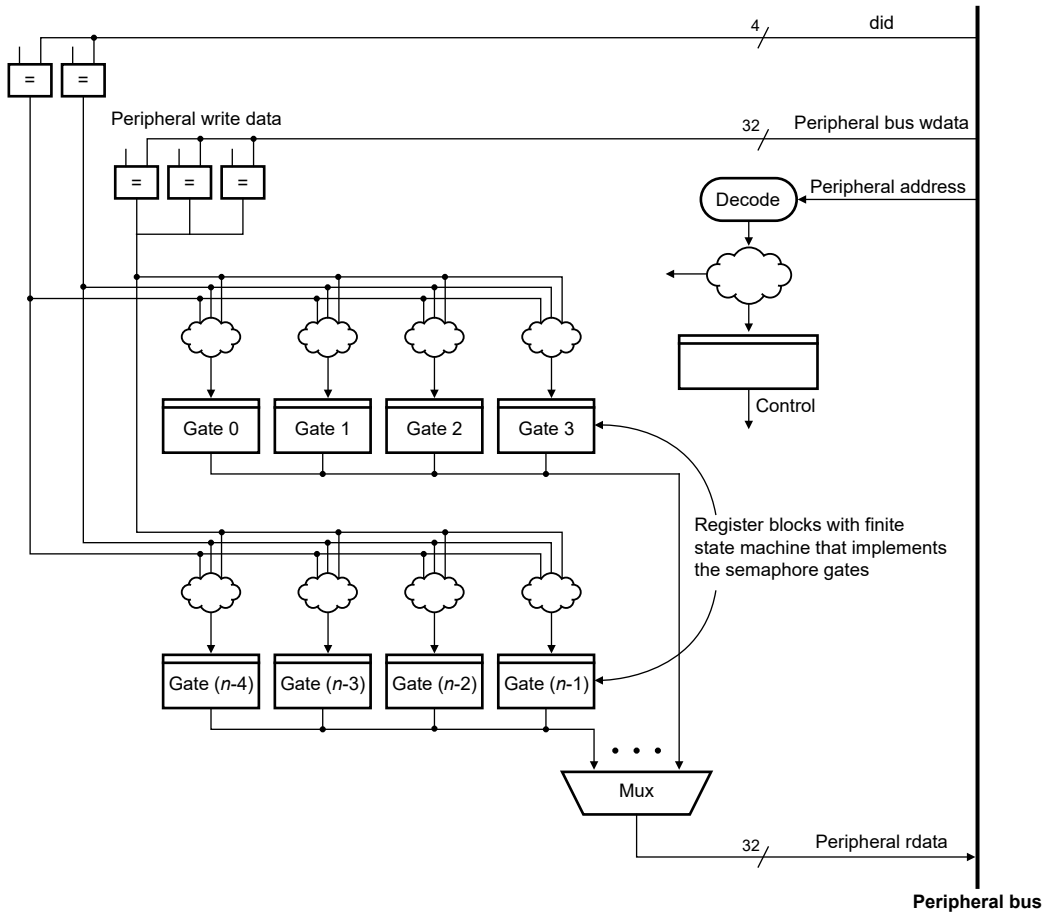


Figure 43. SEMA42 block diagram

17.2.2 Features

SEMA42 implements hardware-enforced semaphores as an IPS-mapped target peripheral device. The feature set includes:

- Support for 16 hardware-enforced gates in a multi-domain configuration that supports up to 15 domains.
- Each hardware gate appears as a 16-state, 4-bit state machine.
- Support for secure reset mechanisms to clear the contents of individual gates, as well as a clear-all capability.
- Memory-mapped target peripheral that offers programming-model accesses.

17.3 Functional description

The intent of the hardware gate implementation is to specify a protocol where the locking domain must unlock the gate. However, some systems may require a reset function to re-initialize the state of any gate(s) without requiring a system-level reset. To support this special gate reset requirement, SEMA42 implements a secure reset mechanism that allows you to initialize a hardware gate (or all the gates) by following a specific dual-write access pattern. The secure-gate reset:

- Uses a technique similar to that required for the servicing of a software watchdog timer
- Requires two consecutive writes with pre-defined data patterns from the same domain

You must do this to force the clearing of the specified gate(s). The required access pattern as follows:

1. A domain performs a 16-bit write to the RSTGT memory location. The most significant byte (RSTGT_W[RSTGDP]) must be E2h. The value of least significant byte is irrelevant for this reference and can be anything.
2. The same domain then performs a second 16-bit write to the RSTGT location. For this write, the upper byte (RSTGT_W[RSTGDP]) is the logical complement of the first data pattern (1Dh) and the lower byte (RSTGT_W[RSTGTN]) specifies the gate(s) to be reset. This gate field can specify a single gate or all gates to be cleared. If the same domain writes incorrect data on the second access or another domain performs the second write access, SEMA42 stops the special gate reset sequence and does not assert an error signal.
3. Reads of the RSTGT location return information on the 2-bit reset state machine (RSTGT_R[RSTGSM]) that implements these functions:
   - The domain performing the reset (RSTGT_R[RSTGMS])
   - The last-cleared gate number(s) (RSTGT_R[RSTGTN])

   Reads of the RSTGT register do not affect the secure-reset finite state machine in any manner.

17.3.1 Multi-core programming: software gates

Multi-processor systems require a function that can be used to safely and easily provide a locking mechanism for system software to control access to shared data structures, shared hardware resources, and so on. The software uses the gating mechanisms to serialize (and synchronize) accesses to shared data and/or resources to prevent race conditions and preserve memory coherency between different processes and domains.

Consider the following description of a typical use-case: dm*X* enters a section of code, where shared data values are to be updated. The domain must first acquire a semaphore. Think of this as the locking (or closing) of a software gate. After the gate locks, a properly-architected software system does not allow other processes (or domains) to execute the same code segment or modify the shared data structure protected by the gate. In other words, the system locks out other processes/domains. Many software implementations include a spin-wait loop within the lock function until the gate locks. After domain X obtains the lock, domain X continues execution and updates the data values protected by the particular lock. After domain X completes the updates, it unlocks (or opens) the software gate, allowing other processes/domains access to the updated data values.

A correctly-implemented system solution must follow these important rules:

- A gate variable must protect all writes to shared data values or shared hardware resources.
- After a domain locks a gate, the system must block other processes/domains from accessing the shared data or resources. Software conventions enforce this.
- The domain that locks a particular gate is the only domain that can open (unlock) that gate.

Information in the hardware gate identifying the locking domain can be extremely useful for system-level debugging.

17.3.2 16 Hardware-enforced gates

Gates appear as a 16-entry byte-size array with read and write accesses.

Domains lock gates by writing "domainID_number+1" to the appropriate gate and must read back the gate value to verify that the lock operation succeeded.

After the gate locks, the locking domain unlocks the gate by writing zeroes.

- 16-state implementation

  If gate = 0h, then state = unlocked

  if gate = 1h, then state = locked by domain (initiator) 0

  if gate = 2h, then state = locked by domain (initiator) 1

  …

  if gate = Fh, then state = locked by domain (initiator) 14

SEMA42 uses the logical domain number and the specified data patterns as reference attributes to validate all write operation.

After a gate locks, the locking domain must unlock the gate by writing zeroes.

17.3.3 State machine of the GATE*n* registers

This section describes more about the SEMA42 functional operation and specific details of the state machines of the GATE*n* registers.

As described previously, each of the GATE*n* registers implements a 4-bit, 16-state machine. The following figure shows a simplified diagram of the state transitions for each gate.

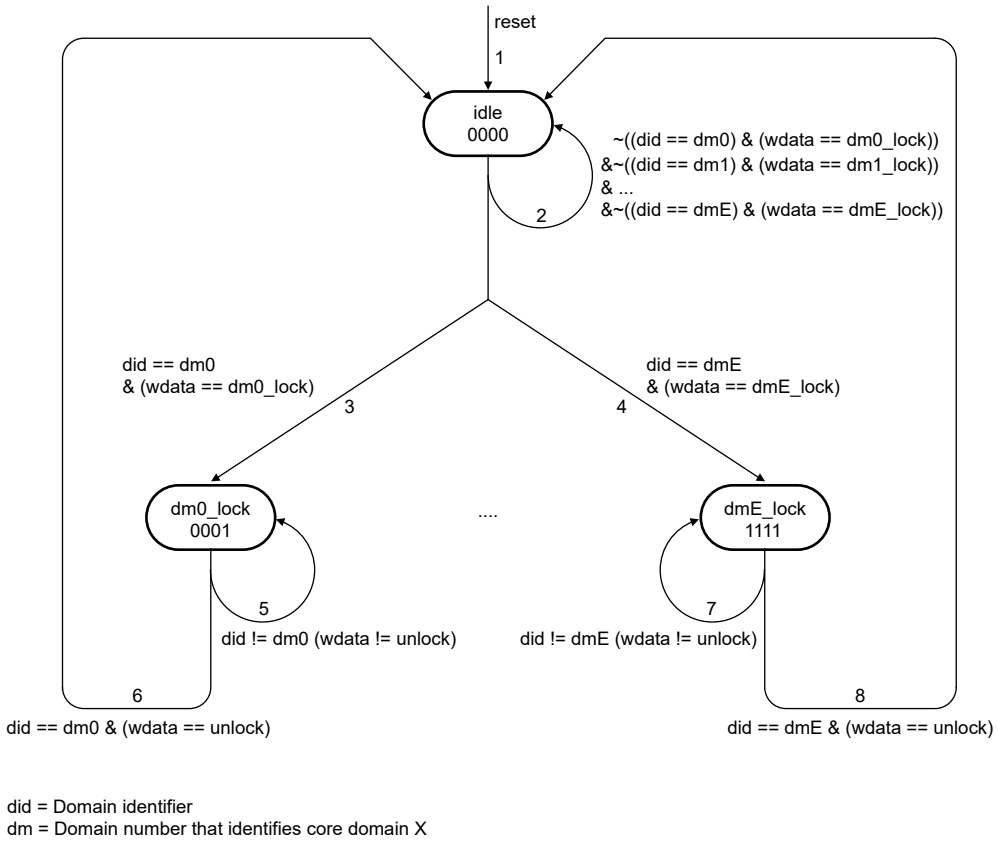


Figure 44. GATE*n* state machine

In the figure above, "dmE" represents domain 14 (E in hexadecimal). The platform passes the domain number to SEMA42.

The following table defines the GATE*n* state transitions.

Table 66. GATE*n* state transitions

Current state	Next state	Transition	Description
–	idle	1	Any reset, whether a system reset or a software-initiated gate reset, unconditionally forces the gate into the idle state.
idle	idle	2	Unless a write of the appropriate lock value from the corresponding domain occurs, the gate remains in the idle state.
idle	dm0_lock	3	When domain 0h initiates a write of the dm0_lock data value, the gate transitions into the dm0_lock state.

Table continues on the next page...

Table 66. GATE*n* state transitions (continued)

Current state	Next state	Transition	Description
idle	dmE_lock	4	When domain Eh initiates a write of the dmE_lock value, the gate transitions into the dmE_lock state.
dm0_lock	dm0_lock	5	When in this state, the gate remains here if any attempted write is not from domain 0h with the unlock data value.
dm0_lock	idle	6	The gate returns to the idle (unlocked) state after a write from domain 0h with the unlock data value occurs.
dmE_lock	dmE_lock	7	When in this state, the gate remains here if any attempted write is not from domain Eh with the unlock data value.
dmE_lock	idle	8	The gate returns to the idle (unlocked) state after a write from domain Eh with the unlock data value occurs.

SEMA42 uses these gate data values:

- The lock data value is (domain number) + 1.
- The unlock data value is 00h.

17.3.4 Clocking

Type of clock	Description
Module clock (clk)	Functions the module and controls the access to the registers.

17.3.5 Interrupts

This module has no interrupts.

17.4 External signals

This module has no external signals.

17.5 Initialization

This module does not require initialization.

17.6 Memory map/register definition

You can access these registers only in Supervisor mode. User accesses terminate with an error.

17.6.1 SEMA42 register descriptions

17.6.1.1 SEMA42 memory map

SEMA42 base address: 4046_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h - Fh	Gate (GATE0 - GATE15) [1]	8	RW	00h
42h	Reset Gate Read (RSTGT_R)	16	R	0000h
42h	Reset Gate Write (RSTGT_W)	16	W	See section

1. In this array, the index and offset values of the registers do not increment in direct alignment. For details, see the register description.

17.6.1.2 Gate (GATE0 - GATE15)

Offset

For n = 0 to 15:

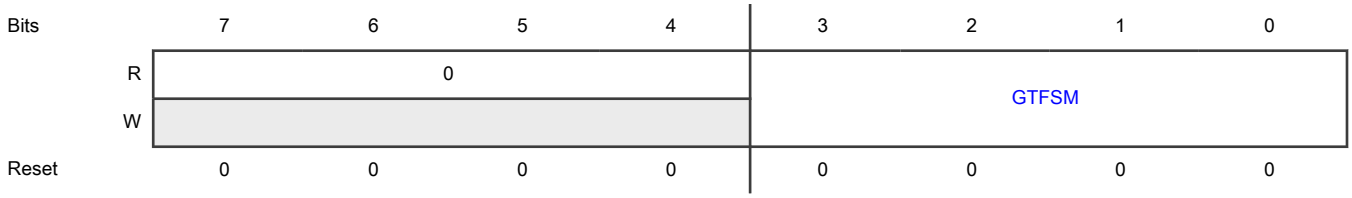
Register	Offset
GATEn	0h + (n + 3 - 2 × (n mod 4))

Function

Implements each semaphore gate in a 4-bit finite state machine, right-justified in a byte data structure. The hardware uses the logical domain-identifier number in conjunction with the data patterns to validate all attempted write operations. Only domain initiators can modify the gate registers. After a gate locks, only the locking domain must open (unlock) the gate.

You can read multiple gate values in a single access. However, you can update only a single gate at a time, via a write operation. If you attempt to write a byte-wide value that is neither the unlock value (00h) nor the appropriate lock value (domainID_number + 1), SEMA42 considers this as "no operation" and does not change any gate state. Attempts to write multiple gates in a single-aligned access with a size larger than 8 bits (byte) generate an error termination and do not allow any gate state changes.

Diagram



Fields

Field	Function
7-4 —	Reserved
3-0 GTFSM	Gate Finite State Machine Indicates the state of the gate for the last domain that locked the gate. This can be useful during system debug.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	The hardware gate has a 16-state implementation, defined as: 0000b - The gate is unlocked (free). 0001b - Domain 0 locked the gate. 0010b - Domain 1 locked the gate. 0011b - Domain 2 locked the gate. 0100b - Domain 3 locked the gate. 0101b - Domain 4 locked the gate. 0110b - Domain 5 locked the gate. 0111b - Domain 6 locked the gate. 1000b - Domain 7 locked the gate. 1001b - Domain 8 locked the gate. 1010b - Domain 9 locked the gate. 1011b - Domain 10 locked the gate. 1100b - Domain 11 locked the gate. 1101b - Domain 12 locked the gate. 1110b - Domain 13 locked the gate. 1111b - Domain 14 locked the gate.

17.6.1.3 Reset Gate Read (RSTGT_R)

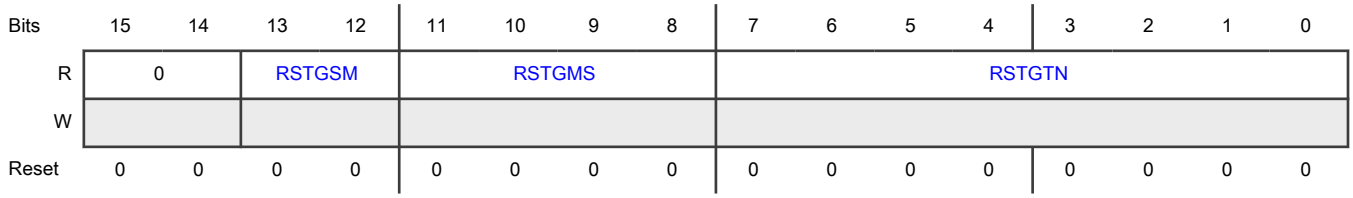
Offset

Register	Offset
RSTGT_R	42h

Function

Describes the specific hardware gate to be reset and records the logic number of domain. Reset Gate Write (RSTGT_W) also describe the same register showing the fields when you write it.

Diagram



Fields

Field	Function
15-14 —	Reserved
13-12 RSTGSM	Reset Gate Finite State Machine Indicates the encoded state machine value when you read the register. RSTGSM = 10b is valid for only a single machine cycle, so a read can never return this value. SEMA42 maintains the reset state machine in a 2-bit, 3-state implementation, defined as follows: 00b - Idle, waiting for the first data pattern write. 01b - Waiting for the second data pattern write 10b - The 2-write sequence has completed. Generate the specified gate reset(s). After the reset is performed, this machine returns to the idle (waiting for first data pattern write) state. 11b - This state encoding is never used and therefore reserved.
11-8 RSTGMS	Reset Gate Domain Records the logical number of the domain performing the gate reset function. The logical number is the domain number. This domain number is determined by the XRDC's initiator Domain Assignment Controller (XRDC_MDAC). To succeed, this function requires that the same domain initiate the two consecutive writes to this register. SEMA42 updates the field each time a write to this register occurs.
7-0 RSTGTN	Reset Gate Number Specifies the specific hardware gate to be reset. The second write updates this field. If RSTGTN < 64, SEMA42 resets the single gate defined by RSTGTN. Otherwise, SEMA42 resets all the gates.

17.6.1.4 Reset Gate Write (RSTGT_W)

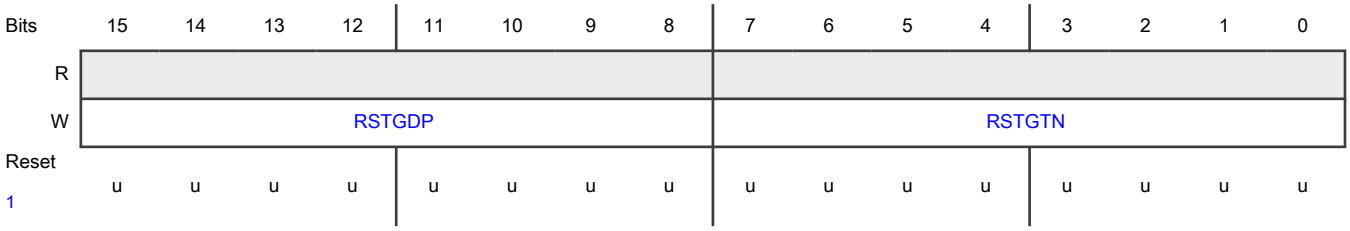
Offset

Register	Offset
RSTGT_W	42h

Function

Specifies the hardware gate to reset when data patterns are specified.

Diagram



1. Reset value is not applicable for writes.

Fields

Field	Function
15-8 RSTGDP	Reset Gate Data Pattern You access this field with the specified data patterns on the two consecutive writes to enable the gate-reset mechanism. For the first write, RSTGDP must be E2h. For the second write, RSTGDP must be 1Dh.
7-0 RSTGTN	Reset Gate Number Specifies the specific hardware gate to be reset. The second write updates this field. If RSTGTN < 64, SEMA42 resets the single gate defined by RSTGTN. Otherwise, SEMA42 resets all the gates.

17.7 Glossary

IPS Protocol used for peripheral accesses to the programming model

dm*X* Domain X

Chapter 18
Crossbar Integrity Checker (XBIC)

18.1 Chip-specific XBIC information

18.1.1 XBIC instances and configuration

This chip has up to seven instances of XBIC. The following tables describes the instances and their configuration.

Table 67. XBIC instances

Instance	S32K388/S32K389	S32K358/ S32K356/S32K348/ S32K338/S32K328	S32K344/S32K324/S32K314/S32K342/ S32K322/S32K341	S32K310/ S32K311/ S32K312
XBIC_0	Yes	Yes	Yes	Yes
XBIC_1	Yes	Yes	Yes	No
XBIC_2	Yes	Yes	Yes	No
XBIC_3	Yes	No	Yes	No
XBIC_4	Yes	Yes	No	No
XBIC_5	Yes	No	No	No
XBIC_6	Yes	No	No	No

Table 68. XBIC configuration for S32K388/S32K389

Instance	Available on crossbar	Initiator and target assignments			
XBIC_0	AXBS_0 (main)				
		Initiator port	**Initiator module**	**Target port**	**Target module**
		M0	Cortex-M7_0 AHBM	S0	S32K388: Flash memory port 0 S32K389: Flash memory 0 port 0
		M1	DMA	S1	S32K388: Flash memory port 1 S32K389: Flash memory 1 port 0
		M2	HSE_B/AES_ACCEL	S2	PRAM_0
		M3	GMAC_0	S3	S32K388: Cortex-M7 TCM/ PRAM_2 S32K389: Cortex-M7 TCM/ PRAM_2 / PRAM_3
		M4	Cortex-M7_1 AHBM	S4	S32K388: Flash memory Port 2

<table>
<tr><th>Instance</th><th>Available on crossbar</th><th>Initiator and target assignments</th></tr>
<tr><td></td><td></td><td>
<table>
<tr><td></td><td></td><td></td><td></td><td>S32K389: Flash memory 0 Port 1</td></tr>
<tr><td>M5</td><td>Cortex-M7_2 AHBM</td><td></td><td>S5</td><td>QuadSPI</td></tr>
<tr><td>M6</td><td>GMAC_1</td><td></td><td>S6</td><td>PRAM_1</td></tr>
<tr><td>M7</td><td>Cortex-M7_3 AHBM</td><td></td><td>S7</td><td>S32K388: Flash memory port 3
S32K389: Flash memory 1 port 1</td></tr>
</table>
</td></tr>
<tr><td>XBIC_1</td><td>AXBS_1 (peripheral)</td><td>
<table>
<tr><th>Initiator port</th><th>Initiator module</th><th></th><th>Target port</th><th>Target module</th></tr>
<tr><td>M0</td><td>Cortex-M7_0 AHBP</td><td></td><td>S0</td><td>AIPS_0</td></tr>
<tr><td>M1</td><td>DMA</td><td></td><td>S1</td><td>AIPS_1</td></tr>
<tr><td>M2</td><td>HSE_B/ ACE</td><td></td><td>S2</td><td>AIPS_2</td></tr>
<tr><td>M3</td><td>Cortex-M7_1 AHBP</td><td></td><td>S3</td><td>ACE target</td></tr>
<tr><td>M4</td><td>Cortex-M7_2 AHBP</td><td></td><td></td><td></td></tr>
<tr><td>M5</td><td>Cortex-M7_3 AHBP</td><td></td><td></td><td></td></tr>
</table>
</td></tr>
<tr><td>XBIC_2</td><td>AXBS_2 (eDMA)</td><td>
<table>
<tr><th>Initiator port</th><th>Initiator module</th><th></th><th>Target port</th><th>Target module</th></tr>
<tr><td>M0</td><td>eDMA</td><td></td><td>S0</td><td>System AXBS</td></tr>
<tr><td></td><td></td><td></td><td>S1</td><td>Peripheral AXBS</td></tr>
</table>
</td></tr>
<tr><td>XBIC_3</td><td>AXBS_3 (Cortex-M7 TCM)</td><td>
<table>
<tr><th>Initiator port</th><th>Initiator module</th><th></th><th>Target port</th><th>Target module</th></tr>
<tr><td>M0</td><td>TCM PRAM AXBS</td><td></td><td>S0</td><td>Cortex-M7_0 TCM</td></tr>
<tr><td></td><td></td><td></td><td>S1</td><td>Cortex-M7_1 TCM</td></tr>
<tr><td></td><td></td><td></td><td>S2</td><td>Cortex-M7_2 TCM</td></tr>
<tr><td></td><td></td><td></td><td>S3</td><td>Cortex-M7_3 TCM</td></tr>
</table>
</td></tr>
<tr><td>XBIC_4</td><td>AXBS_4 (TCM PRAM)</td><td>
<table>
<tr><th>Initiator port</th><th>Initiator module</th><th></th><th>Target port</th><th>Target module</th></tr>
</table>
</td></tr>
</table>

Table 68. XBIC configuration for S32K388/S32K389 (continued)

Instance	Available on crossbar	Initiator and target assignments			
		M0	System AXBS	S0	PRAM_2
				S1	TCM AXBS
				S2	PRAM3 (Applicable for S32K389 only)
XBIC_5	AXBS_5 (ACE HSE_B AXBS)	**Initiator port**	**Initiator module**	**Target port**	**Target module**
		M0	HSE_B	S0	Main AXBS
		M1	ACE	S1	Peripheral AXBS
XBIC_6	AXBS_6 (ACE AXBS)	**Initiator port**	**Initiator module**	**Target port**	**Target module**
		M0	ACE M0	S0	ACE HSE AXBS
		M1	ACE M1		

NOTE

For AES_ACCEL initiator, only SRAM slave is applicable. Accessing other slaves from AES_ACCEL can result in unpredictable system behavior.

Table 69. XBIC configuration for S32K358/S32K356/S32K348/S32K338/S32K328

Instance	Available on crossbar	Initiator and target assignments			
XBIC_0	AXBS_0 (main)	**Initiator port**	**Initiator module**	**Target port**	**Target module**
		M0	Cortex-M7_0 AHBM	S0	Flash memory port 0
		M1	DMA	S1	Flash memory port 1
		M2	HSE_B	S2	PRAM_0
		M3	GMAC	S3	Cortex-M7 TCM/ PRAM_2
		M4	Cortex-M7_1 AHBM	S4	Flash memory Port 2
		M5	Cortex-M7_2 AHBM	S5	QuadSPI
		M6	uSDHC	S6	PRAM_1
				S7	Flash memory port 3

Table 69. XBIC configuration for S32K358/S32K356/S32K348/S32K338/S32K328 (continued)

Instance	Available on crossbar	Initiator and target assignments			
XBIC_1	AXBS_1 (peripheral)				
		Initiator port	**Initiator module**	**Target port**	**Target module**
		M0	Cortex-M7_0 AHBP	S0	AIPS_0
		M1	DMA	S1	AIPS_1
		M2	HSE_B	S2	AIPS_2
		M3	Cortex-M7_1 AHBP		
		M4	Cortex-M7_2 AHBP		
XBIC_2	AXBS_2 (eDMA)				
		Initiator port	**Initiator module**	**Target port**	**Target module**
		M0	eDMA	S0	System AXBS
				S1	Peripheral AXBS
XBIC_3	AXBS_3 (Cortex-M7 TCM)	Not applicable			
XBIC_4	AXBS_4 (Cortex-M7 TCM PRAM)				
		Initiator port	**Initiator module**	**Target port**	**Target module**
		M0	System AXBS	S0	PRAM_2
				S1	Cortex-M7_0 TCM
				S2	Cortex-M7_1 TCM
				S3	Cortex-M7_2 TCM

Table 70. XBIC configuration for S32K344/S32K324/S32K314/S32K342/S32K322/S32K341

Instance	Available on crossbar	Initiator and target assignments			
XBIC_0	AXBS_0 (main)				
		Initiator port	**Initiator module**	**Target port**	**Target module**
		M0	Cortex-M7_0 AHBM	S0	Flash memory port 0
		M1	AXBS_2 S0	S1	Flash memory port 1
		M2	HSE_B	S2	PRAM_0
		M3	EMAC	S3	Cortex-M7 TCM
		M4[1]	Cortex-M7_1 AHBM	S4	Flash memory Port 2
				S5	QuadSPI

Instance	Available on crossbar	Initiator and target assignments
		S6 PRAM_1
XBIC_1	AXBS_1 (peripheral)	**Initiator port** / **Initiator module** / **Target port** / **Target module** M0 Cortex-M7_0 AHBP — S0 AIPS_0 M1 AXBS_2 S1 — S1 AIPS_1 M2 HSE_B — S2 AIPS_2 M3[1] Cortex-M7_1 AHBP 1. These ports are reserved for S32K314.
XBIC_2	AXBS_2 (eDMA)	**Initiator port** / **Initiator module** / **Target port** / **Target module** M0 eDMA — S0 AXBS_0 M1 S1 AXBS_1 M1
XBIC_3	AXBS_3 (Cortex-M7 TCM) [2]	**Initiator port** / **Initiator module** / **Target port** / **Target module** M0 AXBS_0 S3 — S0 Cortex-M7_0 TCM S1[1] Cortex-M7_1 TCM

1. Base address: 4040_0000h. This instance follows the memory map given in section 'XBIC memory map'.

Table 71. XBIC configuration for S32K312 and S32K311

Instance	Available on crossbar	Initiator and target assignments
XBIC_0	AXBS_0 (main)	**Initiator port** / **Initiator module** / **Target port** / **Target module** M0 Cortex-M7_0 AHBM — S0 Flash memory port 0 M1 eDMA — S1 Flash memory port 1 M2 HSE_B — S2 PRAM_0 M3 Cortex-M7_0 AHBP — S3 Cortex-M7 TCM M4 Reserved — S4 AIPS_0 S5 AIPS_1

The errors detected are connected to FCCU. See the memory map file attached to this document for details.

NOTE

- Reset value of MCR register for S32K311, S32K342, S32K322, S32K341, S32K344, S32K324, and S32K314 is "FFFF_0000h".
- For S32K388/S32K389/S32K358/S32K356/S32K348/S32K338/S32K328 specific reset value, see MCR register description.

18.1.2 Target master IDs

See "Chip-specific XRDC information" for master IDs.

18.1.3 XBIC_ESR[VLD] behavior across S32K3xx chips

In S32K388, S32K389, S32K358, S32K356, S32K348, S32K338 and S32K328, the XBIC_ESR[VLD] bit is W1C and writing 1'b1 clears this bit as mentioned in the XBIC_ESR[VLD] description. In rest of the chips, this bit field is reserved and accessing this bit results in transfer error.

18.2 Overview

XBIC is a safety module that verifies the integrity of crossbar transfers.

18.2.1 Block diagram

The chip routes crossbar transfer attribute information for all mapped initiator and target ports to XBIC, which calculates and checks the EDC value of the attribute information, as shown in the following diagram.

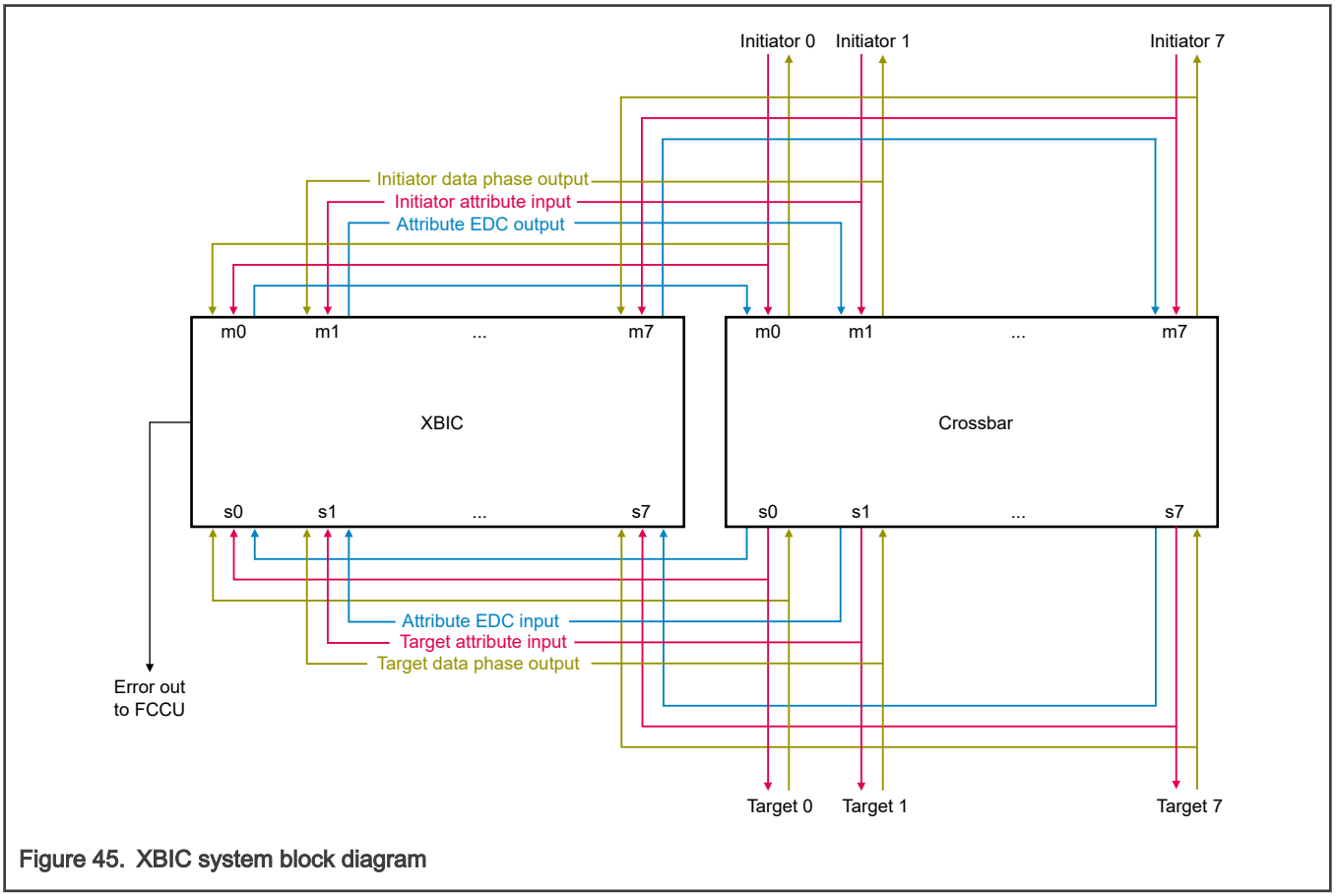


Figure 45. XBIC system block diagram

The above figure illustrates one of many possible XBIC and crossbar configurations. See the Chip-specific XBIC information section for actual port mappings.

18.2.2 Features

XBIC includes the following features:

- Verification of attribute information for crossbar transfers [1],[2]
  - EDC detects single-bit and double-bit errors
- Verification of feedback information for each data phase during crossbar transfers
- Error injection for testing
  - Programmable initiator and target port specifiers
  - Programmable 8-bit toggle vector to insert error in initiator EDC value
  - Address, EDC syndrome, initiator, and target port information are captured when error is detected
- Crossbar transfer attribute integrity check programmable on a per-target-port basis
- Feedback integrity check programmable on a per-initiator-port basis

18.3 Functional description

XBIC verifies the integrity of the crossbar interface on an individual port basis according to the configuration specified via the MCR register. When XBIC detects an error, it reports relevant information and sends an error signal to the FCCU module, but does not generate a bus error. XBIC integrity checking is independent of end-to-end ECC, which monitors the integrity of the transfer address and data.

During the address phase of a transfer, XBIC verifies the crossbar attribute information using an 8-bit EDC, which detects any single-bit or double-bit errors. When XBIC detects an error (an attribute integrity error), due to either hardware fault or error injection, it reports information related to the error in ESR and EAR.

During the data phase of a data transfer, XBIC verifies the integrity of response signals from target to intiator as they pass through the crossbar. When XBIC detects an error in the response signals (a feedback integrity error), it reports the XBIC target and intiator ports in ESR[DPSE0] - ESR[DPSE7] and ESR[DPME0] - ESR[DPME7], respectively.

During the data phase, XBIC sends an alarm to the FCCU module if a intiator port attempts back-to-back accesses in which:

1. The first access terminates normally.
2. The second attempts access to an address space not mapped to any target on the crossbar.

The resultant 'absent target error' causes the crossbar to generate a bus error response to the requesting intiator. XBIC detects the bus error response as a difference because the bus error did not originate from the target port.

You can program XBIC to inject EDC errors for testing purposes. Error injection targets a single target port and a single intiator port, as specified by the configuration settings in the EIR register. When XBIC inserts an error, it changes the EDC syndrome, causing the XBIC to assert an error indication to the FCCU module. Otherwise, transfers are unaffected by error injection. This enables verification of the check logic without compromising the integrity of the data transfer. After you enable error injection function by writing 1 to EIR[EIE], XBIC induces errors on all subsequent targeted transactions until you write a 0 to the field. After the FCCU error indication asserts, it remains asserted even after you write 0 to EIR[EIE]. The error indication deasserts after FCCU specifically clears the existing error. After FCCU clears the XBIC error, additional error injection testing can continue.

To trace a fault reported by the XBIC to the FCCU:

1. Note the error reported by the FCCU. For example, "NCF[46]".
2. Locate the source module and error description in the attached fault mapping spreadsheet. For example:

[1] See Table 72 for a list of crossbar attribute signals verified.
[2] The chip verifies read and write data separately, via the end-to-end ECC architecture.

- Channel number: NCF[46]
- Source module: AXBS_1 integrity checker
- Description: Instruction crossbar error indication to FCCU if syndrome calculated using EDC on the data is not zero

3. Refer to the Chip-specific XBIC information section for the source XBIC module to determine the specific XBIC module instanceâ€""XBIC_1", for example.
4. Determine the type of error (attribute integrity error or feedback integrity error) and the XBIC port(s) involved by reading the information reported in the ESR register of the reported XBIC instance.
5. For attribute integrity check errors, read the XBIC EAR register for the target address of the requested transfer.
6. Refer to the Chip-specific XBIC information section and possibly the Chip-specific AXBS information section for port mapping of XBIC ports to AXBS ports.

Decode a single-bit error syndrome value reported in ESR[SYN] by finding the value in the following table. Any syndrome value not included in the table indicates a multi-bit error.

Table 72. Hexadecimal attribute single-bit error syndromes

Signal	SYN	Signal	SYN	Signal	SYN	Signal	SYN
hwrite	07	hbstrb[7]	70	hdecor[31]	25	hdecor[15]	23
htrans[0]	0b	hbstrb[6]	32	hdecor[30]	68	hdecor[14]	51
hsize[2]	0d	hbstrb[5]	52	hdecor[29]	c7	hdecor[13]	54
hsize[1]	0e	hbstrb[4]	a8	hdecor[28]	83	hdecor[12]	61
hsize[0]	13	hbstrb[3]	43	hdecor[27]	85	hdecor[11]	e3
hprot[5]	15	hbstrb[2]	45	hdecor[26]	86	hdecor[10]	e6
hprot[4]	16	hbstrb[1]	4c	hdecor[25]	89	hdecor[9]	f8
hprot[3]	19	hbstrb[0]	a4	hdecor[24]	8a	hdecor[8]	38
hprot[2]	1a	hmaster[3]	a2	hdecor[23]	8c	hdecor[7]	58
hprot[1]	1c	hmaster[2]	b0	hdecor[22]	49	hdecor[6]	37
hprot[0]	91	hmaster[1]	c1	hdecor[21]	92	hdecor[5]	f1
hburst[2]	a1	hmaster[0]	c2	hdecor[20]	94	hdecor[4]	3b
hburst[1]	64	hslave[2]	c4	hdecor[19]	98	hdecor[3]	3d
hburst[0]	29	hslave[1]	c8	hdecor[18]	46	hdecor[2]	3e
hmastlock	2a	hslave[0]	d0	hdecor[17]	34	hdecor[1]	4f
hunalign	2c	hdecorated	e0	hdecor[16]	4a	hdecor[0]	6e
edc[7]	80	edc[6]	40	edc[5]	20	edc[4]	10
edc[3]	08	edc[2]	04	edc[1]	02	edc[0]	01

18.3.1 Interrupts

This module has no interrupts.

18.4 External signals

XBIC has no external interface signals.

18.5 Initialization

This module does not require initialization.

18.6 XBIC register descriptions

The XBIC programming model consists of four 32-bit registers. Software can access this model only in supervisor mode using 32-bit (word) accesses. Each of the following generates a transfer error back to the requesting intiator. Such errors could cause core exceptions apart from other problems.

- Access size other than 32-bit
- Access to an undefined (reserved) address
- Access in user mode

18.6.1 XBIC memory map

XBIC_AXBS base address: 4020_4000h

XBIC_AXBS_ACE base address: 4040_C000h

XBIC_AXBS_ACE_HSE base address: 4000_8000h

XBIC_AXBS_PERI base address: 4020_8000h

XBIC_AXBS_PRAM_TCM base address: 4040_8000h

XBIC_AXBS_TCM base address: 4040_0000h

XBIC_AXBS_eDMA base address: 4040_4000h

Offset	Register	Width (In bits)	Access	Reset value
0h	XBIC Module Control (MCR)	32	RW	See section
4h	XBIC Error Injection Attributes (EIR)	32	RW	0000_0000h
8h	XBIC Error Status Attributes (ESR)	32	RW	0000_0000h
Ch	XBIC Error Address (EAR)	32	R	0000_0000h

18.6.2 XBIC Module Control (MCR)

Offset

Register	Offset
MCR	0h

Function

Use this register to turn attribute integrity checking and feedback integrity checking on or off on a per-port basis.

- Turn on attribute integrity checking on one or more XBIC target ports by ensuring that the associated SE*n* field(s) have a value of 1. For example, setting field SE*0* enables attribute integrity checking for target port 0. The default (reset) behavior is for attribute integrity checking to be performed for all target ports. XBIC performs EDC-based checks on all transfer requests targeting the selected target port(s). The signals verified are transfer attribute signals going from initiator to target. When XBIC detects an attribute integrity error, it reports relevant information in the ESR and EAR registers.
- Turn on feedback integrity checking on one or more XBIC initiator ports by ensuring that the associated ME*n* field(s) have a value of 1. For example, setting field ME*0* to 1 enables feedback integrity checking for initiator port 0. The default (reset) behavior is for feedback integrity checking to be performed for all initiator ports. XBIC checks target-to-initiator feedback signals for transfer requests originating from the selected XBIC initiator port(s). If any feedback signal value is different at the initiator and target ports during the data phase, XBIC reports the relevant initiator and target ports in the ESR register.

Each field in this register references a specific initiator or target port using XBIC port numbering. Referring to Figure 45, "target port 0" refers to XBIC port "s0" in the figure, "target port 1" refers to port "s1", and so on. Similarly, "initiator port 0" refers to XBIC port "m0", "initiator port 1" refers to port "m1", and so on. See the "Chip-specific XBIC information" section in this document for the mapping of XBIC instances to AXBS instances and XBIC ports to AXBS ports. See the "Chip-specific AXBS information" section in this document for the device component(s) mapped to each port of an AXBS instance.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SE0	SE1	SE2	SE3	SE4	SE5	SE6	SE7	ME0	ME1	ME2	ME3	ME4	ME5	ME6	ME7
W																
Reset	See Register reset values.															

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W																
Reset	See Register reset values.															

Register reset values

Register	Reset value
MCR	XBIC_AXBS: FFFF_0000h XBIC_AXBS_ACE: 80C0_0000h XBIC_AXBS_ACE_HSE: C0C0_0000h XBIC_AXBS_PERI: F0FC_0000h XBIC_AXBS_PRAM_TCM: E080_0000h XBIC_AXBS_TCM: F080_0000h XBIC_AXBS_eDMA: C0C0_0000h

Fields

Field	Function
31	target port EDC Error Detection Enable

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>SE0</td><td>0b - Attribute integrity checking disabled for target port 0
1b - Attribute integrity checking enabled for target port 0</td></tr>
<tr><td>30
SE1</td><td>target port EDC Error Detection Enable
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>XBIC_AXBS</td><td>MCR</td><td>—</td></tr>
<tr><td>XBIC_AXBS_ACE</td><td>—</td><td>MCR</td></tr>
<tr><td>XBIC_AXBS_ACE_HSE</td><td>MCR</td><td>—</td></tr>
<tr><td>XBIC_AXBS_PERI</td><td>MCR</td><td>—</td></tr>
<tr><td>XBIC_AXBS_PRAM_TCM</td><td>MCR</td><td>—</td></tr>
<tr><td>XBIC_AXBS_TCM</td><td>MCR</td><td>—</td></tr>
<tr><td>XBIC_AXBS_eDMA</td><td>MCR</td><td>—</td></tr>
</table>
0b - Attribute integrity checking disabled for target port 1
1b - Attribute integrity checking enabled for target port 1</td></tr>
<tr><td>29
SE2</td><td>target port EDC Error Detection Enable
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>XBIC_AXBS</td><td>MCR</td><td>—</td></tr>
<tr><td>XBIC_AXBS_ACE</td><td>—</td><td>MCR</td></tr>
<tr><td>XBIC_AXBS_ACE_HSE</td><td>—</td><td>MCR</td></tr>
<tr><td>XBIC_AXBS_PERI</td><td>MCR</td><td>—</td></tr>
<tr><td>XBIC_AXBS_PRAM_TCM</td><td>MCR</td><td>—</td></tr>
</table></td></tr>
</table>

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>XBIC_AXBS_TCM</td><td>MCR</td><td>—</td></tr>
<tr><td>XBIC_AXBS_eDMA</td><td>—</td><td>MCR</td></tr>
</table>
0b - Attribute integrity checking disabled for target port 2
1b - Attribute integrity checking enabled for target port 2
</td></tr>
<tr><td>28
SE3</td><td>target port EDC Error Detection Enable
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>XBIC_AXBS</td><td>MCR</td><td>—</td></tr>
<tr><td>XBIC_AXBS_ACE</td><td>—</td><td>MCR</td></tr>
<tr><td>XBIC_AXBS_ACE_HSE</td><td>—</td><td>MCR</td></tr>
<tr><td>XBIC_AXBS_PERI</td><td>MCR</td><td>—</td></tr>
<tr><td>XBIC_AXBS_PRAM_TCM</td><td>—</td><td>MCR</td></tr>
<tr><td>XBIC_AXBS_TCM</td><td>MCR</td><td>—</td></tr>
<tr><td>XBIC_AXBS_eDMA</td><td>—</td><td>MCR</td></tr>
</table>
0b - Attribute integrity checking disabled for target port 3
1b - Attribute integrity checking enabled for target port 3
</td></tr>
<tr><td>27
SE4</td><td>target port EDC Error Detection Enable
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>XBIC_AXBS</td><td>MCR</td><td>—</td></tr>
</table>
</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>XBIC_AXBS_ACE</td><td>—</td><td>MCR</td></tr>
<tr><td>XBIC_AXBS_ACE_HSE</td><td>—</td><td>MCR</td></tr>
<tr><td>XBIC_AXBS_PERI</td><td>—</td><td>MCR</td></tr>
<tr><td>XBIC_AXBS_PRAM_TCM</td><td>—</td><td>MCR</td></tr>
<tr><td>XBIC_AXBS_TCM</td><td>—</td><td>MCR</td></tr>
<tr><td>XBIC_AXBS_eDMA</td><td>—</td><td>MCR</td></tr>
</table>
0b - Attribute integrity checking disabled for target port 4
1b - Attribute integrity checking enabled for target port 4
</td></tr>
<tr><td>26
SE5</td><td>target port EDC Error Detection Enable
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>XBIC_AXBS</td><td>MCR</td><td>—</td></tr>
<tr><td>XBIC_AXBS_ACE</td><td>—</td><td>MCR</td></tr>
<tr><td>XBIC_AXBS_ACE_HSE</td><td>—</td><td>MCR</td></tr>
<tr><td>XBIC_AXBS_PERI</td><td>—</td><td>MCR</td></tr>
<tr><td>XBIC_AXBS_PRAM_TCM</td><td>—</td><td>MCR</td></tr>
<tr><td>XBIC_AXBS_TCM</td><td>—</td><td>MCR</td></tr>
<tr><td>XBIC_AXBS_eDMA</td><td>—</td><td>MCR</td></tr>
</table>
0b - Attribute integrity checking disabled for target port 5
1b - Attribute integrity checking enabled for target port 5
</td></tr>
<tr><td>25
SE6</td><td>target port EDC Error Detection Enable</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>XBIC_AXBS</td><td>MCR</td><td>—</td></tr>
<tr><td>XBIC_AXBS_ACE</td><td>—</td><td>MCR</td></tr>
<tr><td>XBIC_AXBS_ACE_HSE</td><td>—</td><td>MCR</td></tr>
<tr><td>XBIC_AXBS_PERI</td><td>—</td><td>MCR</td></tr>
<tr><td>XBIC_AXBS_PRAM_TCM</td><td>—</td><td>MCR</td></tr>
<tr><td>XBIC_AXBS_TCM</td><td>—</td><td>MCR</td></tr>
<tr><td>XBIC_AXBS_eDMA</td><td>—</td><td>MCR</td></tr>
</table>
0b - Attribute integrity checking disabled for target port 6
1b - Attribute integrity checking enabled for target port 6
</td></tr>
<tr><td>24
SE7</td><td>
target Port EDC Error Detection Enable
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>XBIC_AXBS</td><td>MCR</td><td>—</td></tr>
<tr><td>XBIC_AXBS_ACE</td><td>—</td><td>MCR</td></tr>
<tr><td>XBIC_AXBS_ACE_HSE</td><td>—</td><td>MCR</td></tr>
<tr><td>XBIC_AXBS_PERI</td><td>—</td><td>MCR</td></tr>
<tr><td>XBIC_AXBS_PRAM_TCM</td><td>—</td><td>MCR</td></tr>
<tr><td>XBIC_AXBS_TCM</td><td>—</td><td>MCR</td></tr>
<tr><td>XBIC_AXBS_eDMA</td><td>—</td><td>MCR</td></tr>
</table>
</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>0b - Attribute integrity checking disabled for target port 7
1b - Attribute integrity checking enabled for target port 7</td></tr>
<tr><td>23
ME0</td><td>initiator Port Enable For Feedback Integrity Check
0b - Feedback integrity checking disabled for initiator port 0
1b - Feedback integrity checking enabled for initiator port 0</td></tr>
<tr><td>22
ME1</td><td>initiator Port Enable For Feedback Integrity Check
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>XBIC_AXBS</td><td>MCR</td><td>—</td></tr><tr><td>XBIC_AXBS_ACE</td><td>MCR</td><td>—</td></tr><tr><td>XBIC_AXBS_ACE_HSE</td><td>MCR</td><td>—</td></tr><tr><td>XBIC_AXBS_PERI</td><td>MCR</td><td>—</td></tr><tr><td>XBIC_AXBS_PRAM_TCM</td><td>—</td><td>MCR</td></tr><tr><td>XBIC_AXBS_TCM</td><td>—</td><td>MCR</td></tr><tr><td>XBIC_AXBS_eDMA</td><td>MCR</td><td>—</td></tr></table>
0b - Feedback integrity checking disabled for initiator port 1
1b - Feedback integrity checking enabled for initiator port 1</td></tr>
<tr><td>21
ME2</td><td>initiator Port Enable For Feedback Integrity Check
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>XBIC_AXBS</td><td>MCR</td><td>—</td></tr><tr><td>XBIC_AXBS_ACE</td><td>—</td><td>MCR</td></tr><tr><td>XBIC_AXBS_ACE_HSE</td><td>—</td><td>MCR</td></tr></table></td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>XBIC_AXBS_PERI</td><td>MCR</td><td>—</td></tr><tr><td>XBIC_AXBS_PRAM_TCM</td><td>—</td><td>MCR</td></tr><tr><td>XBIC_AXBS_TCM</td><td>—</td><td>MCR</td></tr><tr><td>XBIC_AXBS_eDMA</td><td>—</td><td>MCR</td></tr></table> 0b - Feedback integrity checking disabled for initiator port 2 1b - Feedback integrity checking enabled for initiator port 2
20 ME3	initiator Port Enable For Feedback Integrity Check **NOTE** This field is not supported in every instance. The following table includes only supported registers. <table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>XBIC_AXBS</td><td>MCR</td><td>—</td></tr><tr><td>XBIC_AXBS_ACE</td><td>—</td><td>MCR</td></tr><tr><td>XBIC_AXBS_ACE_HSE</td><td>—</td><td>MCR</td></tr><tr><td>XBIC_AXBS_PERI</td><td>MCR</td><td>—</td></tr><tr><td>XBIC_AXBS_PRAM_TCM</td><td>—</td><td>MCR</td></tr><tr><td>XBIC_AXBS_TCM</td><td>—</td><td>MCR</td></tr><tr><td>XBIC_AXBS_eDMA</td><td>—</td><td>MCR</td></tr></table> 0b - Feedback integrity checking disabled for initiator port 3 1b - Feedback integrity checking enabled for initiator port 3
19 ME4	initiator Port Enable For Feedback Integrity Check **NOTE** This field is not supported in every instance. The following table includes only supported registers.

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>XBIC_AXBS</td><td>MCR</td><td>—</td></tr>
<tr><td>XBIC_AXBS_ACE</td><td>—</td><td>MCR</td></tr>
<tr><td>XBIC_AXBS_ACE_HSE</td><td>—</td><td>MCR</td></tr>
<tr><td>XBIC_AXBS_PERI</td><td>MCR</td><td>—</td></tr>
<tr><td>XBIC_AXBS_PRAM_TCM</td><td>—</td><td>MCR</td></tr>
<tr><td>XBIC_AXBS_TCM</td><td>—</td><td>MCR</td></tr>
<tr><td>XBIC_AXBS_eDMA</td><td>—</td><td>MCR</td></tr>
</table>
0b - Feedback integrity checking disabled for initiator port 4
1b - Feedback integrity checking enabled for initiator port 4
</td></tr>
<tr><td>18
ME5</td><td>initiator Port Enable For Feedback Integrity Check
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>XBIC_AXBS</td><td>MCR</td><td>—</td></tr>
<tr><td>XBIC_AXBS_ACE</td><td>—</td><td>MCR</td></tr>
<tr><td>XBIC_AXBS_ACE_HSE</td><td>—</td><td>MCR</td></tr>
<tr><td>XBIC_AXBS_PERI</td><td>MCR</td><td>—</td></tr>
<tr><td>XBIC_AXBS_PRAM_TCM</td><td>—</td><td>MCR</td></tr>
<tr><td>XBIC_AXBS_TCM</td><td>—</td><td>MCR</td></tr>
<tr><td>XBIC_AXBS_eDMA</td><td>—</td><td>MCR</td></tr>
</table>
0b - Feedback integrity checking disabled for initiator port 5
1b - Feedback integrity checking enabled for initiator port 5
</td></tr>
<tr><td>17</td><td>initiator Port Enable For Feedback Integrity Check</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>ME6</td><td>
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>XBIC_AXBS</td><td>MCR</td><td>—</td></tr>
<tr><td>XBIC_AXBS_ACE</td><td>—</td><td>MCR</td></tr>
<tr><td>XBIC_AXBS_ACE_HSE</td><td>—</td><td>MCR</td></tr>
<tr><td>XBIC_AXBS_PERI</td><td>—</td><td>MCR</td></tr>
<tr><td>XBIC_AXBS_PRAM_TCM</td><td>—</td><td>MCR</td></tr>
<tr><td>XBIC_AXBS_TCM</td><td>—</td><td>MCR</td></tr>
<tr><td>XBIC_AXBS_eDMA</td><td>—</td><td>MCR</td></tr>
</table>
0b - Feedback integrity checking disabled for initiator port 6
1b - Feedback integrity checking enabled for initiator port 6
</td></tr>
<tr><td>16
ME7</td><td>
initiator Port Enable For Feedback Integrity Check
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>XBIC_AXBS</td><td>MCR</td><td>—</td></tr>
<tr><td>XBIC_AXBS_ACE</td><td>—</td><td>MCR</td></tr>
<tr><td>XBIC_AXBS_ACE_HSE</td><td>—</td><td>MCR</td></tr>
<tr><td>XBIC_AXBS_PERI</td><td>—</td><td>MCR</td></tr>
<tr><td>XBIC_AXBS_PRAM_TCM</td><td>—</td><td>MCR</td></tr>
<tr><td>XBIC_AXBS_TCM</td><td>—</td><td>MCR</td></tr>
<tr><td>XBIC_AXBS_eDMA</td><td>—</td><td>MCR</td></tr>
</table>
</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Feedback integrity checking disabled for initiator port 7 1b - Feedback integrity checking enabled for initiator port 7
15-0 —	Reserved

18.6.3 XBIC Error Injection Attributes (EIR)

Offset

Register	Offset
EIR	4h

Function

Use this register to configure the XBIC error injection function and turn it on or off. When enabled, the XBIC error injection function inserts an attribute integrity error when the targeted initiator requests a transaction of the targeted target port. The inserted error changes the calculated EDC syndrome value, causing XBIC to:

- Capture transfer information in the ESR and EAR registers
- Assert an error signal to the FCCU module

Otherwise, XBIC error injection does not affect transfers.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	EIE	0														
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	SLV			MST				SYN							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 EIE	Error Injection Enable 0b - Error injection disabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Error injection enabled
30-15 —	Reserved
14-12 SLV	Target Port Specifies the target port for error injection—other target ports are unaffected. Specify the target port by its XBIC target port number (0–7). See the "Chip-specific XBIC information" section in this document for the mapping of XBIC instances to AXBS instances and XBIC ports to AXBS ports. See the "Chip-specific AXBS information" section in this document for the device component(s) mapped to each port of an AXBS instance.
11-8 MST	Target initiator ID Specifies the target initiator port for error injection—transfers with other intiators are unaffected. Specify the target initiator port using the logical initiator ID number of the target bus initiator. See the "Chip-specific XBIC information" section in this document for the initiator IDs and their corresponding components.
7-0 SYN	Syndrome XBIC performs an exclusive OR operation on the value in this field and the calculated syndrome, to generate an error with the specified syndrome. A value of zero does not generate an error. See Table 72 for a list of transfer attribute single-bit error syndromes, noting that the values given in the table are hexadecimal.

18.6.4 XBIC Error Status Attributes (ESR)

Offset

Register	Offset
ESR	8h

Function

In this register, XBIC reports information about the most recent transfer with an error detected. If XBIC detects an attribute integrity check error, it reports:

- The target port identifier (SLV)
- The initiator port identifier (MST) and the error syndrome (SYN)

If XBIC detects a mismatch among feedback signals during the data phase:

- The DPSE0 - DPSE7 field with a value of 1 indicates the XBIC target port. In the DPSE0-DPSE7 field descriptions, the target port number refers to the XBIC target port number. Referring to Figure 45, "target port 0" refers to XBIC port "s0" in the figure, "target port 1" refers to port "s1", and so on. See the "Chip-specific XBIC information" section in this document for the mapping of XBIC instances to AXBS instances and XBIC ports to AXBS ports. See the "Chip-specific AXBS information" section in this document for the device component(s) mapped to each port of an AXBS instance.
- The DPME0 - DPME7 field with a value of 1 indicates the XBIC initiator port. In the DPME0-DPME7 field descriptions, the initiator port number refers to the XBIC initiator port number. Referring to Figure 45, "initiator port 0" refers to XBIC port "m0" in

the figure, "initiator port 1" refers to port "m1", and so on. See the "Chip-specific XBIC information" section in this document for the mapping of XBIC instances to AXBS instances and XBIC ports to AXBS ports. See the "Chip-specific AXBS information" section in this document for the device component(s) mapped to each port of an AXBS instance.

XBIC sets this register to all 0s only on reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	VLD	DPSE 0	DPSE 1	DPSE 2	DPSE 3	DPSE 4	DPSE 5	DPSE 6	DPSE 7	DPME 0	DPME 1	DPME 2	DPME 3	DPME 4	DPME 5	DPME 6
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DPME 7	SLV			MST				SYN							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 VLD	Error Status Valid 0b - No error detected—other fields of the ESR and EAR registers are invalid 1b - Error detected—all fields of the ESR and EAR registers are valid
30 DPSE0	Data Phase target Port Error 0b - No feedback integrity error detected on target port 0 1b - Feedback integrity error detected on target port 0
29 DPSE1	Data Phase target Port Error 0b - No feedback integrity error detected on target port 1 1b - Feedback integrity error detected on target port 1
28 DPSE2	Data Phase target Port Error 0b - No feedback integrity error detected on target port 2 1b - Feedback integrity error detected on target port 2
27 DPSE3	Data Phase target Port Error 0b - No feedback integrity error detected on target port 3 1b - Feedback integrity error detected on target port 3
26 DPSE4	Data Phase target Port Error 0b - No feedback integrity error detected on target port 4

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Feedback integrity error detected on target port 4
25 DPSE5	Data Phase target Port Error 0b - No feedback integrity error detected on target port 5 1b - Feedback integrity error detected on target port 5
24 DPSE6	Data Phase target Port Error 0b - No feedback integrity error detected on target port 6 1b - Feedback integrity error detected on target port 6
23 DPSE7	Data Phase target Port Error 0b - No feedback integrity error detected on target port 7 1b - Feedback integrity error detected on target port 7
22 DPME0	Data Phase initiator Port Error 0b - No feedback integrity error detected on initiator port 0 1b - Feedback integrity error detected on initiator port 0
21 DPME1	Data Phase initiator Port Error 0b - No feedback integrity error detected on initiator port 1 1b - Feedback integrity error detected on initiator port 1
20 DPME2	Data Phase initiator Port Error 0b - No feedback integrity error detected on initiator port 2 1b - Feedback integrity error detected on initiator port 2
19 DPME3	Data Phase initiator Port Error 0b - No feedback integrity error detected on initiator port 3 1b - Feedback integrity error detected on initiator port 3
18 DPME4	Data Phase initiator Port Error 0b - No feedback integrity error detected on initiator port 4 1b - Feedback integrity error detected on initiator port 4
17 DPME5	Data Phase initiator Port Error 0b - No feedback integrity error detected on initiator port 5 1b - Feedback integrity error detected on initiator port 5
16 DPME6	Data Phase initiator Port Error 0b - No feedback integrity error detected on initiator port 6 1b - Feedback integrity error detected on initiator port 6

Table continues on the next page...

Table continued from the previous page...

Field	Function
15 DPME7	Data Phase initiator Port Error 0b - No feedback integrity error detected on initiator port 7 1b - Feedback integrity error detected on initiator port 7
14-12 SLV	target Port target port targeted by the most recent transfer with an attribute integrity check error detected. The value in this field is the XBIC target port number (0–7). See the "Chip-specific XBIC information" section in this document for the mapping of XBIC instances to AXBS instances and XBIC ports to AXBS ports. See the "Chip-specific AXBS information" section in this document for the device component(s) mapped to each port of an AXBS instance.
11-8 MST	initiator ID initiator port that requested the most recent transfer with an attribute integrity check error detected. The value in this field is the logical initiator ID number of the bus initiator. See the "Chip-specific XBIC information" section in this document for the initiator IDs and their corresponding components.
7-0 SYN	Syndrome Syndrome calculated for the most recent transfer with an attribute integrity check error detected. For single-bit errors, identify the signal in error by matching the SYN value in Table 72, noting that the syndrome (SYN) values in the table are hexadecimal.

18.6.5 XBIC Error Address (EAR)

Offset

Register	Offset
EAR	Ch

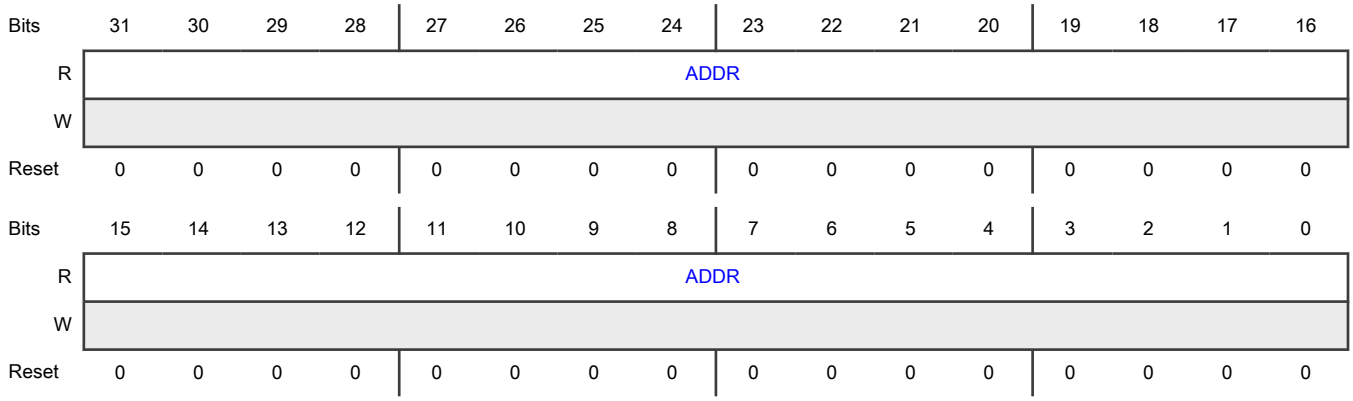
Function

In this register, XBIC reports the address of the most recent transfer with an attribute integrity check error detected—either because of a hardware fault or error injection. XBIC sets this register to all 0s only on reset.

NOTE

An attempted write to this read-only register results in a transfer error.

Diagram



Fields

Field	Function
31-0 ADDR	Error Address Address of the most recent transfer with an attribute integrity check error detected.

18.7 Glossary

EDC Error Detection Code

hdecor[31:0] Non-standard AHB address phase signal for transporting optional decorated storage instruction information

hdecorated Non-standard AHB address phase signal for transporting optional decorated storage instruction information

Chapter 19
Extended Resource Domain Controller (XRDC)

19.1 Chip-specific XRDC information

19.1.1 MDAC configuration

All MDACs with DID = 0 use the default DID parameter.

Table 73. MDAC configuration

Submodule instance	Configuration	XRDC MDACFG#	Bus initiator	Initiator ID value	Default DID	PID	Nonsecure input	Applicability
XRDC_MDAC0	Processor	1	Cortex-M7_0, AXI, AHBP	0h	0h	PID0	0b	S32K310, S32K311, S32K312, S32K344, S32K324, S32K314, S32K342, S32K322, S32K341, S32K388, S32K389, S32K358, S32K356, S32K348, S32K338, S32K328
			Cortex-M7_0 debug	8h				S32K388, S32K389, S32K358, S32K356, S32K348, S32K338, S32K328
XRDC_MDAC1	Nonprocessor	1	eDMA AHB	2h	0h	—	1b	All
XRDC_MDAC4	Processor	1	Cortex-M7_1, AXI, AHBP	1h	0h	PID4	0b	S32K322, S32K324, S32K388, S32K389, S32K358, S32K356, S32K348, S32K338, S32K328
			Cortex-M7_1 debug	9h				S32K388, S32K389, S32K348,S32

Table continues on the next page...

Table 73. MDAC configuration (continued)

Submodule instance	Configuration	XRDC MDACFG#	Bus initiator	Initiator ID value	Default DID	PID	Nonsecure input	Applicability
								K338, S32K328
XRDC_MDAC5	Nonprocessor	1	EMAC AHB	4h	0h	—	1b	S32K342, S32K322, S32K314, S32K324, S32K344
			GMAC_0 AHB					S32K338, S32K348, S32K328, S32K358, S32K356, S32K388, S32K389
XRDC_MDAC6	Processor	1	Cortex-M7_2, AXI, AHBP Cortex-M7_2 debug	7h Bh	0h	PID6	1b	S32K338, S32K358, S32K356, S32K388, S32K389
XRDC_MDAC7	Nonprocessor	1	uSDHC AHB	6h	0h	—	1b	S32K338, S32K348, S32K328, S32K358, S32K356
			GMAC_1 AHB					S32K388, S32K389
XRDC_MDAC8	Processor	1	Cortex-M7_3, AXI, AHBP, debug	5h	0h	PID8	0b	S32K388, S32K389
XRDC_MDAC9	Nonprocessor	1	AES_ACCEL	8h, 9h	0h	—	1b	S32K388, S32K389

19.1.2 MRC configuration

Table 74. MRC configuration

Submodule instance	Region format	Number of region descriptors	Slaves protected (port number)	Applicability
XRDC_MRC0	Auto	16[1] or 8	PFLASH_0 (0) PFLASH_1 (1) PFLASH_2 (2)[1]	S32K310, S32K311, S32K312, S32K342, S32K322, S32K314, S32K324, S32K344, S32K358, S32K356,

Table continues on the next page...

Table 74. MRC configuration (continued)

Submodule instance	Region format	Number of region descriptors	Slaves protected (port number)	Applicability
			PFLASH_3 (3) PFLASH_WR (3)	S32K348, S32K338, S32K328, S32K388
XRDC_MRC0	Auto	16 or 8	PFLASH_0 (P0) PFLASH_0 (P1) PFLASH_0_WR	S32K389
XRDC_MRC1	Auto	16[1] or 8	PRAM0 (0) PRAM1[1] (1)	S32K310, S32K311, S32K312, S32K342, S32K322, S32K314, S32K324, S32K344, S32K358, S32K356, S32K348, S32K338, S32K328, S32K388, S32K389
XRDC_MRC2	Auto	4	QuadSPI (0)	S32K342, S32K322, S32K314, S32K324, S32K344, S32K328, S32K338, S32K348, S32K358, S32K356, S32K388, S32K389
XRDC_MRC3	Auto	16	TCM backdoor/ PRAM2(0) TCM backdoor / PRAM2 (0) / PRAM3 (0)[2]	S32K358, S32K356, S32K348, S32K338, S32K328, S32K388, S32K389
XRDC_MRC4	Auto	1	AES_ACCEL backdoor	S32K388, S32K389
XRDC_MRC5	Auto	1	PFLASH_1 (P0) PFLASH_1 (P1) PFLASH_1_WR	S32K389

1. Applicable to S32K342, S32K341, S32K322, S32K314, S32K324, S32K344, S32K328, S32K338, S32K348, S32K358, S32K356, S32K388
2. Applicable to S32K389

19.1.3 PAC configuration

Table 75. PAC configuration

Module name	Slaves protected	Applicability
XRDC_PAC0	AIPS_0	S32K310, S32K311, S32K312, S32K342, S32K341, S32K322, S32K314, S32K324, S32K344, S32K358, S32K356, S32K348, S32K338, S32K328, S32K388, S32K389

Table continues on the next page...

Table 75. PAC configuration (continued)

Module name	Slaves protected	Applicability
XRDC_PAC1	AIPS_1	S32K310, S32K311, S32K312, S32K342, S32K341, S32K322, S32K314, S32K324, S32K344, S32K358, S32K356, S32K348, S32K338, S32K328, S32K388, S32K389
XRDC_PAC2	AIPS_2	S32K342, S32K341, S32K322, S32K314, S32K324, S32K344, S32K358, S32K356, S32K348, S32K338, S32K328, S32K388, S32K389

NOTE

For PDAC registers assignment to peripherals, see the memory map file attached to this document.

19.1.4 Number of Domain ID

Table 76. Number of Domain ID

Chip	Number of Domain ID supported	Supported values
S32K310, S32K311, S32K312	2	0, 1
S32K322, S32K342, S32K314, S32K324, S32K344	3	0, 1, 2
S32K358, S32K356, S32K348, S32K338, S32K328	4	0, 1, 2, 3
S32K388, S32K389	5	0, 1, 2, 3, 4

19.1.5 Domain Error Word registers (DERR_W*x*_0-18) mapping

The mapping of the domain error capture registers is as follows:

Table 77. Domain Error Word registers mapping

Register	Corresponding MRC/PAC	Available in chips
DERR_W*x*_0	MRC0	S32K310, S32K311, S32K312, S32K342, S32K341, S32K322, S32K314, S32K324, S32K344, S32K358, S32K356, S32K348, S32K338, S32K328, S32K388, S32K389
DERR_W*x*_1	MRC1	S32K310, S32K311, S32K312, S32K342, S32K341, S32K322, S32K314, S32K324, S32K344, S32K358, S32K356, S32K348, S32K338, S32K328, S32K388, S32K389
DERR_W*x*_2	MRC2	S32K342, S32K341, S32K322, S32K314, S32K324, S32K344, S32K358, S32K356, S32K348, S32K338, S32K328, S32K388, S32K389
DERR_W*x*_3	MRC3	S32K358, S32K356, S32K348, S32K338, S32K328, S32K388, S32K389
DERR_W*x*_4	MRC4	S32K388, S32K389
DERR_W*x*_16	PAC0	S32K310, S32K311, S32K312, S32K342, S32K341, S32K322, S32K314, S32K324, S32K344, S32K358, S32K356, S32K348, S32K338, S32K328, S32K388, S32K389
DERR_W*x*_17	PAC1	S32K310, S32K311, S32K312, S32K342, S32K341, S32K322, S32K314, S32K324, S32K344, S32K358, S32K356, S32K348, S32K338, S32K328, S32K388, S32K389

Table continues on the next page...

Table 77. Domain Error Word registers mapping (continued)

Register	Corresponding MRC/PAC	Available in chips
DERR_W*x*_18	PAC2	S32K342, S32K341, S32K322, S32K314, S32K324, S32K344, S32K358, S32K356, S32K348, S32K338, S32K328, S32K388, S32K389

Where *x*: 0, 1, 2, 3.

If the above registers are accessed in chips wherein they are not present, a bus error gets reported.

19.1.6 Exceptions and violations

A write attempt by a noncore bus master outside the defined ranges leads to an exception in case the XRDC region is defined to prevent noncore master access. The chip generates a bus error if you violate XRDC policies.

19.1.7 Configuration using SBAF

SBAF must protect access to its own resources. Hence, XRDC is configured during initialization. SBAF provides the mechanism for you to configure XRDC during boot. You must program the configuration data in the application flash memory region.

19.1.7.1 PDAC default configuration

SBAF configures and lock the following peripherals for its exclusive use. Read and write access to these peripheral are not provided to application domains.

Table 78. PDAC default configuration

Peripheral	Peripheral PDAC number	Remarks
Flash memory controller alternate	187	HSE_B uses the alternate interface exclusively.
Flash memory alternate	188	HSE_B uses the alternate interface exclusively.
Flash memory 1 controller alternate[1]	352	HSE_B uses the alternate interface exclusively.
Flash memory 1 alternate[1]	353	HSE_B uses the alternate interface exclusively.

1. Applicable to S32K389.

19.1.8 Configuration when HSE_B firmware-feature flag is cleared

When the HSE_B firmware-feature flag is cleared, SBAF:

- Does not permit XRDC configuration.
- Locks the aforementioned configurations.

19.1.9 HWCFG0 register reset value in different chips

Table 79. HWCFG0 reset value

Chip	Reset value
S32K310, S32K311, S32K312, S32K322	0x11010301
S32K341, S32K342, S32K314, S32K344, S32K324	0x12020502
S32K358, S32K356, S32K348, S32K338, S32K328	0x12030703
S32K388	0x12040904
S32K389	0x12050904

19.2 Overview

XRDC manages access control between initiators (cores and noncore initiators) and targets (memories and peripherals) by placing them in virtual groups called domains.

Conceptually, a domain is one or more initiators and memories and peripherals, that are isolated from others. It may help to look at a domain as a permissions group within a computing environment. All initiators in a domain have the same access to chip resources such as memory and peripherals. See Introduction to domains for more information on domains.

The protection provided by XRDC access control is in addition to the local memory protection unit contained within each core.

NOTE

Terminology in this chapter has been updated as follows:

Table 80. Updated terms

Updated term	Deprecated term
Initiator	Master
Target	Slave

19.2.1 Features

- Enables you to partition chip resources (initiator and target) into access-controlled domain.
  - Each domain has a unique DID .
  - The DID is an attribute of every system bus transaction.
- Provides a four-level hierarchical access control scheme for defining an ACP for each target in a domain. See Access control model for more information.
  - Memory region descriptors define access policies for address ranges within memories.
  - Peripheral access control registers define access policies for individual peripherals.
- Supports optional hardware semaphores to dynamically modify access rights for target resources.

19.2.2 Block diagram

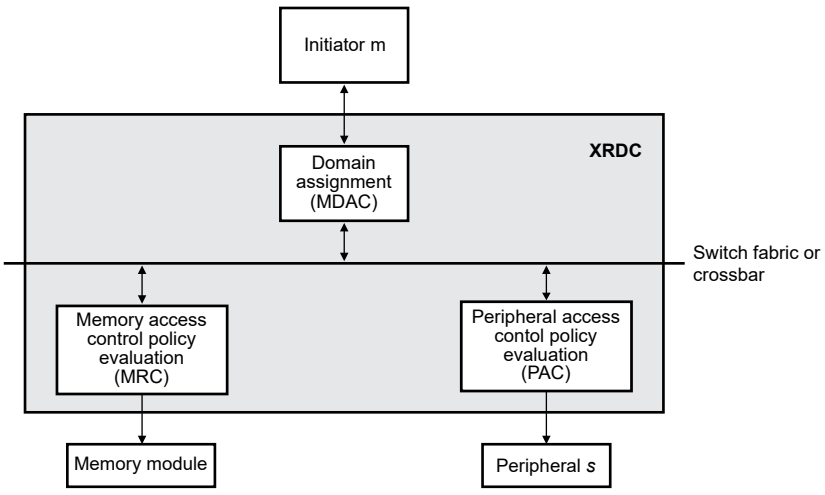


Figure 46. Block diagram

19.2.3 Block descriptions

Block	Description
Domain assignment	A process that adds information to transactions, including: • DID • Privileged attribute • Secure attribute Domain assignment is performed by the MDAC submodule. See: • Domain assignment • Master domain assignment controller (MDAC)
Initiator	A core or noncore (for example, DMA) module that can initiate transactions with memory or peripheral resources.
Memory	Non-volatile memory, RAM, or other memory.
Memory access control policy evaluation	A process that determines whether the domain associated with a transaction has access rights to a memory location. The process is performed by the MRC submodule. See: • Memory region ACP evaluation • Memory region controller (MRC)
Peripheral	A nonmemory resource module within the chip—an ADC, timer module, or communications module, for example.
Peripheral access control policy evaluation	A process that determines whether the domain associated with a transaction has access rights to a peripheral. The process is performed by the PAC submodule. See Peripheral access controller (PAC).
Crossbar	The chip's module and I/O interconnect infrastructure.

19.2.4 Indexes used in this chapter

Table 81. Indexes used in this chapter

Index	Description
c	Memory controller number. For example, MRC*c*.
d	Domain number. For example, PDAC_W0_6[D*d*ACP].
m	Initiator number. For example, PID*m*.
r	Memory region number. For example, RGD_W0_*r*.
s	PDAC slot number. For example, PDAC_W0_*s*[D0ACP].
w	Word number. In a group of registers consisting of consecutive 32-bit registers, *w* is the 0-indexed register number. For example, MRGD_W*w*_0.

19.2.5 Exceptions and violations

A write attempt by a noncore bus initiator outside the defined ranges leads to an exception in case the XRDC region is defined to prevent noncore initiator access. The chip generates a bus error if you violate XRDC policies.

19.3 Modes of operation

XRDC does not support any special modes of operation.

19.4 External signal description

XRDC does not have any external signals.

19.5 Functional description

19.5.1 Introduction to domains

A domain typically consists of one or more initiators, along with the memories and peripherals those initiators are allowed to access. Because it is access-controlled, a domain acts as an independent computing environment.

Domains enable applications to coexist on the same silicon with a firewall between them, enforcing absolute interference protection.

Generally, you create each domain to meet a specific need. Examples of XRDC domain usage include:

- Isolation of real-time from non-real-time applications to ensure resource availability
- Isolation of safety-critical code from non-safety-critical code
- Isolation of third-party untrusted applications from trusted software
- Isolation of memory regions to ensure data security or to prevent accidental overwrites
- Limiting write access for a specific area of system memory to a single DMA module instance
- Limiting read access for a specific area of system memory to a specific processor core

You can assign a core to multiple domains but only one of those domains can be active at a given time.

Within each XRDC instance, each domain has a unique identifier, known as its DID. If an XRDC instance has 16 DIDs, that means the instance has 16 available domains.

You control which initiators can access a peripheral by configuring the domain ACP for each peripheral. Similarly, you control which initiators can access a memory region by configuring the ACP for each memory region.

19.5.2 Submodules

XRDC implements its functionality through its hardware submodules:

- Master domain assignment controller (MDAC)
- Memory region controller (MRC)
- Peripheral access controller (PAC)

19.5.2.1 Master domain assignment controller (MDAC)

The MDAC submodule performs domain assignment logic. XRDC contains an MDAC submodule for each XRDC-protected initiator. Each MDAC assigns a DID to every transaction from its associated initiator. You configure the domain assignment activity for each MDAC through a set of registers, in one of two formats:

- DFMT0 core domain assignment registers
- DFMT1 noncore domain assignment registers

To understand the role of domains in XRDC protection, see Introduction to domains.

For a full explanation of the domain assignment process, see Domain assignment.

19.5.2.2 Memory region controller (MRC)

The MRC submodule performs memory region access control. Each XRDC instance contains the number of MRCs indicated in HWCFG0[NMRC]. Each MRC is associated with one or more memories (see the chip-specific XRDC information for details). The MRC controls memory access using memory region descriptors. MRCFG*c*[NMRGD] indicates the number of region descriptors available for MRC *c*.

Each memory region descriptor defines a memory address range and a configurable access control policy for each domain using a set of four or five 32-bit registers (see Memory ACP evaluation registers).

Memory region descriptors also support including an optional hardware semaphore in the ACP evaluation for memory regions shared by multiple domains (see Hardware semaphores and dynamic access rights).

For a full explanation of the memory region ACP evaluation process, see Memory region ACP evaluation.

19.5.2.3 Peripheral access controller (PAC)

The PAC submodule provides domain access control for all peripherals connected to a single peripheral bus. Each XRDC contains the number of PACs indicated in HWCFG0[NPAC]. Each PAC supports up to 128 peripheral slots (see Finding the PDAC slot number for a peripheral). You configure the ACP for each peripheral using a set of PDAC_W*w_s* registers (see Peripheral ACP evaluation registers).

Peripheral access control also enable a hardware semaphore to be included in the access control policy evaluation for peripherals shared by multiple domains. See Hardware semaphores and dynamic access rights for more details.

For a full explanation of the peripheral ACP evaluation process, see Peripheral ACP evaluation process.

19.5.3 Transaction protection

During application execution, high-level chip modules such as cores or DMA, Ethernet, known as initiators, initiate transaction requests to memory and peripheral resources. XRDC adds protection capabilities to ensure the requesting initiator accesses only the resources that it is authorized to access. These capabilities support security and safety requirements.

XRDC provides this protection by adding two steps to the unprotected transaction flow, as shown in XRDC transaction flow.

XRDC transaction processing differs depending on:

- The type of initiator making the request (see Transaction request sources).
- The type of target receiving the request (see Transaction targets).

NOTE

See chip-specific XRDC information for the domain ID of each initiator on chip.

19.5.4 XRDC transaction flow

Table 82. XRDC transaction flow

Step	Operation	Performed by	Description
1	Transaction request	Initiator	An initiator requests a read or write transaction targeting memory or a peripheral.
2	Domain assignment	XRDC	XRDC MDAC submodule intercepts the request and performs domain assignment, which adds this metadata to the transaction: • DID

Table continues on the next page...

Table 82. XRDC transaction flow (continued)

Step	Operation	Performed by	Description
			• Privileged attribute • Secure attribute
3	Interconnect	Chip	The chip transmits the domain-assigned transaction request across the interconnect (crossbar).
4	ACP evaluation	XRDC	XRDC MRC or PAC submodule intercepts the transaction request and evaluates it against the target's ACP to determine whether the requesting initiator has sufficient access rights to the target. If it does, the transaction continues. Otherwise, XRDC generates an access violation error and the transaction terminates.
5	Transaction response	Target resource	If the previous step does not generate an access violation error, the target resource processes the transaction request and transmits data (for read transactions) and transaction status information (for all transactions) back across the interconnect to the requesting initiator. XRDC is not involved in the flow of information from the target resource back to the requesting initiator.

19.5.5 Domain assignment

19.5.5.1 Overview

Domain assignment associates a DID with each transaction request from an initiator. To determine the DID for a transaction, XRDC evaluates the domain-specific configuration data in the set of MDAC registers (MDA_W*n*_*m*_DFMT0 or MDA_W*n*_*m*_DFMT1) associated with the requesting initiator. The exact process depends on the source of the request (see Transaction request sources).

If the value of MDACFG*m*[NMDAR] is 1, which means a single W*w* register for a given initiator, the specified domain identifier is used directly. In case this value is more than 1, which means there are multiple W*w* registers for a given initiator, MDAC evaluates the conditional terms to determine a "hit". For all W*w* hits, their corresponding domain identifiers are logically summed together (boolean OR). Use cases are typically expected to hit in a single W*w*. Special care is needed if none of the conditional terms hit in any Wn evaluation; in this case, the generated DID = 0 and you must be aware of any potential access rights granted for this DID.

The number of MDAC registers can vary for each initiator. See the chip-specific information. MDACFG*m*[NMDAR] indicates the number of MDA registers, where *m* is the initiator number. You need *m* to locate the registers relevant for domain assignment. See the chip-specific XRDC information for more on initiator numbers. See:

- Register settings for DFMT0 direct domain assignment transactions
- Register settings for DFMT0 PID-based transactions
- Register settings for DFMT1 direct domain assignment transactions

Domain assignment also assigns the secure and privileged attributes to the transaction.

19.5.5.2 PID-based domain assignment

To provide more flexibility in routing core tasks to chip resources in different access-controlled domains, XRDC supports the use of a PID. If the core initiator contains a built-in PID register, indicated by HWCFG2[PIDP*m*] = 1 , XRDC reflects the core PID value in PID*m*[PID], and bit 5 of that field indicates the secure attribute for the transaction request. Otherwise, an application can mimic PID-based domain assignment by writing a value to that field.

19.5.5.3 Transaction request sources

The domain assignment process for an XRDC-protected transaction request depends on the source of the request.

Table 83. Transaction request sources

Request source	Topic	Brief description
Core initiator	DFMT0 direct domain assignment example	Direct domain assignment using a DFMT0 master domain assignment register.
Core initiator	DFMT0 PID-based domain assignment example	PID-based domain assignment using a DFMT0 master domain assignment register.
Core initiator	DFMT1 direct domain assignment example-single MDA	Direct domain assignment using a DFMT1 master domain assignment register with single MDA.

19.5.5.4 DFMT0 core domain assignment registers

Table 84. DFMT0 core domain assignment registers

Register	Index
MDACFG*m*	*m* = initiator number. See the chip-specific XRDC information for the available list of initiators with their IDs.
MDA_W*w*_*m*_DFMT0[1]	*m* = initiator number. *w* = word (see MDA register structure). MDACFG*m*[NMDAR] indicates the number of MDA registers (words) per initiator.
PID*m*	*m* = initiator number.

1. See MDA register structure.

19.5.5.5 DFMT1 noncore domain assignment registers

Table 85. DFMT1 noncore domain assignment registers

Register	Index
MDACFG*m*	*m* = initiator number. See the chip-specific XRDC information for the available list of initiators and their IDs.
MDA_W*w*_*m*_DFMT1[1]	*m* = initiator number. *w* = word. MDACFG*m*[NMDAR] indicates the number of MDA registers (words) per initiator.

1. See MDA register structure.

19.5.5.6 MDA register structure

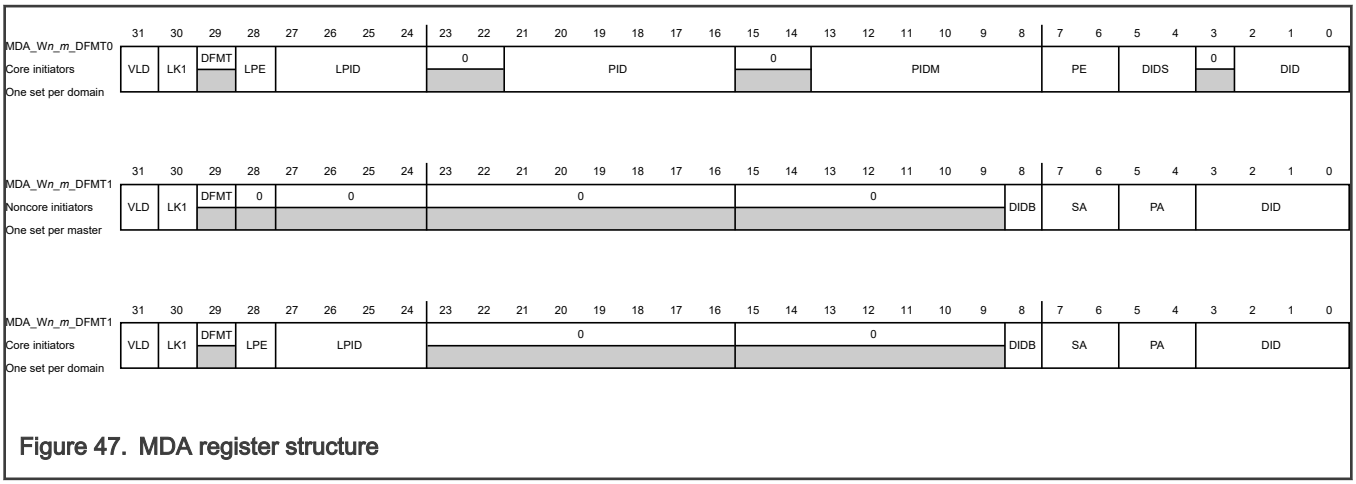


Figure 47. MDA register structure

NOTE

This XRDC configuration does not have LPE and LPID fields and should be considered as reserved fields.

19.5.6 ACP evaluation

19.5.6.1 Overview

When XRDC is not enabled, all peripherals and memories allow unrestricted access. XRDC allows you to limit that access to requests from a particular domain or domains with an application-specified ACP. XRDC intercepts the transaction request and evaluates it against the target's ACP to determine whether the requesting initiator has sufficient access rights to the target, based on the:

- Domain ID assignment (DID)
- Privileged attribute
- Secured attribute

XRDC obtains the target resource's domain ACP from the associated Domain Access Control Policy (D*d*ACP) field (see Domain ACP specification) in the appropriate register set:

- Peripheral ACP evaluation registers
- Memory ACP evaluation registers

If ACP evaluation determines that the transaction request has sufficient access rights to the target resource, XRDC allows the transaction to continue. Otherwise, it terminates the request with an error and captures the address and attribute information in the appropriate error registers.

The exact process depends on the target of the request (see Transaction targets).

XRDC optionally supports the inclusion of a hardware semaphore to dynamically alter the ACP of a memory region or peripheral (see Hardware semaphores and dynamic access rights).

19.5.6.2 Transaction targets

The ACP evaluation for an XRDC-protected transaction request depends on the target of the request.

Table 86. Transaction targets

Request target	Topic	Brief description
Peripheral	Peripheral ACP evaluation example	Process for a transaction request to a target peripheral that is within a protected domain, with ACP evaluation configured by PDAC_W*w*_*s*.
Memory	Memory ACP evaluation example	Process for a transaction request to a target memory region that is within a protected domain, with ACP evaluation configured by MRGD_W*w*_*r*.

19.5.6.3 Peripheral ACP evaluation registers

Table 87. Peripheral ACP evaluation registers

Register	Index	Brief description
PDAC_W0_*s*[1]	*s* = peripheral slot	Specifies the ACP for an XRDC-protected peripheral, and an optional hardware semaphore.
PDAC_W1_*s*[1]	*s* = peripheral slot	Enables the set of PDAC registers for the associated peripheral and locks the set. Typically, you configure the peripheral by writing to the PDAC registers and then limiting their respective domains from making updates to the D*d*ACP fields or by locking the set for all updates.

1. See PDAC register structure.

19.5.6.4 PDAC register structure

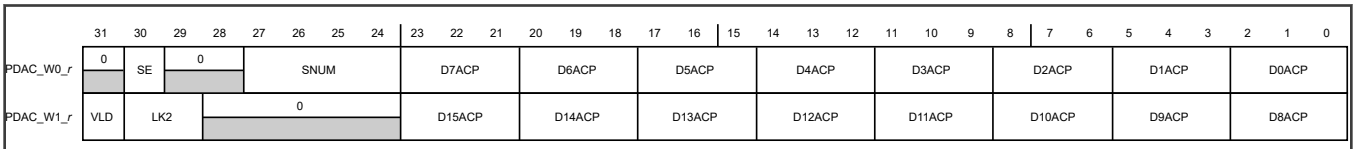


Figure 48. PDAC register structure

19.5.6.5 Memory ACP evaluation registers

When XRDC is enabled (CR[GVLD] = 1), you cannot access any XRDC-protected memory unless you configure at least one set of memory region descriptors (see MRGD_W*w*_*r*) that includes the targeted memory.

Table 88. Memory ACP evaluation registers

Register	Index	Brief description
MRCFG*c*	*c* = memory controller instance	Indicates the number of memory regions per memory controller (NMRGD). Each memory region is configured by a set of memory region descriptor registers (MRGD_W*w*_*r*) described below.
MRGD_W0_*r*[1]	*r* = memory region	Specifies starting address for the memory region.
MRGD_W1_*r*[1]	*r* = memory region	Specifies ending address for the memory region.
MRGD_W2_*r*[1]	*r* = memory region	Specifies the ACP for each supported domain, and an optional hardware semaphore.

Table continues on the next page...

Table 88. Memory ACP evaluation registers (continued)

Register	Index	Brief description
MRGD_W3_*r*[1]	*r* = memory region	Enables the set of MRGD registers for the associated region and locks the set. Typically, you define the memory region by writing to the MRGD registers and then limiting their respective domains from updating the D*d*ACP fields or by locking the set for all updates.

1. See MRGD register structure.

19.5.6.6 MRGD register structure

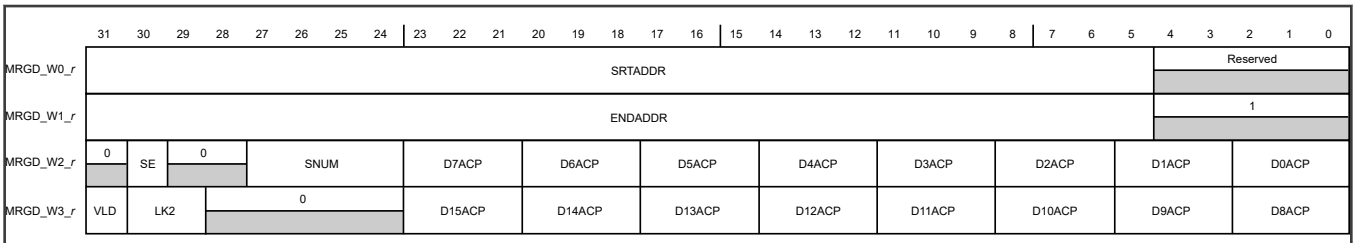


Figure 49. MRGD register structure

19.5.6.7 Access control model

XRDC supports a four-level hierarchical access control model. This model combines the traditional privileged (also known as supervisor) and user access levels with an additional signal for the secure attribute of each memory reference, as shown in Access control model levels.

Each level has different access control policies that specify the read and write accessibility for a target. XRDC combines the privileged and secure attributes with the DID assigned to each system bus transaction to form the hardware basis for the access control mechanism. You specify the ACP for target resources using the D*d*ACP fields (see Domain ACP specification) found in the configuration registers shown in Peripheral ACP evaluation registers and Memory ACP evaluation registers.

You can dynamically control access to shared memory regions and target peripherals with the optional inclusion of a hardware semaphore (see Hardware semaphores and dynamic access rights). If you enable this semaphore for a given address space or peripheral, XRDC allows writes to the target resource only if the requesting domain owns the semaphore.

For cores that support only the three-state access control model (Secure Privileged, Secure User, Nonsecure User), XRDC forces the nonsecure output signal from the MDAC submodule to 0 in privileged mode. This change enables precise state transitions between user and privileged modes. Specifically, the MDAC logic for initiator *m* generates the nonsecure attribute output signal as a function of the Three-State Model (PID*m*[TSM]) and PID Present (HWCFG2[PIDP*m*]) fields, as shown in Generation of secure attribute).

19.5.6.7.1 Access control model levels

Table 89. Access control model levels

Secure	Nonsecure	Privileged (supervisor)	Not privileged (user)	Level
Yes	Not applicable	Yes	Not applicable	Most restricted
Yes	Not applicable	Not applicable	Yes	More restricted
Not applicable	Yes	Yes	Not applicable	Less restricted
Not applicable	Yes	Not applicable	Yes	Least restricted

19.5.6.7.2 Generation of secure attribute

This table shows how XRDC generates the secure attribute for a transaction.

Table 90. Generation of secure attribute

Access levels [1]	PID*m*[TSM] (b)	PID present [2]	Secure attribute determined by
4	0	No	Initiator secure attribute
4	0	Yes	Initiator secure attribute && ~initiator privileged attribute
3	1	No	PID*m*[5] && ~initiator privileged attribute
3	1	Yes	Local secure attribute && ~initiator privileged attribute

1. XRDC assumes a core initiator supports the four-level access model. If a core supports only the three-state access control model, you must write 1 to PID*m*[TSM] before loading any nonsecure value into the PID.
2. Indicated in HWCFG2–3[PIDP*m*].

19.5.6.8 Domain ACP specification

Table 91. Domain ACP specification

D*d*ACP	Allowable accesses			
	Secure Privileged	Secure User	Nonsecure Privileged	Nonsecure User
111b	R, W	R, W	R, W	R, W
110b	R, W	R, W	R, W	None
101b	R, W	R, W	R	R
100b	R, W	R, W	R	None
011b	R, W	R, W	None	None
010b	R, W	None	None	None
001b	R	R	None	None
000b	None	None	None	None

19.5.6.9 Memory region ACP evaluation

During ACP evaluation of a memory transaction request, if the target memory location falls within the address range specified by any XRDC memory region descriptor, then XRDC identifies the request as a memory region hit. For each memory region hit, XRDC compares the DID, privileged attribute, and secured attribute of the transaction to the associated domain ACP (MRGD_W{2,3}_*r*[D*d*ACP]) in the memory region descriptor. See ACP evaluation for additional details on this function.

The following conditions cause XRDC to report an access error:

- The target memory location does not fall within any defined memory regions; in other words, the transaction request is not a hit.
- The transaction request does not have the appropriate access permissions for the region, which triggers a domain violation.
- The transaction request hits multiple (overlapping) regions, and all of those regions signal access violations.

Unimplemented domain identifiers default to no access privileges, and therefore the access type of a D*d*ACP field for an unimplemented domain is read-as-zero/writes-ignored (RAZ/WI).

19.5.6.10 Hardware semaphores and dynamic access rights

XRDC memory region descriptors and peripheral access control support an optional hardware semaphore in their access evaluations. This hardware semaphore allows hardware enforcement of dynamic access rights, based on the state of the semaphore for shared memory regions or shared peripherals.

If enabled, the state of the semaphore dynamically modifies the access control policies so that only the domain owning it has write access to the resource. The write permissions for all other domains are revoked based on the semaphore state. If no domain owns the semaphore, the PAC and MRC submodules evaluate DdACP normally.

If you enable a hardware semaphore (by writing 1 to MRGD_W2_*r*[SE] or PDAC_W0_*r*[SE]) then, before the normal DdACP evaluation, XRDC checks the state of the hardware semaphore specified in MRGD_W2_*r*[SNUM] or PDAC_W0_*r*[SNUM].

On a write transaction, if the semaphore is not idle (the semaphore state is non-zero) and the requesting domain does not own the semaphore, the memory or peripheral access terminates with an error. In other words, writes into a semaphore-enabled address space or peripheral are allowed only if the semaphore is idle or the requesting domain owns the semaphore.

19.5.7 XRDC transaction examples

To see the complete transaction process for an XRDC-protected transaction request from an initiator to target:

1. Follow one of these examples of domain assignment:
   - DFMT0 direct domain assignment example
   - DFMT0 PID-based domain assignment example
   - DFMT1 direct domain assignment example-single MDA
2. Then follow one of these examples of ACP evaluation:
   - Peripheral ACP evaluation example
   - Memory ACP evaluation example

19.5.7.1 DFMT0 direct domain assignment example

This configuration assigns a specific domain to all incoming transactions.

- A core initiator has initiator number = 6.
- The core has one MDAC register (MDACFG6[NMDAR] = 1), configured as shown in Register settings for DFMT0 direct domain assignment transactions.

MDAC_W*w*_*m*_DFMT0 registers do not include fields for secure and privileged attributes. Those attributes are part of the transaction data from core initiators and are forwarded with the transaction after domain assignment.

19.5.7.1.1 Register settings for DFMT0 direct domain assignment transactions

Table 92. Register settings for DFMT0 direct domain assignment transactions

Field	MDA_W0_6_DFMT0	Comments
VLD	1	Enables this register for use in domain assignment.
LK1	1	Locks the settings in this register until the next module reset.
DFMT	0	Indicates that this is a DFMT0 register.
PID	00_0000b	Not used because PE = 00b.

Table continues on the next page...

Table 92. Register settings for DFMT0 direct domain assignment transactions (continued)

Field	MDA_W0_6_DFMT0	Comments
PIDM	00_0000b	Not used because PE = 00b.
PE	00b	Disables PID-based filtering.
DIDS	00b	All incoming transactions are to be assigned to the domain specified by the DID field.
DID	001b	Because PE = 00b and DIDS = 00b, all incoming transactions are to be assigned DID value 001b.

19.5.7.1.2 DFMT0 direct domain assignment process

The application configures XRDC as it boots. After application boot completes, a core issues a read request to a target peripheral.

1. XRDC intercepts the request and performs domain assignment using the configuration in MDA_W0_6_DFMT0. In this example, XRDC assigns DID = 001b to all incoming transactions.
2. The transaction proceeds with DID = 001b and the privileged and secure attributes provided by the core.

19.5.7.2 DFMT0 PID-based domain assignment example

This example configuration demonstrates PID-based domain assignment:

- A core initiator with initiator number = 4 processes both safety-critical tasks and routine tasks, using two domains:
  - Domain 0 is reserved for safety-critical tasks (PID = 0–15).
  - Domain 1 is reserved for routine tasks (PID > 15).
- The core has eight MDAC registers (MDACFG4[NMDAR] = 8), configured as follows (see Register settings for DFMT0 PID-based transactions):
  - MDA_W0_4_DFMT0 assigns safety-critical tasks (PID = 0–15) to domain 0.
  - MDA_W1_4_DFMT0 assigns routine tasks (PID = 16–31) to domain 1.
  - MDA_W2_4_DFMT0 assigns routine tasks (PID = 32–63) to domain 1.
  - MDA_W3_4_DFMT0 through MDA_W7_4_DFMT0 are not used.

MDAC_W*w_m*_DFMT0 registers do not include fields for secure and privileged attributes. Those attributes are part of the transaction data from core initiators and are forwarded with the transaction after domain assignment.

19.5.7.2.1 Register settings for DFMT0 PID-based transactions

Table 93. Register settings for PID-based transactions

Field	Registers				
	MDA_W0_4_DFMT0	MDA_W1_4_DFMT0	MDA_W2_4_DFMT0	MDA_W[3–7]_4_DFMT0	Comments
VLD	1	1	1	0	Enables this register for use when assigning domains.
LK1	1	1	1	1	Locks the settings in each register until the next device reset.
DFMT	0	0	0	0	Indicates that these registers apply to domain assignment for core initiators.

Table continues on the next page...

Table 93. Register settings for PID-based transactions (continued)

Field	Registers				
	MDA_W0_4_DFMT0	MDA_W1_4_DFMT0	MDA_W2_4_DFMT0	MDA_W[3–7]_4_DFMT0	Comments
PID	00_0000b	01_0000b	10_0000b	—	Constant match value to be used for PID-based filtering.
PIDM	00_1111b	10_1111b	01_1111b	—	Each 0 bit causes the corresponding PID bit to be considered in domain assignment.
PE	10b	10b	10b	—	Specifies the type of pattern matching used for PID evaluation.
DIDS	00b	00b	00b	—	Assign all incoming transactions with PIDs that pass the filtering criteria to the domain specified by the DID field.
DID	000b	001b	001b	—	This DID value is assigned to incoming transactions with PIDs that pass the filtering criteria.

19.5.7.2.2 DFMT0 PID-based domain assignment process

The application configures XRDC as it boots. After booting completes, a core issues a read request to a target peripheral. The task making the request has PID = 6, indicating that it is a safety-critical task.

1. XRDC intercepts the request and performs domain assignment using each enabled MDA_W*n*_4_DFMT0 register, regardless of whether it has already found a match. In this example, XRDC performs the domain assignments as shown in DFMT0 PID-based domain assignment evaluation.
2. After XRDC completes all domain assignment evaluations, it logically ORs the assigned DIDs to determine the final DID assigned to the transaction. In this example, only one evaluation results in a DID assignment, so there is no OR operation.
3. The transaction proceeds with DID 000b and the privilege and secure attributes provided by the core.

19.5.7.2.3 DFMT0 PID-based domain assignment evaluation

Table 94. DFMT0 PID-based domain assignment evaluation

Register	Evaluation steps	Boolean math	Result
MDA_W0_4_DFMT0	1. Bitwise AND of PID with inverted PIDM.	00_0000b & 11_0000b	00_0000b
	2. Bitwise AND of transaction PID (PID4[PID]) with inverted PIDM.	000110b & 11_0000b	00_0000b
	3. Compare the results of steps 1 and 2.	00_0000b == 000000b	True: Assign DID 0
MDA_W1_4_DFMT0	1. Bitwise AND of PID with inverted PIDM.	01_0000b & 01_0000b	01_0000b
	2. Bitwise AND of transaction PID (PID4[PID]) with inverted PIDM.	00_0110b & 01_0000b	00_0000b
	3. Compare the results of steps 1 and 2.	01_0000b == 00_0000b	False: No DID assignment
MDA_W2_4_DFMT0	1. Bitwise AND of PID with inverted PIDM.	10_0000b & 10_0000b	10_0000b

Table continues on the next page...

Table 94. DFMT0 PID-based domain assignment evaluation (continued)

Register	Evaluation steps	Boolean math	Result
	2. Bitwise AND of transaction PID (PID4[PID]) with inverted PIDM.	00_0110b & 10_0000b	00_0000b
	3. Compare the results of steps 1 and 2.	10_0000b == 00_0000b	False: No DID assignment

19.5.7.3 DFMT1 direct domain assignment example-single MDA

This configuration assigns a specific domain to all incoming transactions.

- An initiator has initiator number = 6.
- The initiator has one MDAC register (MDACFG6[NMDAR] = 1) as shown in Register settings for DFMT1 direct domain assignment transactions.

19.5.7.3.1 Register settings for DFMT1 direct domain assignment transactions

Table 95. Register settings for DFMT1 direct domain assignment transactions

Field	MDA_W0_6_DFMT1	Comments
VLD	1	Enables this register for use in domain assignment.
LK1	1	Locks the settings in this register until the next chip reset.
DFMT	1	Indicates that this is a DFMT1 register.
DIDB	0b	All incoming transactions are to be assigned to the domain that the DID field specifies.
SA	10b	All incoming transactions retain their Secure attribute value.
PA	10b	All incoming transactions retain their Privileged attribute value.
DID	001b	Because DIDB = 0b, all incoming transactions are to be assigned DID value 001b.

19.5.7.3.2 DFMT1 direct domain assignment process

The application configures XRDC as it boots. After application boot completes, an initiator issues a read request to a target peripheral.

1. XRDC intercepts the request and assigns a domain using the configuration in MDA_W0_6_DFMT1. In this example, XRDC assigns DID = 001b to all incoming transactions.
2. The transaction proceeds with DID = 001b and the privileged and secure attributes provided by the initiator.

19.5.7.4 Peripheral ACP evaluation example

This example configuration demonstrates ACP evaluation for a target peripheral.

- The core initiator must have exclusive access to the ADC0 peripheral for safety-critical tasks in domain 0.
- The core initiator supports 8 domains (HWCFG0[NDID] = 111b).
- ADC occupies PDAC slot 40 in the chip, as defined in an example memory map file (see Finding the PDAC slot number for a peripheral).
- Given 8 domains and PDAC slot 40, the PDAC registers associated with ADC0 are PDAC_W0_40 and PDAC_W1_40.

The following sections describe the register configurations for this example.

19.5.7.4.1 Finding the PDAC slot number for a peripheral

This topic shows how to find the PDAC slot number for a peripheral, but it is a generic example. Memory map file organization and appearance can vary.

To find the PDAC slot number for a peripheral:

1. Open the memory map file attached to this document and view the peripherals page.
2. Locate the peripheral in the "Instance" column.
3. The PDAC slot number for the peripheral is at the intersection of the "PDAC slot number" column and the peripheral row.

In this table, for peripheral ADC_0, the PDAC slot number is 40. Therefore, the PDAC registers for ADC0 are PDAC_W*w*_40.

Table 96. Finding the PDAC slot number for a peripheral

Instance	Description	Start address	End address	PDAC slot number	AIPS instance
LCU_0	Logic Control Unit 0	0x40098000	0x4009BFFF	38	AIPS_0
LCU_1	Logic Control Unit 1	0x4009C000	0x4009FFFF	39	AIPS_0
ADC_0	Analog-to-digital converter 0	0x400A0000	0x400A3FFF	40	AIPS_0
ADC_1	Analog-to-digital converter 1	0x400A4000	0x400A7FFF	41	AIPS_0
ADC_2	Analog-to-digital converter 2	0x400A8000	0x400ABFFF	42	AIPS_0

19.5.7.4.2 Register settings for peripheral ACP evaluation

This table shows the PDAC_W*w*_40 settings for a safety-critical task assigned to domain 0.

Table 97. Register settings for peripheral ACP evaluation

Register	Field	Value (b)	Comments
PDAC_W0_40	SE	0	The hardware semaphore (see the SEMA42 chapter) is disabled.
	SNUM	(don't care)	The hardware semaphore is not used in this example.
	D7ACP	000	Domain 7 has no access to the peripheral.
	D6ACP	000	Domain 6 has no access to the peripheral.
	D5ACP	000	Domain 5 has no access to the peripheral.
	D4ACP	000	Domain 4 has no access to the peripheral.
	D3ACP	000	Domain 3 has no access to the peripheral.
	D2ACP	000	Domain 2 has no access to the peripheral.
	D1ACP	000	Domain 1 has no access to the peripheral.
	D0ACP	010	Only privileged, secure transactions from domain 0 have access.
PDAC_W1_40	VLD	1	Use this register set in domain ACP evaluations.
	LK2	11	Lock the settings in this register until the next device reset.

19.5.7.4.3 Peripheral ACP evaluation process

XRDC performs the following process for ACP evaluation:

1. When the application is running, the core issues a read request to a target peripheral. The task making the request has PID = 0, indicating that it is a safety-critical task. The transaction request is privileged and secured.
2. Between the chip interconnect and ADC0, XRDC intercepts the request and compares its DID, privileged attribute, and secured attribute to the configuration in PDAC_W0_40 and PDAC_W1_40.
3. Because the ADC0 D0ACP field is 010b for privileged, secured access, XRDC grants access to the transaction.
4. The transaction proceeds normally without any further intervention from XRDC.

19.5.7.5 Memory ACP evaluation example

This example configuration demonstrates ACP evaluation for a target memory. Following are the desired features:

- The core initiator must have exclusive access to the memory region for safety-critical tasks in domain 0.
- The target memory is the address range 1B00_0000h–1B00_1FFFh, protected by memory controller 0 (MRC0), for example.
- Access to the entire memory range will be controlled by the memory region descriptor defined by MRGD_W*w*_0.
- The requested transaction is secure privileged.

With the configuration settings shown in Register settings for memory ACP evaluation, XRDC grants access and the transaction proceeds normally. There is no further XRDC intervention.

19.5.7.5.1 Register settings for memory ACP evaluation

Table 98. Register settings for memory ACP evaluation

Register	Field	Value	Comments
MRGD_W0_0	SRTADDR	1B00_0000h	Starting address of the memory region.
MRGD_W1_0	ENDADDR	1B00_1FFFh	Ending address of the memory region.
MRGD_W2_0	SE	0	The hardware semaphore (see the SEMA42 chapter) is disabled.
	SNUM	(don't care)	The hardware semaphore is not used in this example.
	D7ACP	000b	Domain 7 has no access to the peripheral.
	D6ACP	000b	Domain 6 has no access to the peripheral.
	D5ACP	000b	Domain 5 has no access to the peripheral.
	D4ACP	000b	Domain 4 has no access to the peripheral.
	D3ACP	000b	Domain 3 has no access to the peripheral.
	D2ACP	000b	Domain 2 has no access to the peripheral.
	D1ACP	000b	Domain 1 has no access to the peripheral.
	D0ACP	010b	Only privileged, secure transactions from domain 0 have access.
MRGD_W3_0	VLD	1	Use this register set in domain ACP evaluations.
	LK2	11b	Lock the settings in this register until the next device reset.

19.5.7.5.2 Memory ACP evaluation process

In this example, XRDC performs the following process for ACP evaluation:

1. When the application is running, the core issues a read request to address 1B00_0100h. The transaction request is secure privileged.
2. Between the chip interconnect and the memory, XRDC intercepts the request and compares its DID, memory location of the address, privileged attribute, and secured attribute to the configuration in MRGD_W0_0, MRGD_W1_0, MRGD_W2_0 and MRGD_W3_0.
3. Because the address 1B00_0100h is in the memory range 1B00_0000h–1B00_1FFFh and its D0ACP field is 010b for privileged, secured access, XRDC grants access to the transaction.
4. The transaction proceeds normally without any no further intervention from XRDC.

19.5.8 Clocking

This module has no clocking considerations.

19.5.9 Interrupts

This module outputs an interrupt signal which can be connected to interrupt controller. Check chip-specific interrupt assignment for details. Interrupt is asserted on detection of access violation by any checker, and it remains asserted until DERRLOC registers are cleared.

19.6 Initialization information

Out of reset, XRDC is disabled (CR[GVLD] = 0), which allows secure privileged startup code to configure the entire programming model.

19.6.1 Initialization procedure

1. Read the hardware configuration registers to obtain the implemented XRDC hardware capabilities for the chip:
   - HWCFG0
   - HWCFG1
   - HWCFG2
   - MDACFG*m* (one for each supported bus initiator—number indicated in HWCFG0[NMSTR])
   - MRCFG*c* (one for each supported memory controller—number indicated in HWCFG0[NMRC])
2. Use the information retrieved in step 1 and the desired domain architecture to configure:
   - Domain assignments (MDA_W*w_m*_DFMT0 and MDA_W*w_m*_DFMT1)
   - Memory region descriptors (MRGD_W*w_n*)
   - Peripheral access control (PDAC_W*w_s*)

   Ensure that you enable the necessary registers and register sets using the appropriate VLD fields. Also, you can limit access to these registers or lock them after you configure them, by using the appropriate LK1 fields.
3. Enable XRDC (write 1 to CR[GVLD]).

XRDC is now fully operational.

19.6.2 Minimize access errors

When you configure and enable XRDC, it begins generating DIDs for transaction requests and evaluating access rights at the target memory and peripheral resources. Because of the distributed design hierarchy and the pipelined nature of the hardware system bus fabric, it can take multiple cycles for a generated DID to propagate. Until that happens, XRDC uses the initiator's default DID. Depending on the programmed ACPs, the default DID might generate an access error response.

If XRDC generates incorrect error responses, you can use the following approaches to minimize or eliminate these extraneous access errors:

- Minimize the amount of system bus traffic when XRDC is enabled (CR[GVLD] = 1).
- Ensure that all target memory addresses provide sufficient access rights for any default DIDs and for the newly programmed DIDs. After XRDC is fully operational, as confirmed by a read of HWCFG1, you can remove the permissions for the default DIDs.
- Try to have the bus initiator that programs and configures XRDC use the same DID, that is, its default DID, for both initialization and configuration, besides normal system operation. You do this when you define the DID assignments for the system.

19.7 Application information

19.7.1 Master domain assignments

The typical use case for master domain assignments is to include one or more core bus initiators in a single domain, possibly combined with other noncore bus initiator modules such as DMA. This configuration may be static or may be changed dynamically to select between a small number of domains. HWCFG0[NDID] indicates the maximum number of supported domains. XRDC also supports the optional use of PIDs to create multiple classes of cores, each in different domains.

For example, you can group critical tasks—safety-critical, performance-critical, and so on—into one domain and all other tasks into a second domain. Typically, you assign the DID at initialization, but you can also reconfigure domain assignment while the application is running.

A core bus initiator typically has multiple MDA_W*w_m*_DFMT0 registers associated with it.

A noncore bus initiator typically has a single MDA_W*w_m*_DFMT1 register associated with it.

The master domain assignment, memory region descriptor, and peripheral domain access control registers have lock fields that enable you to limit access to, or to lock, the registers. These actions protect the configuration.

19.7.2 Memory region descriptor management

There are two important concepts to consider for managing the memory region descriptors.

Each MRC*c* configuration is chip-specific. See the chip-specific XRDC information for the number of implemented memory region descriptors (MRGD_W*w_n*) in a given MRC*c* instance, and the specific port numbers associated with the target memories being monitored.

Second, as detailed in Memory region ACP evaluation, after you enable the XRDC, a memory reference must hit one or more of the configured regions. Otherwise, the transaction results in an access violation. Two other conditions also result in access errors:

- The access hits a single region descriptor and that region signals a domain violation.
- The access hits multiple (overlapping) regions and all regions signal violations.

The second condition reflects that XRDC gives priority to permission granting over access denying for overlapping regions. This approach provides more flexibility to system software in memory region descriptor assignments.

19.7.3 Domain error capture management

19.7.3.1 Domain error capture registers

When an MRC or PAC detects a domain access violation, XRDC captures information about the transaction in the following registers:

Table 99. Domain error capture registers

Register[field]	Index	Information
DERRLOC*d*[MRCINST]	*d* = DID	Domain error location for MRC instances, with asserted bits indicating which MRC instance numbers are reporting an error

Table continues on the next page...

Table 99. Domain error capture registers (continued)

Register[field]	Index	Information
DERRLOC*d*[PACINST]	*d* = DID	Domain error location for PAC instances, with asserted bits indicating which PAC instance numbers are reporting an error
DERR_W0_*i*	*i* = instance number	Transaction target address
DERR_W1_*i*	*i* = instance number	Additional information about the transaction
DERR_W3_*i*	*i* = instance number	Reset and rearm domain error capture for the instance

19.7.3.2 Handling domain access violation errors

When an MRC or PAC instance detects a domain access violation, it reports the error by asserting the associated bit in DERRLOCd[MRCINST] or [PACINST], and XRDC asserts the error interrupt output. To retrieve information about the error, the error handler must:

1. Read each DERRLOC*d* register until it finds a non-zero MRCINST or PACINST value.

   The index of the DERRLOC*d* register is the DID for the domain in which the error occurred.

2. Configure the domain assignment for the master executing the exception handler (MDA_Ww_m_DFMTf[DID]) to assign the DID that corresponds to the DERRLOC*d* register index, such that the DERR_Ww_i registers show the error for that domain.

3. Read HWCFG1[DID] to be sure the error handler is now operating in the correct domain. In other words, make sure HWCFG1[DID] equals the value written to MDA_Ww_m_DFMTf[DID].

4. Find the number of an MRC or PAC instance reporting an error by parsing DERRLOC*d*[MRCINST] and DERRLOC*d*[PACINST] for an asserted bit.

   There may be multiple access violations, across multiple MRC or PAC instances, pending for a given domain. To quickly find the lowest numbered instance reporting an access violation, execute a "find first one bit" instruction (alternatively known as "count leading zeroes") on the MRCINST and PACINST fields.

5. Retrieve the error address (DERR_W0_*i*[EADDR]).

6. Retrieve the error information (DERR_W1_*i*).

   More than one error may have occurred in the MRC or PAC instance, as indicated by the error status (DERR_W1_*i*[EST] = 11b). If more than one error has occurred in the instance, XRDC captures data only for the first error.

7. Use the error address and information to handle the error (whatever that may require).

8. Reset the DERR_W*w_i* registers and rearm error capture (write 1b to DERR_W3_*i*[RECR]).

   Rearming error capture deasserts the instance bit in DERRLOC*d*.

9. Repeat steps 1 and 8 for each asserted bit in PACINST and MRCINST until there are no more asserted bits.

Domain error retrieval illustrates an example error retrieval.

19.7.3.3 Domain error retrieval

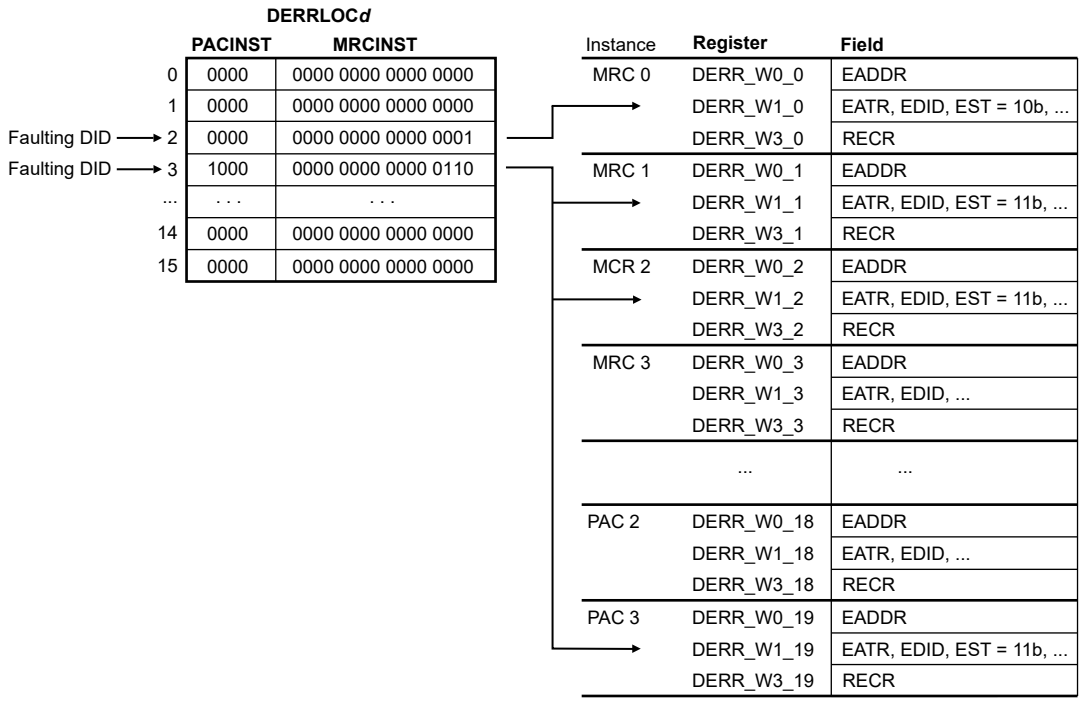


Figure 50. Domain error retrieval

19.8 Memory map and register definitions

19.8.1 Register organization

XRDC registers are partitioned into these groups:

- Basic hardware control and configuration
- Domain errors (including location and details)
- Master domain assignments
- Peripheral domain access controls
- Memory region descriptors

19.8.2 Register access guidelines

The following guidelines apply to XRDC register access:

- You can access the XRDC registers only in secure, privileged access mode.
- Unless stated otherwise, the registers support 8-, 16-, and 32-bit reads, and 32-bit writes.
- Unless stated otherwise, XRDC terminates the following access attempts with an error:
  - Accesses in a different access mode
  - Unsupported write data size
  - Writes to read-only resources
  - Writes to reserved address spaces

- Accesses to these memory map holes return an error:
  - Any access to a register that does not exist
  - Holes in the 0–F0h and DERR to PID register space
  - For MDAC, gaps in the master domain assignment (MDA_W*w*_*m*_DFMT*n*) registers
  - For MRCs, any gap in the MRGD_W*w*_*r* registers (for example, if there are four memory region descriptors, attempted access to a fifth descriptor fails)
- Accesses to these memory map holes do not return a bus error:
  - MRCs: Memory region descriptors occupy only four words but have an additional four words of address available: words 4–7
  - PDAC: Registers associated with unimplemented PDAC slots
- Read accesses to these memory map holes do not return a bus error:
  - Offset F8h and FCh
  - Offset 100–13Fh
  - Offset 140–14Fh
  - Offset 200–23Ch

19.8.3 XRDC register descriptions

19.8.3.1 XRDC memory map

XRDC base address: 4027_8000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Control (CR)	32	RW	0000_008Ah
F0h	Hardware Configuration 0 (HWCFG0)	32	R	1205_0904h
F4h	Hardware Configuration 1 (HWCFG1)	32	R	See section
F8h	Hardware Configuration 2 (HWCFG2)	32	R	0000_0000h
100h	Master Domain Assignment Configuration (MDACFG0)	8	R	01h
101h	Master Domain Assignment Configuration (MDACFG1)	8	R	81h
102h	Master Domain Assignment Configuration (MDACFG2)	8	R	81h
103h	Master Domain Assignment Configuration (MDACFG3)	8	R	01h
104h	Master Domain Assignment Configuration (MDACFG4)	8	R	01h
105h	Master Domain Assignment Configuration (MDACFG5)	8	R	81h
106h	Master Domain Assignment Configuration (MDACFG6)	8	R	01h
107h	Master Domain Assignment Configuration (MDACFG7)	8	R	81h
108h	Master Domain Assignment Configuration (MDACFG8)	8	R	01h
109h	Master Domain Assignment Configuration (MDACFG9)	8	R	81h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
140h	Memory Region Configuration (MRCFG0)	8	R	10h
141h	Memory Region Configuration (MRCFG1)	8	R	10h
142h	Memory Region Configuration (MRCFG2)	8	R	04h
143h	Memory Region Configuration (MRCFG3)	8	R	10h
144h	Memory Region Configuration (MRCFG4)	8	R	04h
145h	Memory Region Configuration (MRCFG5)	8	R	10h
200h - 210h	Domain Error Location (DERRLOC0 - DERRLOC4)	32	R	0000_0000h
400h	Domain Error Word 0 (DERR_W0_0)	32	R	0000_0000h
404h	Domain Error Word 1 (DERR_W1_0)	32	R	0000_0000h
40Ch	Domain Error Word 3 (DERR_W3_0)	32	RW	0000_0000h
410h	Domain Error Word 0 (DERR_W0_1)	32	R	0000_0000h
414h	Domain Error Word 1 (DERR_W1_1)	32	R	0000_0000h
41Ch	Domain Error Word 3 (DERR_W3_1)	32	RW	0000_0000h
420h	Domain Error Word 0 (DERR_W0_2)	32	R	0000_0000h
424h	Domain Error Word 1 (DERR_W1_2)	32	R	0000_0000h
42Ch	Domain Error Word 3 (DERR_W3_2)	32	RW	0000_0000h
430h	Domain Error Word 0 (DERR_W0_3)	32	R	0000_0000h
434h	Domain Error Word 1 (DERR_W1_3)	32	R	0000_0000h
43Ch	Domain Error Word 3 (DERR_W3_3)	32	RW	0000_0000h
440h	Domain Error Word 0 (DERR_W0_4)	32	R	0000_0000h
444h	Domain Error Word 1 (DERR_W1_4)	32	R	0000_0000h
44Ch	Domain Error Word 3 (DERR_W3_4)	32	RW	0000_0000h
450h	Domain Error Word 0 (DERR_W0_5)	32	R	0000_0000h
454h	Domain Error Word 1 (DERR_W1_5)	32	R	0000_0000h
45Ch	Domain Error Word 3 (DERR_W3_5)	32	RW	0000_0000h
500h	Domain Error Word 0 (DERR_W0_16)	32	R	0000_0000h
504h	Domain Error Word 1 (DERR_W1_16)	32	R	0000_0000h
50Ch	Domain Error Word 3 (DERR_W3_16)	32	RW	0000_0000h
510h	Domain Error Word 0 (DERR_W0_17)	32	R	0000_0000h
514h	Domain Error Word 1 (DERR_W1_17)	32	R	0000_0000h
51Ch	Domain Error Word 3 (DERR_W3_17)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
520h	Domain Error Word 0 (DERR_W0_18)	32	R	0000_0000h
524h	Domain Error Word 1 (DERR_W1_18)	32	R	0000_0000h
52Ch	Domain Error Word 3 (DERR_W3_18)	32	RW	0000_0000h
700h	Process Identifier (PID0)	32	RW	0000_0000h
70Ch	Process Identifier (PID3)	32	RW	0000_0000h
710h	Process Identifier (PID4)	32	RW	0000_0000h
718h	Process Identifier (PID6)	32	RW	0000_0000h
720h	Process Identifier (PID8)	32	RW	0000_0000h
800h	Master Domain Assignment (MDA_W0_0_DFMT0)	32	RW	0000_0000h
820h	Master Domain Assignment (MDA_W0_1_DFMT1)	32	RW	2000_0000h
840h	Master Domain Assignment (MDA_W0_2_DFMT1)	32	RW	2000_0000h
860h	Master Domain Assignment (MDA_W0_3_DFMT0)	32	RW	0000_0000h
880h	Master Domain Assignment (MDA_W0_4_DFMT0)	32	RW	0000_0000h
8A0h	Master Domain Assignment (MDA_W0_5_DFMT1)	32	RW	2000_0000h
8C0h	Master Domain Assignment (MDA_W0_6_DFMT0)	32	RW	0000_0000h
8E0h	Master Domain Assignment (MDA_W0_7_DFMT1)	32	RW	2000_0000h
900h	Master Domain Assignment (MDA_W0_8_DFMT0)	32	RW	0000_0000h
920h	Master Domain Assignment (MDA_W0_9_DFMT1)	32	RW	2000_0000h
1010h	Peripheral Domain Access Control Word 0 (PDAC_W0_2)	32	RW	0000_0000h
1014h	Peripheral Domain Access Control Word 1 (PDAC_W1_2)	32	RW	0000_0000h
1018h	Peripheral Domain Access Control Word 0 (PDAC_W0_3)	32	RW	0000_0000h
101Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_3)	32	RW	0000_0000h
10E0h	Peripheral Domain Access Control Word 0 (PDAC_W0_28)	32	RW	0000_0000h
10E4h	Peripheral Domain Access Control Word 1 (PDAC_W1_28)	32	RW	0000_0000h
1100h	Peripheral Domain Access Control Word 0 (PDAC_W0_32)	32	RW	0000_0000h
1104h	Peripheral Domain Access Control Word 1 (PDAC_W1_32)	32	RW	0000_0000h
1108h	Peripheral Domain Access Control Word 0 (PDAC_W0_33)	32	RW	0000_0000h
110Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_33)	32	RW	0000_0000h
1110h	Peripheral Domain Access Control Word 0 (PDAC_W0_34)	32	RW	0000_0000h
1114h	Peripheral Domain Access Control Word 1 (PDAC_W1_34)	32	RW	0000_0000h
1118h	Peripheral Domain Access Control Word 0 (PDAC_W0_35)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
111Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_35)	32	RW	0000_0000h
1120h	Peripheral Domain Access Control Word 0 (PDAC_W0_36)	32	RW	0000_0000h
1124h	Peripheral Domain Access Control Word 1 (PDAC_W1_36)	32	RW	0000_0000h
1130h	Peripheral Domain Access Control Word 0 (PDAC_W0_38)	32	RW	0000_0000h
1134h	Peripheral Domain Access Control Word 1 (PDAC_W1_38)	32	RW	0000_0000h
1138h	Peripheral Domain Access Control Word 0 (PDAC_W0_39)	32	RW	0000_0000h
113Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_39)	32	RW	0000_0000h
1140h	Peripheral Domain Access Control Word 0 (PDAC_W0_40)	32	RW	0000_0000h
1144h	Peripheral Domain Access Control Word 1 (PDAC_W1_40)	32	RW	0000_0000h
1148h	Peripheral Domain Access Control Word 0 (PDAC_W0_41)	32	RW	0000_0000h
114Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_41)	32	RW	0000_0000h
1150h	Peripheral Domain Access Control Word 0 (PDAC_W0_42)	32	RW	0000_0000h
1154h	Peripheral Domain Access Control Word 1 (PDAC_W1_42)	32	RW	0000_0000h
1160h	Peripheral Domain Access Control Word 0 (PDAC_W0_44)	32	RW	0000_0000h
1164h	Peripheral Domain Access Control Word 1 (PDAC_W1_44)	32	RW	0000_0000h
1168h	Peripheral Domain Access Control Word 0 (PDAC_W0_45)	32	RW	0000_0000h
116Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_45)	32	RW	0000_0000h
1170h	Peripheral Domain Access Control Word 0 (PDAC_W0_46)	32	RW	0000_0000h
1174h	Peripheral Domain Access Control Word 1 (PDAC_W1_46)	32	RW	0000_0000h
1178h	Peripheral Domain Access Control Word 0 (PDAC_W0_47)	32	RW	0000_0000h
117Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_47)	32	RW	0000_0000h
1188h	Peripheral Domain Access Control Word 0 (PDAC_W0_49)	32	RW	0000_0000h
118Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_49)	32	RW	0000_0000h
1190h	Peripheral Domain Access Control Word 0 (PDAC_W0_50)	32	RW	0000_0000h
1194h	Peripheral Domain Access Control Word 1 (PDAC_W1_50)	32	RW	0000_0000h
1198h	Peripheral Domain Access Control Word 0 (PDAC_W0_51)	32	RW	0000_0000h
119Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_51)	32	RW	0000_0000h
11A0h	Peripheral Domain Access Control Word 0 (PDAC_W0_52)	32	RW	0000_0000h
11A4h	Peripheral Domain Access Control Word 1 (PDAC_W1_52)	32	RW	0000_0000h
1400h	Peripheral Domain Access Control Word 0 (PDAC_W0_128)	32	RW	0000_0000h
1404h	Peripheral Domain Access Control Word 1 (PDAC_W1_128)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1408h	Peripheral Domain Access Control Word 0 (PDAC_W0_129)	32	RW	0000_0000h
140Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_129)	32	RW	0000_0000h
1410h	Peripheral Domain Access Control Word 0 (PDAC_W0_130)	32	RW	0000_0000h
1414h	Peripheral Domain Access Control Word 1 (PDAC_W1_130)	32	RW	0000_0000h
1418h	Peripheral Domain Access Control Word 0 (PDAC_W0_131)	32	RW	0000_0000h
141Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_131)	32	RW	0000_0000h
1420h	Peripheral Domain Access Control Word 0 (PDAC_W0_132)	32	RW	0000_0000h
1424h	Peripheral Domain Access Control Word 1 (PDAC_W1_132)	32	RW	0000_0000h
1428h	Peripheral Domain Access Control Word 0 (PDAC_W0_133)	32	RW	0000_0000h
142Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_133)	32	RW	0000_0000h
1430h	Peripheral Domain Access Control Word 0 (PDAC_W0_134)	32	RW	0000_0000h
1434h	Peripheral Domain Access Control Word 1 (PDAC_W1_134)	32	RW	0000_0000h
1438h	Peripheral Domain Access Control Word 0 (PDAC_W0_135)	32	RW	0000_0000h
143Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_135)	32	RW	0000_0000h
1440h	Peripheral Domain Access Control Word 0 (PDAC_W0_136)	32	RW	0000_0000h
1444h	Peripheral Domain Access Control Word 1 (PDAC_W1_136)	32	RW	0000_0000h
1448h	Peripheral Domain Access Control Word 0 (PDAC_W0_137)	32	RW	0000_0000h
144Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_137)	32	RW	0000_0000h
1450h	Peripheral Domain Access Control Word 0 (PDAC_W0_138)	32	RW	0000_0000h
1454h	Peripheral Domain Access Control Word 1 (PDAC_W1_138)	32	RW	0000_0000h
1458h	Peripheral Domain Access Control Word 0 (PDAC_W0_139)	32	RW	0000_0000h
145Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_139)	32	RW	0000_0000h
1460h	Peripheral Domain Access Control Word 0 (PDAC_W0_140)	32	RW	0000_0000h
1464h	Peripheral Domain Access Control Word 1 (PDAC_W1_140)	32	RW	0000_0000h
1468h	Peripheral Domain Access Control Word 0 (PDAC_W0_141)	32	RW	0000_0000h
146Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_141)	32	RW	0000_0000h
1470h	Peripheral Domain Access Control Word 0 (PDAC_W0_142)	32	RW	0000_0000h
1474h	Peripheral Domain Access Control Word 1 (PDAC_W1_142)	32	RW	0000_0000h
1478h	Peripheral Domain Access Control Word 0 (PDAC_W0_143)	32	RW	0000_0000h
147Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_143)	32	RW	0000_0000h
1480h	Peripheral Domain Access Control Word 0 (PDAC_W0_144)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1484h	Peripheral Domain Access Control Word 1 (PDAC_W1_144)	32	RW	0000_0000h
1488h	Peripheral Domain Access Control Word 0 (PDAC_W0_145)	32	RW	0000_0000h
148Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_145)	32	RW	0000_0000h
1490h	Peripheral Domain Access Control Word 0 (PDAC_W0_146)	32	RW	0000_0000h
1494h	Peripheral Domain Access Control Word 1 (PDAC_W1_146)	32	RW	0000_0000h
1498h	Peripheral Domain Access Control Word 0 (PDAC_W0_147)	32	RW	0000_0000h
149Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_147)	32	RW	0000_0000h
14A0h	Peripheral Domain Access Control Word 0 (PDAC_W0_148)	32	RW	0000_0000h
14A4h	Peripheral Domain Access Control Word 1 (PDAC_W1_148)	32	RW	0000_0000h
14A8h	Peripheral Domain Access Control Word 0 (PDAC_W0_149)	32	RW	0000_0000h
14ACh	Peripheral Domain Access Control Word 1 (PDAC_W1_149)	32	RW	0000_0000h
14B8h	Peripheral Domain Access Control Word 0 (PDAC_W0_151)	32	RW	0000_0000h
14BCh	Peripheral Domain Access Control Word 1 (PDAC_W1_151)	32	RW	0000_0000h
14C0h	Peripheral Domain Access Control Word 0 (PDAC_W0_152)	32	RW	0000_0000h
14C4h	Peripheral Domain Access Control Word 1 (PDAC_W1_152)	32	RW	0000_0000h
14C8h	Peripheral Domain Access Control Word 0 (PDAC_W0_153)	32	RW	0000_0000h
14CCh	Peripheral Domain Access Control Word 1 (PDAC_W1_153)	32	RW	0000_0000h
14D0h	Peripheral Domain Access Control Word 0 (PDAC_W0_154)	32	RW	0000_0000h
14D4h	Peripheral Domain Access Control Word 1 (PDAC_W1_154)	32	RW	0000_0000h
14D8h	Peripheral Domain Access Control Word 0 (PDAC_W0_155)	32	RW	0000_0000h
14DCh	Peripheral Domain Access Control Word 1 (PDAC_W1_155)	32	RW	0000_0000h
14E0h	Peripheral Domain Access Control Word 0 (PDAC_W0_156)	32	RW	0000_0000h
14E4h	Peripheral Domain Access Control Word 1 (PDAC_W1_156)	32	RW	0000_0000h
14E8h	Peripheral Domain Access Control Word 0 (PDAC_W0_157)	32	RW	0000_0000h
14ECh	Peripheral Domain Access Control Word 1 (PDAC_W1_157)	32	RW	0000_0000h
14F0h	Peripheral Domain Access Control Word 0 (PDAC_W0_158)	32	RW	0000_0000h
14F4h	Peripheral Domain Access Control Word 1 (PDAC_W1_158)	32	RW	0000_0000h
14F8h	Peripheral Domain Access Control Word 0 (PDAC_W0_159)	32	RW	0000_0000h
14FCh	Peripheral Domain Access Control Word 1 (PDAC_W1_159)	32	RW	0000_0000h
1500h	Peripheral Domain Access Control Word 0 (PDAC_W0_160)	32	RW	0000_0000h
1504h	Peripheral Domain Access Control Word 1 (PDAC_W1_160)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1508h	Peripheral Domain Access Control Word 0 (PDAC_W0_161)	32	RW	0000_0000h
150Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_161)	32	RW	0000_0000h
1510h	Peripheral Domain Access Control Word 0 (PDAC_W0_162)	32	RW	0000_0000h
1514h	Peripheral Domain Access Control Word 1 (PDAC_W1_162)	32	RW	0000_0000h
1518h	Peripheral Domain Access Control Word 0 (PDAC_W0_163)	32	RW	0000_0000h
151Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_163)	32	RW	0000_0000h
1520h	Peripheral Domain Access Control Word 0 (PDAC_W0_164)	32	RW	0000_0000h
1524h	Peripheral Domain Access Control Word 1 (PDAC_W1_164)	32	RW	0000_0000h
1528h	Peripheral Domain Access Control Word 0 (PDAC_W0_165)	32	RW	0000_0000h
152Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_165)	32	RW	0000_0000h
1530h	Peripheral Domain Access Control Word 0 (PDAC_W0_166)	32	RW	0000_0000h
1534h	Peripheral Domain Access Control Word 1 (PDAC_W1_166)	32	RW	0000_0000h
1538h	Peripheral Domain Access Control Word 0 (PDAC_W0_167)	32	RW	0000_0000h
153Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_167)	32	RW	0000_0000h
1540h	Peripheral Domain Access Control Word 0 (PDAC_W0_168)	32	RW	0000_0000h
1544h	Peripheral Domain Access Control Word 1 (PDAC_W1_168)	32	RW	0000_0000h
1548h	Peripheral Domain Access Control Word 0 (PDAC_W0_169)	32	RW	0000_0000h
154Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_169)	32	RW	0000_0000h
1550h	Peripheral Domain Access Control Word 0 (PDAC_W0_170)	32	RW	0000_0000h
1554h	Peripheral Domain Access Control Word 1 (PDAC_W1_170)	32	RW	0000_0000h
1558h	Peripheral Domain Access Control Word 0 (PDAC_W0_171)	32	RW	0000_0000h
155Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_171)	32	RW	0000_0000h
1568h	Peripheral Domain Access Control Word 0 (PDAC_W0_173)	32	RW	0000_0000h
156Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_173)	32	RW	0000_0000h
1578h	Peripheral Domain Access Control Word 0 (PDAC_W0_175)	32	RW	0000_0000h
157Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_175)	32	RW	0000_0000h
1588h	Peripheral Domain Access Control Word 0 (PDAC_W0_177)	32	RW	0000_0000h
158Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_177)	32	RW	0000_0000h
1590h	Peripheral Domain Access Control Word 0 (PDAC_W0_178)	32	RW	0000_0000h
1594h	Peripheral Domain Access Control Word 1 (PDAC_W1_178)	32	RW	0000_0000h
1598h	Peripheral Domain Access Control Word 0 (PDAC_W0_179)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
159Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_179)	32	RW	0000_0000h
15A0h	Peripheral Domain Access Control Word 0 (PDAC_W0_180)	32	RW	0000_0000h
15A4h	Peripheral Domain Access Control Word 1 (PDAC_W1_180)	32	RW	0000_0000h
15A8h	Peripheral Domain Access Control Word 0 (PDAC_W0_181)	32	RW	0000_0000h
15ACh	Peripheral Domain Access Control Word 1 (PDAC_W1_181)	32	RW	0000_0000h
15B0h	Peripheral Domain Access Control Word 0 (PDAC_W0_182)	32	RW	0000_0000h
15B4h	Peripheral Domain Access Control Word 1 (PDAC_W1_182)	32	RW	0000_0000h
15B8h	Peripheral Domain Access Control Word 0 (PDAC_W0_183)	32	RW	0000_0000h
15BCh	Peripheral Domain Access Control Word 1 (PDAC_W1_183)	32	RW	0000_0000h
15C0h	Peripheral Domain Access Control Word 0 (PDAC_W0_184)	32	RW	0000_0000h
15C4h	Peripheral Domain Access Control Word 1 (PDAC_W1_184)	32	RW	0000_0000h
15C8h	Peripheral Domain Access Control Word 0 (PDAC_W0_185)	32	RW	0000_0000h
15CCh	Peripheral Domain Access Control Word 1 (PDAC_W1_185)	32	RW	0000_0000h
15D0h	Peripheral Domain Access Control Word 0 (PDAC_W0_186)	32	RW	0000_0000h
15D4h	Peripheral Domain Access Control Word 1 (PDAC_W1_186)	32	RW	0000_0000h
15D8h	Peripheral Domain Access Control Word 0 (PDAC_W0_187)	32	RW	0000_0000h
15DCh	Peripheral Domain Access Control Word 1 (PDAC_W1_187)	32	RW	0000_0000h
15E0h	Peripheral Domain Access Control Word 0 (PDAC_W0_188)	32	RW	0000_0000h
15E4h	Peripheral Domain Access Control Word 1 (PDAC_W1_188)	32	RW	0000_0000h
15E8h	Peripheral Domain Access Control Word 0 (PDAC_W0_189)	32	RW	0000_0000h
15ECh	Peripheral Domain Access Control Word 1 (PDAC_W1_189)	32	RW	0000_0000h
15F0h	Peripheral Domain Access Control Word 0 (PDAC_W0_190)	32	RW	0000_0000h
15F4h	Peripheral Domain Access Control Word 1 (PDAC_W1_190)	32	RW	0000_0000h
15F8h	Peripheral Domain Access Control Word 0 (PDAC_W0_191)	32	RW	0000_0000h
15FCh	Peripheral Domain Access Control Word 1 (PDAC_W1_191)	32	RW	0000_0000h
1600h	Peripheral Domain Access Control Word 0 (PDAC_W0_192)	32	RW	0000_0000h
1604h	Peripheral Domain Access Control Word 1 (PDAC_W1_192)	32	RW	0000_0000h
1608h	Peripheral Domain Access Control Word 0 (PDAC_W0_193)	32	RW	0000_0000h
160Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_193)	32	RW	0000_0000h
1610h	Peripheral Domain Access Control Word 0 (PDAC_W0_194)	32	RW	0000_0000h
1614h	Peripheral Domain Access Control Word 1 (PDAC_W1_194)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1618h	Peripheral Domain Access Control Word 0 (PDAC_W0_195)	32	RW	0000_0000h
161Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_195)	32	RW	0000_0000h
1620h	Peripheral Domain Access Control Word 0 (PDAC_W0_196)	32	RW	0000_0000h
1624h	Peripheral Domain Access Control Word 1 (PDAC_W1_196)	32	RW	0000_0000h
1628h	Peripheral Domain Access Control Word 0 (PDAC_W0_197)	32	RW	0000_0000h
162Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_197)	32	RW	0000_0000h
1630h	Peripheral Domain Access Control Word 0 (PDAC_W0_198)	32	RW	0000_0000h
1634h	Peripheral Domain Access Control Word 1 (PDAC_W1_198)	32	RW	0000_0000h
1638h	Peripheral Domain Access Control Word 0 (PDAC_W0_199)	32	RW	0000_0000h
163Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_199)	32	RW	0000_0000h
1640h	Peripheral Domain Access Control Word 0 (PDAC_W0_200)	32	RW	0000_0000h
1644h	Peripheral Domain Access Control Word 1 (PDAC_W1_200)	32	RW	0000_0000h
1648h	Peripheral Domain Access Control Word 0 (PDAC_W0_201)	32	RW	0000_0000h
164Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_201)	32	RW	0000_0000h
1650h	Peripheral Domain Access Control Word 0 (PDAC_W0_202)	32	RW	0000_0000h
1654h	Peripheral Domain Access Control Word 1 (PDAC_W1_202)	32	RW	0000_0000h
1658h	Peripheral Domain Access Control Word 0 (PDAC_W0_203)	32	RW	0000_0000h
165Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_203)	32	RW	0000_0000h
1660h	Peripheral Domain Access Control Word 0 (PDAC_W0_204)	32	RW	0000_0000h
1664h	Peripheral Domain Access Control Word 1 (PDAC_W1_204)	32	RW	0000_0000h
1668h	Peripheral Domain Access Control Word 0 (PDAC_W0_205)	32	RW	0000_0000h
166Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_205)	32	RW	0000_0000h
1670h	Peripheral Domain Access Control Word 0 (PDAC_W0_206)	32	RW	0000_0000h
1674h	Peripheral Domain Access Control Word 1 (PDAC_W1_206)	32	RW	0000_0000h
1678h	Peripheral Domain Access Control Word 0 (PDAC_W0_207)	32	RW	0000_0000h
167Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_207)	32	RW	0000_0000h
1680h	Peripheral Domain Access Control Word 0 (PDAC_W0_208)	32	RW	0000_0000h
1684h	Peripheral Domain Access Control Word 1 (PDAC_W1_208)	32	RW	0000_0000h
1688h	Peripheral Domain Access Control Word 0 (PDAC_W0_209)	32	RW	0000_0000h
168Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_209)	32	RW	0000_0000h
1690h	Peripheral Domain Access Control Word 0 (PDAC_W0_210)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1694h	Peripheral Domain Access Control Word 1 (PDAC_W1_210)	32	RW	0000_0000h
1698h	Peripheral Domain Access Control Word 0 (PDAC_W0_211)	32	RW	0000_0000h
169Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_211)	32	RW	0000_0000h
16A0h	Peripheral Domain Access Control Word 0 (PDAC_W0_212)	32	RW	0000_0000h
16A4h	Peripheral Domain Access Control Word 1 (PDAC_W1_212)	32	RW	0000_0000h
16A8h	Peripheral Domain Access Control Word 0 (PDAC_W0_213)	32	RW	0000_0000h
16ACh	Peripheral Domain Access Control Word 1 (PDAC_W1_213)	32	RW	0000_0000h
16B0h	Peripheral Domain Access Control Word 0 (PDAC_W0_214)	32	RW	0000_0000h
16B4h	Peripheral Domain Access Control Word 1 (PDAC_W1_214)	32	RW	0000_0000h
16B8h	Peripheral Domain Access Control Word 0 (PDAC_W0_215)	32	RW	0000_0000h
16BCh	Peripheral Domain Access Control Word 1 (PDAC_W1_215)	32	RW	0000_0000h
16C0h	Peripheral Domain Access Control Word 0 (PDAC_W0_216)	32	RW	0000_0000h
16C4h	Peripheral Domain Access Control Word 1 (PDAC_W1_216)	32	RW	0000_0000h
16C8h	Peripheral Domain Access Control Word 0 (PDAC_W0_217)	32	RW	0000_0000h
16CCh	Peripheral Domain Access Control Word 1 (PDAC_W1_217)	32	RW	0000_0000h
16D8h	Peripheral Domain Access Control Word 0 (PDAC_W0_219)	32	RW	0000_0000h
16DCh	Peripheral Domain Access Control Word 1 (PDAC_W1_219)	32	RW	0000_0000h
16E0h	Peripheral Domain Access Control Word 0 (PDAC_W0_220)	32	RW	0000_0000h
16E4h	Peripheral Domain Access Control Word 1 (PDAC_W1_220)	32	RW	0000_0000h
16E8h	Peripheral Domain Access Control Word 0 (PDAC_W0_221)	32	RW	0000_0000h
16ECh	Peripheral Domain Access Control Word 1 (PDAC_W1_221)	32	RW	0000_0000h
16F8h	Peripheral Domain Access Control Word 0 (PDAC_W0_223)	32	RW	0000_0000h
16FCh	Peripheral Domain Access Control Word 1 (PDAC_W1_223)	32	RW	0000_0000h
1700h	Peripheral Domain Access Control Word 0 (PDAC_W0_224)	32	RW	0000_0000h
1704h	Peripheral Domain Access Control Word 1 (PDAC_W1_224)	32	RW	0000_0000h
1708h	Peripheral Domain Access Control Word 0 (PDAC_W0_225)	32	RW	0000_0000h
170Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_225)	32	RW	0000_0000h
1710h	Peripheral Domain Access Control Word 0 (PDAC_W0_226)	32	RW	0000_0000h
1714h	Peripheral Domain Access Control Word 1 (PDAC_W1_226)	32	RW	0000_0000h
1718h	Peripheral Domain Access Control Word 0 (PDAC_W0_227)	32	RW	0000_0000h
171Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_227)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1728h	Peripheral Domain Access Control Word 0 (PDAC_W0_229)	32	RW	0000_0000h
172Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_229)	32	RW	0000_0000h
1730h	Peripheral Domain Access Control Word 0 (PDAC_W0_230)	32	RW	0000_0000h
1734h	Peripheral Domain Access Control Word 1 (PDAC_W1_230)	32	RW	0000_0000h
1738h	Peripheral Domain Access Control Word 0 (PDAC_W0_231)	32	RW	0000_0000h
173Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_231)	32	RW	0000_0000h
1740h	Peripheral Domain Access Control Word 0 (PDAC_W0_232)	32	RW	0000_0000h
1744h	Peripheral Domain Access Control Word 1 (PDAC_W1_232)	32	RW	0000_0000h
1748h	Peripheral Domain Access Control Word 0 (PDAC_W0_233)	32	RW	0000_0000h
174Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_233)	32	RW	0000_0000h
1750h	Peripheral Domain Access Control Word 0 (PDAC_W0_234)	32	RW	0000_0000h
1754h	Peripheral Domain Access Control Word 1 (PDAC_W1_234)	32	RW	0000_0000h
1758h	Peripheral Domain Access Control Word 0 (PDAC_W0_235)	32	RW	0000_0000h
175Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_235)	32	RW	0000_0000h
1760h	Peripheral Domain Access Control Word 0 (PDAC_W0_236)	32	RW	0000_0000h
1764h	Peripheral Domain Access Control Word 1 (PDAC_W1_236)	32	RW	0000_0000h
1770h	Peripheral Domain Access Control Word 0 (PDAC_W0_238)	32	RW	0000_0000h
1774h	Peripheral Domain Access Control Word 1 (PDAC_W1_238)	32	RW	0000_0000h
1780h	Peripheral Domain Access Control Word 0 (PDAC_W0_240)	32	RW	0000_0000h
1784h	Peripheral Domain Access Control Word 1 (PDAC_W1_240)	32	RW	0000_0000h
1788h	Peripheral Domain Access Control Word 0 (PDAC_W0_241)	32	RW	0000_0000h
178Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_241)	32	RW	0000_0000h
1790h	Peripheral Domain Access Control Word 0 (PDAC_W0_242)	32	RW	0000_0000h
1794h	Peripheral Domain Access Control Word 1 (PDAC_W1_242)	32	RW	0000_0000h
1798h	Peripheral Domain Access Control Word 0 (PDAC_W0_243)	32	RW	0000_0000h
179Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_243)	32	RW	0000_0000h
17A0h	Peripheral Domain Access Control Word 0 (PDAC_W0_244)	32	RW	0000_0000h
17A4h	Peripheral Domain Access Control Word 1 (PDAC_W1_244)	32	RW	0000_0000h
17A8h	Peripheral Domain Access Control Word 0 (PDAC_W0_245)	32	RW	0000_0000h
17ACh	Peripheral Domain Access Control Word 1 (PDAC_W1_245)	32	RW	0000_0000h
17B0h	Peripheral Domain Access Control Word 0 (PDAC_W0_246)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
17B4h	Peripheral Domain Access Control Word 1 (PDAC_W1_246)	32	RW	0000_0000h
17B8h	Peripheral Domain Access Control Word 0 (PDAC_W0_247)	32	RW	0000_0000h
17BCh	Peripheral Domain Access Control Word 1 (PDAC_W1_247)	32	RW	0000_0000h
17C0h	Peripheral Domain Access Control Word 0 (PDAC_W0_248)	32	RW	0000_0000h
17C4h	Peripheral Domain Access Control Word 1 (PDAC_W1_248)	32	RW	0000_0000h
17C8h	Peripheral Domain Access Control Word 0 (PDAC_W0_249)	32	RW	0000_0000h
17CCh	Peripheral Domain Access Control Word 1 (PDAC_W1_249)	32	RW	0000_0000h
17D0h	Peripheral Domain Access Control Word 0 (PDAC_W0_250)	32	RW	0000_0000h
17D4h	Peripheral Domain Access Control Word 1 (PDAC_W1_250)	32	RW	0000_0000h
17D8h	Peripheral Domain Access Control Word 0 (PDAC_W0_251)	32	RW	0000_0000h
17DCh	Peripheral Domain Access Control Word 1 (PDAC_W1_251)	32	RW	0000_0000h
17E0h	Peripheral Domain Access Control Word 0 (PDAC_W0_252)	32	RW	0000_0000h
17E4h	Peripheral Domain Access Control Word 1 (PDAC_W1_252)	32	RW	0000_0000h
17E8h	Peripheral Domain Access Control Word 0 (PDAC_W0_253)	32	RW	0000_0000h
17ECh	Peripheral Domain Access Control Word 1 (PDAC_W1_253)	32	RW	0000_0000h
17F0h	Peripheral Domain Access Control Word 0 (PDAC_W0_254)	32	RW	0000_0000h
17F4h	Peripheral Domain Access Control Word 1 (PDAC_W1_254)	32	RW	0000_0000h
17F8h	Peripheral Domain Access Control Word 0 (PDAC_W0_255)	32	RW	0000_0000h
17FCh	Peripheral Domain Access Control Word 1 (PDAC_W1_255)	32	RW	0000_0000h
1800h	Peripheral Domain Access Control Word 0 (PDAC_W0_256)	32	RW	0000_0000h
1804h	Peripheral Domain Access Control Word 1 (PDAC_W1_256)	32	RW	0000_0000h
1808h	Peripheral Domain Access Control Word 0 (PDAC_W0_257)	32	RW	0000_0000h
180Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_257)	32	RW	0000_0000h
1810h	Peripheral Domain Access Control Word 0 (PDAC_W0_258)	32	RW	0000_0000h
1814h	Peripheral Domain Access Control Word 1 (PDAC_W1_258)	32	RW	0000_0000h
1818h	Peripheral Domain Access Control Word 0 (PDAC_W0_259)	32	RW	0000_0000h
181Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_259)	32	RW	0000_0000h
1820h	Peripheral Domain Access Control Word 0 (PDAC_W0_260)	32	RW	0000_0000h
1824h	Peripheral Domain Access Control Word 1 (PDAC_W1_260)	32	RW	0000_0000h
1828h	Peripheral Domain Access Control Word 0 (PDAC_W0_261)	32	RW	0000_0000h
182Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_261)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1830h	Peripheral Domain Access Control Word 0 (PDAC_W0_262)	32	RW	0000_0000h
1834h	Peripheral Domain Access Control Word 1 (PDAC_W1_262)	32	RW	0000_0000h
1838h	Peripheral Domain Access Control Word 0 (PDAC_W0_263)	32	RW	0000_0000h
183Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_263)	32	RW	0000_0000h
1840h	Peripheral Domain Access Control Word 0 (PDAC_W0_264)	32	RW	0000_0000h
1844h	Peripheral Domain Access Control Word 1 (PDAC_W1_264)	32	RW	0000_0000h
1848h	Peripheral Domain Access Control Word 0 (PDAC_W0_265)	32	RW	0000_0000h
184Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_265)	32	RW	0000_0000h
1850h	Peripheral Domain Access Control Word 0 (PDAC_W0_266)	32	RW	0000_0000h
1854h	Peripheral Domain Access Control Word 1 (PDAC_W1_266)	32	RW	0000_0000h
1858h	Peripheral Domain Access Control Word 0 (PDAC_W0_267)	32	RW	0000_0000h
185Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_267)	32	RW	0000_0000h
1860h	Peripheral Domain Access Control Word 0 (PDAC_W0_268)	32	RW	0000_0000h
1864h	Peripheral Domain Access Control Word 1 (PDAC_W1_268)	32	RW	0000_0000h
1868h	Peripheral Domain Access Control Word 0 (PDAC_W0_269)	32	RW	0000_0000h
186Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_269)	32	RW	0000_0000h
1870h	Peripheral Domain Access Control Word 0 (PDAC_W0_270)	32	RW	0000_0000h
1874h	Peripheral Domain Access Control Word 1 (PDAC_W1_270)	32	RW	0000_0000h
1878h	Peripheral Domain Access Control Word 0 (PDAC_W0_271)	32	RW	0000_0000h
187Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_271)	32	RW	0000_0000h
1880h	Peripheral Domain Access Control Word 0 (PDAC_W0_272)	32	RW	0000_0000h
1884h	Peripheral Domain Access Control Word 1 (PDAC_W1_272)	32	RW	0000_0000h
1888h	Peripheral Domain Access Control Word 0 (PDAC_W0_273)	32	RW	0000_0000h
188Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_273)	32	RW	0000_0000h
1890h	Peripheral Domain Access Control Word 0 (PDAC_W0_274)	32	RW	0000_0000h
1894h	Peripheral Domain Access Control Word 1 (PDAC_W1_274)	32	RW	0000_0000h
1898h	Peripheral Domain Access Control Word 0 (PDAC_W0_275)	32	RW	0000_0000h
189Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_275)	32	RW	0000_0000h
18A0h	Peripheral Domain Access Control Word 0 (PDAC_W0_276)	32	RW	0000_0000h
18A4h	Peripheral Domain Access Control Word 1 (PDAC_W1_276)	32	RW	0000_0000h
18A8h	Peripheral Domain Access Control Word 0 (PDAC_W0_277)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
18ACh	Peripheral Domain Access Control Word 1 (PDAC_W1_277)	32	RW	0000_0000h
18B0h	Peripheral Domain Access Control Word 0 (PDAC_W0_278)	32	RW	0000_0000h
18B4h	Peripheral Domain Access Control Word 1 (PDAC_W1_278)	32	RW	0000_0000h
18B8h	Peripheral Domain Access Control Word 0 (PDAC_W0_279)	32	RW	0000_0000h
18BCh	Peripheral Domain Access Control Word 1 (PDAC_W1_279)	32	RW	0000_0000h
18C0h	Peripheral Domain Access Control Word 0 (PDAC_W0_280)	32	RW	0000_0000h
18C4h	Peripheral Domain Access Control Word 1 (PDAC_W1_280)	32	RW	0000_0000h
18C8h	Peripheral Domain Access Control Word 0 (PDAC_W0_281)	32	RW	0000_0000h
18CCh	Peripheral Domain Access Control Word 1 (PDAC_W1_281)	32	RW	0000_0000h
18D0h	Peripheral Domain Access Control Word 0 (PDAC_W0_282)	32	RW	0000_0000h
18D4h	Peripheral Domain Access Control Word 1 (PDAC_W1_282)	32	RW	0000_0000h
18D8h	Peripheral Domain Access Control Word 0 (PDAC_W0_283)	32	RW	0000_0000h
18DCh	Peripheral Domain Access Control Word 1 (PDAC_W1_283)	32	RW	0000_0000h
18E0h	Peripheral Domain Access Control Word 0 (PDAC_W0_284)	32	RW	0000_0000h
18E4h	Peripheral Domain Access Control Word 1 (PDAC_W1_284)	32	RW	0000_0000h
18E8h	Peripheral Domain Access Control Word 0 (PDAC_W0_285)	32	RW	0000_0000h
18ECh	Peripheral Domain Access Control Word 1 (PDAC_W1_285)	32	RW	0000_0000h
18F0h	Peripheral Domain Access Control Word 0 (PDAC_W0_286)	32	RW	0000_0000h
18F4h	Peripheral Domain Access Control Word 1 (PDAC_W1_286)	32	RW	0000_0000h
18F8h	Peripheral Domain Access Control Word 0 (PDAC_W0_287)	32	RW	0000_0000h
18FCh	Peripheral Domain Access Control Word 1 (PDAC_W1_287)	32	RW	0000_0000h
1908h	Peripheral Domain Access Control Word 0 (PDAC_W0_289)	32	RW	0000_0000h
190Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_289)	32	RW	0000_0000h
1910h	Peripheral Domain Access Control Word 0 (PDAC_W0_290)	32	RW	0000_0000h
1914h	Peripheral Domain Access Control Word 1 (PDAC_W1_290)	32	RW	0000_0000h
1918h	Peripheral Domain Access Control Word 0 (PDAC_W0_291)	32	RW	0000_0000h
191Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_291)	32	RW	0000_0000h
1920h	Peripheral Domain Access Control Word 0 (PDAC_W0_292)	32	RW	0000_0000h
1924h	Peripheral Domain Access Control Word 1 (PDAC_W1_292)	32	RW	0000_0000h
1928h	Peripheral Domain Access Control Word 0 (PDAC_W0_293)	32	RW	0000_0000h
192Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_293)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1930h	Peripheral Domain Access Control Word 0 (PDAC_W0_294)	32	RW	0000_0000h
1934h	Peripheral Domain Access Control Word 1 (PDAC_W1_294)	32	RW	0000_0000h
1938h	Peripheral Domain Access Control Word 0 (PDAC_W0_295)	32	RW	0000_0000h
193Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_295)	32	RW	0000_0000h
1940h	Peripheral Domain Access Control Word 0 (PDAC_W0_296)	32	RW	0000_0000h
1944h	Peripheral Domain Access Control Word 1 (PDAC_W1_296)	32	RW	0000_0000h
1948h	Peripheral Domain Access Control Word 0 (PDAC_W0_297)	32	RW	0000_0000h
194Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_297)	32	RW	0000_0000h
1950h	Peripheral Domain Access Control Word 0 (PDAC_W0_298)	32	RW	0000_0000h
1954h	Peripheral Domain Access Control Word 1 (PDAC_W1_298)	32	RW	0000_0000h
1978h	Peripheral Domain Access Control Word 0 (PDAC_W0_303)	32	RW	0000_0000h
197Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_303)	32	RW	0000_0000h
1980h	Peripheral Domain Access Control Word 0 (PDAC_W0_304)	32	RW	0000_0000h
1984h	Peripheral Domain Access Control Word 1 (PDAC_W1_304)	32	RW	0000_0000h
1998h	Peripheral Domain Access Control Word 0 (PDAC_W0_307)	32	RW	0000_0000h
199Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_307)	32	RW	0000_0000h
19B8h	Peripheral Domain Access Control Word 0 (PDAC_W0_311)	32	RW	0000_0000h
19BCh	Peripheral Domain Access Control Word 1 (PDAC_W1_311)	32	RW	0000_0000h
19D0h	Peripheral Domain Access Control Word 0 (PDAC_W0_314)	32	RW	0000_0000h
19D4h	Peripheral Domain Access Control Word 1 (PDAC_W1_314)	32	RW	0000_0000h
19D8h	Peripheral Domain Access Control Word 0 (PDAC_W0_315)	32	RW	0000_0000h
19DCh	Peripheral Domain Access Control Word 1 (PDAC_W1_315)	32	RW	0000_0000h
19F0h	Peripheral Domain Access Control Word 0 (PDAC_W0_318)	32	RW	0000_0000h
19F4h	Peripheral Domain Access Control Word 1 (PDAC_W1_318)	32	RW	0000_0000h
19F8h	Peripheral Domain Access Control Word 0 (PDAC_W0_319)	32	RW	0000_0000h
19FCh	Peripheral Domain Access Control Word 1 (PDAC_W1_319)	32	RW	0000_0000h
1A00h	Peripheral Domain Access Control Word 0 (PDAC_W0_320)	32	RW	0000_0000h
1A04h	Peripheral Domain Access Control Word 1 (PDAC_W1_320)	32	RW	0000_0000h
1A08h	Peripheral Domain Access Control Word 0 (PDAC_W0_321)	32	RW	0000_0000h
1A0Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_321)	32	RW	0000_0000h
1A18h	Peripheral Domain Access Control Word 0 (PDAC_W0_323)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1A1Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_323)	32	RW	0000_0000h
1A20h	Peripheral Domain Access Control Word 0 (PDAC_W0_324)	32	RW	0000_0000h
1A24h	Peripheral Domain Access Control Word 1 (PDAC_W1_324)	32	RW	0000_0000h
1A28h	Peripheral Domain Access Control Word 0 (PDAC_W0_325)	32	RW	0000_0000h
1A2Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_325)	32	RW	0000_0000h
1A30h	Peripheral Domain Access Control Word 0 (PDAC_W0_326)	32	RW	0000_0000h
1A34h	Peripheral Domain Access Control Word 1 (PDAC_W1_326)	32	RW	0000_0000h
1A40h	Peripheral Domain Access Control Word 0 (PDAC_W0_328)	32	RW	0000_0000h
1A44h	Peripheral Domain Access Control Word 1 (PDAC_W1_328)	32	RW	0000_0000h
1A48h	Peripheral Domain Access Control Word 0 (PDAC_W0_329)	32	RW	0000_0000h
1A4Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_329)	32	RW	0000_0000h
1A50h	Peripheral Domain Access Control Word 0 (PDAC_W0_330)	32	RW	0000_0000h
1A54h	Peripheral Domain Access Control Word 1 (PDAC_W1_330)	32	RW	0000_0000h
1A58h	Peripheral Domain Access Control Word 0 (PDAC_W0_331)	32	RW	0000_0000h
1A5Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_331)	32	RW	0000_0000h
1A60h	Peripheral Domain Access Control Word 0 (PDAC_W0_332)	32	RW	0000_0000h
1A64h	Peripheral Domain Access Control Word 1 (PDAC_W1_332)	32	RW	0000_0000h
1A68h	Peripheral Domain Access Control Word 0 (PDAC_W0_333)	32	RW	0000_0000h
1A6Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_333)	32	RW	0000_0000h
1A70h	Peripheral Domain Access Control Word 0 (PDAC_W0_334)	32	RW	0000_0000h
1A74h	Peripheral Domain Access Control Word 1 (PDAC_W1_334)	32	RW	0000_0000h
1A78h	Peripheral Domain Access Control Word 0 (PDAC_W0_335)	32	RW	0000_0000h
1A7Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_335)	32	RW	0000_0000h
1A88h	Peripheral Domain Access Control Word 0 (PDAC_W0_337)	32	RW	0000_0000h
1A8Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_337)	32	RW	0000_0000h
1A90h	Peripheral Domain Access Control Word 0 (PDAC_W0_338)	32	RW	0000_0000h
1A94h	Peripheral Domain Access Control Word 1 (PDAC_W1_338)	32	RW	0000_0000h
1A98h	Peripheral Domain Access Control Word 0 (PDAC_W0_339)	32	RW	0000_0000h
1A9Ch	Peripheral Domain Access Control Word 1 (PDAC_W1_339)	32	RW	0000_0000h
1AA0h	Peripheral Domain Access Control Word 0 (PDAC_W0_340)	32	RW	0000_0000h
1AA4h	Peripheral Domain Access Control Word 1 (PDAC_W1_340)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1AA8h	Peripheral Domain Access Control Word 0 (PDAC_W0_341)	32	RW	0000_0000h
1AACh	Peripheral Domain Access Control Word 1 (PDAC_W1_341)	32	RW	0000_0000h
1AB0h	Peripheral Domain Access Control Word 0 (PDAC_W0_342)	32	RW	0000_0000h
1AB4h	Peripheral Domain Access Control Word 1 (PDAC_W1_342)	32	RW	0000_0000h
1AB8h	Peripheral Domain Access Control Word 0 (PDAC_W0_343)	32	RW	0000_0000h
1ABCh	Peripheral Domain Access Control Word 1 (PDAC_W1_343)	32	RW	0000_0000h
1AC0h	Peripheral Domain Access Control Word 0 (PDAC_W0_344)	32	RW	0000_0000h
1AC4h	Peripheral Domain Access Control Word 1 (PDAC_W1_344)	32	RW	0000_0000h
1AC8h	Peripheral Domain Access Control Word 0 (PDAC_W0_345)	32	RW	0000_0000h
1ACCh	Peripheral Domain Access Control Word 1 (PDAC_W1_345)	32	RW	0000_0000h
1AD0h	Peripheral Domain Access Control Word 0 (PDAC_W0_346)	32	RW	0000_0000h
1AD4h	Peripheral Domain Access Control Word 1 (PDAC_W1_346)	32	RW	0000_0000h
1AD8h	Peripheral Domain Access Control Word 0 (PDAC_W0_347)	32	RW	0000_0000h
1ADCh	Peripheral Domain Access Control Word 1 (PDAC_W1_347)	32	RW	0000_0000h
2000h	Memory Region Descriptor Word 0 (MRGD_W0_0)	32	RW	0000_0001h
2004h	Memory Region Descriptor Word 1 (MRGD_W1_0)	32	RW	0000_001Fh
2008h	Memory Region Descriptor Word 2 (MRGD_W2_0)	32	RW	0000_0000h
200Ch	Memory Region Descriptor Word 3 (MRGD_W3_0)	32	RW	0000_0000h
2020h	Memory Region Descriptor Word 0 (MRGD_W0_1)	32	RW	0000_0001h
2024h	Memory Region Descriptor Word 1 (MRGD_W1_1)	32	RW	0000_001Fh
2028h	Memory Region Descriptor Word 2 (MRGD_W2_1)	32	RW	0000_0000h
202Ch	Memory Region Descriptor Word 3 (MRGD_W3_1)	32	RW	0000_0000h
2040h	Memory Region Descriptor Word 0 (MRGD_W0_2)	32	RW	0000_0001h
2044h	Memory Region Descriptor Word 1 (MRGD_W1_2)	32	RW	0000_001Fh
2048h	Memory Region Descriptor Word 2 (MRGD_W2_2)	32	RW	0000_0000h
204Ch	Memory Region Descriptor Word 3 (MRGD_W3_2)	32	RW	0000_0000h
2060h	Memory Region Descriptor Word 0 (MRGD_W0_3)	32	RW	0000_0001h
2064h	Memory Region Descriptor Word 1 (MRGD_W1_3)	32	RW	0000_001Fh
2068h	Memory Region Descriptor Word 2 (MRGD_W2_3)	32	RW	0000_0000h
206Ch	Memory Region Descriptor Word 3 (MRGD_W3_3)	32	RW	0000_0000h
2080h	Memory Region Descriptor Word 0 (MRGD_W0_4)	32	RW	0000_0001h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
2084h	Memory Region Descriptor Word 1 (MRGD_W1_4)	32	RW	0000_001Fh
2088h	Memory Region Descriptor Word 2 (MRGD_W2_4)	32	RW	0000_0000h
208Ch	Memory Region Descriptor Word 3 (MRGD_W3_4)	32	RW	0000_0000h
20A0h	Memory Region Descriptor Word 0 (MRGD_W0_5)	32	RW	0000_0001h
20A4h	Memory Region Descriptor Word 1 (MRGD_W1_5)	32	RW	0000_001Fh
20A8h	Memory Region Descriptor Word 2 (MRGD_W2_5)	32	RW	0000_0000h
20ACh	Memory Region Descriptor Word 3 (MRGD_W3_5)	32	RW	0000_0000h
20C0h	Memory Region Descriptor Word 0 (MRGD_W0_6)	32	RW	0000_0001h
20C4h	Memory Region Descriptor Word 1 (MRGD_W1_6)	32	RW	0000_001Fh
20C8h	Memory Region Descriptor Word 2 (MRGD_W2_6)	32	RW	0000_0000h
20CCh	Memory Region Descriptor Word 3 (MRGD_W3_6)	32	RW	0000_0000h
20E0h	Memory Region Descriptor Word 0 (MRGD_W0_7)	32	RW	0000_0001h
20E4h	Memory Region Descriptor Word 1 (MRGD_W1_7)	32	RW	0000_001Fh
20E8h	Memory Region Descriptor Word 2 (MRGD_W2_7)	32	RW	0000_0000h
20ECh	Memory Region Descriptor Word 3 (MRGD_W3_7)	32	RW	0000_0000h
2100h	Memory Region Descriptor Word 0 (MRGD_W0_8)	32	RW	0000_0001h
2104h	Memory Region Descriptor Word 1 (MRGD_W1_8)	32	RW	0000_001Fh
2108h	Memory Region Descriptor Word 2 (MRGD_W2_8)	32	RW	0000_0000h
210Ch	Memory Region Descriptor Word 3 (MRGD_W3_8)	32	RW	0000_0000h
2120h	Memory Region Descriptor Word 0 (MRGD_W0_9)	32	RW	0000_0001h
2124h	Memory Region Descriptor Word 1 (MRGD_W1_9)	32	RW	0000_001Fh
2128h	Memory Region Descriptor Word 2 (MRGD_W2_9)	32	RW	0000_0000h
212Ch	Memory Region Descriptor Word 3 (MRGD_W3_9)	32	RW	0000_0000h
2140h	Memory Region Descriptor Word 0 (MRGD_W0_10)	32	RW	0000_0001h
2144h	Memory Region Descriptor Word 1 (MRGD_W1_10)	32	RW	0000_001Fh
2148h	Memory Region Descriptor Word 2 (MRGD_W2_10)	32	RW	0000_0000h
214Ch	Memory Region Descriptor Word 3 (MRGD_W3_10)	32	RW	0000_0000h
2160h	Memory Region Descriptor Word 0 (MRGD_W0_11)	32	RW	0000_0001h
2164h	Memory Region Descriptor Word 1 (MRGD_W1_11)	32	RW	0000_001Fh
2168h	Memory Region Descriptor Word 2 (MRGD_W2_11)	32	RW	0000_0000h
216Ch	Memory Region Descriptor Word 3 (MRGD_W3_11)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
2180h	Memory Region Descriptor Word 0 (MRGD_W0_12)	32	RW	0000_0001h
2184h	Memory Region Descriptor Word 1 (MRGD_W1_12)	32	RW	0000_001Fh
2188h	Memory Region Descriptor Word 2 (MRGD_W2_12)	32	RW	0000_0000h
218Ch	Memory Region Descriptor Word 3 (MRGD_W3_12)	32	RW	0000_0000h
21A0h	Memory Region Descriptor Word 0 (MRGD_W0_13)	32	RW	0000_0001h
21A4h	Memory Region Descriptor Word 1 (MRGD_W1_13)	32	RW	0000_001Fh
21A8h	Memory Region Descriptor Word 2 (MRGD_W2_13)	32	RW	0000_0000h
21ACh	Memory Region Descriptor Word 3 (MRGD_W3_13)	32	RW	0000_0000h
21C0h	Memory Region Descriptor Word 0 (MRGD_W0_14)	32	RW	0000_0001h
21C4h	Memory Region Descriptor Word 1 (MRGD_W1_14)	32	RW	0000_001Fh
21C8h	Memory Region Descriptor Word 2 (MRGD_W2_14)	32	RW	0000_0000h
21CCh	Memory Region Descriptor Word 3 (MRGD_W3_14)	32	RW	0000_0000h
21E0h	Memory Region Descriptor Word 0 (MRGD_W0_15)	32	RW	0000_0001h
21E4h	Memory Region Descriptor Word 1 (MRGD_W1_15)	32	RW	0000_001Fh
21E8h	Memory Region Descriptor Word 2 (MRGD_W2_15)	32	RW	0000_0000h
21ECh	Memory Region Descriptor Word 3 (MRGD_W3_15)	32	RW	0000_0000h
2200h	Memory Region Descriptor Word 0 (MRGD_W0_16)	32	RW	0000_0001h
2204h	Memory Region Descriptor Word 1 (MRGD_W1_16)	32	RW	0000_001Fh
2208h	Memory Region Descriptor Word 2 (MRGD_W2_16)	32	RW	0000_0000h
220Ch	Memory Region Descriptor Word 3 (MRGD_W3_16)	32	RW	0000_0000h
2220h	Memory Region Descriptor Word 0 (MRGD_W0_17)	32	RW	0000_0001h
2224h	Memory Region Descriptor Word 1 (MRGD_W1_17)	32	RW	0000_001Fh
2228h	Memory Region Descriptor Word 2 (MRGD_W2_17)	32	RW	0000_0000h
222Ch	Memory Region Descriptor Word 3 (MRGD_W3_17)	32	RW	0000_0000h
2240h	Memory Region Descriptor Word 0 (MRGD_W0_18)	32	RW	0000_0001h
2244h	Memory Region Descriptor Word 1 (MRGD_W1_18)	32	RW	0000_001Fh
2248h	Memory Region Descriptor Word 2 (MRGD_W2_18)	32	RW	0000_0000h
224Ch	Memory Region Descriptor Word 3 (MRGD_W3_18)	32	RW	0000_0000h
2260h	Memory Region Descriptor Word 0 (MRGD_W0_19)	32	RW	0000_0001h
2264h	Memory Region Descriptor Word 1 (MRGD_W1_19)	32	RW	0000_001Fh
2268h	Memory Region Descriptor Word 2 (MRGD_W2_19)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
226Ch	Memory Region Descriptor Word 3 (MRGD_W3_19)	32	RW	0000_0000h
2280h	Memory Region Descriptor Word 0 (MRGD_W0_20)	32	RW	0000_0001h
2284h	Memory Region Descriptor Word 1 (MRGD_W1_20)	32	RW	0000_001Fh
2288h	Memory Region Descriptor Word 2 (MRGD_W2_20)	32	RW	0000_0000h
228Ch	Memory Region Descriptor Word 3 (MRGD_W3_20)	32	RW	0000_0000h
22A0h	Memory Region Descriptor Word 0 (MRGD_W0_21)	32	RW	0000_0001h
22A4h	Memory Region Descriptor Word 1 (MRGD_W1_21)	32	RW	0000_001Fh
22A8h	Memory Region Descriptor Word 2 (MRGD_W2_21)	32	RW	0000_0000h
22ACh	Memory Region Descriptor Word 3 (MRGD_W3_21)	32	RW	0000_0000h
22C0h	Memory Region Descriptor Word 0 (MRGD_W0_22)	32	RW	0000_0001h
22C4h	Memory Region Descriptor Word 1 (MRGD_W1_22)	32	RW	0000_001Fh
22C8h	Memory Region Descriptor Word 2 (MRGD_W2_22)	32	RW	0000_0000h
22CCh	Memory Region Descriptor Word 3 (MRGD_W3_22)	32	RW	0000_0000h
22E0h	Memory Region Descriptor Word 0 (MRGD_W0_23)	32	RW	0000_0001h
22E4h	Memory Region Descriptor Word 1 (MRGD_W1_23)	32	RW	0000_001Fh
22E8h	Memory Region Descriptor Word 2 (MRGD_W2_23)	32	RW	0000_0000h
22ECh	Memory Region Descriptor Word 3 (MRGD_W3_23)	32	RW	0000_0000h
2300h	Memory Region Descriptor Word 0 (MRGD_W0_24)	32	RW	0000_0001h
2304h	Memory Region Descriptor Word 1 (MRGD_W1_24)	32	RW	0000_001Fh
2308h	Memory Region Descriptor Word 2 (MRGD_W2_24)	32	RW	0000_0000h
230Ch	Memory Region Descriptor Word 3 (MRGD_W3_24)	32	RW	0000_0000h
2320h	Memory Region Descriptor Word 0 (MRGD_W0_25)	32	RW	0000_0001h
2324h	Memory Region Descriptor Word 1 (MRGD_W1_25)	32	RW	0000_001Fh
2328h	Memory Region Descriptor Word 2 (MRGD_W2_25)	32	RW	0000_0000h
232Ch	Memory Region Descriptor Word 3 (MRGD_W3_25)	32	RW	0000_0000h
2340h	Memory Region Descriptor Word 0 (MRGD_W0_26)	32	RW	0000_0001h
2344h	Memory Region Descriptor Word 1 (MRGD_W1_26)	32	RW	0000_001Fh
2348h	Memory Region Descriptor Word 2 (MRGD_W2_26)	32	RW	0000_0000h
234Ch	Memory Region Descriptor Word 3 (MRGD_W3_26)	32	RW	0000_0000h
2360h	Memory Region Descriptor Word 0 (MRGD_W0_27)	32	RW	0000_0001h
2364h	Memory Region Descriptor Word 1 (MRGD_W1_27)	32	RW	0000_001Fh

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
2368h	Memory Region Descriptor Word 2 (MRGD_W2_27)	32	RW	0000_0000h
236Ch	Memory Region Descriptor Word 3 (MRGD_W3_27)	32	RW	0000_0000h
2380h	Memory Region Descriptor Word 0 (MRGD_W0_28)	32	RW	0000_0001h
2384h	Memory Region Descriptor Word 1 (MRGD_W1_28)	32	RW	0000_001Fh
2388h	Memory Region Descriptor Word 2 (MRGD_W2_28)	32	RW	0000_0000h
238Ch	Memory Region Descriptor Word 3 (MRGD_W3_28)	32	RW	0000_0000h
23A0h	Memory Region Descriptor Word 0 (MRGD_W0_29)	32	RW	0000_0001h
23A4h	Memory Region Descriptor Word 1 (MRGD_W1_29)	32	RW	0000_001Fh
23A8h	Memory Region Descriptor Word 2 (MRGD_W2_29)	32	RW	0000_0000h
23ACh	Memory Region Descriptor Word 3 (MRGD_W3_29)	32	RW	0000_0000h
23C0h	Memory Region Descriptor Word 0 (MRGD_W0_30)	32	RW	0000_0001h
23C4h	Memory Region Descriptor Word 1 (MRGD_W1_30)	32	RW	0000_001Fh
23C8h	Memory Region Descriptor Word 2 (MRGD_W2_30)	32	RW	0000_0000h
23CCh	Memory Region Descriptor Word 3 (MRGD_W3_30)	32	RW	0000_0000h
23E0h	Memory Region Descriptor Word 0 (MRGD_W0_31)	32	RW	0000_0001h
23E4h	Memory Region Descriptor Word 1 (MRGD_W1_31)	32	RW	0000_001Fh
23E8h	Memory Region Descriptor Word 2 (MRGD_W2_31)	32	RW	0000_0000h
23ECh	Memory Region Descriptor Word 3 (MRGD_W3_31)	32	RW	0000_0000h
2400h	Memory Region Descriptor Word 0 (MRGD_W0_32)	32	RW	0000_0001h
2404h	Memory Region Descriptor Word 1 (MRGD_W1_32)	32	RW	0000_001Fh
2408h	Memory Region Descriptor Word 2 (MRGD_W2_32)	32	RW	0000_0000h
240Ch	Memory Region Descriptor Word 3 (MRGD_W3_32)	32	RW	0000_0000h
2420h	Memory Region Descriptor Word 0 (MRGD_W0_33)	32	RW	0000_0001h
2424h	Memory Region Descriptor Word 1 (MRGD_W1_33)	32	RW	0000_001Fh
2428h	Memory Region Descriptor Word 2 (MRGD_W2_33)	32	RW	0000_0000h
242Ch	Memory Region Descriptor Word 3 (MRGD_W3_33)	32	RW	0000_0000h
2440h	Memory Region Descriptor Word 0 (MRGD_W0_34)	32	RW	0000_0001h
2444h	Memory Region Descriptor Word 1 (MRGD_W1_34)	32	RW	0000_001Fh
2448h	Memory Region Descriptor Word 2 (MRGD_W2_34)	32	RW	0000_0000h
244Ch	Memory Region Descriptor Word 3 (MRGD_W3_34)	32	RW	0000_0000h
2460h	Memory Region Descriptor Word 0 (MRGD_W0_35)	32	RW	0000_0001h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
2464h	Memory Region Descriptor Word 1 (MRGD_W1_35)	32	RW	0000_001Fh
2468h	Memory Region Descriptor Word 2 (MRGD_W2_35)	32	RW	0000_0000h
246Ch	Memory Region Descriptor Word 3 (MRGD_W3_35)	32	RW	0000_0000h
2600h	Memory Region Descriptor Word 0 (MRGD_W0_48)	32	RW	0000_0001h
2604h	Memory Region Descriptor Word 1 (MRGD_W1_48)	32	RW	0000_001Fh
2608h	Memory Region Descriptor Word 2 (MRGD_W2_48)	32	RW	0000_0000h
260Ch	Memory Region Descriptor Word 3 (MRGD_W3_48)	32	RW	0000_0000h
2620h	Memory Region Descriptor Word 0 (MRGD_W0_49)	32	RW	0000_0001h
2624h	Memory Region Descriptor Word 1 (MRGD_W1_49)	32	RW	0000_001Fh
2628h	Memory Region Descriptor Word 2 (MRGD_W2_49)	32	RW	0000_0000h
262Ch	Memory Region Descriptor Word 3 (MRGD_W3_49)	32	RW	0000_0000h
2640h	Memory Region Descriptor Word 0 (MRGD_W0_50)	32	RW	0000_0001h
2644h	Memory Region Descriptor Word 1 (MRGD_W1_50)	32	RW	0000_001Fh
2648h	Memory Region Descriptor Word 2 (MRGD_W2_50)	32	RW	0000_0000h
264Ch	Memory Region Descriptor Word 3 (MRGD_W3_50)	32	RW	0000_0000h
2660h	Memory Region Descriptor Word 0 (MRGD_W0_51)	32	RW	0000_0001h
2664h	Memory Region Descriptor Word 1 (MRGD_W1_51)	32	RW	0000_001Fh
2668h	Memory Region Descriptor Word 2 (MRGD_W2_51)	32	RW	0000_0000h
266Ch	Memory Region Descriptor Word 3 (MRGD_W3_51)	32	RW	0000_0000h
2680h	Memory Region Descriptor Word 0 (MRGD_W0_52)	32	RW	0000_0001h
2684h	Memory Region Descriptor Word 1 (MRGD_W1_52)	32	RW	0000_001Fh
2688h	Memory Region Descriptor Word 2 (MRGD_W2_52)	32	RW	0000_0000h
268Ch	Memory Region Descriptor Word 3 (MRGD_W3_52)	32	RW	0000_0000h
26A0h	Memory Region Descriptor Word 0 (MRGD_W0_53)	32	RW	0000_0001h
26A4h	Memory Region Descriptor Word 1 (MRGD_W1_53)	32	RW	0000_001Fh
26A8h	Memory Region Descriptor Word 2 (MRGD_W2_53)	32	RW	0000_0000h
26ACh	Memory Region Descriptor Word 3 (MRGD_W3_53)	32	RW	0000_0000h
26C0h	Memory Region Descriptor Word 0 (MRGD_W0_54)	32	RW	0000_0001h
26C4h	Memory Region Descriptor Word 1 (MRGD_W1_54)	32	RW	0000_001Fh
26C8h	Memory Region Descriptor Word 2 (MRGD_W2_54)	32	RW	0000_0000h
26CCh	Memory Region Descriptor Word 3 (MRGD_W3_54)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
26E0h	Memory Region Descriptor Word 0 (MRGD_W0_55)	32	RW	0000_0001h
26E4h	Memory Region Descriptor Word 1 (MRGD_W1_55)	32	RW	0000_001Fh
26E8h	Memory Region Descriptor Word 2 (MRGD_W2_55)	32	RW	0000_0000h
26ECh	Memory Region Descriptor Word 3 (MRGD_W3_55)	32	RW	0000_0000h
2700h	Memory Region Descriptor Word 0 (MRGD_W0_56)	32	RW	0000_0001h
2704h	Memory Region Descriptor Word 1 (MRGD_W1_56)	32	RW	0000_001Fh
2708h	Memory Region Descriptor Word 2 (MRGD_W2_56)	32	RW	0000_0000h
270Ch	Memory Region Descriptor Word 3 (MRGD_W3_56)	32	RW	0000_0000h
2720h	Memory Region Descriptor Word 0 (MRGD_W0_57)	32	RW	0000_0001h
2724h	Memory Region Descriptor Word 1 (MRGD_W1_57)	32	RW	0000_001Fh
2728h	Memory Region Descriptor Word 2 (MRGD_W2_57)	32	RW	0000_0000h
272Ch	Memory Region Descriptor Word 3 (MRGD_W3_57)	32	RW	0000_0000h
2740h	Memory Region Descriptor Word 0 (MRGD_W0_58)	32	RW	0000_0001h
2744h	Memory Region Descriptor Word 1 (MRGD_W1_58)	32	RW	0000_001Fh
2748h	Memory Region Descriptor Word 2 (MRGD_W2_58)	32	RW	0000_0000h
274Ch	Memory Region Descriptor Word 3 (MRGD_W3_58)	32	RW	0000_0000h
2760h	Memory Region Descriptor Word 0 (MRGD_W0_59)	32	RW	0000_0001h
2764h	Memory Region Descriptor Word 1 (MRGD_W1_59)	32	RW	0000_001Fh
2768h	Memory Region Descriptor Word 2 (MRGD_W2_59)	32	RW	0000_0000h
276Ch	Memory Region Descriptor Word 3 (MRGD_W3_59)	32	RW	0000_0000h
2780h	Memory Region Descriptor Word 0 (MRGD_W0_60)	32	RW	0000_0001h
2784h	Memory Region Descriptor Word 1 (MRGD_W1_60)	32	RW	0000_001Fh
2788h	Memory Region Descriptor Word 2 (MRGD_W2_60)	32	RW	0000_0000h
278Ch	Memory Region Descriptor Word 3 (MRGD_W3_60)	32	RW	0000_0000h
27A0h	Memory Region Descriptor Word 0 (MRGD_W0_61)	32	RW	0000_0001h
27A4h	Memory Region Descriptor Word 1 (MRGD_W1_61)	32	RW	0000_001Fh
27A8h	Memory Region Descriptor Word 2 (MRGD_W2_61)	32	RW	0000_0000h
27ACh	Memory Region Descriptor Word 3 (MRGD_W3_61)	32	RW	0000_0000h
27C0h	Memory Region Descriptor Word 0 (MRGD_W0_62)	32	RW	0000_0001h
27C4h	Memory Region Descriptor Word 1 (MRGD_W1_62)	32	RW	0000_001Fh
27C8h	Memory Region Descriptor Word 2 (MRGD_W2_62)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
27CCh	Memory Region Descriptor Word 3 (MRGD_W3_62)	32	RW	0000_0000h
27E0h	Memory Region Descriptor Word 0 (MRGD_W0_63)	32	RW	0000_0001h
27E4h	Memory Region Descriptor Word 1 (MRGD_W1_63)	32	RW	0000_001Fh
27E8h	Memory Region Descriptor Word 2 (MRGD_W2_63)	32	RW	0000_0000h
27ECh	Memory Region Descriptor Word 3 (MRGD_W3_63)	32	RW	0000_0000h
2800h	Memory Region Descriptor Word 0 (MRGD_W0_64)	32	RW	0000_0001h
2804h	Memory Region Descriptor Word 1 (MRGD_W1_64)	32	RW	0000_001Fh
2808h	Memory Region Descriptor Word 2 (MRGD_W2_64)	32	RW	0000_0000h
280Ch	Memory Region Descriptor Word 3 (MRGD_W3_64)	32	RW	0000_0000h
2820h	Memory Region Descriptor Word 0 (MRGD_W0_65)	32	RW	0000_0001h
2824h	Memory Region Descriptor Word 1 (MRGD_W1_65)	32	RW	0000_001Fh
2828h	Memory Region Descriptor Word 2 (MRGD_W2_65)	32	RW	0000_0000h
282Ch	Memory Region Descriptor Word 3 (MRGD_W3_65)	32	RW	0000_0000h
2840h	Memory Region Descriptor Word 0 (MRGD_W0_66)	32	RW	0000_0001h
2844h	Memory Region Descriptor Word 1 (MRGD_W1_66)	32	RW	0000_001Fh
2848h	Memory Region Descriptor Word 2 (MRGD_W2_66)	32	RW	0000_0000h
284Ch	Memory Region Descriptor Word 3 (MRGD_W3_66)	32	RW	0000_0000h
2860h	Memory Region Descriptor Word 0 (MRGD_W0_67)	32	RW	0000_0001h
2864h	Memory Region Descriptor Word 1 (MRGD_W1_67)	32	RW	0000_001Fh
2868h	Memory Region Descriptor Word 2 (MRGD_W2_67)	32	RW	0000_0000h
286Ch	Memory Region Descriptor Word 3 (MRGD_W3_67)	32	RW	0000_0000h
2A00h	Memory Region Descriptor Word 0 (MRGD_W0_80)	32	RW	0000_0001h
2A04h	Memory Region Descriptor Word 1 (MRGD_W1_80)	32	RW	0000_001Fh
2A08h	Memory Region Descriptor Word 2 (MRGD_W2_80)	32	RW	0000_0000h
2A0Ch	Memory Region Descriptor Word 3 (MRGD_W3_80)	32	RW	0000_0000h
2A20h	Memory Region Descriptor Word 0 (MRGD_W0_81)	32	RW	0000_0001h
2A24h	Memory Region Descriptor Word 1 (MRGD_W1_81)	32	RW	0000_001Fh
2A28h	Memory Region Descriptor Word 2 (MRGD_W2_81)	32	RW	0000_0000h
2A2Ch	Memory Region Descriptor Word 3 (MRGD_W3_81)	32	RW	0000_0000h
2A40h	Memory Region Descriptor Word 0 (MRGD_W0_82)	32	RW	0000_0001h
2A44h	Memory Region Descriptor Word 1 (MRGD_W1_82)	32	RW	0000_001Fh

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
2A48h	Memory Region Descriptor Word 2 (MRGD_W2_82)	32	RW	0000_0000h
2A4Ch	Memory Region Descriptor Word 3 (MRGD_W3_82)	32	RW	0000_0000h
2A60h	Memory Region Descriptor Word 0 (MRGD_W0_83)	32	RW	0000_0001h
2A64h	Memory Region Descriptor Word 1 (MRGD_W1_83)	32	RW	0000_001Fh
2A68h	Memory Region Descriptor Word 2 (MRGD_W2_83)	32	RW	0000_0000h
2A6Ch	Memory Region Descriptor Word 3 (MRGD_W3_83)	32	RW	0000_0000h
2A80h	Memory Region Descriptor Word 0 (MRGD_W0_84)	32	RW	0000_0001h
2A84h	Memory Region Descriptor Word 1 (MRGD_W1_84)	32	RW	0000_001Fh
2A88h	Memory Region Descriptor Word 2 (MRGD_W2_84)	32	RW	0000_0000h
2A8Ch	Memory Region Descriptor Word 3 (MRGD_W3_84)	32	RW	0000_0000h
2AA0h	Memory Region Descriptor Word 0 (MRGD_W0_85)	32	RW	0000_0001h
2AA4h	Memory Region Descriptor Word 1 (MRGD_W1_85)	32	RW	0000_001Fh
2AA8h	Memory Region Descriptor Word 2 (MRGD_W2_85)	32	RW	0000_0000h
2AACh	Memory Region Descriptor Word 3 (MRGD_W3_85)	32	RW	0000_0000h
2AC0h	Memory Region Descriptor Word 0 (MRGD_W0_86)	32	RW	0000_0001h
2AC4h	Memory Region Descriptor Word 1 (MRGD_W1_86)	32	RW	0000_001Fh
2AC8h	Memory Region Descriptor Word 2 (MRGD_W2_86)	32	RW	0000_0000h
2ACCh	Memory Region Descriptor Word 3 (MRGD_W3_86)	32	RW	0000_0000h
2AE0h	Memory Region Descriptor Word 0 (MRGD_W0_87)	32	RW	0000_0001h
2AE4h	Memory Region Descriptor Word 1 (MRGD_W1_87)	32	RW	0000_001Fh
2AE8h	Memory Region Descriptor Word 2 (MRGD_W2_87)	32	RW	0000_0000h
2AECh	Memory Region Descriptor Word 3 (MRGD_W3_87)	32	RW	0000_0000h
2B00h	Memory Region Descriptor Word 0 (MRGD_W0_88)	32	RW	0000_0001h
2B04h	Memory Region Descriptor Word 1 (MRGD_W1_88)	32	RW	0000_001Fh
2B08h	Memory Region Descriptor Word 2 (MRGD_W2_88)	32	RW	0000_0000h
2B0Ch	Memory Region Descriptor Word 3 (MRGD_W3_88)	32	RW	0000_0000h
2B20h	Memory Region Descriptor Word 0 (MRGD_W0_89)	32	RW	0000_0001h
2B24h	Memory Region Descriptor Word 1 (MRGD_W1_89)	32	RW	0000_001Fh
2B28h	Memory Region Descriptor Word 2 (MRGD_W2_89)	32	RW	0000_0000h
2B2Ch	Memory Region Descriptor Word 3 (MRGD_W3_89)	32	RW	0000_0000h
2B40h	Memory Region Descriptor Word 0 (MRGD_W0_90)	32	RW	0000_0001h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
2B44h	Memory Region Descriptor Word 1 (MRGD_W1_90)	32	RW	0000_001Fh
2B48h	Memory Region Descriptor Word 2 (MRGD_W2_90)	32	RW	0000_0000h
2B4Ch	Memory Region Descriptor Word 3 (MRGD_W3_90)	32	RW	0000_0000h
2B60h	Memory Region Descriptor Word 0 (MRGD_W0_91)	32	RW	0000_0001h
2B64h	Memory Region Descriptor Word 1 (MRGD_W1_91)	32	RW	0000_001Fh
2B68h	Memory Region Descriptor Word 2 (MRGD_W2_91)	32	RW	0000_0000h
2B6Ch	Memory Region Descriptor Word 3 (MRGD_W3_91)	32	RW	0000_0000h
2B80h	Memory Region Descriptor Word 0 (MRGD_W0_92)	32	RW	0000_0001h
2B84h	Memory Region Descriptor Word 1 (MRGD_W1_92)	32	RW	0000_001Fh
2B88h	Memory Region Descriptor Word 2 (MRGD_W2_92)	32	RW	0000_0000h
2B8Ch	Memory Region Descriptor Word 3 (MRGD_W3_92)	32	RW	0000_0000h
2BA0h	Memory Region Descriptor Word 0 (MRGD_W0_93)	32	RW	0000_0001h
2BA4h	Memory Region Descriptor Word 1 (MRGD_W1_93)	32	RW	0000_001Fh
2BA8h	Memory Region Descriptor Word 2 (MRGD_W2_93)	32	RW	0000_0000h
2BACh	Memory Region Descriptor Word 3 (MRGD_W3_93)	32	RW	0000_0000h
2BC0h	Memory Region Descriptor Word 0 (MRGD_W0_94)	32	RW	0000_0001h
2BC4h	Memory Region Descriptor Word 1 (MRGD_W1_94)	32	RW	0000_001Fh
2BC8h	Memory Region Descriptor Word 2 (MRGD_W2_94)	32	RW	0000_0000h
2BCCh	Memory Region Descriptor Word 3 (MRGD_W3_94)	32	RW	0000_0000h
2BE0h	Memory Region Descriptor Word 0 (MRGD_W0_95)	32	RW	0000_0001h
2BE4h	Memory Region Descriptor Word 1 (MRGD_W1_95)	32	RW	0000_001Fh
2BE8h	Memory Region Descriptor Word 2 (MRGD_W2_95)	32	RW	0000_0000h
2BECh	Memory Region Descriptor Word 3 (MRGD_W3_95)	32	RW	0000_0000h

19.8.3.2 Control (CR)

Offset

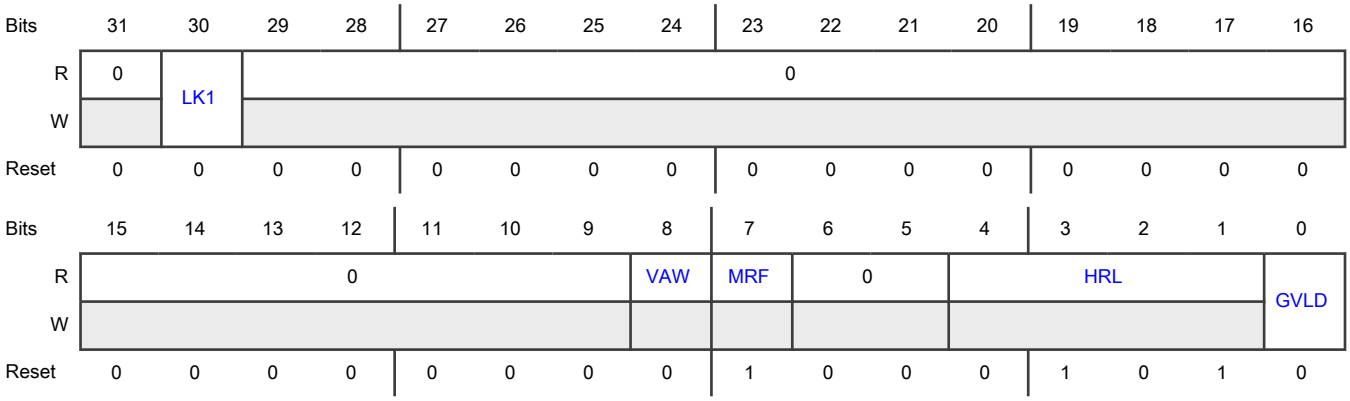
Register	Offset
CR	0h

Function

Provides XRDC status and enables XRDC operation.

Access: Secure privileged read/write

Diagram



Fields

Field	Function
31 —	Reserved
30 LK1	Lock Prohibits writes to this register. • If unlocked, this register accepts any secure privileged write. • If locked, you cannot write to this register and it remains read-only until after the next reset. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Unlocked 1b - Locks When writing 0b - No effect 1b - Locks
29-9 —	Reserved
8 VAW	Virtualization Aware Indicates whether domain assignments support the optional inclusion of a logical partition identifier, which is also known as an operating system number or Arm virtual machine identifier (VMID). 0b - Not virtualization-aware 1b - Virtualization-aware

Table continues on the next page...

Table continued from the previous page...

Field	Function
7 MRF	Memory Region Format Indicates the format of memory region descriptors. 0b - Reserved 1b - SMPU family format
6-5 —	Reserved
4-1 HRL	Hardware Revision Level Indicates the XRDC hardware revision level, which is associated with a set of functional characteristics of the module.
0 GVLD	Global Valid (XRDC Global Enable/Disable) Enables XRDC. When XRDC is disabled, all bus initiators can access all targets. 0b - Disables 1b - Enables

19.8.3.3 Hardware Configuration 0 (HWCFG0)

Offset

Register	Offset
HWCFG0	F0h

Function

Indicates XRDC configuration details, including:

- XRDC module ID
- Number of implemented domains
- Number of bus initiators
- Number of MRCs
- Number of PACs

Attempting to write to this register causes an error.

Access: Secure privileged read

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MID				NPAC				NMRC							
W																
Reset	0	0	0	1	0	0	1	0	0	0	0	0	0	1	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	NMSTR								NDID							
W																
Reset	0	0	0	0	1	0	0	1	0	0	0	0	0	1	0	0

Fields

Field	Function
31-28 MID	Module ID
27-24 NPAC	Number Of PACs Indicates the number of PACs minus 1. In other words, the actual number of PACs is NPAC + 1.
23-16 NMRC	Number of MRCs Indicates the number of MRCs minus 1. In other words, the actual number of MRCs is NMRC + 1.
15-8 NMSTR	Number Of Bus Initiators Indicates the number of bus initiators minus 1. In other words, the actual number of bus initiators is NMSTR + 1.
7-0 NDID	Number Of DIDs Indicates the number of domains (DIDs) minus 1. In other words, the actual number of DIDs is NDID + 1.

19.8.3.4 Hardware Configuration 1 (HWCFG1)

Offset

Register	Offset
HWCFG1	F4h

Function

Indicates the DID of the bus initiator making the current transaction request. See Domain error capture management for information about typical usage.

Attempting to write to this register causes an error.

Access: Secure privileged read

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												DID			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	u[1]	u	u	u

1. The reset value is determined by the current configuration of the accessing initiator.

Fields

Field	Function
31-4 —	Reserved
3-0 DID	Domain Identifier Indicates the DID of the requesting bus initiator.

19.8.3.5 Hardware Configuration 2 (HWCFG2)

Offset

Register	Offset
HWCFG2	F8h

Function

For initiators 0–31, indicates whether a given initiator has a built-in PID register as part of its programming model. If not, you must use the corresponding PIDm register to mimic the functionality of a built-in PID register.

Each bit corresponds to the same numbered initiator. For example, if PIDP18 is 1, bus initiator 18 has its own PID register. If PIDP18 is 0, then initiator 18 does not have its own PID register and you must use PID18.

Attempting to write to this register causes an error.

Access: Secure privileged read

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PIDP31	PIDP30	PIDP29	PIDP28	PIDP27	PIDP26	PIDP25	PIDP24	PIDP23	PIDP22	PIDP21	PIDP20	PIDP19	PIDP18	PIDP17	PIDP16
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PIDP15	PIDP14	PIDP13	PIDP12	PIDP11	PIDP10	PIDP9	PIDP8	PIDP7	PIDP6	PIDP5	PIDP4	PIDP3	PIDP2	PIDP1	PIDP0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PIDPn	Process Identifier Present 0b - Does not have PID register 1b - Has PID register

19.8.3.6 Master Domain Assignment Configuration (MDACFG0 - MDACFG9)

Offset

For m = 0 to 9:

Register	Offset
MDACFGm	100h + (m × 1h)

Function

Indicates the number of implemented master domain assignment registers (MDA_W*w*_*m*_DFMT0 or MDA_W*w*_*m*_DFMT1) for initiator *m*, where *m* ranges from 0 to 63. You can read these registers using 8-, 16-, or 32-bit accesses.

If NMDAR is 0, the associated initiator does not exist.

Attempting to write to this register causes an error.

Access: Secure privileged read

Diagram

Bits	7	6	5	4	3	2	1	0
R	NCM	0			NMDAR			
W								
Reset	See Register reset values.							

Register reset values

Register	Reset value
MDACFG0	01h
MDACFG1–MDACFG2	81h
MDACFG3–MDACFG4	01h
MDACFG5	81h
MDACFG6	01h
MDACFG7	81h
MDACFG8	01h
MDACFG9	81h

Fields

Field	Function
7 NCM	Noncore Master If NMDAR is greater than zero, indicates whether initiator *m* uses MDA_W*w*_*m*_DFMT0 or MDA_W*w*_*m*_DFMT1 to configure domain assignment. This field is 0 for a non-existent initiator. 0b - Core initiator or initiator does not exist 1b - Noncore initiator
6-4 —	Reserved
3-0 NMDAR	Number Of Master Domain Assignment Registers Indicates the number of master domain assignment registers (MDA_W*w*_*m*_DFMT0 or MDA_W*w*_*m*_DFMT1) associated with initiator *m*. 0000b - Initiator does not exist 0001b-1000b - Number of registers All other values are reserved.

19.8.3.7 Memory Region Configuration (MRCFG0 - MRCFG5)

Offset

Register	Offset
MRCFG0	140h
MRCFG1	141h
MRCFG2	142h
MRCFG3	143h
MRCFG4	144h
MRCFG5	145h

Function

Indicates the number of memory region descriptors (*n*) for MRC*c*, from 4 to 16 in increments of four, with 0 indicating a non-existent MRC. These registers are organized as byte-sized data arrays and can be read using 8-, 16-, or 32-bit accesses.

Attempting to write to this register causes an error.

Access: Secure Privileged Read

Diagram

Bits	7	6	5	4	3	2	1	0
R	0				NMRGD			
W								
Reset	See Register reset values.							

Register reset values

Register	Reset value
MRCFG0–MRCFG1	10h
MRCFG2	04h
MRCFG3	10h
MRCFG4	04h
MRCFG5	10h

Fields

Field	Function
7-5 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
4-0 NMRGD	Number Of Memory Region Descriptors Indicates the number of memory region descriptors associated with the MRC. 0_0000b - MRC does not exist 0_0100b - 4 0_1000b - 8 0_1100b - 12 1_0000b - 16 All other values are reserved.

19.8.3.8 Domain Error Location (DERRLOC0 - DERRLOC4)

Offset

Register	Offset
DERRLOC0	200h
DERRLOC1	204h
DERRLOC2	208h
DERRLOC3	20Ch
DERRLOC4	210h

Function

Indicates the MRC or PAC instance in domain *d* where an access violation has occurred. Each bit corresponds to the like-numbered instance. For more information, see Domain error capture management.

Attempting to write to this register causes an error.

Access: Secure privileged read

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												PACINST			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MRCINST															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-20 —	Reserved
19-16 PACINST	PAC Instance Indicates the presence of a detected access violation for domain *d* in a PAC instance. Each bit corresponds to the instance index for the DERR_W*w*_*i* registers: bit 16 of the register corresponds to the DERR_W*w*_*16* registers, which show error information for PAC 0, and so on. Multiple bits can be 1 at any time, indicating access violations have been detected across multiple PACs. For each bit in this field: 0b - No access violation error or the PAC instance is not physically present 1b - Access violation detected
15-0 MRCINST	MRC Instance Indicates the presence of a detected access violation for domain *d* in an MRC instance. Each bit corresponds to the like-numbered MRC instance: bit 0 (bit 0 of the register) corresponds to MRC instance 0, and so on. Multiple bits can be 1 at any time, indicating access violations have been detected across multiple MRCs. For each bit in this field: 0b - No access violation error or the MRC instance is not physically present 1b - Access violation detected

19.8.3.9 Domain Error Word 0 (DERR_W0_0 - DERR_W0_18)

Offset

Register	Offset
DERR_W0_0	400h
DERR_W0_1	410h
DERR_W0_2	420h
DERR_W0_3	430h
DERR_W0_4	440h
DERR_W0_5	450h
DERR_W0_16	500h
DERR_W0_17	510h
DERR_W0_18	520h

Function

Indicates the address of an access violation detected by an MRC or a PAC, indexed by the MRC or PAC instance (*i*) that detected the violation, as indicated in DERRLOC*d*. This register is part of a 16-byte set:

- DERR_W0_*i*: Word 0, the first 4 bytes
- DERR_W1_*i*: Word 1, the second 4 bytes
- Word 2, 4 reserved bytes
- DERR_W3_*i*: Word 3, the fourth 4 bytes

The first 16 sets (*i* from 0 to 15) are associated with MRCs and the rest (starting with *i* = 16) are associated with PACs. For more information, see Domain error capture management.

The error capture registers in the memory region controller and peripheral access controller submodules contain physical registers for each domain, but are organized in the DERR_Wn registers to provide the information for the requesting domain only. The registers return the error information for the domain id that is used to read the registers. When no error occurred for this domain id, it returns 0 data.

When the error capture logic is rearmed by writing the required data pattern to DERR_W3_n, this register is cleared.

When XRDC detects an access violation, it captures the error information and disables subsequent updates to the error capture registers until you write to DERR_W3_*i*.

NOTE

If initiators with the same DID cause simultaneous error accesses, the error capture registers record only the error of the lowest target index.

Attempting to write to this register causes an error. Attempting to read the error registers for a non-existent MRC or PAC instance causes an error.

Access: Secure privileged read

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	EADDR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EADDR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 EADDR	Error Address Indicates the target address of the first transaction that causes an access violation after reset or after rearming error capture.

19.8.3.10 Domain Error Word 1 (DERR_W1_0 - DERR_W1_18)

Offset

Register	Offset
DERR_W1_0	404h
DERR_W1_1	414h
DERR_W1_2	424h
DERR_W1_3	434h
DERR_W1_4	444h
DERR_W1_5	454h
DERR_W1_16	504h
DERR_W1_17	514h
DERR_W1_18	524h

Function

Indicates the attributes of an access violation detected by an MRC or a PAC, indexed by the MRC or PAC instance (*i*) that detected the access violation, as indicated in DERRLOC*d*. For more information, see DERR_W0_*i* and Domain error capture management.

The error capture registers in the memory region controller and peripheral access controller submodules contain physical registers for each domain, but are organized in the DERR_Wn registers to provide the information for the requesting domain only. The registers return the error information for the domain id that is used to read the registers. When no error occurred for this domain id, it returns 0 data.

When the error capture logic is rearmed by writing the required data pattern to DERR_W3_n, this register is cleared.

NOTE

If initiators with the same DID cause simultaneous error accesses, the error capture registers record only the error of the lowest target index.

Attempting to write to this register causes an error. Attempting to read the error registers for a non-existent MRC or PAC instance causes an error.

Access: Secure privileged read

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	EST				0				EPORT				0			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				ERW	EATR			0				EDID			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 EST	Error State Indicates the state of access violations for this domain in this instance of the MRC or PAC. After the first access violation to occur after reset or rearming of error capture, XRDC records subsequent errors as an overrun condition without any data captured. 00b-01b - No access violations detected 10b - A single access violation has been detected 11b - Multiple access violations have been detected
29-27 —	Reserved
26-24 EPORT	Error Port Identifies the encoded port number of the MRC that detected the access violation. See the chip-specific configuration details for the MRC port connections. For access violations detected by a PAC, this field is zero.
23-12 —	Reserved
11 ERW	Error Read Or Write Indicates whether the captured access violation occurred on a read or write access. 0b - Read access 1b - Write access
10-8 EATR	Error Attributes Indicates attributes of the access violation. 000b - Secure user mode, instruction fetch access 001b - Secure user mode, data access 010b - Secure privileged mode, instruction fetch access 011b - Secure privileged mode, data access 100b - Nonsecure user mode, instruction fetch access 101b - Nonsecure user mode, data access 110b - Nonsecure privileged mode, instruction fetch access 111b - Nonsecure privileged mode, data access
7-4 —	Reserved
3-0 EDID	Error Domain Identifier Indicates the DID of the access violation.

19.8.3.11 Domain Error Word 3 (DERR_W3_0 - DERR_W3_18)

Offset

Register	Offset
DERR_W3_0	40Ch
DERR_W3_1	41Ch
DERR_W3_2	42Ch
DERR_W3_3	43Ch
DERR_W3_4	44Ch
DERR_W3_5	45Ch
DERR_W3_16	50Ch
DERR_W3_17	51Ch
DERR_W3_18	52Ch

Function

Rearms instance error capture, resets the error capture registers (DERR_W0_*d*, DERR_W1_*d*), and deasserts the instance bit in DERRLOC*d*. After reading the access violation error information, an error handler must write 1 to RECR.

Register write only rearms error capture registers for the domain id that is used to write this register.

This register returns 0000h when read. Attempted reads of an MRC or PAC instance that is not physically present cause an error.

For more information, see Domain error capture management.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				0											
W	RECR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30	Rearm Error Capture Registers
RECR	Resets and rearms the domain error capture registers for this instance, including deasserting the instance bit in DERRLOC*d*.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00b,10b,11b - No effect 01b - Rearms error capture, resets error capture registers, and deasserts instance bit in DERRLOCd
29-0 —	Reserved

19.8.3.12 Process Identifier (PID0 - PID8)

Offset

Register	Offset
PID0	700h
PID3	70Ch
PID4	710h
PID6	718h
PID8	720h

Function

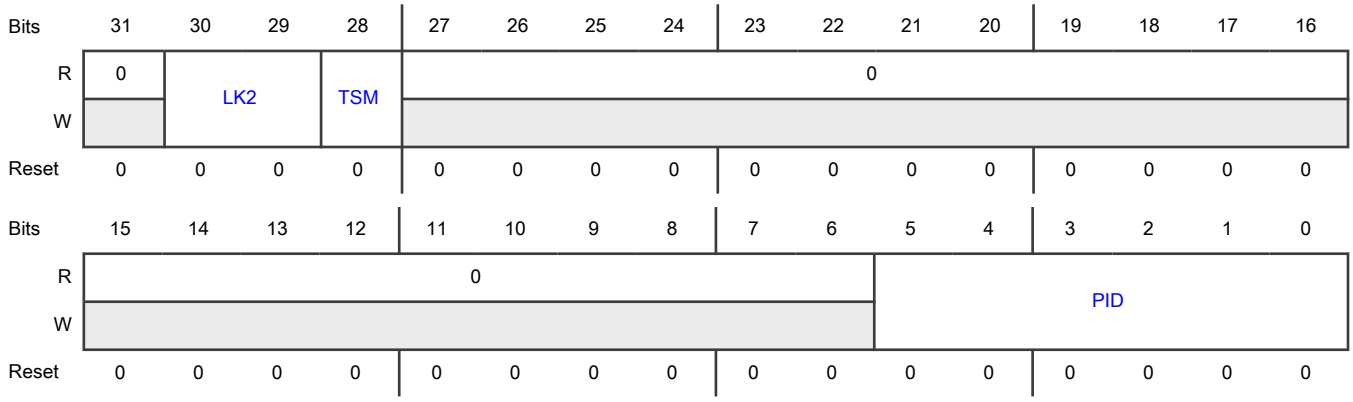
Specifies the PID for the associated core initiator *m*.

Some cores contain a built-in PID register. If the core has a built-in PID register, XRDC populates the PID field with the value from the core PID register. If the core does not have the built-in register, the XRDC PID register allows applications to mimic PID operation for that core by writing the desired PID to the associated register.

HWCFG2 provides a bitmap of the implemented PID*m* registers. Noncore initiators do not have an associated PID register.

For information about PID-based operation, see PID-based domain assignment.

Diagram



Fields

Field	Function
31 —	Reserved
30-29 LK2	Lock Limits or prohibits writes to this register. When you assert a bit in this field, it remains asserted until the next module reset. If the core initiator has a built-in PID register, as indicated in HWCFG2, then a secure privileged read returns 0 for this field. 00b,01b - Any secure privileged write 10b - Secure privileged writes from initiator only 11b - Locks
28 TSM	Three-State Model Specifies that the core initiator supports only the three-state access control model. If you write 1 to this field, it remains asserted until the next reset. For cores that support only the three-state access control model, you must assert this field before loading any nonsecure value into the PID. See Generation of secure attribute.
27-6 —	Reserved
5-0 PID	Process Identifier Specifies the transaction PID for the corresponding core initiator. If the core has a built-in PID register, then a secure privileged read returns the core's PID register value. Bit 5 specifies the secure attribute (0 = secure, 1 = nonsecure) for the transaction.

19.8.3.13 Master Domain Assignment (MDA_W0_0_DFMT0 - MDA_W0_8_DFMT0)

Offset

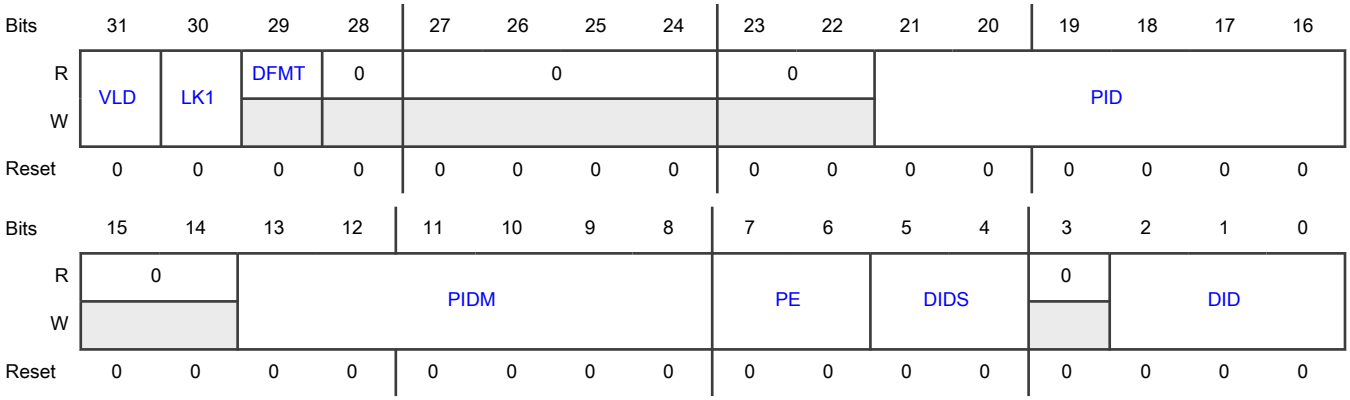
Register	Offset
MDA_W0_0_DFMT0	800h
MDA_W0_3_DFMT0	860h
MDA_W0_4_DFMT0	880h
MDA_W0_6_DFMT0	8C0h
MDA_W0_8_DFMT0	900h

Function

Specifies the information used by the MDAC to assign a core bus initiator to a specific domain (DID). For more information, see Master domain assignment controller (MDAC).

Access: Secure privileged read/write

Diagram



Fields

Field	Function
31 VLD	Valid Specifies whether this domain assignment is valid. In other words, if VLD and CR[GVLD] are both asserted, XRDC uses the configuration in this register in the domain assignment process. If CR[GVLD] is set to 1 and VLD is set to 0 then every transaction from this initiator will be assigned a DID of 0. This field has no effect unless XRDC is enabled (CR[GVLD] = 1). 0b - Invalid 1b - Valid
30 LK1	Lock Prohibits writes to this register. • If unlocked, this register accepts any secure privileged write. • If locked, you cannot write to this register and it remains read-only until after the next reset. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Unlocked 1b - Locks When writing 0b - No effect 1b - Locks

Table continues on the next page...

Table continued from the previous page...

Field	Function
29 DFMT	Domain Format Indicates the domain assignment format. 0b - Core bus initiator domain assignment (DFMT0)
28 —	Reserved
27-24 —	Reserved
23-22 —	Reserved
21-16 PID	Process Identifier Specifies the PID to be combined with the PIDM field and included in the domain assignment process. This field applies only if PID is enabled (PE = 1).
15-14 —	Reserved
13-8 PIDM	Process Identifier Mask Specifies a mask applied to the PID to support including multiple PIDs in the domain hit determination. For each bit asserted in PIDM, the corresponding bit of the PID is ignored in the comparison. This field applies only if PID is enabled (PE = 1).
7-6 PE	Process Identifier Enable Enables the optional inclusion of PID, qualified by PIDM, in the domain hit evaluation. This inclusion supports the definition of inclusive or exclusive sets of masked PID values. 00b-01b - No PID is included 10b - Partial domain hit = (PID & ~PIDM) == (PIDm[PID] & ~PIDM) 11b - Partial domain hit = ~((PID & ~PIDM) == (PIDm[PID] & ~PIDM))
5-4 DIDS	DID Select Selects the source of the DID. 00b - Use the DID field of this register 01b - Use the input DID 10b - Concatenate bits 3–2 of this register with the least significant 2 bits of the input DID (DID_in[1:0]) 11b - Reserved
3	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
2-0 DID	Domain Identifier Specifies the DID. DIDS controls whether and how this value is used.

19.8.3.14 Master Domain Assignment (MDA_W0_1_DFMT1 - MDA_W0_9_DFMT1)

Offset

Register	Offset
MDA_W0_1_DFMT1	820h
MDA_W0_2_DFMT1	840h
MDA_W0_5_DFMT1	8A0h
MDA_W0_7_DFMT1	8E0h
MDA_W0_9_DFMT1	920h

Function

Specifies the information used by the MDAC to assign a bus initiator to a specific domain (DID). For more information, see Master domain assignment controller (MDAC).

Access: Secure privileged read/write

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	VLD	LK1	DFMT	0	0				0							
W																
Reset	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0							DIDB	SA		PA		0	DID		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31	Valid

Table continues on the next page...

Table continued from the previous page...

Field	Function
VLD	Specifies whether this domain assignment is valid. In other words, if VLD and CR[GVLD] are both asserted, XRDC uses the configuration in this register in the domain assignment process. If CR[GVLD] is set to 1 and VLD is set to 0 then every transaction from this initiator will be assigned a DID of 0. This field has no effect unless XRDC is enabled (CR[GVLD] = 1). 0b - Invalid 1b - Valid
30 LK1	Lock Prohibits writes to this register. • If unlocked, this register accepts any secure privileged write. • If locked, you cannot write to this register and it remains read-only until after the next reset. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Unlocked 1b - Locks When writing 0b - No effect 1b - Locks
29 DFMT	Domain Format Indicates the domain assignment format. 1b - Bus initiator domain assignment (DFMT1)
28 —	Reserved
27-24 —	Reserved
23-9 —	Reserved
8 DIDB	DID Bypass Enables bypassing of an input DID as the domain identifier for this initiator. This capability allows noncore initiators (for example, a DMA) to appear as a core. After this field is set to 1, it remains at that value until the next reset. 0b - Bypass DID input. Use DID

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Use DID input
7-6 SA	Secure Attribute Specifies the secure attribute. **NOTE** If SA = 1X or VLD = 0, use the secure attribute input from the initiator. 00b - Force to secure 01b - Force to nonsecure 1xb - Use secure attribute from the initiator
5-4 PA	Privileged Attribute Specifies the privileged (supervisor/user) attribute. **NOTE** If PA = 1X or VLD = 0, use the privileged attribute input from the initiator. 00b - Force to user 01b - Force to privileged 1xb - Use privileged attribute from the initiator
3 —	Reserved
2-0 DID	Domain Identifier Specifies the DID.

19.8.3.15 Peripheral Domain Access Control Word 0 (PDAC_W0_2 - PDAC_W0_347)

Offset

Register	Offset
PDAC_W0_2	1010h
PDAC_W0_3	1018h
PDAC_W0_28	10E0h
PDAC_W0_32	1100h
PDAC_W0_33	1108h
PDAC_W0_34	1110h
PDAC_W0_35	1118h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
PDAC_W0_36	1120h
PDAC_W0_38	1130h
PDAC_W0_39	1138h
PDAC_W0_40	1140h
PDAC_W0_41	1148h
PDAC_W0_42	1150h
PDAC_W0_44	1160h
PDAC_W0_45	1168h
PDAC_W0_46	1170h
PDAC_W0_47	1178h
PDAC_W0_49	1188h
PDAC_W0_50	1190h
PDAC_W0_51	1198h
PDAC_W0_52	11A0h
PDAC_W0_128	1400h
PDAC_W0_129	1408h
PDAC_W0_130	1410h
PDAC_W0_131	1418h
PDAC_W0_132	1420h
PDAC_W0_133	1428h
PDAC_W0_134	1430h
PDAC_W0_135	1438h
PDAC_W0_136	1440h
PDAC_W0_137	1448h
PDAC_W0_138	1450h
PDAC_W0_139	1458h
PDAC_W0_140	1460h
PDAC_W0_141	1468h
PDAC_W0_142	1470h
PDAC_W0_143	1478h
PDAC_W0_144	1480h
PDAC_W0_145	1488h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
PDAC_W0_146	1490h
PDAC_W0_147	1498h
PDAC_W0_148	14A0h
PDAC_W0_149	14A8h
PDAC_W0_151	14B8h
PDAC_W0_152	14C0h
PDAC_W0_153	14C8h
PDAC_W0_154	14D0h
PDAC_W0_155	14D8h
PDAC_W0_156	14E0h
PDAC_W0_157	14E8h
PDAC_W0_158	14F0h
PDAC_W0_159	14F8h
PDAC_W0_160	1500h
PDAC_W0_161	1508h
PDAC_W0_162	1510h
PDAC_W0_163	1518h
PDAC_W0_164	1520h
PDAC_W0_165	1528h
PDAC_W0_166	1530h
PDAC_W0_167	1538h
PDAC_W0_168	1540h
PDAC_W0_169	1548h
PDAC_W0_170	1550h
PDAC_W0_171	1558h
PDAC_W0_173	1568h
PDAC_W0_175	1578h
PDAC_W0_177	1588h
PDAC_W0_178	1590h
PDAC_W0_179	1598h
PDAC_W0_180	15A0h
PDAC_W0_181	15A8h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
PDAC_W0_182	15B0h
PDAC_W0_183	15B8h
PDAC_W0_184	15C0h
PDAC_W0_185	15C8h
PDAC_W0_186	15D0h
PDAC_W0_187	15D8h
PDAC_W0_188	15E0h
PDAC_W0_189	15E8h
PDAC_W0_190	15F0h
PDAC_W0_191	15F8h
PDAC_W0_192	1600h
PDAC_W0_193	1608h
PDAC_W0_194	1610h
PDAC_W0_195	1618h
PDAC_W0_196	1620h
PDAC_W0_197	1628h
PDAC_W0_198	1630h
PDAC_W0_199	1638h
PDAC_W0_200	1640h
PDAC_W0_201	1648h
PDAC_W0_202	1650h
PDAC_W0_203	1658h
PDAC_W0_204	1660h
PDAC_W0_205	1668h
PDAC_W0_206	1670h
PDAC_W0_207	1678h
PDAC_W0_208	1680h
PDAC_W0_209	1688h
PDAC_W0_210	1690h
PDAC_W0_211	1698h
PDAC_W0_212	16A0h
PDAC_W0_213	16A8h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
PDAC_W0_214	16B0h
PDAC_W0_215	16B8h
PDAC_W0_216	16C0h
PDAC_W0_217	16C8h
PDAC_W0_219	16D8h
PDAC_W0_220	16E0h
PDAC_W0_221	16E8h
PDAC_W0_223	16F8h
PDAC_W0_224	1700h
PDAC_W0_225	1708h
PDAC_W0_226	1710h
PDAC_W0_227	1718h
PDAC_W0_229	1728h
PDAC_W0_230	1730h
PDAC_W0_231	1738h
PDAC_W0_232	1740h
PDAC_W0_233	1748h
PDAC_W0_234	1750h
PDAC_W0_235	1758h
PDAC_W0_236	1760h
PDAC_W0_238	1770h
PDAC_W0_240	1780h
PDAC_W0_241	1788h
PDAC_W0_242	1790h
PDAC_W0_243	1798h
PDAC_W0_244	17A0h
PDAC_W0_245	17A8h
PDAC_W0_246	17B0h
PDAC_W0_247	17B8h
PDAC_W0_248	17C0h
PDAC_W0_249	17C8h
PDAC_W0_250	17D0h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
PDAC_W0_251	17D8h
PDAC_W0_252	17E0h
PDAC_W0_253	17E8h
PDAC_W0_254	17F0h
PDAC_W0_255	17F8h
PDAC_W0_256	1800h
PDAC_W0_257	1808h
PDAC_W0_258	1810h
PDAC_W0_259	1818h
PDAC_W0_260	1820h
PDAC_W0_261	1828h
PDAC_W0_262	1830h
PDAC_W0_263	1838h
PDAC_W0_264	1840h
PDAC_W0_265	1848h
PDAC_W0_266	1850h
PDAC_W0_267	1858h
PDAC_W0_268	1860h
PDAC_W0_269	1868h
PDAC_W0_270	1870h
PDAC_W0_271	1878h
PDAC_W0_272	1880h
PDAC_W0_273	1888h
PDAC_W0_274	1890h
PDAC_W0_275	1898h
PDAC_W0_276	18A0h
PDAC_W0_277	18A8h
PDAC_W0_278	18B0h
PDAC_W0_279	18B8h
PDAC_W0_280	18C0h
PDAC_W0_281	18C8h
PDAC_W0_282	18D0h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
PDAC_W0_283	18D8h
PDAC_W0_284	18E0h
PDAC_W0_285	18E8h
PDAC_W0_286	18F0h
PDAC_W0_287	18F8h
PDAC_W0_289	1908h
PDAC_W0_290	1910h
PDAC_W0_291	1918h
PDAC_W0_292	1920h
PDAC_W0_293	1928h
PDAC_W0_294	1930h
PDAC_W0_295	1938h
PDAC_W0_296	1940h
PDAC_W0_297	1948h
PDAC_W0_298	1950h
PDAC_W0_303	1978h
PDAC_W0_304	1980h
PDAC_W0_307	1998h
PDAC_W0_311	19B8h
PDAC_W0_314	19D0h
PDAC_W0_315	19D8h
PDAC_W0_318	19F0h
PDAC_W0_319	19F8h
PDAC_W0_320	1A00h
PDAC_W0_321	1A08h
PDAC_W0_323	1A18h
PDAC_W0_324	1A20h
PDAC_W0_325	1A28h
PDAC_W0_326	1A30h
PDAC_W0_328	1A40h
PDAC_W0_329	1A48h
PDAC_W0_330	1A50h

Table continues on the next page...

Table continued from the previous page...

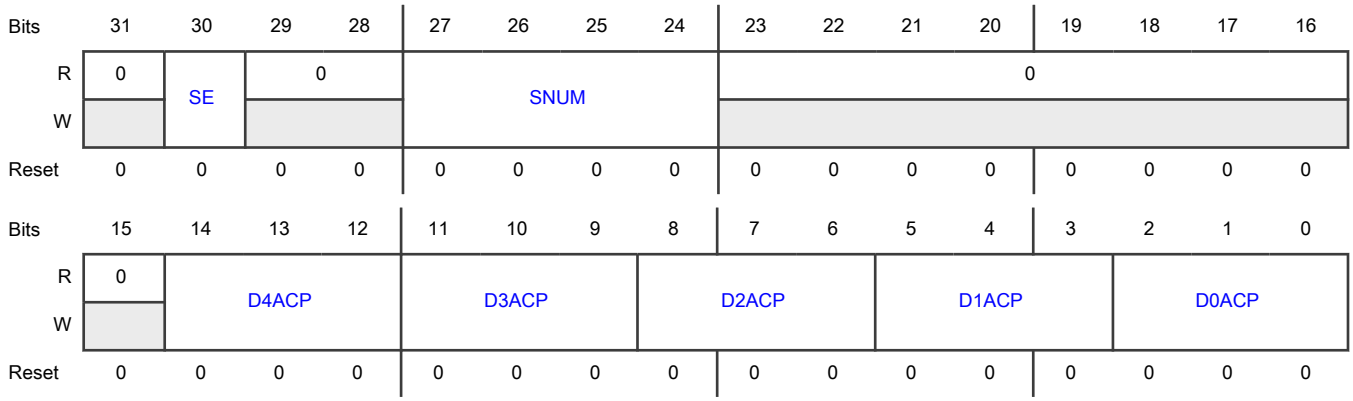
Register	Offset
PDAC_W0_331	1A58h
PDAC_W0_332	1A60h
PDAC_W0_333	1A68h
PDAC_W0_334	1A70h
PDAC_W0_335	1A78h
PDAC_W0_337	1A88h
PDAC_W0_338	1A90h
PDAC_W0_339	1A98h
PDAC_W0_340	1AA0h
PDAC_W0_341	1AA8h
PDAC_W0_342	1AB0h
PDAC_W0_343	1AB8h
PDAC_W0_344	1AC0h
PDAC_W0_345	1AC8h
PDAC_W0_346	1AD0h
PDAC_W0_347	1AD8h

Function

In conjunction with PDAC_W1_*s*, specifies the ACP configuration for peripheral slot *s*. The ACP controls access to the peripheral by all initiators within the domain. For the available ACPs, see Domain ACP specification. For more information, see Peripheral access controller (PAC).

Access: Secure privileged read/write

Diagram



Fields

Field	Function
31 —	Reserved
30 SE	Semaphore Enable Enables the inclusion of the semaphore specified in SNUM in the D*d*ACP evaluation. 0b - Disables 1b - Enables
29-28 —	Reserved
27-24 SNUM	Semaphore Number Specifies the hardware semaphore to include in the D*d*ACP access evaluation. This field applies only if you enable semaphore (write 1 to SE).
23-15 —	Reserved
14-12: D4ACP 11-9: D3ACP 8-6: D2ACP 5-3: D1ACP 2-0: D0ACP	Domain Access Control Policy Specifies the ACP for the associated domain. This field applies only for a supported DID; if the DID is not implemented, the field is read-only zero (no access rights). For field values, see Domain ACP specification.

19.8.3.16 Peripheral Domain Access Control Word 1 (PDAC_W1_2 - PDAC_W1_347)

Offset

Register	Offset
PDAC_W1_2	1014h
PDAC_W1_3	101Ch
PDAC_W1_28	10E4h
PDAC_W1_32	1104h
PDAC_W1_33	110Ch
PDAC_W1_34	1114h
PDAC_W1_35	111Ch
PDAC_W1_36	1124h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
PDAC_W1_38	1134h
PDAC_W1_39	113Ch
PDAC_W1_40	1144h
PDAC_W1_41	114Ch
PDAC_W1_42	1154h
PDAC_W1_44	1164h
PDAC_W1_45	116Ch
PDAC_W1_46	1174h
PDAC_W1_47	117Ch
PDAC_W1_49	118Ch
PDAC_W1_50	1194h
PDAC_W1_51	119Ch
PDAC_W1_52	11A4h
PDAC_W1_128	1404h
PDAC_W1_129	140Ch
PDAC_W1_130	1414h
PDAC_W1_131	141Ch
PDAC_W1_132	1424h
PDAC_W1_133	142Ch
PDAC_W1_134	1434h
PDAC_W1_135	143Ch
PDAC_W1_136	1444h
PDAC_W1_137	144Ch
PDAC_W1_138	1454h
PDAC_W1_139	145Ch
PDAC_W1_140	1464h
PDAC_W1_141	146Ch
PDAC_W1_142	1474h
PDAC_W1_143	147Ch
PDAC_W1_144	1484h
PDAC_W1_145	148Ch
PDAC_W1_146	1494h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
PDAC_W1_147	149Ch
PDAC_W1_148	14A4h
PDAC_W1_149	14ACh
PDAC_W1_151	14BCh
PDAC_W1_152	14C4h
PDAC_W1_153	14CCh
PDAC_W1_154	14D4h
PDAC_W1_155	14DCh
PDAC_W1_156	14E4h
PDAC_W1_157	14ECh
PDAC_W1_158	14F4h
PDAC_W1_159	14FCh
PDAC_W1_160	1504h
PDAC_W1_161	150Ch
PDAC_W1_162	1514h
PDAC_W1_163	151Ch
PDAC_W1_164	1524h
PDAC_W1_165	152Ch
PDAC_W1_166	1534h
PDAC_W1_167	153Ch
PDAC_W1_168	1544h
PDAC_W1_169	154Ch
PDAC_W1_170	1554h
PDAC_W1_171	155Ch
PDAC_W1_173	156Ch
PDAC_W1_175	157Ch
PDAC_W1_177	158Ch
PDAC_W1_178	1594h
PDAC_W1_179	159Ch
PDAC_W1_180	15A4h
PDAC_W1_181	15ACh
PDAC_W1_182	15B4h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
PDAC_W1_183	15BCh
PDAC_W1_184	15C4h
PDAC_W1_185	15CCh
PDAC_W1_186	15D4h
PDAC_W1_187	15DCh
PDAC_W1_188	15E4h
PDAC_W1_189	15ECh
PDAC_W1_190	15F4h
PDAC_W1_191	15FCh
PDAC_W1_192	1604h
PDAC_W1_193	160Ch
PDAC_W1_194	1614h
PDAC_W1_195	161Ch
PDAC_W1_196	1624h
PDAC_W1_197	162Ch
PDAC_W1_198	1634h
PDAC_W1_199	163Ch
PDAC_W1_200	1644h
PDAC_W1_201	164Ch
PDAC_W1_202	1654h
PDAC_W1_203	165Ch
PDAC_W1_204	1664h
PDAC_W1_205	166Ch
PDAC_W1_206	1674h
PDAC_W1_207	167Ch
PDAC_W1_208	1684h
PDAC_W1_209	168Ch
PDAC_W1_210	1694h
PDAC_W1_211	169Ch
PDAC_W1_212	16A4h
PDAC_W1_213	16ACh
PDAC_W1_214	16B4h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
PDAC_W1_215	16BCh
PDAC_W1_216	16C4h
PDAC_W1_217	16CCh
PDAC_W1_219	16DCh
PDAC_W1_220	16E4h
PDAC_W1_221	16ECh
PDAC_W1_223	16FCh
PDAC_W1_224	1704h
PDAC_W1_225	170Ch
PDAC_W1_226	1714h
PDAC_W1_227	171Ch
PDAC_W1_229	172Ch
PDAC_W1_230	1734h
PDAC_W1_231	173Ch
PDAC_W1_232	1744h
PDAC_W1_233	174Ch
PDAC_W1_234	1754h
PDAC_W1_235	175Ch
PDAC_W1_236	1764h
PDAC_W1_238	1774h
PDAC_W1_240	1784h
PDAC_W1_241	178Ch
PDAC_W1_242	1794h
PDAC_W1_243	179Ch
PDAC_W1_244	17A4h
PDAC_W1_245	17ACh
PDAC_W1_246	17B4h
PDAC_W1_247	17BCh
PDAC_W1_248	17C4h
PDAC_W1_249	17CCh
PDAC_W1_250	17D4h
PDAC_W1_251	17DCh

Table continues on the next page...

Table continued from the previous page...

Register	Offset
PDAC_W1_252	17E4h
PDAC_W1_253	17ECh
PDAC_W1_254	17F4h
PDAC_W1_255	17FCh
PDAC_W1_256	1804h
PDAC_W1_257	180Ch
PDAC_W1_258	1814h
PDAC_W1_259	181Ch
PDAC_W1_260	1824h
PDAC_W1_261	182Ch
PDAC_W1_262	1834h
PDAC_W1_263	183Ch
PDAC_W1_264	1844h
PDAC_W1_265	184Ch
PDAC_W1_266	1854h
PDAC_W1_267	185Ch
PDAC_W1_268	1864h
PDAC_W1_269	186Ch
PDAC_W1_270	1874h
PDAC_W1_271	187Ch
PDAC_W1_272	1884h
PDAC_W1_273	188Ch
PDAC_W1_274	1894h
PDAC_W1_275	189Ch
PDAC_W1_276	18A4h
PDAC_W1_277	18ACh
PDAC_W1_278	18B4h
PDAC_W1_279	18BCh
PDAC_W1_280	18C4h
PDAC_W1_281	18CCh
PDAC_W1_282	18D4h
PDAC_W1_283	18DCh

Table continues on the next page...

Table continued from the previous page...

Register	Offset
PDAC_W1_284	18E4h
PDAC_W1_285	18ECh
PDAC_W1_286	18F4h
PDAC_W1_287	18FCh
PDAC_W1_289	190Ch
PDAC_W1_290	1914h
PDAC_W1_291	191Ch
PDAC_W1_292	1924h
PDAC_W1_293	192Ch
PDAC_W1_294	1934h
PDAC_W1_295	193Ch
PDAC_W1_296	1944h
PDAC_W1_297	194Ch
PDAC_W1_298	1954h
PDAC_W1_303	197Ch
PDAC_W1_304	1984h
PDAC_W1_307	199Ch
PDAC_W1_311	19BCh
PDAC_W1_314	19D4h
PDAC_W1_315	19DCh
PDAC_W1_318	19F4h
PDAC_W1_319	19FCh
PDAC_W1_320	1A04h
PDAC_W1_321	1A0Ch
PDAC_W1_323	1A1Ch
PDAC_W1_324	1A24h
PDAC_W1_325	1A2Ch
PDAC_W1_326	1A34h
PDAC_W1_328	1A44h
PDAC_W1_329	1A4Ch
PDAC_W1_330	1A54h
PDAC_W1_331	1A5Ch

Table continues on the next page...

Table continued from the previous page...

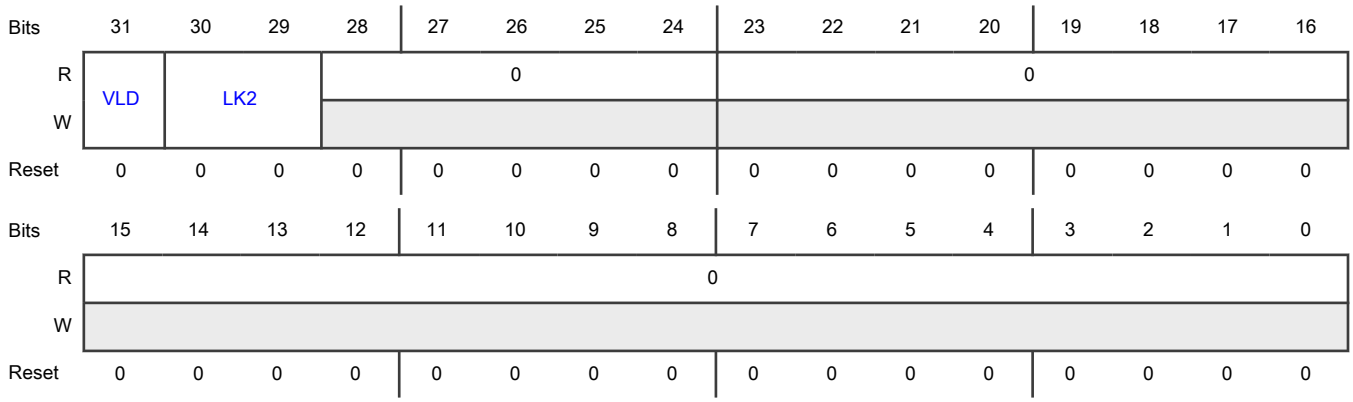
Register	Offset
PDAC_W1_332	1A64h
PDAC_W1_333	1A6Ch
PDAC_W1_334	1A74h
PDAC_W1_335	1A7Ch
PDAC_W1_337	1A8Ch
PDAC_W1_338	1A94h
PDAC_W1_339	1A9Ch
PDAC_W1_340	1AA4h
PDAC_W1_341	1AACh
PDAC_W1_342	1AB4h
PDAC_W1_343	1ABCh
PDAC_W1_344	1AC4h
PDAC_W1_345	1ACCh
PDAC_W1_346	1AD4h
PDAC_W1_347	1ADCh

Function

In conjunction with PDAC_W0_*s*, specifies the ACP configuration for peripheral slot *s*.

Access: Secure privileged read/write

Diagram



Fields

Field	Function
31	Valid

Table continues on the next page...

Table continued from the previous page...

Field	Function
VLD	Specifies whether this domain assignment is valid. In other words, if VLD and CR[GVLD] are both asserted, XRDC uses the configuration in this pair of registers. If either CR[GVLD] or this field is 0, all accesses to the peripheral are allowed. To support a coherent register state, any write to PDAC_W0_*s* forces this field to zero. This field has no effect unless XRDC is enabled (CR[GVLD] = 1). 0b - Invalid 1b - Valid
30-29 LK2	Lock Limits or prohibits writes to the set of PDAC words (PDAC_W0_*s* and PDAC_W1_*s*) for this peripheral slot. When you assert a bit in this field, it remains asserted until the next module reset. 00b-01b - Both words can be written to 10b - Domain d can update only its associated DdACP field—all other fields are read-only 11b - Locks (both words are read-only)
28-24 —	Reserved
23-0 —	Reserved

19.8.3.17 Memory Region Descriptor Word 0 (MRGD_W0_0 - MRGD_W0_95)

Offset

Registers in this array exist only for the following combinations of index values.

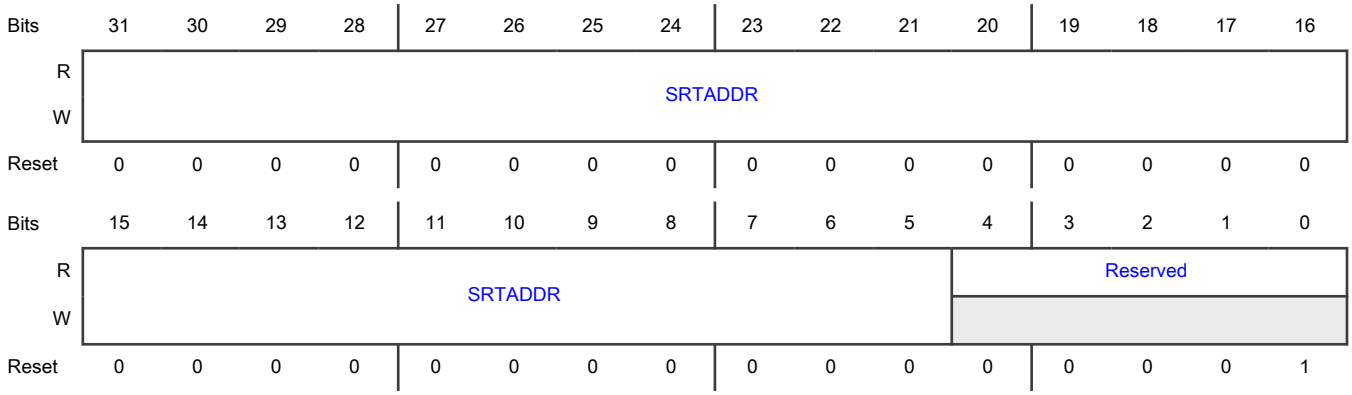
Index *n*	Index *m*
0–1, 3, 5	0–15
2, 4	0–3

Register	Offset
MRGD_W0_(n * 16 + m)	2000h + (n × 200h) + (m × 20h)

Function

Specifies the starting address of memory region *r*.

Diagram



Fields

Field	Function
31-5 SRTADDR	Start Address Specifies the most significant bits of the 0-modulo 32-byte start address of the memory region. The minimum region size is 32 bytes.
4-0 —	Reserved

19.8.3.18 Memory Region Descriptor Word 1 (MRGD_W1_0 - MRGD_W1_95)

Offset

Registers in this array exist only for the following combinations of index values.

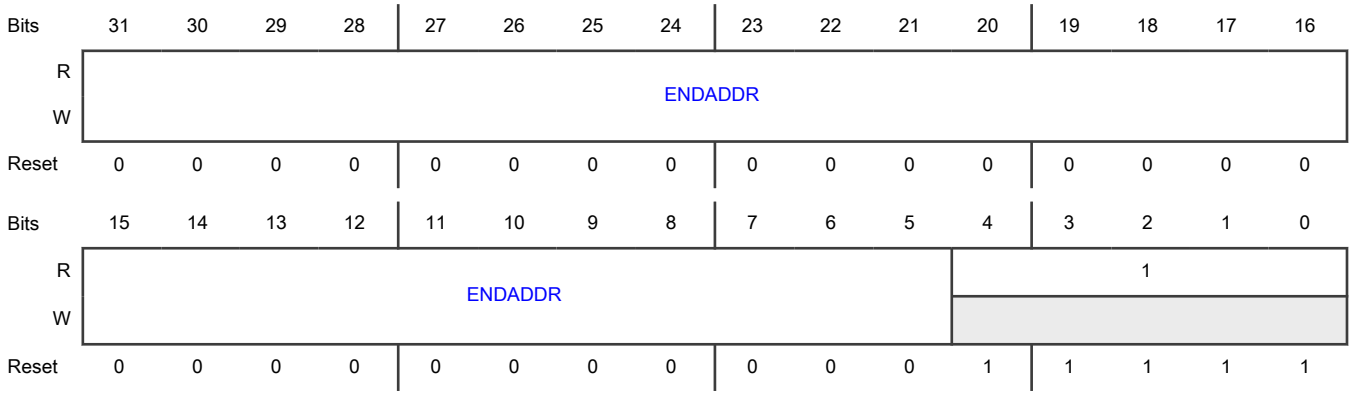
Index *n*	Index *m*
0–1, 3, 5	0–15
2, 4	0–3

Register	Offset
MRGD_W1_(n * 16 + m)	2004h + (n × 200h) + (m × 20h)

Function

Specifies the ending address of memory region *r*.

Diagram



Fields

Field	Function
31-5 ENDADDR	End Address Specifies the most significant bits of the 31-modulo 32-byte end address of memory region *r*.
4-0 —	Reserved

19.8.3.19 Memory Region Descriptor Word 2 (MRGD_W2_0 - MRGD_W2_95)

Offset

Registers in this array exist only for the following combinations of index values.

Index *n*	Index *m*
0–1, 3, 5	0–15
2, 4	0–3

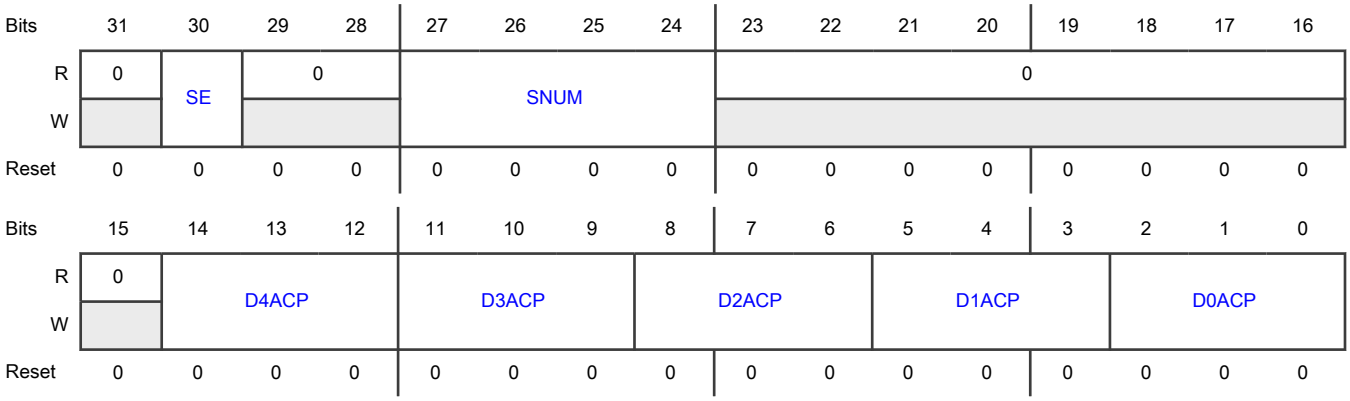
Register	Offset
MRGD_W2_(n * 16 + m)	2008h + (n × 200h) + (m × 20h)

Function

Specifies the ACP for the associated domain. The encodings specify read and write access capabilities based on the four operating states. This field applies only for a supported DID; if the DID is not implemented, the field is read-only zero (no access rights).

For field values, see Domain ACP specification.

Diagram



Fields

Field	Function
31 —	Reserved
30 SE	Semaphore Enable Enables the inclusion of the semaphore specified in SNUM in the D*d*ACP evaluation. 0b - Disables 1b - Enables
29-28 —	Reserved
27-24 SNUM	Semaphore Number Specifies the hardware semaphore to include in the D*d*ACP access evaluation. This field applies only if you enable semaphore (write 1 to SE).
23-15 —	Reserved
14-12: D4ACP 11-9: D3ACP 8-6: D2ACP 5-3: D1ACP 2-0: D0ACP	Domain Access Control Policy Specifies the ACP for the associated domain. This field applies only for a supported DID; if the DID is not implemented, the field is read-only zero (no access rights). For field values, see Domain ACP specification.

19.8.3.20 Memory Region Descriptor Word 3 (MRGD_W3_0 - MRGD_W3_95)

Offset

Registers in this array exist only for the following combinations of index values.

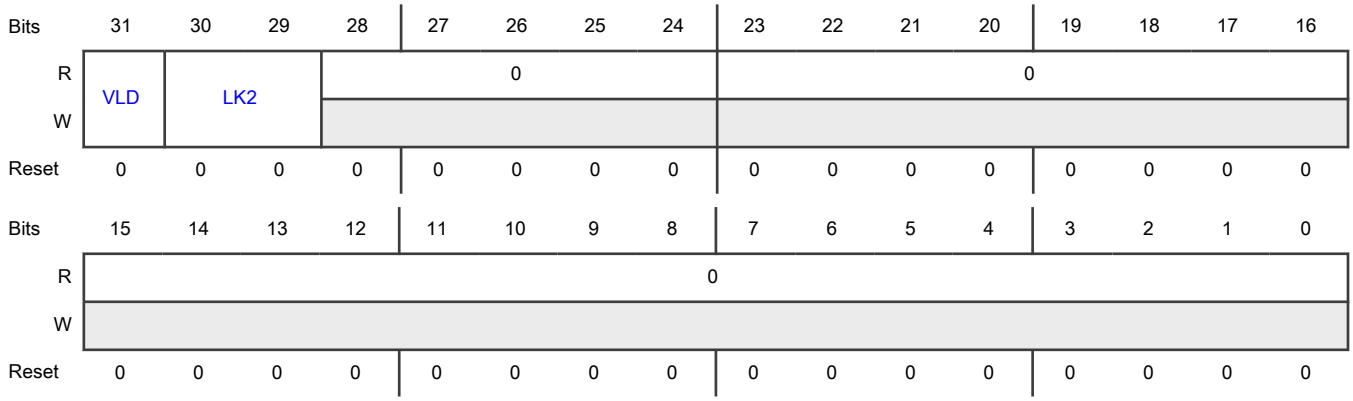
Index *n*	Index *m*
0–1, 3, 5	0–15
2, 4	0–3

Register	Offset
MRGD_W3_(n * 16 + m)	200Ch + (n × 200h) + (m × 20h)

Function

Specifies whether this memory region descriptor is enabled and limits or prohibits writes to it.

Diagram



Fields

Field	Function
31 VLD	Valid Specifies whether this domain assignment is valid. In other words, if VLD and CR[GVLD] are both asserted, XRDC uses the configuration in this set of registers. If CR[GVLD] is 0, all accesses to the memory are allowed. If CR[GVLD] is 1 and VLD is 0, all accesses are blocked. To support a coherent register state, a write to any of the MRGD W0–W2 registers forces this field to zero. This field has no effect unless XRDC is enabled (CR[GVLD] = 1). 0b - Invalid 1b - Valid
30-29 LK2	Lock Limits or prohibits writes to the set of MRGD words (MRGD_W*w_r*) for this memory region. When you assert a bit in this field, it remains asserted until the next module reset. 00b - All words in the set can be written to 01b - Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10b - Domain d can update only its associated DdACP field; all other fields are read-only 11b - Locks (all words are read-only)
28-24 —	Reserved
23-0 —	Reserved

19.9 Glossary

ACP access control policy. The access limitations specified for memory and peripheral resources.

DID domain identifier. A numeric value that identifies a specific domain.

domain An access-controlled virtual group of on-chip initiators (cores and noncore initiators) and targets (memories and peripherals) that comprise an isolated computing environment. All initiators in a domain have the same access to chip resources within that domain.

See Introduction to domains for more information.

initiator A processor core or non-processor module, such as DMA or a communications channel, that can initiate transactions with memory and peripheral resources. This term replaces "master".

LPID logical partition identifier. Also called an operating system ID or VMID, the LPID identifies a virtual initiator (either core or noncore) that runs on a hypervisor.

MDAC Master Domain Assignment Controller. Manages resource assignments and DIDs.

MGR Manager. Manages accesses through the XRDC programming model.

MRC Memory Region Controller. Controls access to memories based on memory region descriptors.

PAC Peripheral Access Controller (also sometimes called PDAC). Controls access to peripherals.

PDAC See PAC.

PID process identifier. A numeric value provided by some core processors to identify the currently active process.

SDAC Deprecated. See MRC.

target A peripheral or memory resource that one or more initiators can access. This term replaces "slave".

transaction A read or write request made by an initiator to a target peripheral or memory.

VMID virtual machine identifier. See LPID.

Chapter 20
Memory and Memory Interfaces

20.1 Introduction

This chapter discusses the configuration of memories and memory interfaces, such as flash memory, flash memory controller, and SRAM.

20.2 Flash memory controller and flash memory modules

For information on this, see the following chapters:

- Flash Memory Controller (PFLASH)
- Embedded Flash Memory

20.3 Related information

Table 100. Related information

Topic	Related chapters	For additional information
System memory map	Memory Map	See the memory map file attached to this document.
Clocking	• Clocking Overview • Clock Generation Module (MC_CGM)	—
Arm Cortex-M7 core	Cortex-M7 Overview	
XRDC	Extended Resource Domain Controller	
Direct-memory access	• Direct Memory Access Multiplexer (DMAMUX) • Enhanced Direct Memory Access (eDMA)	See the DMAMUX map file attached to this document.
EIM	Error Injection Module	—
ERM	Error Reporting Module	—

20.4 SRAM access

In case a master accesses an SRAM with multi-bit ECC errors, the chip may respond as follows:

- Map all such faults to FCCU. The recommended reaction for the fault is to generate the functional reset.
- Map such faults to ERM. If the ERM interrupt is enabled, ERM generates an interrupt.

20.5 Memories

The following table provides information on the types of memories, with their associated configurations, available in S32K3xx family.

Table 101. Memory configuration

Memory	Size	Configuration (words × bits)	ECC or parity	ECC or parity width	Diagnostic information or error report	Applicability
SRAM0	160 KB	32 KB with STANDBY mode retention (4096 × (64 + 8)) 128 KB (16384 × (64 + 8))	ECC (SECDED)	8	ERM	S32K314 S32K324 S32K344
	64 KB	32 KB with STANDBY mode retention (4096 × (64 + 8)) 32 KB (4096 × (64 + 8))	ECC (SECDED)	8	ERM	S32K342 S32K322 S32K341
	96 KB	32 KB with STANDBY mode retention (4096 × (64 + 8)) 64 KB (8192 × (64 + 8))	ECC (SECDED)	8	ERM	S32K312
	32 KB	32 KB with STANDBY mode retention (4096 × (64 + 8))	ECC (SECDED)	8	ERM	S32K311
	16 KB	16 KB with STANDBY mode retention (2048 × (64 + 8))	ECC (SECDED)	8	ERM	S32K310
	256 KB	64 KB with standby mode retention (8192 x (64 +8)) 128 KB (16384 x (64+8)) 64 KB (8192 x (64 +8))	ECC (SECDED)	8	ERM	S32K358 S32K356 S32K388 S32K338 S32K348 S32K328
	512 KB	64 KB with STANDBY mode retention (8192 x (64 +8)) 64 KB (8192 x (64 +8)) 64 KB (8192 x (64 +8)) 64 KB (8192 x (64 +8)) 64 KB (8192 x (64 +8)) 64 KB (8192 x (64 +8)) 64 KB (8192 x (64 +8)) 64 KB (8192 x (64 +8))	ECC (SECDEC)	8	ERM	S32K389
SRAM1	160 KB	32 KB (4096 × (64 + 8)) 128 KB (16384 × (64 + 8))	ECC (SECDED)	8	ERM	S32K314 S32K324 S32K344

Table continues on the next page...

Table 101. Memory configuration (continued)

Memory	Size	Configuration (words × bits)	ECC or parity	ECC or parity width	Diagnostic information or error report	Applicability
	256 KB	128 KB (16384 x (64+8))	ECC (SECDED)	8	ERM	S32K358 S32K356 S32K388 S32K338 S32K348 S32K328
		128 KB (16384 x (64+8))				
	512 KB	64 KB (8192 x (64 +8)) 64 KB (8192 x (64 +8)) 64 KB (8192 x (64 +8)) 64 KB (8192 x (64 +8)) 64 KB (8192 x (64 +8)) 64 KB (8192 x (64 +8)) 64 KB (8192 x (64 +8)) 64 KB (8192 x (64 +8))	ECC (SECDEC)	8	ERM	S32K389
SRAM_2	256 KB	64 KB (8192 x (64 +8))	ECC (SECDED)	8	ERM	S32K358 S32K356 S32K388 S32K338 S32K348 S32K328
		128 KB (16384 x (64+8))				
		64KB (8192 x (64 +8))				
	512 KB	64 KB (8192 x (64 +8)) 64 KB (8192 x (64 +8)) 64 KB (8192 x (64 +8)) 64 KB (8192 x (64 +8)) 64 KB (8192 x (64 +8)) 64 KB (8192 x (64 +8)) 64 KB (8192 x (64 +8)) 64 KB (8192 x (64 +8))	ECC (SECDEC)	8	ERM	S32K389
SRAM_3	384 KB	64 KB (8192 x (64 +8)) 64 KB (8192 x (64 +8)) 64 KB (8192 x (64 +8)) 64 KB (8192 x (64 +8)) 64 KB (8192 x (64 +8)) 64 KB (8192 x (64 +8))	ECC (SECDEC)	8	ERM	S32K389

Table continues on the next page...

Table 101. Memory configuration (continued)

Memory	Size	Configuration (words × bits)	ECC or parity	ECC or parity width	Diagnostic information or error report	Applicability
CM7_1 I-cache data	8 KB	4 KB (512 × (64 + 8))	ECC (SECDED)	8	ERM	S32K324 S32K322
		4 KB (512 × (64 + 8))				
CM7_1 I-cache tag	672 bytes	336 bytes (128 × (21 + 7))	ECC (SECDED)	7	ERM	
		336 bytes (128 × (21 + 7))				
CM7_1 D-cache data	8 KB	1 KB (256 × (32 + 7))	ECC (SECDED)	7	ERM	
		1 KB (256 × (32 + 7))				
		1 KB (256 × (32 + 7))				
		1 KB (256 × (32 + 7))				
		1 KB (256 × (32 + 7))				
		1 KB (256 × (32 + 7))				
		1 KB (256 × (32 + 7))				
		1 KB (256 × (32 + 7))				
CM7_1 D-cache tag	800 bytes	200 bytes (64 × (25 + 7))	ECC (SECDED)	7	ERM	
		200 bytes (64 × (25 + 7))				
		200 bytes (64 × (25 + 7))				
		200 bytes (64 × (25 + 7))				
CM7_0 I-cache data	8 KB	4 KB (512 × (64 + 8))	ECC (SECDED)	8	ERM	S32K314 S32K324
		4 KB (512 × (64 + 8))				
CM7_0 I-cache tag	672 bytes	336 bytes (128 × (21 + 7))	ECC (SECDED)	7	ERM	S32K344 S32K312
		336 bytes (128 × (21 + 7))				
CM7_0 D-cache data	8 KB	1 KB (256 × (32 + 7))	ECC (SECDED)	7	ERM	S32K342 S32K341 S32K311 S32K310 S32K322
		1 KB (256 × (32 + 7))				
		1 KB (256 × (32 + 7))				
		1 KB (256 × (32 + 7))				
		1 KB (256 × (32 + 7))				
		1 KB (256 × (32 + 7))				
		1 KB (256 × (32 + 7))				
		1 KB (256 × (32 + 7))				
CM7_0 D-cache tag	800 bytes	200 bytes (64 × (25 + 7))	ECC (SECDED)	7	ERM	
		200 bytes (64 × (25 + 7))				
		200 bytes (64 × (25 + 7))				

Table continues on the next page...

Table 101. Memory configuration (continued)

Memory	Size	Configuration (words × bits)	ECC or parity	ECC or parity width	Diagnostic information or error report	Applicability
		200 bytes (64 × (25 + 7))				
CM7_0 I-cache data CM7_1 I-cache data [1] CM7_2 I-cache data [2]	16 KB	8 KB (1024 x (64 + 8))	ECC (SECDED)	8	ERM	S32K358 S32K356 S32K388 S32K389 S32K338 S32K348 S32K328
		8 KB (1024 x (64 + 8))				
CM7_3 I-cache data	16 KB	8 KB (1024 x (64 + 8))	ECC (SECDED)	8	ERM	S32K388 S32K389
		8 KB (1024 x (64 + 8))				
CM7_0 I-cache tag CM7_1 I-cache tag [1] CM7_2 I-cache tag [2]	1280 bytes	640 bytes (256 x (20 + 7))	ECC (SECDED)	7	ERM	S32K358 S32K356 S32K338 S32K348 S32K328 S32K388 S32K389
		640 bytes (256 x (20 + 7))				
CM7_3 I-cache tag	1280 bytes	640 bytes (256 x (20 + 7))	ECC (SECDED)	7	ERM	S32K388 S32K389
		640 bytes (256 x (20 + 7))				
CM7_0 D-cache data CM7_1 D-cache data [1] CM7_2 D-cache data [2]	16 KB	2048 bytes (512 x (32 + 7))	ECC (SECDED)	7	ERM	S32K358 S32K356 S32K338 S32K348 S32K328 S32K388 S32K389
		2048 bytes (512 x (32 + 7))				
		2048 bytes (512 x (32 + 7))				
		2048 bytes (512 x (32 + 7))				
		2048 bytes (512 x (32 + 7))				
		2048 bytes (512 x (32 + 7))				
		2048 bytes (512 x (32 + 7))				
		2048 bytes (512 x (32 + 7))				
CM7_3 D-cache data	16 KB	2048 bytes (512 x (32 + 7))	ECC (SECDED)	7	ERM	S32K388 S32K389
		2048 bytes (512 x (32 + 7))				
		2048 bytes (512 x (32 + 7))				
		2048 bytes (512 x (32 + 7))				
		2048 bytes (512 x (32 + 7))				
		2048 bytes (512 x (32 + 7))				

Table continues on the next page...

Table 101. Memory configuration (continued)

Memory	Size	Configuration (words × bits)	ECC or parity	ECC or parity width	Diagnostic information or error report	Applicability
		2048 bytes (512 x (32 + 7))				
		2048 bytes (512 x (32 + 7))				
CM7_0 D-cache tag CM7_1 D-cache tag [1] CM7_2 D-cache tag [2]	1536 bytes	384 bytes (128 x (24 + 7))	ECC (SECDED)	7	ERM	S32K358 S32K356 S32K338 S32K348 S32K328 S32K388 S32K389
		384 bytes (128 x (24 + 7))				
		384 bytes (128 x (24 + 7))				
		384 bytes (128 x (24 + 7))				
CM7_3 D-cache tag	1536 bytes	384 bytes (128 x (24 + 7))	ECC (SECDED)	7	ERM	S32K388 S32K389
		384 bytes (128 x (24 + 7))				
		384 bytes (128 x (24 + 7))				
		384 bytes (128 x (24 + 7))				
DMA TCD	1 KB	1 KB (128 × (64 + 8))	ECC (SECDED)	8	ERM	S32K314 S32K324 S32K344 S32K342 S32K341 S32K322 S32K338 S32K348 S32K328 S32K358 S32K356 S32K388 S32K389
	640 bytes	640 bytes (80 × (64 + 8))	ECC (SECDED)	8	ERM	S32K312 S32K311 S32K310
FlexCAN_0	5 KB	5 KB (640 × (64 + 40))	ECC (SECDED)	40	FlexCAN_0	S32K344 S32K314 S32K324 S32K358 S32K356 S32K338

Table continues on the next page...

Table 101. Memory configuration (continued)

Memory	Size	Configuration (words × bits)	ECC or parity	ECC or parity width	Diagnostic information or error report	Applicability
						S32K328 S32K348 S32K388 S32K389
FlexCAN_0	3968 bytes	3968 bytes (496 × (64 + 40))	ECC (SECDED)	40	FlexCAN_0	S32K311 S32K310 S32K312 S32K322 S32K342 S32K341
FlexCAN_1	1920 bytes	1920 bytes (240 × (64 + 40))	ECC (SECDED)	40	FlexCAN_1	S32K314 S32K324
FlexCAN_2	1920 bytes	1920 bytes (240 × (64 + 40))	ECC (SECDED)	40	FlexCAN_2	S32K344 S32K312 S32K342 S32K341 S32K311 S32K310 S32K322
FlexCAN_3	1152 bytes	1152 bytes (144 × (64 + 40))	ECC (SECDED)	40	FlexCAN_3	S32K314 S32K324 S32K344 S32K312 S32K342 S32K341 S32K322
FlexCAN_4	1152 bytes	1152 bytes (144 × (64 + 40))	ECC(SECDED)	40	FlexCAN_4	S32K314 S32K324
FlexCAN_5	1152 bytes	1152 bytes (144 × (64 + 40))	ECC (SECDED)	40	FlexCAN_5	S32K344 S32K312
FlexCAN_1	5120 bytes	5120 bytes (640 x (64 + 40))	ECC (SECDED)	40	ERM	S32K338 S32K348 S32K328

Table continues on the next page...

Table 101. Memory configuration (continued)

Memory	Size	Configuration (words × bits)	ECC or parity	ECC or parity width	Diagnostic information or error report	Applicability
						S32K358 S32K356 S32K388 S32K389
FlexCAN_2	5120 bytes	5120 bytes (640 x (64 + 40))	ECC (SECDED)	40	ERM	S32K338 S32K348 S32K328 S32K358 S32K356 S32K388 S32K389
FlexCAN_3- FlexCAN_7	1920 bytes	1920 bytes (240 x (64 + 40))	ECC (SECDED)	40	ERM	S32K328 S32K338 S32K348 S32K358 S32K356 S32K388 S32K389
FlexCAN_8 - FlexCAN_11	1920 bytes	1920 bytes (240 x (64 + 40))	ECC (SECDEC)	40	ERM	S32K389
GMAC_TX	17408 bytes	8704 bytes (1024 x (68 + 8)) 8704 bytes (1024 x (68 + 8))	ECC (SECDED)	8	GMAC	S32K328 S32K338 S32K348 S32K358 S32K356
GMAC_0_TX GMAC_1_TX	17408 bytes	8704 bytes (1024 x (68 + 8)) 8704 bytes (1024 x (68 + 8))	ECC (SECDED)	8	GMAC	S32K388 S32K389
GMAC_RX	17408 bytes	8704 bytes (1024 x (68 + 8)) 8704 bytes (1024 x (68 + 8))	ECC (SECDED)	8	GMAC	S32K328 S32K338 S32K348 S32K358 S32K356
GMAC_0_RX GMAC_1_RX	17408 bytes	8704 bytes (1024 x (68 + 8))	ECC (SECDED)	8	GMAC	S32K388 S32K389

Table continues on the next page...

Table 101. Memory configuration (continued)

Memory	Size	Configuration (words × bits)	ECC or parity	ECC or parity width	Diagnostic information or error report	Applicability
		8704 bytes (1024 x (68 + 8))				
GMAC_TSN	1728 bytes	1728 bytes (512 x (27 + 7))	ECC (SECDED)	7	GMAC	S32K328 S32K338 S32K348 S32K358 S32K356
GMAC_0_TSN GMAC_1_TSN	1728 bytes	1728 bytes (512 x (27 + 7))	ECC (SECDED)	7	GMAC	S32K388 S32K389
GMAC_RXPARSER	960 bytes	960 Bytes (80 x (96 + 8)	ECC (SECDED)	8	GMAC	S32K328 S32K338 S32K348 S32K358 S32K356
GMAC_0_RXPARSER GMAC_1_RXPARSER	960 bytes	960 Bytes (80 x (96 + 8)	ECC (SECDED)	8	GMAC	S32K388 S32K389
QuadSPI RAM	1 KB	1 KB (128×64)	No	0	Not applicable	S32K328 S32K338 S32K348 S32K358 S32K356 S32K388 S32K389
EMAC TX	8960 bytes	4480 bytes (1024 × (35 + 7)) 4480 bytes (1024 × (35 + 7))	ECC (SECDED)	7	EMAC	S32K314 S32K324 S32K344 S32K342 S32K341 S32K322
EMAC RX	8960 bytes	4480 bytes (1024 × (35 + 7)) 4480 bytes (1024 × (35 + 7))	ECC (SECDED)	7	EMAC	
EMAC TSN	1664 bytes	1664 bytes (512 × (26 + 7))	ECC (SECDED)	7	EMAC	
EMAC RXPARSER	960 bytes	960 bytes (80 × (96 + 8))	ECC (SECDED)	8	EMAC	
QuadSPI TX	320 bytes	320 bytes (80×32)	No	0	Not applicable	

Table continues on the next page...

Table 101. Memory configuration (continued)

Memory	Size	Configuration (words × bits)	ECC or parity	ECC or parity width	Diagnostic information or error report	Applicability
Boot ROM[3]	160 kB	32 kB(8192x(32+0))	No	0	Not applicable	S32K311 S32K310 S32K312 S32K342 S32K341 S32K344 S32K338 S32K348 S32K328 S32K358 S32K356 S32K388 S32K389
QuadSPI TX	1152 bytes	1152 bytes (256 x (36 + 0))	No	0	Not applicable	S32K328 S32K338 S32K348 S32K358 S32K356 S32K388 S32K389
Cortex-M7 cluster ETF ETMI	1 KB	1 KB (128×64)	No	0	Not applicable	S32K314 S32K324 S32K344
Cortex-M7 cluster ETF ETMD	2 KB	2 KB (128×128)	No	0	Not applicable	
HTM ETF	1 KB	1 KB (128×64)	No	0	Not applicable	
Shared system ETF	2 KB	2 KB (256×64)	No	0	Not applicable	
Cortex-M7 cluster ETF ETMI	2 KB	2 KB (256 x 64)	No	0	Not applicable	S32K328 S32K338 S32K348 S32K358 S32K356
Cortex-M7 cluster ETF ETMI	4 KB	4 KB (512 x 64)	No	0	Not applicable	S32K388 S32K389
Cortex-M7 cluster ETF ETMD	2 KB	2 KB (128 x 128)	No	0	Not applicable	S32K328 S32K338 S32K348

Table continues on the next page...

Table 101. Memory configuration (continued)

Memory	Size	Configuration (words × bits)	ECC or parity	ECC or parity width	Diagnostic information or error report	Applicability
						S32K358 S32K356
Cortex-M7 cluster ETF ETMD	4 KB	4 KB (256 x 128)	No	0	Not applicable	S32K388 S32K389
HTM ETF	1 KB	1 KB (128 x 64)	No	0	Not applicable	S32K338 S32K348 S32K328 S32K358 S32K356
HTM ETF	2 KB	2 KB (256 x 64)	No	0	Not applicable	S32K388 S32K389
Shared System ETF	4 KB	4 KB (512 x 64)	No	0	Not applicable	S32K328 S32K338 S32K348 S32K358 S32K356 S32K388 S32K389
CM7_0_ITCM	32kB	4096x(64+8)	ECC (SECDED)	8	ERM	S32K311 S32K310 S32K312 S32K322 S32K314 S32K324 S32K328 S32K338 S32K388 S32K389 (CM7_0 not in lockstep)
CM7_0_DTCM	64kB	8192x(32+8) 8192x(32+8)	ECC (SECDED)	8	ERM	S32K311 S32K310 S32K312 S32K322 S32K314 S32K324 S32K328 S32K338 S32K388

Table continues on the next page...

Table 101. Memory configuration (continued)

Memory	Size	Configuration (words × bits)	ECC or parity	ECC or parity width	Diagnostic information or error report	Applicability
						S32K389 (CM7_0 not in lockstep)
CM7_1_ITCM	32kB	4096x(64+8)	ECC (SECDED)	8	ERM	S32K322 S32K324 S32K328 S32K338 S32K388 S32K389 (CM7_0 not in lockstep)
CM7_1_DTCM	64kB	8192x(32+8) 8192x(32+8)	ECC (SECDED)	8	ERM	S32K322 S32K324 S32K328 S32K338 S32K388 S32K389 (CM7_0 not in lockstep)
CM7_2_ITCM	64kB	8192*(64+8)	ECC (SECDED)	8	ERM	S32K338 S32K348 S32K328 S32K358 S32K356
CM7_2_DTCM	128kB	16384x(32+8) 16384x(32+8)	ECC (SECDED)	8	ERM	S32K338 S32K348 S32K328 S32K358 S32K356
CM7_2_ITCM	32kB	4096x(64+8)	ECC (SECDED)	8	ERM	S32K388 S32K389
CM7_2_DTCM	64kB	8192x(32+8) 8192x(32+8)	ECC (SECDED)	8	ERM	S32K388 S32K389
CM7_3_ITCM	32kB	4096x(64+8)	ECC (SECDED)	8	ERM	S32K388 S32K389
CM7_3_DTCM	64kB	8192x(32+8) 8192x(32+8)	ECC (SECDED)	8	ERM	S32K388 S32K389

Table continues on the next page...

Table 101. Memory configuration (continued)

Memory	Size	Configuration (words × bits)	ECC or parity	ECC or parity width	Diagnostic information or error report	Applicability
CM7_0_ITCM	64kB	4096x(64+8) 4096x(64+8)	ECC (SECDED)	8	ERM	S32K358 S32K356 S32K348 S32K344 S32K342 S32K341
CM7_0_DTCM	128kB	8192x(32+8) 8192x(32+8) 8192x(32+8) 8192x(32+8)	ECC (SECDED)	8	ERM	S32K358 S32K356 S32K348 S32K344 S32K342 S32K341
ACE FEED_DMA	1 Kb	128x(64+8)	ECC (SECDED)	8		S32K388 S32K389
ACE RESULT_DMA	1 Kb	128x(64+8)	ECC (SECDED)	8		S32K388 S32K389

1. Not applicable for S32K358, S32K356, and S32K348.
2. Not applicable for S32K348 and S32K328.
3. Five instances of this memory

20.6 Recommendations for Arm memories

As per Arm M-7 Safety manual, following considerations must be ensured for proper operation of Arm memories:

- ITCM and DTCM must be properly initialized with correct ECC before any read operation to avoid any code runaway or software malfunction or core lockup.

  NOTE

  ITCM must be initialized with 64-bit writes whereas DTCM can be initialized with 32-bit writes also.

- To safely disable TCM:
  1. Clear ITCMCR[EN] or DTCMCR[EN] as required. See Arm Cortex-M7 Devices Generic User Guide for details on ITCMCR and DTCMCR register.
  2. Execute DSB instruction
  3. Execute ISB instruction

  NOTE

  Care must be taken if disabling the ITCM while executing from it. In this case, software must ensure that the switch-off code is stored in the L2 code memory region from where execution continues when the ITCM is disabled.

- To safely disable the I-cache:
  1. Clear CCR[IC]. See Arm Cortex-M7 Devices Generic User Guide for details on CCR register.
  2. Execute DSB instruction
  3. Execute ISB instruction
- To safely disable the D-cache:

1. Clean and invalidate non-WT locations in D-cache
2. Clear CCR[DC]
3. Execute DSB instruction

See Table 102 for details on memory ECC initialization.

20.7 Memory ECC initialization summary

The table below summarizes memory ECC initialization.

Table 102. Memory ECC initialization summary

Memory	Write access size	Masters		
		CM7_ n	CM7_ m	eDMA
SRAM	64-bits only	System	System	System
CM7_n ITCM	64-bits only	Direct or Backdoor	Backdoor	Not possible
CM7_n DTCM	32-bits or 64-bits	Direct or Backdoor	Backdoor	Backdoor
CM7_m ITCM	64-bits only	Backdoor	Direct or Backdoor	Not possible
CM7_m DTCM	32-bits or 64-bits	Backdoor	Direct or Backdoor	Backdoor

20.8 Glossary

DTCM	Data tightly coupled memory
ECC	Error code correction
ETF	Embedded trace FIFO
ETMD	Embedded trace macrocell-data
ETMI	Embedded trace macrocell-instruction
ITCM	Instruction tightly coupled memory
MTB	Macrocell trace buffer
PKC	Public key cryptography
RXPARSER	Receive parser memory
SECDED	Single error correction double error detection
SRAM	Static random access memory
TSN	Time sensitive network

Chapter 21
Embedded Flash Memory (c40asf)

21.1 Chip-specific c40asf flash memory information

21.1.1 Flash memory configuration and register settings

Table 103 shows flash memory blocks and their associated configuration.

NOTE

- Flash cannot be accessed during RWSC programming.
- User-accessible Code and Data flash spaces have been all erased at the NXP factory.
- Embedded Flash Memory (c40asf) module supports a total memory size up to 8.5 MB of NVM. S32K389 uses two blocks of 8 MB and 4 MB memory size. Therefore, the total memory size supported is 12 MB. Please see the "Memory map and register definition" section for details on FLASH0 and FLASH1 instances.

Table 103. Flash block configuration

Block or address number	Block name	Start address (hex)	End address (hex)	Size	Applicability
0	Code flash memory 0	0040_0000	0047_FFFF	512 KB	S32K310, S32K311
		0040_0000	004F_FFFF	1 MB	S32K3x4, S32K342, S32K312, S32K322, S32K341
		0040_0000	005F_FFFF	2 MB	S32K3x8, S32K356
1	Code flash memory 1	0048_0000	004F_FFFF	512 KB	S32K311
		0050_0000	005F_FFFF	1 MB	S32K3x4, S32K342, S32K312, S32K322
		0060_0000	007F_FFFF	2 MB	S32K3x8, S32K356
2	Code flash memory 2	0060_0000	006F_FFFF	1 MB	S32K3x4
		0080_0000	009F_FFFF	2 MB	S32K3x8, S32K356
3	Code flash memory 3	0070_0000	007F_FFFF	1 MB	S32K3x4
		00A0_0000	00BF_FFFF	2 MB	S32K3x8
4[1]	Data flash memory	1000_0000	1001_FFFF	128 KB	S32K3x8, S32K356, S32K3x4
2[1]				128 KB	S32K342, S32K312, S32K322, S32K341
		1000_0000	1000_FFFF	64 KB	S32K311, S32K310
0 [2]	UTest NVM	1B00_0000	1B00_1FFF	8 KB	S32K3x8, S32K356, S32K3x4, S32K342, S32K312, S32K322, S32K341, S32K311, S32K310

1. Sector operation should be used for data flash during high voltage operation.
2. The address region number is the same as block number for all the blocks except this one. Address region is called UTest NVM address region in this case.

Table 104. Flash block configuration (S32K389)

Block or address number	Block name	Start address (hex)	End address (hex)	Size	Applicability
0	Code flash memory (PFC1 Block 0)	0040_0000	004F_FFFF	1 MB	S32K389
1	Code flash memory (PFC1 Block 1)	0050_0000	005F_FFFF	1 MB	S32K389
2	Code flash memory (PFC0 Block 0)	0060_0000	007F_FFF	2 MB	S32K389
3	Code flash memory (PFC0 Block 1)	0080_0000	009F_FFFF	2 MB	S32K389
4	Code flash memory (PFC1 Block 2)	00A0_0000	00AF_FFFF	1 MB	S32K389
5	Code flash memory (PFC1 Block 3)	00B0_0000	00BF_FFFF	1 MB	S32K389
6	Code flash memory (PFC0 Block 2)	00C0_0000	00DF_FFFF	2 MB	S32K389
7	Code flash memory (PFC0 Block 3)	00E0_0000	00FF_FFFF	2 MB	S32K389
8[1]	Data flash memory (PFC0 Block 3)	1000_0000	1003_FFFF	256 KB	S32K389
9 [2]	UTest NVM (PFC0 Block 0)	1B00_0000	1B00_1FFF	8 KB	S32K389

No application cores except HSE_B can access the address regions shown in Table 105.

Table 105. Secure flash memory configuration

Description	S32K311/ S32K310/ S32K341		S32K312		S32K342/ S32K322		S32K314/ S32K324/ S32K344		S32K328/ S32K338/ S32K348/ S32K358/ S32K356/ S32K388		S32K389		Size
	Start address	End address	Start address	End address	Start address	End address	Start address	End address	Start address	End address	Start address	End address	
Secure BAF code (SBAF_CODE)	004F_4000h	004F_FFFFh	005F_4000h	005F_FFFFh	005F_4000h	005F_FFFFh	007F_4000h	007F_FFFFh	0x00BF_4000	0X00BF_FFFF	0x00FF_4000	0x00FF_FFFF	48 KB

NOTE

For HSE memory configuration, see *'Memory map when HSE firmware usage feature flag is enabled'* section in *'Boot Overview'* chapter.

The flash memory can perform multiple reads in parallel (between different blocks), using the single-, dual-, or quad-read feature. It also includes an internal UTest mode that can generate single- and double-bit ECC errors.

The code flash memory lets you access data paths as per the following table:

Table 106. Data paths

Chip	No. of data paths that can be accessed simultaneously
S32K310, S32K311, S32K312	2
S32K342, S32K344, S32K324, S32K314, S32K322, S32K341	3
S32K388, S32K358, S32K356, S32K348, S32K338, S32K328	4
S32K389	4[1]

1. Parallel dual-read.

You can configure the flash memory sectors as erase- or write-protected by using the flash memory's sector and super-sector locking features.

Data flash and/or Code flash memory blocks can be altered by a code executing from different flash block or SRAM.

Data flash memory is the same as code flash memory—supporting program, erase, and read operations.

After the completion of the program or erase operations, the MCR and MCRS registers notify you as soon as the code flash memory code can be executed. At the end of reset recovery, MCRS[DONE] transitions from 0 to 1.

NOTE

MCRS[RWE] might become 1 during the program or erase operation because of speculative accesses by Cortex-M7 core(s). If there are memory regions where no speculative accesses must be initiated, Arm recommends to configure the on-core MPU to set those regions with these attributes:

- Device or strongly-ordered
- Execute never

NOTE

- ADR[A3] and ADR[A4] field are reserved for S32K312, as code flash memory size in 2 MB for S32K312.
- For S32K311 and S32K312, reset value of FLASH_SSPELOCK and FLASH_XSSPELOCK register is 0x0FFF_FFFF.

21.1.2 MCRE register reset value

The reset value of the MCRE register depends on the chip-specific flash memory configuration. This table provides the flash memory configuration of the chip and the associated register reset values.

Table 107. MCRE register reset value

MCRE fields	Reset value
n1M (bit position 23:21)	100b – Four 1 MB blocks
	010b – Two 1 MB blocks

Table continues on the next page...

Table 107. MCRE register reset value (continued)

MCRE fields	Reset value
n2M (bit position 31:29)	100b – Four 2 MB blocks
n512K (bit position 15:14)	00b – Zero 512 KB blocks
	01b – One 512 KB block
	10b – Two 512 KB blocks
n256K (bit position 7:6)[1]	01b – One 256 KB blocks
MCRE register reset value	S32K310 - 0000_8040h
	S32K311 - 0000_8040h
	S32K341- 0020_0040h
	S32K3x4 - 0080_0040h
	S32K3x2 - 0040_0040h
	S32K3x8 - 8000_0040h
	S32K389- 000_0040h (FLASH_0) 0080_0040h (FLASH_1)

1. For block size less than equal to 256 KB

21.1.3 Debugger considerations while Flash Program/Erase

While flash programming is done via debugger, special considerations should be taken care off. See "Debug Subsystem" chapter for details.

21.2 Introduction

The primary function of the embedded flash memory serves as an electrically programmable and erasable non-volatile memory (NVM) that may be used for instruction or data storage.

21.2.1 Overview

The embedded flash memory is designed for use in embedded MCU and SoC applications. It supports a total memory size up to 8.5 MB of NVM in main space and 8 KB of UTest NVM space per module. Multiple flash modules may be instantiated within an SoC to achieve higher density. The embedded flash memory is addressable by page (256 bits) for read operation and double-word(s) and page and quad page for program operations. Flash memory reads always return 256 bits, although read page buffering may be performed by the PFC. The flash memory is able to do multiple reads in parallel (between different blocks), utilizing the quad read feature.

For details on the embedded flash memory architecture and features, see Functional description.

The following figure shows the top-level diagram and functional organization of the flash memory unit.

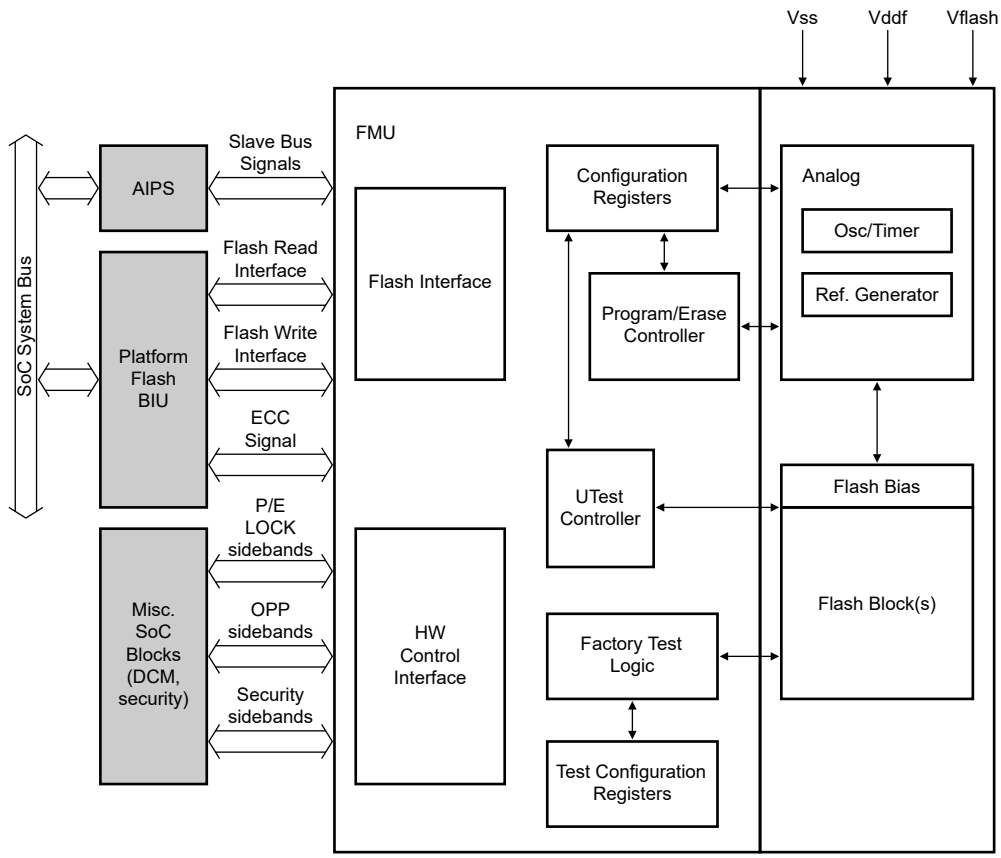


Figure 51. Block diagram

21.2.2 Features

The embedded flash memory includes these distinct features:

- Test information stored in a dedicated non-volatile sector (referred to as the UTest sector)
- OTP space made available in the UTest sector
- Read page size of 256 bits (8 words)
- ECC with single bit correction, double bit detection (all 1s valid) with 64-bit granularity
- Quad page programming (256-bit granularity)
- Hardware programmable sector and super sector program or erase restriction control
- Erasing of selected sector or block erase
- Independent programming of the UTest NVM sector
- Embedded hardware program and erase algorithms
- Support for reading while writing when the accesses are to different blocks
- UTest mode (user-accessible test modes) including array integrity and margin read
- Triple-voted flops for safety-critical flash memory functions (for example, internal trimming, redundancy, mode control, and others)

21.2.3 Modes of operation

Following is a brief description of the embedded flash memory operating modes.

- User mode is the default operating mode of the embedded flash memory. In this mode, it is possible to read and write registers (including register-based interlock writes), read the memory array, program the memory array, and erase the memory array. In this mode program and erase operations are initiated by doing register writes. Program and erase operations are controlled by an internal state machine.
- Low-Power mode turns off all DC current sources within the embedded flash memory and enables VFLASH and VDDF to be power gated. The embedded flash memory is not accessible for read, write, program, or erase when in Low-Power mode. This mode results in the lowest current draw that embedded flash memory can achieve.
- UTest mode is a tiered test mode strategy in which a portion of the factory test modes are made available. This mode is protected but accessible.

21.3 UTest NVM sector

The UTest NVM sector may be enabled by PEADR[PEASAD] during interlock writes. When the UTest NVM space is enabled, all program and erase operations are mapped to the UTest NVM sector. User-mode programming of the UTest NVM sector is enabled only when PEADR[PEASAD] is high.

The UTest NVM sector is an OTP sector (assuming Test mode disable seal is written) . Therefore, performing an erase operation is not permitted.

The UTest NVM sector supports RWW, and is grouped with the sectors in partition 0.

The UTest NVM sector may be locked against program by using hardware program and erase protection input to the flash module (MCRS[TSPELOCK]).

The UTest NVM sector contains specified data that is needed for embedded flash memory or SoC features.

Programming of the UTest NVM space has restrictions that are similar to those of the main space in terms of how ECC is calculated. Only one program operation is allowed per 64-bit ECC segment.

21.4 Test mode disable seal

The UTest NVM sector includes a mechanism to disable factory entry into Test mode selectable by block. Extreme care must be taken when using this feature, because blocks that are selected to be protected in this method will not have possible failures analyzed by factory failure analysts. Once sealed, the UTest sector becomes OTP (erase locked, and OPP enabled). The UTest sector is not accessible in Test Mode.

Protection of this sort prevents all high voltage operations to the flash memory executed by the internal state-machine, as well as reads through this state machine, and reads through the array integrity state machine when using Test mode interfaces.

UTest operations, margin read and array integrity, are also protected by preventing MISR updates on blocks selected for protection, although reads are still performed, and single bit corrections and double bit detections will be logged. Protection through other interfaces is provided by normal User mode protection mechanisms and is device-specific.

The method to disable factory entry into Test mode is to first program the Test mode disable seal location to be 5A4B_3C2Dh. After the next reset is asserted, Test mode is disabled.

It is possible to create a password to enable factory entry into Test mode. This can be programmed into the Test mode disable override passcode. The passcode may not be 0000_0000h, FFFF_FFFFh, or 5555_5555h. These are all invalid passcodes and will not be accepted to override. If it is desired, by the customer, that override never be possible, one of the three invalid passcodes must be put into the Test mode disable override passcode location. Passcode may be entered to authenticate entry, if enabled, by performing a 32-bit register write to register address 90h.

Even if the Test mode disable seal password is written, the UTest NVM sector is always protected and the UTest operations, margin read and array integrity, are always protected.

Only blocks selected in the Test mode disable block select field are controlled by the Test mode disable feature. Therefore, it is possible for customers to selectively pick blocks that have this type of protection and will not be eligible for factory failure

analysis. Bits programmed to 0 in the Test mode disable block select field(s) designate blocks that are controlled by the Test mode disable feature.

In order to let two opportunities to select blocks for the Test mode disable, two regions are available, Test mode disable bock select - 1 and Test mode disable block select - 2. If either Test mode disable block select - 1 or Test mode disable block select - 2 has a 0 programmed, that block will be protected. The bits of these two regions are logically ANDed to define the blocks that are effectively selected for the Test mode disable feature. The bock select field is organized as shown in the next table:

Table 108. Test mode disable block select

Block	Bits used to select blocks
Block 0 disable	Data[0]
Block 1 disable	Data[8]
Block 2 disable	Data[16]
Block 3 disable	Data[24]
Block 4 disable	Data[32]
Block 5 disable	Data[40]

21.5 Functional description

The embedded flash memory module consists of blocks with the following options:

- 2 MB blocks (0, 1, 2, 3 or 4 blocks allowed per module)
- 1 MB blocks (0, 1, 2, 3 or 4 blocks allowed per module)
- 512 KB blocks (0, 1, 2 or 4 blocks allowed per module)
- 256 KB blocks (0, 1, 2 or 4 blocks allowed per module)
- A total of six blocks are allowed per module

Read while Write is allowed within a module, and is determined based on block boundaries. Read while Write partitions are used to determine locations for valid read-while-write (RWW) operations. While the embedded flash memory performs a write (program or erase) to a given partition, it can simultaneously perform a read from any other partition. For program and erase operations, only the address specified by an interlock write determines the partition being written (sector or super sector locking does not determine the RWW partitions being written).

The main address space is also divided into sectors and super sectors to implement independent erase and program protection. The UTest NVM space also exists as a sector and has independent program protection. The UTest NVM sector is included to support systems that require non-volatile memory (NVM) for security or to store system initialization information.

A number of MCR bits are protected against write when another field, or set of fields, is in a specific state. These write locks are covered on a bit-by-bit basis in Module Configuration Register. The write locks do not consider the effects of trying to write two or more bits simultaneously. This module does not allow fields to be written simultaneously that put the device into an illegal state. This is implemented through a priority mechanism among the fields, and the next table shows this mechanism.

Table 109. MCR field set and clear priority levels

Priority level	MCR field(s)
1	ERS
2	PGM
3	EHV

If two or more MCR fields are written simultaneously, only the field with the lowest priority number is accepted for modification. For example, writing 1 to ERS and PGM simultaneously results in 1 being written only to ERS.

Each read of the embedded flash memory retrieves a page, or eight consecutive words (256 bits), of information. The address for each word retrieved within a page differs from the other addresses in the page only by address bits [4:2]. The flash memory page read architecture easily supports both cache and burst mode at the PFC BIU level for high-speed read applications.

The embedded flash memory supports fault tolerance through error correction code (ECC) and error detection. ECC implemented within the embedded flash memory corrects single bit failures and detects double bit failures.

Program and erase of the embedded flash memory requires multiple system clock cycles to complete. Program and erase may be aborted.

The embedded flash memory may operate in various modes as described in the following sections.

21.5.1 User mode

In User mode, the embedded flash memory may be read and written (register writes and interlock writes), programmed or erased. The following sub-sections define all actions that may be performed in User mode.

21.5.1.1 Read and write

The default state of the embedded flash memory is read. The main and UTest NVM address spaces can be read only in the read state. The flash registers are always available for read, except when the embedded flash memory is in Low-Power mode. The module enters the read state on reset, and remains in the read state under two sets of conditions:

- The read state is active when the embedded flash memory is enabled (User mode read).
- The read state is active when MCR[PGM] or MCR[ERS] are 1 and a high voltage operation is ongoing (RWW).

NOTE

Reads done to the partition(s) being operated on (either erased or programmed) result in MCRS[RWE] becoming 1.

In embedded flash memory, flash core reads return 256 bits (1 page). Register reads return 32 bits of data.

NOTE

Flash core reads are performed through the PFC BIU. In many cases, the BIU does read page buffering to allow sequential reads to be done with higher performance. This could provide a data coherency issue that must be handled with software. Data coherency may be an issue after a program or erase operation, as well as UTest NVM sector operations.

In user mode, registers may be written (including interlock writes using registers).

Register reads to unmapped register address space return all 0s.

Register writes to unmapped register address space have no effect.

Interlock writes that are attempted during a high voltage operation (MCR[EHV] = 1 or MCRS[DONE] =0) result in the interlock write being ignored, and address and data will not be updated.

21.5.1.2 Program

A flash memory program sequence operates on any page within the flash core. Within a page, up to eight words may be altered in a single program operation. Also, up to four pages can be altered in a single program operation. Whenever the array is programmed, the ECC bits also get programmed.

ECC is handled on a 64-bit boundary. Thus, if only one word in any given 64-bit ECC segment is programmed, the adjoining word in that segment should not be programmed, because ECC calculation is already complete for that 64-bit segment. Attempts to program the adjoining word may likely result in an operation failure. It is recommended that all programming operations ranger from 64 bits to 1024 bits, and be 64-bit aligned. The programming operation should completely fill selected ECC segments within the page. Only one program is allowed per 64-bit ECC segment between erases.

Caution

In rare cases, over-programming of a 64-bit ECC segment may be done (EEPROM emulation in data flash region). In this case, approved EEPROM emulation drivers must be used, and they must limit the number of over-program operations to three times (four total programs between erases).

Programming changes the value stored in an array bit from logic 1 to logic 0 only. Programming cannot change a stored logic 0 to a logic 1.

NOTE

If a logic 0 is attempted to be over-programmed by a logic 1, the resulting operation fails (MCRS[PEG] = 0), and the 0s that are interlocked are merged (ORed) with the 0s that are already present in the 64-bit ECC segment, unless the block is designated as an over-orogram protected block.

Addresses in locked sectors cannot be programmed. Values may be programmed in any or all of eight words, within a page, with a single program sequence. Up to four pages can be programmed at once on a quad-page boundary. The program operation consists of the following sequence of events:

1. Write the address to be programmed using logical address registers located in the PFC. The result of this write will also be reflected physically in the PEADR register.

   NOTE

   PEADR writes are initiated by writes that first occur in the PFC. Please see the platform flash memory controller chapter for more information. PEADR reads are allowed, and represent the flash memory physical address.

   NOTE

   Ensure the sector that contains the address to be programmed is unlocked. Sector and super sector lock status is latched at this step and does not change until the next program or erase sequence is started.

2. Data to be programmed must be written in the appropriate DATAX register (where X is 0 to 31). All unwritten data words default to FFFF_FFFFh.
3. Change the value in MCR[PGM] from a 0 to a 1.
4. Write a 1 to MCR[EHV] to start the internal program sequence or skip to step 8 to terminate.
5. Wait until MCRS[DONE] becomes 1.

   NOTE

   Since MCRS[DONE] clears with MCR[EHV] being set, it may not be possible for software to read MCRS[DONE] as 0 between step 4 and step 5, depending on the operation selected.

6. Confirm MCRS[PEG] = 1.
7. Write 0 to MCR[EHV].
8. Write 0 to MCR[PGM] to terminate the program sequence.

Program may be initiated with the writing of the PEADR register (enabled with a write done in the PFC block). The PEADR write to initiate the program sequence determines the quad-page address to be programmed. Data must also be written to the DATA register(s). This first write is referred to as an interlock write. Unwritten locations default to a data value of FFFF_FFFFh.

An interlock write must be performed before writing 1 to MCR[PGM], and MCR[PGM] must be set before writing 1 to MCR[EHV] during a program sequence. A program sequence may be terminated by writing 0 to MCR[PGM] prior to writing 1 to MCR[EHV].

While MCR[EHV] is 1 and MCRS[DONE] is 0, MCR[EHV] may be cleared, resulting in a program abort. A program abort forces the embedded flash memory to step 8 of the program sequence. An aborted program results in PEG becoming 0, indicating a failed operation. The data space being operated on before the abort contains indeterminate data.

Caution

Aborting a program operation leaves the FC addresses being programmed in an indeterminate data state. This may be recovered by executing an erase on the affected block.

21.5.1.2.1 Program hardware locking

Hardware locking which affects the program and erase operations is available for UTest sector and all array blocks.

21.5.1.2.2 Over-program protection enable

One-time programming can be enabled. If over-program protection is enabled, any double word that has already been programmed cannot be programmed again. The over-program protection enable does not affect the erase operation. Attempts to over-program result in MCRS[PEG] becoming 0. If any double word within a quad-page has an over-program protection violation, MCRS[PEG] becomes 0, and no double words are programmed.

One-time programming can be enabled for the UTest NVM sector.

21.5.1.3 Erase

An erase changes the value stored in all bits of the selected sector or block to logic 1, and operates on a sector or a block in the main address space. The erase sequence is fully automated within the flash memory. Locked sectors cannot be erased. The erase sequence consists of the following events:

1. Write to the sector or block address to be erased using logical address registers located in the PFC. The result of this write is reflected physically in the PEADR register. One, and only one, DATA register must also be written. This is referred to as an erase interlock write.

   NOTE

   PEADR writes are initiated by writes that first occur in the PFC. Please see the platform memory flash controller chapter for more information. PEADR reads are allowed, and represent the flash memory physical address.

   NOTE

   The selected block or sector must be unlocked prior to initiating an erase with the appropriate sector or super sector lock registers. Sector and super sector lock status is latched at this step and does not change until the next program or erase sequence starts. If a block is selected for erase, all the sectors and super sectors must be unlocked within that block.

2. Change the value in MCR[ERS] from a 0 to a 1, and at the same time select the size of erase using MCR[ESS].
3. Write a 1 to MCR[EHV] to start the internal erase sequence or skip to step 6 to terminate.
4. Wait until MCRS[DONE] becomes 1.

   NOTE

   Since MCRS[DONE] clears with MCR[EHV] being set, it may not be possible for software to read MCRS[DONE] as 0 between step 3 and step 4, depending on the operation selected.

5. Confirm MCRS[PEG] = 1.
6. Write 0 to MCR[EHV].
7. Write 0 to MCR[ERS] to terminate the erase, and if set MCR[ESS] should also be cleared (else the field auto clears).

Erase may be initiated with the writing of the PEADR register (enabled with a write done in the Platform Flash Controller block). The PEADR write to initiate the erase sequence determines the sector or block to be erased. One write must also occur to a DATA register. Data word written during an erase sequence interlock write is ignored. This write is referred to as an erase interlock write.

An erase interlock write must occur before writing 1 to MCR[ERS] (and optionally MCR[ESS]), and MCR[ERS] must be set before writing 1 to MCR[EHV] during an erase sequence. The erase sequence may be terminated by writing 0 to MCR[ERS] prior to writing 1 to MCR[EHV].

While MCR[EHV] is 1 and MCRS[DONE] is 0, MCR[EHV] may be cleared, resulting in an erase abort. An erase abort forces the embedded flash memory to step 7 of the erase sequence. An aborted erase results in PEG becoming 0, indicating a failed operation. The sector or block being operated on before the abort contain indeterminate data.

21.5.1.3.1 Erase hardware locking

Hardware locking which affects the erase and program operations is available for UTest sector and all array blocks. For details, see Program hardware locking.

21.5.1.4 Express program

A mechanism is made available to provide an express method to program up to 1024 bits of flash memory. When activated, all ongoing program and erase operations through the register interface are automatically aborted (MCR[EHV] becomes 0 automatically) and the express interface is given priority. When using the express interface, sequence requirements are ignored, although a XPEADR write is required to start the express program sequence.

The flash express program mechanism provides a quicker way to get information programmed into a block of flash memory when an application needs an immediate diary access, or a crash log access.

Addresses in locked sectors cannot be express programmed.

The express program operation consists of the following sequence of events:

1. Write the address to be programmed using the logical address register located in the PFC. The result of this write will also be reflected physically in the XPEADR register.

   NOTE

   XPEADR writes are initiated by writes that first occur to the PFC. Please see the platform flash controller chapter for more information. XPEADR reads are allowed, and represent the flash memory physical address.

   NOTE

   Ensure the sector that contains the address to be programmed is unlocked. Sector and super sector lock status is latched at this step and does not change until the next program, erase or express program is started.

   NOTE

   If XMCR[XPGM] interrupts a program or erase setup after MCR[EHV] is written to 1, the MCR and MCRS registers indicate if the operation was automatically aborted (MCR[EHV] = 0, MCRS[PEG] = 0, MCR[PGM] =1 or MCR[ERS] = 1), or if the operation was successful (MCR[EHV] = 1, MCRS[PEG] = 1, MCR[PGM] =1 or MCR[ERS] = 1). If express program interrupts a program or erase setup prior to MCR[EHV] being written to a 1, the MCR and PEADR registers will be preserved to show the state of the setup upon interruption. If fields MCR[PGM] = 1 or MCR[ERS] = 1 at interruption, the user must terminate the event by writing 0 to these fields. However, if MCR[PGM] or MCR[ERS] were not 1 at interruption, PEADR is preserved and does not need to be rewritten, but the PDATA registers are required to be rewritten for the interrupted operation. After the PDATA write(s) are repeated to finish the interlock write, the remainder of the program and erase sequence can be done to restart the interrupted operation (including doing a terminate after PGM or ERS are written to 1, if desired).

   NOTE

   Before starting the express program, ensure that UTest operations (array integrity or user margin read) are not ongoing (or about to be started). UT0[UTE] must be low before requesting the express program, and remain low during express program operations. Hardware locks are in place to ensure this occurs.

2. Wait until XMCR[XDOK] goes high.
3. Write the DATA registers with the data desired to be programmed. Up to 1024b (128B) may be programmed with express program. Only DATA registers that are written will receive program pulses.
4. Change the value in XMCR[XPGM] from a 0 to a 1.
5. Write a 1 to XMCR[XEHV] to start the internal express program sequence.

6. Wait until XMCR[XDONE] becomes 1.
7. Confirm XMCR[XPEG] = 1.
8. Write 0 to XMCR[XEHV].
9. Write 0 to XMCR[XPGM] to terminate the express program sequence.

21.5.2 Low-Power mode

In Low-Power mode, the embedded flash memory is put into a state where VDDF power gating and VFLASH power gating is allowed. It is the lowest current mode for the flash memory. No reads from or writes to the embedded flash memory are possible when in this mode. Most power dissipation is due to leakage in this mode.

Prior to entering Low-Power mode, it is required that all program, erase, and UTest operations be stopped.

When in Low-Power mode, register access is prevented. Flash core accesses are also prevented until power mode is exited. Flash core reads and writes may occur after power mode is exited.

The embedded flash memory returns to a post-reset state in all cases.

21.5.3 Program and erase watchdog timer

The embedded flash memory contains an internal watchdog timer that is used to prevent denial of service issues and events.

The watchdog timer is active during two periods of a program and erase event: program/erase setup and program/erase cleanup. If it is desired that program and erase operations also be checked with a watchdog timer, a system resource may be used for this purpose.

Program/erase setup always starts at the time of the PEADR write. An express program setup starts with a write to XPEADR and XMCR[XDOK] = 1. In the event of an Express program interrupt of main or alternate interface setup, the timer starts at the time when the express program operation completes (write of XMCR[XPGM] = 0). The watchdog stops when either MCR[EHV] is set to start the operation, or when the operation is terminated. Termination can occur when MCR[PGM] or MCR[ERS] are written to 0 (prior to MCR[EHV] being written to 1), or MCRS[PES] written to 0. If the express program interrupts the program/erase setup, this also stops the timer on that interface, and express program maintains priority. The above statements also apply to the alternate interface.

NOTE

If PEADR writes occur while UT0[UTE] is 1, the watchdog feature does not activate for setup. Watchdog timer is not active during UTest operations.

Program/erase cleanup always starts at the time of MCRS[DONE] = 1 on completion or an abort. In the event of an express program auto-abort of main or alternate interface, the cleanup starts at the time of MCRS[DONE] becoming 1 and the express program operation completed (XMCR[XPGM] written to 0). The watchdog stops when MCR[PGM] or MCR[ERS] are written to 0 at the end of the operation. If express program interrupts the program/erase cleanup, the timer on that interface also stops. The above statements also apply to the alternate interface.

Watchdog timers exist for each interface (main, alternate and express).

If a watchdog timeout occurs, the watchdog interrupt will be reflected in MCRS[WDI], AMCRS[AWDI] or XMCR[XWDI]. Interrupts can be enabled on the main and alternate interfaces using MCR[WDIE] or AMCR[AWDIE]. When a watchdog timeout occurs and an interrupt register asserts, the PEID restrictions on that interface are released. At that point, any master can complete the operation, and PEID locking is no longer enforced. If the watchdog timeout occurs, the EHV bit on that interface (MCR[EHV], AMCR[AEHV], and XMCR[XEHV]) are locked for writing to a 1. In this instance, any master can take the interface to a program and erase terminate. This enables accepted MCRS[WDI], AMCRS[AWDI] or XMCR[XWDI] to become 0, and the operation may restart with a new interlock write.

NOTE

If the watchdog timeout occurs before PGM becomes 1 (on any interface), or ERS becomes 1 (on any interface), the sequence must be taken to the point that the PGM or ERS bit for that interface is written and then the field for that interface can be cleared to create the terminate event and to clear the watchdog interrupt.

By default, the watchdog is enabled. Through the alternate MCR, the watchdog may be disabled (AMCR[WDD]), and the watchdog timeouts maybe adjusted (AMCR[WDT]) for all timeouts and all interfaces.

21.5.4 UTest mode

UTest mode enables customers to do specific tests that check the integrity of the embedded flash memory.

NOTE

When entering UTest mode, a best practice is to ensure all error flags and address error reporting in ADR are cleared to their reset value to ensure robust software and procedure execution.

NOTE

UTest mode diagnostic features ECC Logic Check, EDC after ECC Logic Check, Address Encode Logic Check, and Read Reference and Voltage Check, require serialization of flash reads starting with mode setup and ending with mode exit. This may be accomplished by executing diagnostic code from system RAM (with single reads to flash tagged address) as well as ensuring any flash reads outside of this diagnostic code are idle (single core).

21.5.4.1 Array integrity self check

Array integrity is checked using a predefined address sequence (based on UT0[AIS]), and this operation is executed on selected blocks.

The data to be read is customer-specific user code programmed into the flash memory and the correct MISR signature is calculated based on that code.

Any random or nonrandom code is valid. After the operation completes, the results of the reads can be checked by reading the MISR value, to determine if an incorrect read or ECC detection was noted. Array integrity MISR value is calculated after ECC detection and correction. Array integrity requires that the read wait states control registers in the CTL register be set to match the system frequency being used.

The array integrity self check consists of the following steps:

1. Enable UTest mode.
2. Select the block to receive the array integrity check by performing an interlock write to that block. Write logical address register in the PFC (which will be reflected in the PEADR register) and 1 DATA register (PDATA is ignored). The block selected for array integrity check does not need to be unlocked.

   NOTE

   Blocks protected with the Test mode disable seal are still read as part of the array integrity sequence. The resulting read on sealed blocks is not captured in the MISR, but single bit correction, double bit detection and breakpoints are still honored.

   NOTE

   It is not possible to perform array integrity operations on the UTest NVM sector.

3. If desired, write 1 to UT0[AIS] for sequential addressing only.

   NOTE

   For normal integrity checks of the flash memory, sequential addressing is recommended. If it is required to more thoroughly check the read path (in a diagnostic mode), AIS shall remain 0 to examine more read transitions. This sequence takes more time.

4. Seed the MISR registers (UM0 - UM9) with desired values.
5. If breakpoints are desired, write 1 to UT0[AIBPE], and ensure that MCRS[EER] and MCRS[SBC] are 0. If it is desired to break on a single bit correction, ensure that UT0[SBCE] = 1.
6. Write 1 to UT0[AIE].

a. If desired, the array integrity operation may be aborted before UT0[AID] goes high. This may be done by writing 0 to UT0[AIE] and then continuing to the next step. Note that in the event of an aborted array integrity check, MISRs contain a signature for the portion of the operation that was completed prior to the abort and are not deterministic. Before performing another array integrity operation, the UM0, UM1, UM2, UM3, UM4, UM5, UM6, UM7, UM8, and UM9 registers may need to be initialized to the desired seed value by doing register writes.

b. If desired, the array integrity operation may be suspended before UT0[AID] goes high. This may be done by writing 1 to UT0[AISUS] to request an array integrity suspend. After UT0[AISUS] becomes 1, the user should wait for UT0[AID] to become 1, which indicates that the flash memory has entered the suspend state, and normal reads to the flash memory may be done. After UT0[AID] becomes 1, UT0[AISUS] may be written to 0 to resume the array integrity sequence.

NOTE

User mode array reads requested during the array integrity test are ignored to ensure that the array integrity operation is not corrupted. The memory array does not respond to array read requests during this time. User mode array reads may be executed if suspended or at a breakpoint.

7. Wait until UT0[AID] becomes 1.
8. If breakpoints are enabled, check if UT0[NAIBP] = 1. If the value of this field is 1, MCRS[EER], MCRS[SBC], and ADDR may be checked to determine the cause of the break and the address of the break. Prior to resuming the operation, write 1 to clear MCRS[SBC] or MCRS[EER]. Then, the operation may be resumed by writing 0 to UT0[NAIBP]. Continue to wait until UT0[AID] becomes 1. If breakpoints are not enabled, or if UT0[NAIBP] = 0 when UT0[AID] becomes 1, the operation is complete. Continue to the next step.
9. Read values in MISRs (UM0 - UM9) to ensure correct signature.

NOTE

Array integrity reads may be done to unselected (or non-present) locations. Reads done to these locations do not update the MISRs, MCRS[EER] and MCRS[SBC].

10. Write 0 to UT0[AIE].

21.5.4.2 User margin read

User margin read may be done using the array integrity interface, and has all the associated features of the array integrity interface (MISR and breakpoints).

User margin reads are done at a read margin level, checking for erased bits or programmed bits encroaching on the nominal read level.

User margin read requires that the Read Wait States control registers in the CTL register be written to match the system frequency being used. Margin ECC corrections and detections are noted during the user margin read test. Margin read MISR value is calculated after ECC detection and correction.

The data to be read is customer-specific user code programmed into the flash memory.

Any random or non-random code is valid. After the operation completes, the margin read results can be checked by reading MCRS[EER] and the MCRS[SBC] to determine if zero, one, or two bits are being detected by the margin read, as well as checking the MISR signature.

The use model for margin read is in the event of a user-detected single bit correction (through user reads), a margin read may be done to check for a possible second bit falling within the selected margin levels.

The user margin read consists of the following steps:

1. Enable UTest mode.
2. Select the block to receive the array integrity check by performing an interlock write to that block. Write logical address register in the PFC (which will be reflected in the PEADR register) and 1 DATA register (PDATA is ignored). The block selected for user margin read check does not need to be unlocked.

NOTE

Blocks protected with the Test mode disable seal are still read as part of the margin read sequence. The resulting read on sealed blocks is not captured in the MISR, but single bit correction, double bit detection and breakpoints are still honored.

NOTE

It is not possible to perform margin read operations on the UTest NVM sector.

3. Write 1 to UT0[AIS] for sequential addressing only.

NOTE

For margin read checks of the flash memory, sequential addressing is recommended. Writing 0 to UT0[AIS] is possible for margin reads, but using the sequence takes more time, and is not recommended.

4. Seed the MISR registers (UM0 - UM9) with desired values.
5. Ensure that MCRS[EER] and MCRS[SBC] are 0.
6. To enable single bit correction reporting during margin read, write UT0[SBCE] = 1.
7. Write 1 to UT0[MRE].
8. Write UT0[MRV] to the desired value depending on if it is desired to do one's margin or zero's margin.
9. Write 1 to UT0[AIE].

NOTE

User mode array reads requested during the margin read test are ignored to ensure that the margin read operation is not corrupted. The memory array does not respond to array read requests during this time. User mode array reads may be executed if suspended or at a breakpoint.

NOTE

During margin read operations, with UT0[AID] = 0, it is not recommended to attempt write operations to MCRS[SBC] and MCRS[EER]. It is recommended that these fields only be written during suspend, breakpoints or at the completion of the margin read operation.

10. Wait until UT0[AID] becomes 1.
11. If breakpoints are enabled or a suspend was requested during margin read, the operation may be at a breakpoint or a suspend state. See Array integrity self check for more information.
12. Read values in the MISRs (UM0 - UM9) to ensure correct signature.

NOTE

Margin reads may be done to unselected (or non-present) locations. Reads done to these locations do not update the MISRs, MCRS[EER] and MCRS[SBC].

13. Write 0 to UT0[AIE].

21.5.4.3 ECC logic check

ECC logic can be checked by providing data to be read in UD0[EDATA], UD1[EDATA] and/or UD2[EDATAC]. Array reads can then be performed, ensuring expected results.

The ECC logic check consists of the following steps:

1. Enable UTest mode.
2. Write 1 to UT0[EIE].
3. Write to UD0[EDATA], UD1[EDATA] and/or UD2[EDATAC] to provide data and check bit values to be read. Single bit corrections or double bit detections can be simulated by properly selecting data and check bit combinations.

4. Write the page address to receive data provided in step 3 into ADR.
5. Write to UD2[ED3], UD2[ED2], UD2[ED1] and/or UD2[ED0] to select the double words on the page to receive the check.
6. Reads can now be done through the BIU using an array read request. In the event of a BIU read requested from an address that matches the address in ADR, expected data and corrections or detections are observed based on data written into UD0[EDATA], UD1[EDATA] and/or UD2[EDATAC]. MCRS[EER] and MCRS[SBC] can be checked to evaluate the status of reads done.

NOTE

In the event of an ECC error or single bit correction, during the ECC logic check (UT0[EIE] = 1), ADR is not loaded, and the address tagged to receive the ECC logic check values is preserved.

7. Once completed, write UT0[EIE] to 0.

21.5.4.4 EDC after ECC logic check

EDC after ECC logic can be checked by providing data to be read in UD3[EDDATA], UD4[EDDATA] and/or UD5[EDDATAC]. Array reads can then be performed, ensuring expected results.

The EDC after ECC logic check consists of the following steps:

1. Enable UTest mode.
2. Write 1 to UT0[EDIE].
3. Write to UD3[EDDATA], UD4[EDDATA] and/or UD5[EDDATAC] to provide data and check bit values to be read.
4. Write the page address to receive data provided in step 3 into ADR.
5. Write to UD5[EDD3], UD5[EDD2], UD5[EDD1] and/or UD5[EDD0] to select the double words on the page to receive the check.
6. Reads can now be done through the BIU using an array read request. In the event of a BIU read requested from an address that matches the address in ADR, expected EDC after ECC errors are observed based on data written into UD3[EDDATA], UD4[EDDATA] and/or UD5[EDDATAC]. MCRS[EEE] can be checked to evaluate the status of reads done.
7. Once completed, write UT0[EDIE] to 0.

21.5.4.5 Address encode logic check

Address encode logic can be checked by inverting the address encode information from the memory array in UA0[AEI] and UA1[AEI]. Array reads can then be performed, ensuring expected results.

The Address encode check consists of the following steps:

1. Enable UTest mode.
2. Write 1 to UT0[AEIE].
3. Write to UA0[AEI] and/or UA1[AEI] to provide address bit(s) to be inverted.
4. Write the page address to receive inverted addresses provided in step 3 into ADR.
5. Reads can now be done through the BIU using an array read request. In the event of a BIU read requested from an address that matches the address in ADR, expected address encode errors are observed based on address invert values written into UA0[AEI] and/or UA1[AEI]. MCRS[AEE] can be checked to evaluate the status of reads done.
6. Once completed, write UT0[AEIE] to 0.

21.5.4.6 Read reference and voltage check

Read reference and voltage detection logic can be checked by writing 1 to UT0[RRIE]. Array reads can be then be performed, ensuring expected results.

The read reference and voltage check consists of the following steps:

1. Enable UTest mode.
2. Write 1 to UT0[RRIE].
3. Write the page address to receive a read reference error MCRS[RRE] and a read voltage error MCRS[RVE] into ADR.
4. Reads can now be done through the BIU using an array read request. In the event of a BIU read requested from an address that matches the address in ADR, expected read reference and read voltage errors are observed based on ADR registers. MCRS[RRE] and MCRS[RVE] can be checked to evaluate the status of reads done.
5. Once completed, write UT0[RRIE] to 0.

21.5.5 Data flash memory requirements for EEPROM emulation

The embedded flash memory may be used to emulate an EEPROM utilizing software drivers, strategic over-programming of double words in the flash memory and following the below requirement and best practices in EEPROM software drivers.

It is required that the EEPROM emulation driver must build in fault tolerance allowing for the ability to “skip” records. In the case of an unsuccessful program, the ability to retry programming in the next available record location is required (record retirement). In the case of an unsuccessful sector erase, the ability to retire a sector is required (sector retirement).

In addition, following are best practices for the EEPROM emulation software driver to be considered:

- Choose a record scheme which allows for grouping of EEPROM data contents that are updated at one time such that unchanged data is not needlessly copied while minimizing record qualifier and status overhead. A variable length record scheme may be used to allow for grouping of data that is written together and limiting total record overhead.
- Copy only the valid records during a sector change and re-constitution of EEPROM data set to avoid needless copying of data within flash memory. This avoids rebuilding the entire data set.
- Only redundantly storing critical data which cannot be recovered in two sectors if attempting to protect against record or sector failures.
- Load level total program/erase cycles applied to flash memory locations allocated for EEPROM emulation by using a round-robin sector scheme.

When combining round-robin with sector retirement requirement, three or more sectors must be allocated for EEPROM emulation inclusive of extra spare sector(s).

For details, see NXP application notes published on this topic.

21.6 Initialization information

A reset is the highest priority operation for the embedded flash memory and terminates all other operations.

The embedded flash memory uses reset to initialize register and status fields to their default reset values. If the embedded flash memory is executing a program or erase operation (MCR[PGM] = 1 or MCR[ERS] = 1) and a reset is issued, the operation aborts and the embedded flash memory disables the high voltage logic without causing any damage to the high voltage circuits. Reset aborts all operations and forces the embedded flash memory into User mode, ready to receive accesses.

After reset is requested, MCRS[DONE] becomes 0, and remains low during reset and reset recovery.

At the end of reset recovery, MCRS[DONE] transitions from 0 to 1.

After a reset completes, register reads may be performed.

NOTE

Registers that require updating from UTest NVM information, or other inputs, may not read updated values until MCRS[DONE] becomes 1.

During reset recovery, register writes are not allowed until MCRS[DONE] becomes 1 to indicate reset recovery is complete.

Caution

Resetting during a program or erase operation leaves the FC blocks being programmed or erased in an indeterminate data state. This may be recovered by executing an erase on the affected blocks.

21.7 Memory map and register description

21.7.1 c40asf flash memory register descriptions

The embedded flash memory map consists of a flash memory array (which includes main array space and UTest NVM space) and a region of registers associated with the programming model that enable flash memory array operation and modification.

The address space consists of up to 6 blocks (with restrictions), and blocks can be 2 MB, 1 MB, 512 KB or 256KB in size.

21.7.1.1 c40asf_flash memory map

FLASH0 base address: 402E_C000h

FLASH1 base address: 4058_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Module Configuration (MCR)	32	RW	00FF_0000h
4h	Module Configuration Status (MCRS)	32	RW	0000_C100h
8h	Extended Module Configuration (MCRE)	32	R	0000_0000h
Ch	Module Control (CTL)	32	RW	0000_0600h
10h	Address (ADR)	32	RW	0000_0000h
14h	Program and Erase Address (PEADR)	32	R	0000_0000h
50h	Sector Program and Erase Hardware Lock (SPELOCK)	32	R	FFFF_FFFFh
54h	Super Sector Program and Erase Hardware Lock (SSPELOCK)	32	R	See section
70h	Express Sector Program and Erase Hardware Lock (XSPELOCK)	32	R	FFFF_FFFFh
74h	Express Super Sector Program and Erase Hardware Lock (XSSPELOCK)	32	R	See section
90h	Test Mode Disable Password Check (TMD)	32	RW	0000_0000h
94h	UTest 0 (UT0)	32	RW	0000_0001h
98h - B8h	UMISRn (UM0 - UM8)	32	RW	0000_0000h
BCh	UMISR9 (UM9)	32	RW	0000_0000h
D0h	UTest Data 0 (UD0)	32	RW	0000_0000h
D4h	UTest Data 1 (UD1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
D8h	UTest Data 2 (UD2)	32	RW	0000_0000h
DCh	UTest Data 3 (UD3)	32	RW	0000_0000h
E0h	UTest Data 4 (UD4)	32	RW	0000_0000h
E4h	UTest Data 5 (UD5)	32	RW	0000_0000h
E8h	UTest Address 0 (UA0)	32	RW	0000_0000h
ECh	UTest Address 1 (UA1)	32	RW	0000_0000h
F0h	Express Module Configuration (XMCR)	32	RW	00FF_C000h
F4h	Express Program Address (XPEADR)	32	R	0000_0000h
100h - 17Ch	Program Data (DATA0 - DATA31)	32	RW	FFFF_FFFFh

21.7.1.2 Module Configuration (MCR)

Offset

Register	Offset
MCR	0h

Function

NOTE

- A number of Module Configuration Register (MCR) bits are locked against write by other bits. These locks are discussed in relationship to each bit in this section. Simultaneously writing bits which lock each other out is also discussed in Functional description.
- See Functional description for information about simultaneous MCR writes, and priority levels.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								PEID							
W																
Reset	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PECIE	0		WDIE	0			PGM	0		ESS	ERS	0			EHV
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-16 PEID	Program and Erase Master/Domain ID This field shows the ID of the master that has started the Program or Erase sequence (as well as Array Integrity and User Margin Reads). The ID is latched when the sequence has started (writing of the PEADR register). Upon completion of an operation (Program (MCR[PGM] written to 0 and MCRS[PES] equals 0), Erase (MCR[ERS] written to 0 and MCRS[PES] equals 0), Array Integrity (UT0[AIE] written to 0), User Margin Read(UT0[AIE] written to 0)), or Program and Erase Sequence clear (MCRS[PES] cleared to 0) the PEID field will return to an all ones state. See the chip-specific information for a list of Master IDs.
15 PECIE	Program/Erase Complete Interrupt Enable PECIE provides a mechanism to trigger an interrupt request upon the assertion of the DONE status due to a high voltage event (program or erase) finishing (normal, abort). If PECIE is written while not in a high voltage event, the interrupt will not immediately trigger, but will trigger after the next high voltage event is completed. Writing (and reading) of this bit will be restricted to the master that updated the PEADR register as captured in the PEID register at the start of the Program or Erase sequence. **NOTE** See the chip-specific information for interrupt details 0b - Interrupt request not generated when MCRS[DONE] is 1 1b - Interrupt request generated when MCRS[DONE] is 1
14-13 —	Reserved
12 WDIE	Watch Dog Interrupt Enable WDIE provides a mechanism to trigger an interrupt request upon the assertion of the WDI bit on the main interface. If WDIE is asserted, when WDI asserts, an interrupt from the flash will trigger to the system. The interrupt from the flash will mirror exactly the WDI bit. WDIE does not affect the register bit WDI. Writing (and reading) of this bit will be restricted to the master that updated the PEADR register as captured in the PEID register at the start of the Program or Erase sequence. **NOTE** See the chip-specific information for interrupt details 0b - Watchdog interrupt not enabled 1b - Watchdog interrupt enabled
11-9 —	Reserved
8 PGM	Program

Table continues on the next page...

Table continued from the previous page...

Field	Function
	PGM is used as part of the setup for a program operation. A 0 to 1 transition of PGM after program interlock write(s) is part of the program sequence. A 1 to 0 transition of PGM ends the program sequence. PGM can be set while in user mode read (ERS is low and UTE is low), and after the interlock write is completed. PES must also be low to enable PGM to be set. PGM can be cleared only when EHV is low and DONE is high. PGM is cleared on reset. Writing (and reading) of this bit will be restricted to the master that updated the PEADR register as captured in the PEID register at the start of the Program or Erase sequence. 0b - Flash memory not executing a program sequence 1b - Flash memory executing a program sequence
7-6 —	Reserved
5 ESS	Erase Size Select ESS is used to qualify the embedded flash memory erase operation for either sector erase or block erase. If ESS is selected for block erase, only the main array of that block will be erased. **NOTE** Block erase is for Factory use only, under controlled conditions. See Appendix A (Electrical Specifications) for more information on environmental condition requirements, and cycle limit. **NOTE** If interlock write occurs to UTest sector, ESS will be locked from setting to a 1. Block Erase of UTest sector is not supported, only sector erase is allowed. Erase of UTest space is not allowed if sealed, although ERS can be set and a PEG error will occur in response to the HV request. ESS can only be written to a 1 at the same time that ERS is written to a 1. If ESS is set, ESS will auto clear with ERS clearing (independent of the value written to ESS). ESS can not be cleared by writing a zero. ESS is cleared on reset. Writing (and reading) of this bit will be restricted to the master that updated the PEADR register as captured in the PEID register at the start of the Program or Erase sequence. 0b - Flash memory erase is on a sector 1b - Flash memory erase is on a block
4 ERS	Erase ERS is used as part of the setup for an erase operation. A 0 to 1 transition of ERS after an erase interlock write is part of the erase sequence. A 1 to 0 transition of ERS ends the erase sequence. ERS can only be set in user mode read (PGM is low and UTE is low), and after the interlock write is completed. PES must also be low to enable ERS to be set. ERS can be cleared only when EHV is low and DONE is high. ERS is cleared on reset. Writing (and reading) of this bit will be restricted to the master that updated the PEADR register as captured in the PEID register at the start of the Program or Erase sequence. 0b - Flash memory not executing an erase sequence

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Flash memory executing an erase sequence
3-1 —	Reserved
0 EHV	Enable High Voltage The EHV bit enables the embedded flash memory for a high voltage program/erase operation. EHV is cleared on reset. EHV must be set after the PGM or ERS bit is set in a program or erase sequence. Writing (and reading) of this bit will be restricted to the master that updated the PEADR register as captured in the PEID register at the start of the Program or Erase sequence. EHV may be set, initiating a program/erase under one of the following conditions: • Erase (ERS = 1, PEP = 0, PES = 0, WDI = 0) • Program (PGM = 1, PEP = 0, PES = 0, WDI = 0) Clearing EHV while DONE is low will abort the current program/erase high voltage operation. An abort causes the value of PEG to be cleared, indicating a failed program/erase; address locations being operated on by the aborted operation contain indeterminate data after an abort. EHV may not be written to a 1 after an abort is requested (EHV being cleared) and before DONE transitions high. EHV may not be written to a 1 after it is cleared as part of a program or erase cleanup, until after the next PEADR write. **NOTE** Aborting a high voltage operation leaves addresses in an indeterminate data state. This may be recovered by executing an erase on the affected sectors. 0b - Flash memory is not enabled to perform a high voltage operation. 1b - Flash memory is enabled to perform a high voltage operation.

21.7.1.3 Module Configuration Status (MCRS)

Offset

Register	Offset
MCRS	4h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	EER	SBC	AEE	EEE	0		RVE	RRE	0			RWE	0		PEP	PES
W	W1C	W1C	W1C	W1C			W1C	W1C				W1C			W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DONE	PEG	0	WDI	0		EPEG	TSPELOCK	0							RE
W																
Reset	1	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0

Fields

Field	Function
31 EER	ECC Event Error This bit provides information on previous reads (either user initiated reads or internally initiated reset reads).If a double bit detection occurred, the EER bit is set to a '1'. This bit must then be cleared, or a reset must occur before it returns to a 0 state. This bit may not be set by software. In the event of a single bit detection and correction, this bit is not set. If EER is not set, or remains 0, this indicates that all previous reads (from the last reset, or clearing of EER) are correct. Since this bit is an error flag, it must be cleared to a 0 by writing a 1 to the register location. A write of 0 has no effect. 0b - Reads occurring normally 1b - ECC error occurred during a previous read
30 SBC	Single Bit Correction SBC provides information on previous reads, if the SBCE is set. If a single bit correction occurred, the SBC bit is set to a 1. This bit must then be cleared, or a reset must occur before this bit returns to a 0 state. If SBC is not set, or remains 0, this indicates that all previous reads (from the last reset, or clearing of SBC) did not require a correction. Since this bit is a status flag, it must be cleared to a 0 by writing a 1 to the register location. A write of 0 has no effect. 0b - Reads occurring without corrections 1b - Single bit correction occurred during a previous read
29 AEE	Address Encode Error AEE provides information on previous reads monitoring the address encode feature. On every read request to the flash, the incoming address is compared to an encoded address (row, column, and block) coming back from the memory array using the read data sense amplifier timing. If these two values do not match (zero selected, multiple selected, wrong selected), or the timing is incorrect, an address encode error will be recorded. If an address encode mismatch is detected, this bit is set to indicate that previous reads requested may have been corrupted. Since this bit is a status flag, it must be cleared to a 0 by writing a 1 to the register location. A write of 0 has no effect. 0b - Reads are occurring without address encode mismatches 1b - Previous read may be corrupted based on address encode mismatch

Table continues on the next page...

Table continued from the previous page...

Field	Function
28 EEE	EDC after ECC Error EEE provides information on previous reads monitoring the EDC after ECC feature. On every read request to the flash, ECC is recalculated serially, and if there is a mismatch between the ECC calculations (taking into account corrections or detections) a late error will be reported. If an EDC after ECC mismatch is detected, this bit is set to indicate that previous reads requested may have been corrupted. Since this bit is a status flag, it must be cleared to a 0 by writing a 1 to the register location. A write of 0 has no effect. 0b - Reads are occurring without EDC after ECC mismatches 1b - Previous read may be corrupted based on ECC calculation errors
27-26 —	Reserved
25 RVE	Read Voltage Error RVE provides information on previous reads monitoring the read voltage. If the read voltage is detected to be out of range, this bit is set to indicate that previous reads requested may have been corrupted. Since this bit is a status flag, it must be cleared to a 0 by writing a 1 to the register location. A write of 0 has no effect. 0b - Reads are occurring without voltage issues 1b - A previous read may have been corrupted due to read voltage being out of range
24 RRE	Read Reference Error RRE provides information on previous reads monitoring the read reference. If the read reference is detected to be out of range, this bit is set to indicate that previous reads requested may have been corrupted. Since this bit is a status flag, it must be cleared to a 0 by writing a 1 to the register location. A write of 0 has no effect. 0b - Reads occur without reference issues 1b - Previous read may be corrupted because of read reference being out of range
23-21 —	Reserved
20 RWE	Read-While-Write Event Error This bit provides information on previous read-while-write (RWW) reads. If an RWW error occurs, this bit is set to 1. The bit must then be cleared, or a reset must occur before it returns to a 0 state. If RWE is not set, or remains 0, this indicates that all previous RWW reads (from the last reset, or clearing of RWE) are correct. Since this bit is an error flag, it must be cleared to a 0 by writing a 1 to the register location. A write of 0 has no effect. 0b - Reads occur normally 1b - RWW error occurred during a previous read
19-18 —	Reserved
17	Program and Erase Protection Error

Table continues on the next page...

Table continued from the previous page...

Field	Function
PEP	PEP provides information about program and erase operations with respect to protection errors. A protection error occurs if a program is attempted to a locked sector or super sector, or if an erase is selected to a locked sector or super sector. This is evaluated prior to the operation beginning, and if an error is detected, high voltage operations (either a Program or Erase) will not be attempted for this request. **NOTE** If a location has both OTP and Lock protection, the response from the NVM will be PEP=1 only. If PEP is asserted, it must be cleared prior to attempting another high voltage operation. Since this bit is a status flag, it must be cleared to a 0 by writing a 1 to the register location. A write of 0 has no effect. 0b - Program and erase protection errors do not exist 1b - Previous program or erase protection error encountered
16 PES	Program and Erase Sequence Error PES provides information about program and erase operations with respect to sequence errors. A sequence error occurs when the program or erase sequence is not followed exactly. If an “out-of-order” event is detected, PES will assert, and the remainder of the sequence will not be accepted. PES monitoring begins when the PEADR register is written and ends when EHV is set to start the operation. PES monitoring only applies to Program and Erase (Sector and Block). It does not apply to Express Program, or UTest operations. Following is a complete list of sequence error conditions: • Any specific DATA register written twice. • More than one DATA register write for ERS operations. • Attempts to write PGM or ERS register out of sequence. • Attempts to write PGM after ERS, or ERS after PGM. • Attempts to write DATA registers after PGM or ERS are written. • Attempts to write EHV out of sequence. Writes to the MCR that are protected by the MCR Priority Levels (i.e. simultaneous PGM, ERS and EHV writes) will not result in a PES condition. Attempts to write MCR or DATA registers by a master that does not match the PEID, will not result in a PES condition. Those reads and writes will be blocked, and the master in control of the interface may continue with its' sequence. Clearing of PES (due to a sequence error) has the same effect as the clearing of PGM or ERS after a program or erase operation. PEADR and DATA registers are cleared, PEID is cleared, and the operation must be re-started from the beginning. If PGM or ERS are already set, they must be cleared before PES is cleared. PES can only be cleared once the PGM or ERS bit for the sequence in progress is cleared. Since this bit is a status flag, it must be cleared to a 0 by writing a 1 to the register location. A write of 0 has no effect. 0b - Program and erase sequence errors do not exist 1b - Previous program or erase sequence encountered an error
15 DONE	State Machine Status

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DONE indicates whether the embedded flash memory is performing a high voltage operation. DONE is set to a 1 on termination of the embedded flash memory reset. DONE is read only. DONE is set to a 1 at the end of program and erase high voltage sequences. **NOTE** This field transitions from a 0 to 1 during reset and remains at 1 after reset. 0b - Performing a high voltage operation 1b - Not executing a high voltage operation
14 PEG	Program/Erase Good The value of PEG is updated automatically during the program and erase high voltage operations. Aborting a program/erase high voltage operation causes PEG to be cleared, indicating the sequence failed. PEG is set to a 1 when the embedded flash memory is reset. PEG is read only. The value of PEG is valid only when PGM = 1 or ERS = 1 and after DONE transitions from 0 to 1 due to an abort or the completion of a program/erase operation. PEG is valid until PGM/ERS makes a 1 to 0 transition or EHV makes a 0 to 1 transition. **NOTE** 1. If program or erase operations are attempted on sector(s) that are locked, the response from embedded flash memory is PEG = 1, indicating that the operation was successful, and the contents of the sector(s) are properly protected from the program or erase operation. PEG = 1 is also true if an abort occurs during an HV request to a locked sector. 2. If a program or erase operation is attempted to a location marked as OTP, the response from the embedded flash memory is PEG = 0, indicating that the operation was not allowed. The value interlocked is not programmed, since desired double word was already programmed with a previous program operation. Erase is prevented on OTP locations. 0b - Program or erase operation failed 1b - Program or erase operation successful
13 —	Reserved
12 WDI	Watch Dog Interrupt WDI is a status register to indicate that the Watchdog Timer for Program or Erase had expired. WDI is status only, and will be automatically cleared once the operation that caused the watchdog timeout is terminated or clean up from the operation is completed (clearing of PGM (with PES low), ERS (with PES low) or PES bit). 0b - Normal Operation, Watchdog Timer has not expired. 1b - Program Watchdog Timer has expired.
11-10 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
9 EPEG	ECC Enabled Program/Erase Good EPEG is a qualifier to PEG to indicate if a passing program or erase required ECC Enabled verifies to pass. In the event of a failed operation (PEG=0), EPEG will always remain 0. The value of EPEG is updated automatically during the program and erase high voltage operations. EPEG is set to a 0 when the embedded flash memory is reset. EPEG is read only. The value of EPEG is valid only when PGM = 1 or ERS = 1 and after DONE transitions from 0 to 1 due to an abort or the completion of a program/erase operation. EPEG is valid until PGM/ERS makes a 1 to 0 transition or EHV makes a 0 to 1 transition. 0b - Program or erase operation did not require ECC Enabled verifies. 1b - Program or erase operation required ECC Enabled verifies to pass.
8 TSPELOCK	UTest NVM Program and Erase Lock TSPELOCK reflects the status of the hardware program and erase protection input to the flash module. The TSPELOCK register is latched on the PEADR[PEASAD] write (UTest Sector Interlock), is not writable, and is status only. During high voltage events, the status is locked. When not interlocked this bit will read 1. The default value of the TSPELOCK bits is program and erase protected. 0b - Corresponding sector not locked, and may be programmed or erased 1b - Corresponding sector protected from the program and erase sequences
7-1 —	Reserved
0 RE	Reset Error This bit provides information on previous resets. Checks are done within the flash and if a coherency issue or ECC error is detected during the reset reads used for trimming on the flash, this status flag will assert. This bit is status only, and is not writable. 0b - Reset occurred without errors 1b - Reset error encountered

21.7.1.4 Extended Module Configuration (MCRE)

Offset

Register	Offset
MCRE	8h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	n2M				0				n1M				0			
W																
Reset [1]	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	n512K		0						n256K		0					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

1. The reset value of this register is set at the factory and varies among chips. See the chip-specific c40asf information for details.

Fields

Field	Function
31-29 n2M	Number of 2 MB Blocks **NOTE** This field is not supported in every instance. The following table includes only supported registers.

Instance	Field supported in	Field not supported in
FLASH0	MCRE	—
FLASH1	—	MCRE

000b - Zero 2 MB blocks
001b - One 2 MB block
010b - Two 2 MB blocks
011b - Three 2 MB blocks
100b - Four 2 MB blocks
101b - Reserved
110b - Reserved
111b - Reserved

28-24 —	Reserved
23-21	Number of 1 MB Blocks

Table continues on the next page...

Table continued from the previous page...

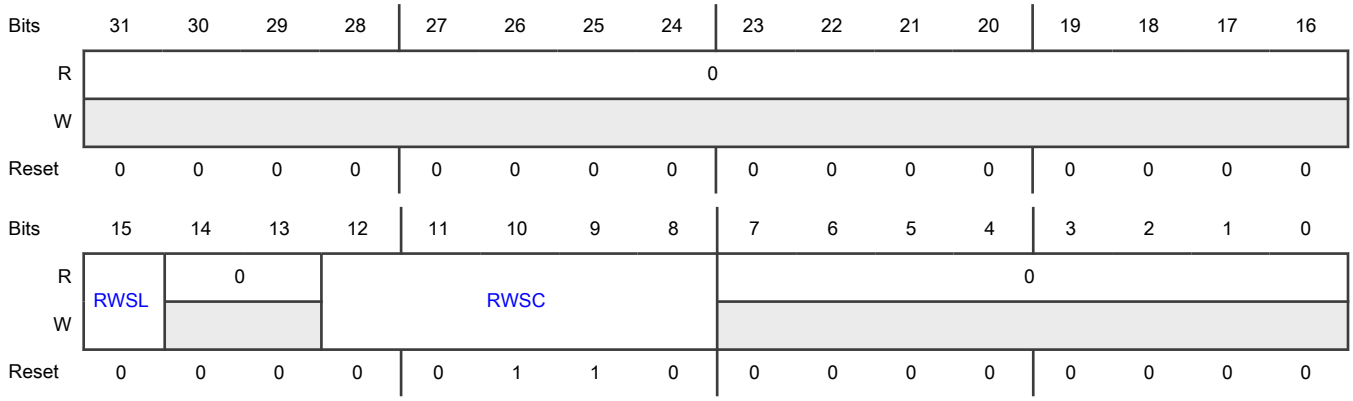
Field	Function
n1M	000b - Zero 1 MB blocks 001b - One 1 MB block 010b - Two 1 MB blocks 011b - Three 1 MB blocks 100b - Four 1 MB blocks 101b - Reserved 110b - Reserved 111b - Reserved
20-16 —	Reserved
15-14 n512K	Number of 512 KB Blocks 00b - Zero 512 KB blocks 01b - One 512 KB block 10b - Two 512 KB blocks 11b - Four 512 KB blocks
13-8 —	Reserved
7-6 n256K	Number of 256 KB Blocks 00b - Zero 256 KB blocks 01b - One 256 KB block 10b - Two 256 KB blocks 11b - Four 256 KB blocks
5-0 —	Reserved

21.7.1.5 Module Control (CTL)

Offset

Register	Offset
CTL	Ch

Diagram



Fields

Field	Function
31-16 —	Reserved
15 RWSL	Read Wait State Lock Read Wait State Lock. The RWSL bit locks the read wait state control register (RWSC). If RWSL is set, then the RWSC field can not be written. RWSL is not clearable by a register write, it is only cleared upon reset. Once RWSL is written to a 1, it will remain a 1 until the next reset and RWSC will be locked from writing. RWSC and RWSL are allowed to be written simultaneously. RWSL does not affect readability of RWSC. 0b - RWSC not locked and available for writing 1b - RWSC locked and unavailable for writing
14-13 —	Reserved
12-8 RWSC	Wait State Control This field controls the number of wait states to be added to the best-case flash array access time for reads. RWSC is only writable when RWSL is a 0. The best case flash array access time for reads is one cycle. This field must be set to a value corresponding to the operation frequency of the Flash and the actual read access time of the Flash to the flash boundary. Total read latency will be SoC dependant based on the how the cores are coupled within the platform. The frequency information, and total latency details are documented in the SoC specification. Higher operating frequencies require non-zero settings for this field for proper flash operation. **NOTE** • Updates to this configuration field that control reads must take place only when the flash memory is idle. Changing configuration settings while flash memory access is in progress can lead to non-deterministic behavior. • Values of RWSC[4:0] not listed here are reserved.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE 0_0000b - Reserved 0_0001b - One additional wait state is added. 0_0010b - Two additional wait states are added. 0_0011b - Three additional wait states are added. 0_0100b - Four additional wait states are added. 0_0101b - Five additional wait states are added. 0_0110b - Six additional wait states are added. 0_0111b - Seven additional wait states are added. 0_1000b - Eight additional wait states are added. 0_1001b - Reserved 1_1111b - Reserved
7-0 —	Reserved

21.7.1.6 Address (ADR)

Offset

Register	Offset
ADR	10h

Function

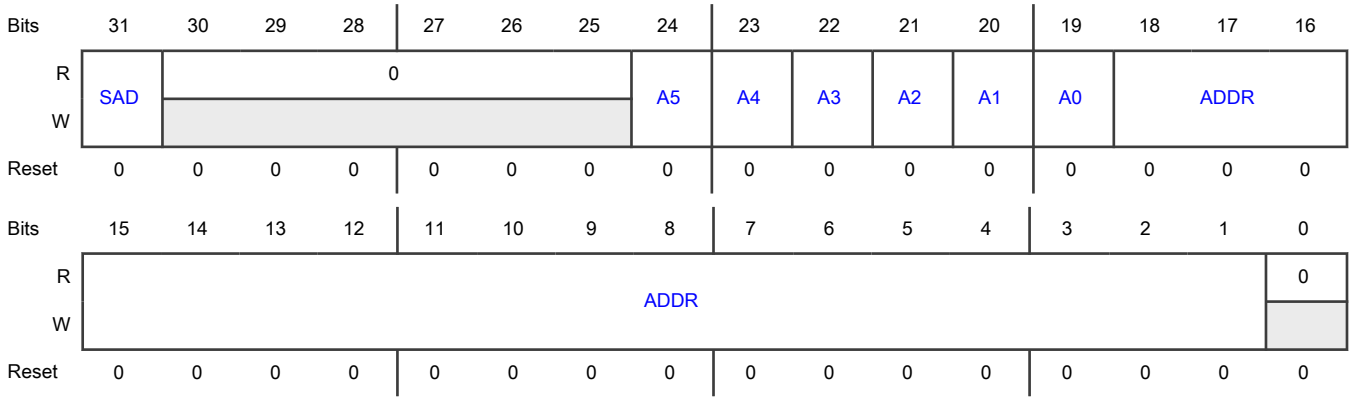
The Address register (ADR) provides the first failing address in the event of a failure (ECC or PGM/Erase state machine), as well as single bit correction information. The address register is also writable for UTest Mode operations. The address is identified using a combination of bits that identify the memory region and offset address.

NOTE

- The address given is an offset from the base address of the address space.
- The block numbering scheme that corresponds to the offset address is based on the flash memory's *internal* addressing scheme, which may be different than the chip's system addressing scheme.

Understanding the mapping between the flash address, which is specified relative to the flash module's internal addressing scheme, and the location within the chip's system memory map requires a clear understanding of the flash memory layout. See the chip-specific information for a detailed explanation of the chip's flash memory layout and how to map system addresses to the flash module's internal addressing.

Diagram



Fields

Field	Function
31 SAD	UTest NVM Address Qualifies the address captured during an ECC event error, single bit correction, or state machine operation. See the description of the ADDR field for more information. This field is not writeable if UTE = 1, and is ignored if it's in UTest mode. 0b - Address captured or to be accessed is from the main array space 1b - Address captured or to be accessed is from the UTest NVM array space
30-25 —	Reserved Reserved; reset to 0
24 A5	Address Region 5 Qualifies the address field (ADDR) to region 5 If the value of this field is 1, the ADDR field maps to region 5. See the description of the ADDR field for more information. If the region is not present, this field is locked to 0. 0b - Address captured or to be accessed is not from region 5 1b - Address captured or to be accessed is from region 5
23 A4	Address Region 4 Qualifies the address field (ADDR) to region 4 If the value of this field is 1, the ADDR field maps to region 4. See the description of the ADDR field for more information. If the region is not present, this field is locked to 0. 0b - Address captured or to be accessed is not from region 4 1b - Address captured or to be accessed is from region 4
22 A3	Address Region 3 Qualifies the address field (ADDR) to region 3

Table continues on the next page...

Table continued from the previous page...

Field	Function
	If the value of this field is 1, the ADDR field maps to region 3. See the description for the ADDR field for more information. If the region is not present, this field is locked to 0. 0b - Address captured or to be accessed is not from region 3 1b - Address captured or to be accessed is from region 3
21 A2	Address Region 2 Qualifies the address field (ADDR) to region 2 If the value of this field is 1, the ADDR field maps to region 2. See the description of the ADDR field for more information. If the region is not present, this field is locked to 0. 0b - Address captured or to be accessed is not from region 2 1b - Address captured or to be accessed is from region 2
20 A1	Address Region 1 Qualifies the address field (ADDR) to the region If the value of this field is 1, the ADDR field maps to region 1. See the description for the ADDR field for more information. If the region is not present, this field is locked to 0. 0b - Address captured or to be accessed is not from region 1 1b - Address captured or to be accessed is from region 1
19 A0	Address Region 0 Qualifies the ADDR field to region 0. If A0 = 1, then the ADDR field maps to that region. If the region is not present, this field is locked to 0. For details, see the field description for ADDR. 0b - Address captured or to be accessed is not from region 0 1b - Address captured or to be accessed is from region 0
18-1 ADDR	Address The ADR register provides the first failing address in the event of ECC event error (EER set), single bit correction (SBC set), as well as providing the address of a failure that may have occurred in a state machine operation (PEG cleared). ECC event errors take priority over single bit corrections, which take priority over state machine errors. This is especially valuable in the event of an RWW operation, where the read senses an ECC error or single bit correction, and the state machine fails simultaneously. The failing address for ECC event error is held until EER is cleared. The failing address for single bit correction is held until an ECC event error, or until SBC is cleared. State machine address is held until an ECC event error or single bit correction event occurs, or until the next state machine event (PEG cleared) occurs. This address is always a double word address that selects 64 bits. The ADR register is writable, and can be used in the UTest ECC Logic Test, UTest EDC after ECC Logic Test, and UTest Address Encode Logic Test. If any of these tests are enabled (UT0[EIE] = 1, UT0[EDIE] = 1, or UT0[AEIE] = 1) then the ADR register will not update for ECC event error, single bit correction or state machine errors. If MCRS[EER] or MCRS[SBC] are set, the ADR register is locked from writing. MCRS[PEG] does not affect the writeability of the ADR register.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	ADDR[18:1] is an offset from a base address of 0h for each block region. The A0, A1, A2, A3, A4, and A5 bits qualify the block size region the ADDR field.
0 —	Reserved Reserved; reset to 0

21.7.1.7 Program and Erase Address (PEADR)

Offset

Register	Offset
PEADR	14h

Function

The Program and Erase Address register (PEADR) is used to provide the address to be programmed or the location of the sector or block to be erased. The program and erase address register is read only in user mode (user mode writes occur with writes done to the Platform Flash Controller). The address is identified using a combination of bits that identify the memory region and offset address.

NOTE

- The address provided is an offset from the base address of the address space.
- The block numbering scheme that corresponds to the offset address is based on the flash memory's *internal* addressing scheme, which may be different from the chip's system addressing scheme.

Understanding the mapping between the flash address, which is specified relative to the flash module's internal addressing scheme, and the location within the chip's system memory map requires a clear understanding of the flash memory layout. See the chip-specific information for a detailed explanation of the chip's flash memory layout and how to map system addresses to the flash module's internal addressing.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PEASAD	0						PEA5	PEA4	PEA3	PEA2	PEA1	PEA0	PEADDR		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PEADDR											0				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 PEASAD	UTest NVM Program and Erase Address Qualifies the address field (PEADDR) to the region. If PESAD = 1, the PEADDR field maps to that region. See the description of the PEADDR field for more information. This field is not writeable if UTE = 1 and is ignored if it becomes 1 while in UTest mode. 0b - Address accessed is from the main array space 1b - Address accessed is from the UTest NVM array space
30-25 —	Reserved Reserved; reset to 0
24 PEA5	Program and Erase Address Region 5 Qualifies the address field (PEADDR) to region 5. If PEA5 = 1, the PEADDR field maps to region 5. See the description of the PEADDR field for more information. If region 5 is not present, this field is locked to 0. 0b - Address accessed is not from region 5 1b - Address accessed is from region 5
23 PEA4	Program and Erase Address Region 4 Qualifies the address field (PEADDR) to region 4. If PEA4 = 1, the PEADDR field maps to region 4. See the description of the PEADDR field for more information. If region 4 is not present, this field is locked to 0. 0b - Address accessed is not from region 4 1b - Adrress accessed is from region 4
22 PEA3	Program and Erase Address Region 3 Qualifies the address field (PEADDR) to region 3. If PEA3 = 1, the PEADDR field maps to region 3. See the description for PEADDR for more information. If region 3 is not present, this field is locked to 0. 0b - Address accessed is not from region 3 1b - Address accessed is from region 3
21 PEA2	Program and Erase Address Region 2 Qualifies the address field (PEADDR) to the region. If PEA2 = 1, the PEADDR field maps to region 2. See the description of the PEADDR field for more information. If region 2 is not present, this field is locked to 0. 0b - Address accessed is not from region 2 1b - Address accessed is from region 2

Table continues on the next page...

Table continued from the previous page...

Field	Function
20 PEA1	Program and Erase Address Region 1 Qualifies the address field (PEADDR) to the region. If PEA1 = 1, the PEADDR field maps to region 1. See the description of the PEADDR field for more information. If region 1 is not present, this field is locked to 0. 0b - Address accessed is not from region 1 1b - Address accessed is from region 1
19 PEA0	Program and Erase Address Region 0 Qualifies the address field (PEADDR) to the region. If PEA0 = 1, the PEADDR field maps to region 0. See the description of the PEADDR field for more information. If region 0 is not present, this field is locked to 0. 0b - Address accessed is not from region 0 1b - Address accessed is from region 0
18-5 PEADDR	Program and Erase Address The PEADDR register provides offset address location to be programmed or the sector to be erased in the case of sector erase. This address is always a quad page address that selects 1024 bits. The PEADDR register is read only in user mode and represents the flash physical address status. PEADDR may be updated once (and only once) per Program or Erase event. PEADDR writes initiate a Program, Erase, Array Integrity or User Margin Read request. Once PEADDR is updated, it will be locked for the full program and erase operation, until MCR[PGM] or MCR[ERS] is cleared at the end of the operation (as long as MCRS[PES] is zero). If MCRS[PES] is set, PEADR is locked until this bit is also cleared. **NOTE** If Express Program has been initiated (XPEADR written), the PEADR is locked from writing until Express Program is completed (XPGM written to 0). Upon completion of an operation (Program (MCR[PGM] written to 0 and MCRS[PES] equals 0), Erase (MCR[ERS] written to 0 and MCRS[PES] equals 0), Array Integrity (UT0[AIE] written to 0), User Margin Read(UT0[AIE] written to 0)), or Program and Erase Sequence clear (MCRS[PES] written to 0) the PEADR registers will return to an all zeros state. When MCR[EHV] or UT0[AIE] are set, PEADDR bits will not be writeable. Once PEADDR is updated, it will have read restrictions based on the PEID master that did the original write in the Platform Flash Controller. PEADDR[18:5] is an offset from a base address of 0h for each block region. The PEA0, PEA1, PEA2, PEA3, PEA4 and PEA5 qualify the block size region the PEADDR field.
4-0 —	Reserved Reserved; reset to 0

21.7.1.8 Sector Program and Erase Hardware Lock (SPELOCK)

Offset

Register	Offset
SPELOCK	50h

Function

The Sector Program and Erase Lock register provides a means to protect sectors from being programed and erased on the main interface. The last 256 KB of all blocks have sector lock protection capability. The rest of the block (if applicable) is protected with super sector protection capability. This register shows the status of the pelock hardware input for the block that is interlocked. For more information on hardware program and erase protection see Program hardware locking and Erase hardware locking .

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SPELOCK															
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SPELOCK															
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-0 SPELOCK	Sector Program and Erase Lock [31:0] SPELOCK reflects the status of the hardware program and erase protection input to the flash module for the block interlocked on the main interface. The SPELOCK register is latched on the PEADR write, is not writable, and is status only. SPELOCK will return back to all ones at the same time that PEADR is cleared. When not interlocked this register will read all ones. During high voltage events, the status is locked. The default value of the SPELOCK bits is program and erase protected. In the event that sectors are not present (due to configuration or total memory size), the SPELOCK bits default to locked, and will remain 1. 0 - The corresponding sector is not locked, and may be programed or erased. 1 - The corresponding sector is protected from program or erase.

21.7.1.9 Super Sector Program and Erase Hardware Lock (SSPELOCK)

Offset

Register	Offset
SSPELOCK	54h

Function

The Program and Erase Lock register provides a means to protect super sectors from being programed and erased on the main interface. Super Sector protection is available on block space greater than 256 KB. For 256 KB blocks, this register is not applicable. For 512 KB blocks, the first half of the block is protected with super sector granularity. For 1 MB blocks, the first 768 KB is protected with super sector granularity. For 2 MB blocks, the first 1792 KB is protected with super sector granularity. This register shows the status of the pelock hardware input for the block interlocked. For more information on hardware program and erase protection see Program hardware locking and Erase hardware locking .

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				SSPELOCK											
W																
Reset	See Register reset values.															

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SSPELOCK															
W																
Reset	See Register reset values.															

Register reset values

Register	Reset value
SSPELOCK	FLASH0: 0FFF_FFFFh FLASH1: 0000_0FFFh

Fields

Field	Function
31-28 —	Reserved Reserved; reset to 0
27-0 SSPELOCK	Super Sector Program and Erase Lock [27:0] SSPELOCK reflects the status of the hardware program and erase protection input to the flash module for the block interlocked on the main interface.

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>The SSPELOCK register is latched on the PEADR write, is not writable, and is status only. SSPELOCK will return back to all ones at the same time that PEADR is cleared. When not interlocked this register will read all ones. During high voltage events, the status is locked.

The default value of the SSPELOCK bits is program and erase protected. In the event that super sectors are not present (due to configuration or total memory size), or super sectors not present due to block size, the SSPELOCK bits default locked, and will remain 1.

0 - The corresponding super sector is not locked, and may be programmed or erased.

1 - The corresponding super sector is protected from program or erase.

NOTE
This field is not supported in every instance. The following table includes only supported registers.

<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>FLASH0</td><td>SSPELOCK</td><td>—</td></tr>
<tr><td>FLASH1</td><td>SSPELOCK[11–0]</td><td>SSPELOCK[27–12]</td></tr>
</table>
</td></tr>
</table>

21.7.1.10 Express Sector Program and Erase Hardware Lock (XSPELOCK)

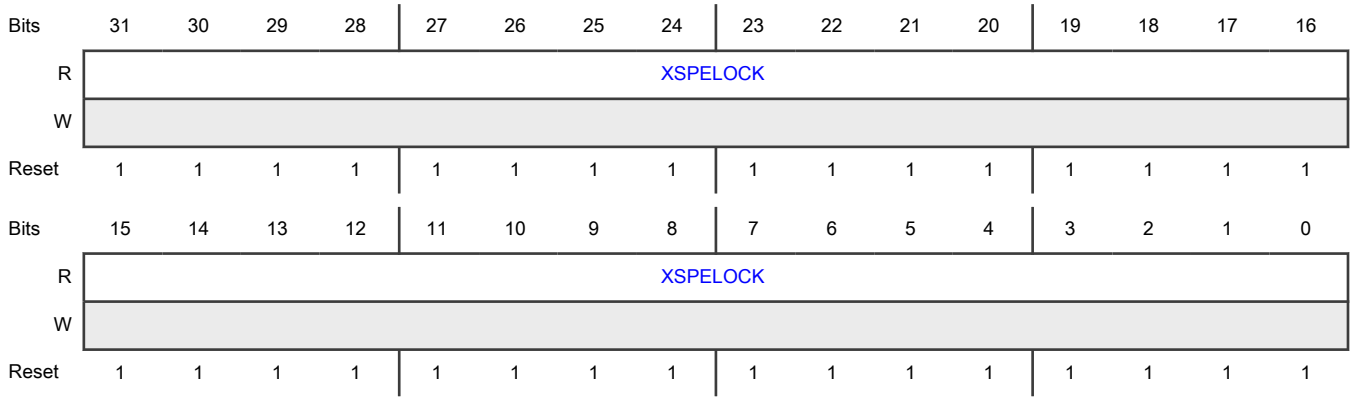
Offset

Register	Offset
XSPELOCK	70h

Function

The Express Program and Erase Lock register provides a means to protect sectors from being programed and erased on the express program interface. The last 256 KB of all blocks have sector lock protection capability. The rest of the block (if applicable) is protected with super sector protection capability. This register shows the status of the pelock hardware input for the block that is interlocked. For more information on hardware program and erase protection see Program hardware locking and Erase hardware locking .

Diagram



Fields

Field	Function
31-0 XSPELOCK	Express Sector Program and Erase Lock [31:0] XSPELOCK reflects the status of the hardware program and erase protection input to the flash module for the block interlocked on the express program interface. The XSPELOCK register is latched on the XPEADR write, is not writable, and is status only. XSPELOCK will return back to all ones at the same time that XPEADR is cleared. When not interlocked this register will read all ones. During high voltage events, the status is locked. The default value of the XSPELOCK bits is program and erase protected. In the event that sectors are not present (due to configuration or total memory size), the XSPELOCK bits default to locked, and will remain 1. 0 - The corresponding sector is not locked, and may be programmed or erased. 1 - The corresponding sector is protected from program or erase.

21.7.1.11 Express Super Sector Program and Erase Hardware Lock (XSSPELOCK)

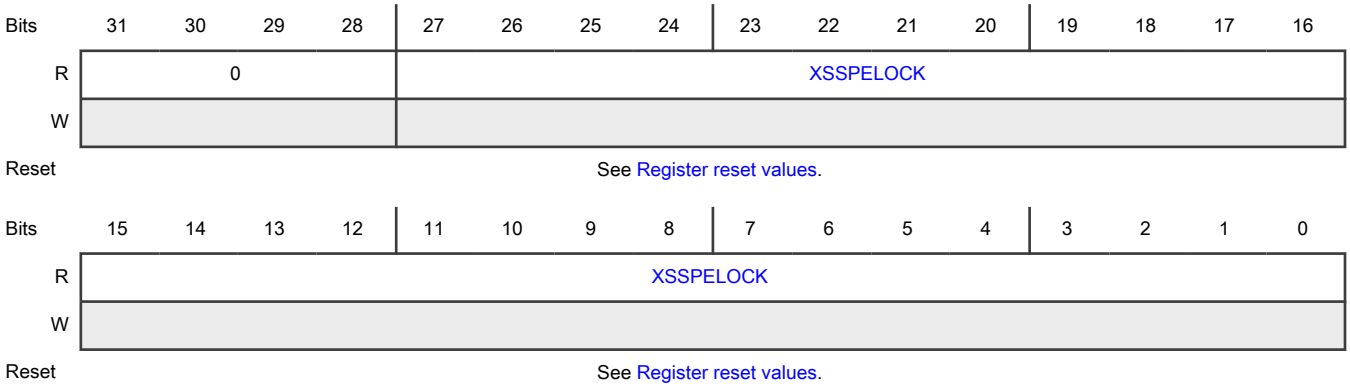
Offset

Register	Offset
XSSPELOCK	74h

Function

The Express Program and Erase Lock register provides a means to protect super sectors from being programed and erased on the express program interface. Super Sector protection is available on block space greater than 256 KB. For 256 KB blocks, this register is not applicable. For 512 KB blocks, the first half of the block is protected with super sector granularity. For 1 MB blocks, the first 768 KB is protected with super sector granularity. For 2 MB blocks, the first 1792 KB is protected with super sector granularity. This register shows the status of the pelock hardware input for the block interlocked. For more information on hardware program and erase protection see Program hardware locking and Erase hardware locking .

Diagram



Register reset values

Register	Reset value
XSSPELOCK	FLASH0: 0FFF_FFFFh FLASH1: 0000_0FFFh

Fields

Field	Function
31-28 —	Reserved Reserved; reset to 0
27-0 XSSPELOCK	Express Super Sector Program and Erase Lock [27:0] XSSPELOCK reflects the status of the hardware program and erase protection input to the flash module for the block interlocked on the express program interface. The XSSPELOCK register is latched on the XPEADR write, is not writable, and is status only. XSSPELOCK will return back to all ones at the same time that XPEADR is cleared. When not interlocked this register will read all ones. During high voltage events, the status is locked. The default value of the XSSPELOCK bits is program and erase protected. In the event that super sectors are not present (due to configuration or total memory size), or super sectors not present due to block size, the XSSPELOCK bits default to locked, and will remain 1. 0 - The corresponding super sector is not locked, and may be programmed or erased. 1 - The corresponding super sector is protected from program or erase. **NOTE** This field is not supported in every instance. The following table includes only supported registers.

Instance	Field supported in	Field not supported in
FLASH0	XSSPELOCK	—

Field	Function

Instance	Field supported in	Field not supported in
FLASH1	XSSPELOCK[11–0]	XSSPELOCK[27–12]

21.7.1.12 Test Mode Disable Password Check (TMD)

Offset

Register	Offset
TMD	90h

Function

Provides a means to supply a challenge password to disable the Test mode disable seal block select.

Writes to the register have no effect except for a password challenge. Reads to this register always return 0.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PWD															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PWD															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PWD	Password Challenge

21.7.1.13 UTest 0 (UT0)

Offset

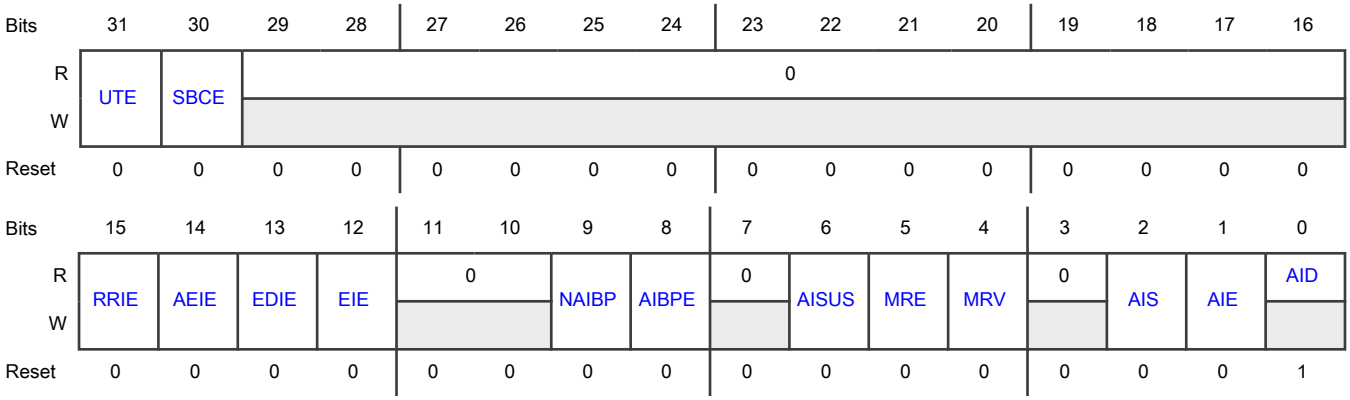
Register	Offset
UT0	94h

Function

Provides a means to control UTest.

UTest mode provides the ability to perform test features on the flash memory. This register is only writeable when the flash memory is put into UTest mode by writing a passcode.

Diagram



Fields

Field	Function
31 UTE	UTest Enable This status bit indicates when U-Test is enabled. All bits in UT0, UM0, UM1, UM2, UM3, UM4, UM5, UM6, UM7, UM8, UM9, UD0, UD1, UD2, UD3, UD4, UD5, UA0 and UA1 registers are locked when this bit is 0. This bit is not writeable to a 1, but may be cleared. The reset value is 0. The method to set this bit is to provide a password, and if the password matches, the UTE bit is set to reflect a status of enabled, and is enabled until it is cleared by a register write. The UTE password (set UTE=1) and UTE clearing to 0 will only be accepted if Program or Erase are not in progress (PEADR write through clearing of PGM/ERS/PES), Alternate Program or Erase are not in progress (APEADR write through clearing of APGM/AERS/APES), AI/MR Operation are not in progress (PEADR write through clearing of AIE), and an Express Program operation is not in progress (XPEADR write through the clearing of XPGM). UTE can only be cleared if AID = 1, MRE, AIE, EIE, EDIE, AEIE, and RRIE are all equal to 0. While successfully clearing UTE, writes to set AIE, MRE, EIE, EDIE, AEIE or RRIE will be ignored. For UTE, the password F9F9_9999h must be written to the UT0 register, and this must be a 32bit write. 0b - U-Test mode is not enabled. 1b - U-Test mode is enabled.
30	Single Bit Correction Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
SBCE	SBCE enables single bit correction results to be observed in SBC. ECC corrections that occur when SBCE is cleared will not be logged. 0b - Disabled 1b - Enabled
29-16 —	Reserved Reserved; reset to 0
15 RRIE	Read Reference Input Enable RRIE enables the force of an error on the read reference detection circuit. This is useful in the Read Reference Error check. If this bit is set, the read reference error register MCRS[RRE] will set when an address match is achieved to the ADR register. RRIE is not simultaneously writable to a 1 as UTE is being successfully cleared to a 0. 0b - Read reference input disabled 1b - Read reference input enabled
14 AEIE	Address Encode Invert Enable AEIE enables the input register AEI to invert the address encode value received from the array, and force a mismatch in the address encode comparison. This is useful in the Address Encode logic check. If this bit is set, address encode information from the memory array will be inverted based on values in the AEI register when an address match is achieved to the ADR register. AEIE is not simultaneously writable to a 1 as UTE is being successfully cleared to a 0. 0b - Address encode invert is disabled 1b - Address encode values are inverted based on UA0[AEI]
13 EDIE	EDC after ECC Data Input Enable EDIE enables the input registers EDDATA and EDDATAC to be the source of data to the EDC after ECC comparator or EDC after ECC encoder. This is useful in the EDC after ECC logic check. If this bit is set, data read via a BIU read request will be from the EDDATA and EDDATAC registers when an address match is achieved to the ADR register. EDIE is not simultaneously writable to a 1 as UTE is being successfully cleared to a 0. 0b - EDC after ECC data input is disabled 1b - Data read is from UD3[EDDATA] and UD5[EDDATAC]
12 EIE	ECC Data Input Enable EIE enables the input registers EDATA and EDATAC to be the source of data to the ECC logic (instead of from the memory array). This is useful in the ECC logic check. If this bit is set, data read via a BIU read request will be from the EDATA and EDATAC registers when an address match is achieved to the ADR register. EIE is not simultaneously writable to a 1 as UTE is being successfully cleared to a 0. 0b - ECC data input is disabled 1b - Data read is from UD0[EDATA] and UD2[EDATAC]

Table continues on the next page...

Table continued from the previous page...

Field	Function
11-10 —	Reserved
9 NAIBP	Next Array Integrity Break Point If AIBPE is set, NAIBP will be set once a single bit correction (if enabled) or double bit detection is noted during the Array Integrity test. NAIBP is not writable to 1, but may be cleared to 0. Clearing NAIBP to 0, will enable the Array Integrity operation to be re-started after a breakpoint is encountered. NAIPB may only be cleared to 0 if both EER = 0 and SBC = 0. If the Array Integrity operation completes without encountering another correction or detection, AID will be set with NAIBP remaining 0. 0b - Array integrity state machine is not currently at a breakpoint 1b - Array integrity state machine is at a breakpoint
8 AIBPE	Array Integrity Break Point Enable To enable breakpoints during an array integrity test, AIBPE may be set. See NAIBP description for more information about array integrity breakpoints. 0b - Array integrity breakpoints disabled 1b - Array integrity breakpoints enabled during array integrity checks
7 —	Reserved Reserved; reset to 0
6 AISUS	Array Integrity Suspend AISUS enables a suspend of an Array Integrity Operation. Array Integrity may be suspend by setting AISUS, and resumed by clearing AISUS. AISUS is writeable to a 1 only when AID is low. AISUS is clearable to a 0 only when AID is high. Attempting to write AISUS and AIE on the same clock cycle will result in only AIE getting written. 0b - Array integrity sequence not suspended. 1b - Array integrity sequence is suspended.
5 MRE	Margin Read Enable MRE combined with MRV enables user margin reads to be done. Normal user reads are not affected by MRE, although user reads while the margin read operation is ongoing are not supported. MRE is not writable if AID is low, or if AISUS is high, or if NAIBP is high. MRE is not simultaneously writable to a 1 as UTE is being successfully cleared to a 0. 0b - Margin reads are not enabled. 1b - Margin reads are enabled.
4 MRV	Margin Read Value MRV selects the margin level that is being checked. Margin can be checked to an erased level (MRV=1) or to a programmed level (MRV=0). In order for this value to be valid, MRE must also be set. MRV is not writable if AID is low, or if AISUS is high, or if NAIBP is high.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Zero's margin reads are requested. 1b - One's margin reads are requested.
3 —	Reserved
2 AIS	Array Integrity Sequence AIS determines the address sequence to be used during array integrity checks. The default sequence (AIS = 0) is meant to replicate sequences that normal user code follows, and thoroughly checks the read propagation paths. This sequence is proprietary. The alternative sequence (AIS = 1) is logically sequential. It should be noted that the time to run a sequential sequence is shorter than the time to run the proprietary sequence. AIS is not writeable if AIE is high. 0b - Array integrity sequence is proprietary sequence 1b - Array integrity sequence is sequential
1 AIE	Array Integrity Enable AIE set to one starts the array integrity check done on all selected blocks. The address sequence is determined by AIS, and the MISR (UM0 through UM9) can be checked after the operation is complete, to determine if a correct signature has been obtained. Once an array integrity operation is requested (AIE=1), it may be terminated by clearing AIE if the operation has finished (AID = 1) or aborted by clearing AIE if the operation is ongoing (AID = 0). AIE is locked from writing unless an interlock write occurred (PEADR write and DATA write). AIE is not simultaneously writable to a 1 as UTE is being successfully cleared to a 0. 0b - Array integrity checks not enabled 1b - Array integrity checks enabled
0 AID	Array Integrity Done AID is cleared upon an array integrity check being enabled (to signify the operation is ongoing). Once completed, AID is set to indicate that the array integrity check is complete. At this time the MISR (UMR registers) can be checked. AID may also assert if breakpoints are enabled (AIBPE is set), an abort is requested or a suspend is requested. AID cannot be written, and is status only. 0b - Array integrity check ongoing 1b - Array integrity check complete

21.7.1.14 UMISRn (UM0 - UM8)

Offset

For a = 0 to 8:

Register	Offset
UMa	98h + (a × 4h)

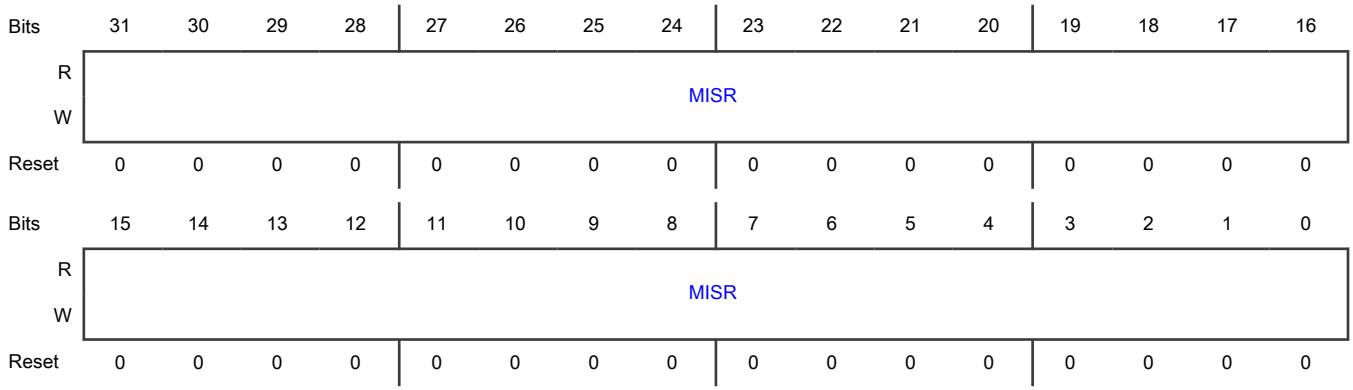
Function

Provides a means to evaluate array integrity. UM9 to UM0 are a set of multiple-input signature registers (MISRs) that hold the 289-bit MISR value as shown in the next table.

Table 110. MISR mapping

UM register	MISR bits
UM0	MISR[31:0]
UM1	MISR[63:32]
UM2	MISR[95:64]
UM3	MISR[127:96]
UM4	MISR[159:128]
UM5	MISR[191:160]
UM6	MISR[223:192]
UM7	MISR[255:224]
UM8	MISR[287:256]
UM9	MISR[288]

Diagram



Fields

Field	Function
31-0 MISR	MISR[31:0] The MISR registers accumulate a signature from an array integrity event. The MISR captures all data fields, as well as ECC fields, and the ECC error signal. Data (read and ECC) captured in the MISR is after the ECC logic, and represents corrected data (if required). The MISR can be seeded to any value by writing the MISR registers.

Table continues on the next page...

Field	Function
	NOTE Writing the MISR register during an array integrity operation (including suspend or breakpoint) although possible, is not recommended. A write of the MISR registers at this point may alter the final signature in an unpredictable way. Writing the MISR registers prior to an Array Integrity operation (to seed the MISR) is allowed. The MISR register provides a means to calculate a MISR during array integrity operations. The MISR can be represented by the following polynomial: $x^{289} + x^{7} + x^{6} + x^{5} + x^{4} + x^{2} + 1$ The MISR is calculated by taking the previous MISR value and then "exclusive ORing" the new data. In addition the most significant bit (in this case it is MISR[288]), is then "exclusive ORed" into input of MISR[7], MISR[6], MISR[5], MISR[4], MISR[2] and MISR[0]. The result of the "exclusive OR" is shifted left on each read. The MISR register is used in array integrity operations. If during address sequencing, reads extend into an invalid address location (in other words, greater than the maximum address for a given array size) then reads are still executed to the array, but the results from the array read are not deterministic. In this instance, the MISR registers are not recalculated, and the previous value is retained. Once the AIE register is cleared, the MISR register will return to an all 0's state.

21.7.1.15 UMISR9 (UM9)

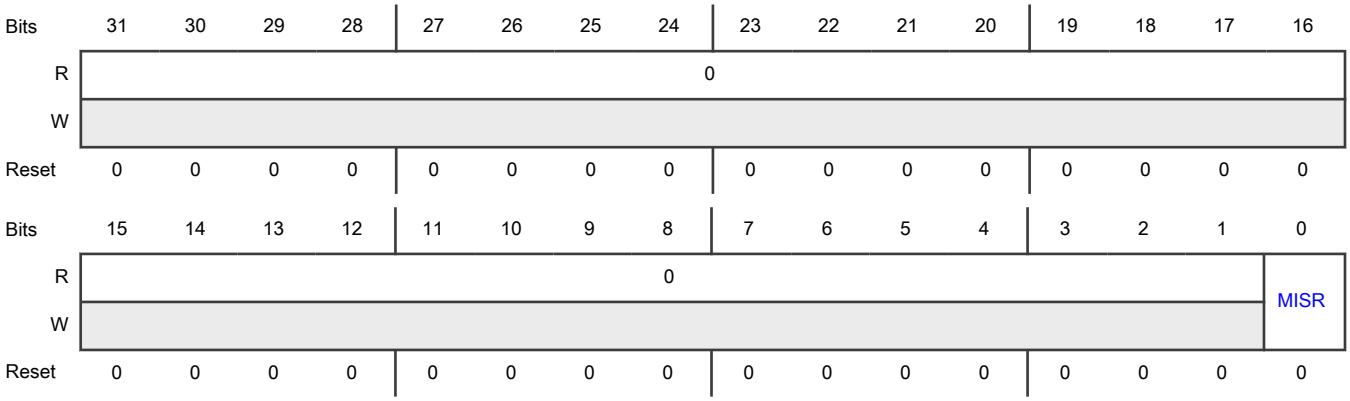
Offset

Register	Offset
UM9	BCh

Function

Provides a means to evaluate array integrity.

Diagram



Fields

Field	Function
31-1 —	Reserved Reserved; reset to 0
0 MISR	MISR[288] The MISR registers accumulate a signature from an array integrity event. The MISR captures all data fields, as well as ECC fields, and the ECC error signal. Data (read and ECC) captured in the MISR is after the ECC logic, and represents corrected data (if required). The MISR can be seeded to any value by writing the MISR registers. **NOTE** Writing the MISR register during an array integrity operation (including suspend or breakpoint) although possible, is not recommended. A write of the MISR registers at this point may alter the final signature in an unpredictable way. Writing the MISR registers prior to an Array Integrity operation (to seed the MISR) is allowed. The MISR register provides a means to calculate a MISR during array integrity operations. The MISR can be represented by the following polynomial: $x^{289} + x^{7} + x^{6} + x^{5} + x^{4} + x^{2} + 1$ The MISR is calculated by taking the previous MISR value and then "exclusive ORing" the new data. In addition the most significant bit (in this case it is MISR[288]), is then "exclusive ORed" into input of MISR[7], MISR[6], MISR[5], MISR[4], MISR[2] and MISR[0]. The result of the "exclusive OR" is shifted left on each read. The MISR register is used in array integrity operations. If during address sequencing, reads extend into an invalid address location (in other words, greater than the maximum address for a given array size) then reads are still executed to the array, but the results from the array read are not deterministic. In this instance, the MISR registers are not recalculated, and the previous value is retained. Once the AIE register is cleared, the MISR register will return to an all 0's state.

21.7.1.16 UTest Data 0 (UD0)

Offset

Register	Offset
UD0	D0h

Function

Provides a means to enable data to be substituted during UTest procedures.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	EDATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	EDATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 EDATA	ECC Data [31:0] Enables checks of ECC logic by allowing data bits to be input into the ECC logic and then read out by doing array reads or array integrity checks. EDATA corresponds to the 32 array bits representing word 0 of all double words in the page selected in ADR.

21.7.1.17 UTest Data 1 (UD1)

Offset

Register	Offset
UD1	D4h

Function

Provides a means to enable data to be substituted during UTest procedures.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	EDATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	EDATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 EDATA	ECC Data [63:32] Enables checks of ECC logic by allowing data bits to be input into the ECC logic and then read out by doing array reads or array integrity checks. UD0[EDATA] corresponds to the 32 array bits representing word 1 of all double words in the page selected in ADR.

21.7.1.18 UTest Data 2 (UD2)

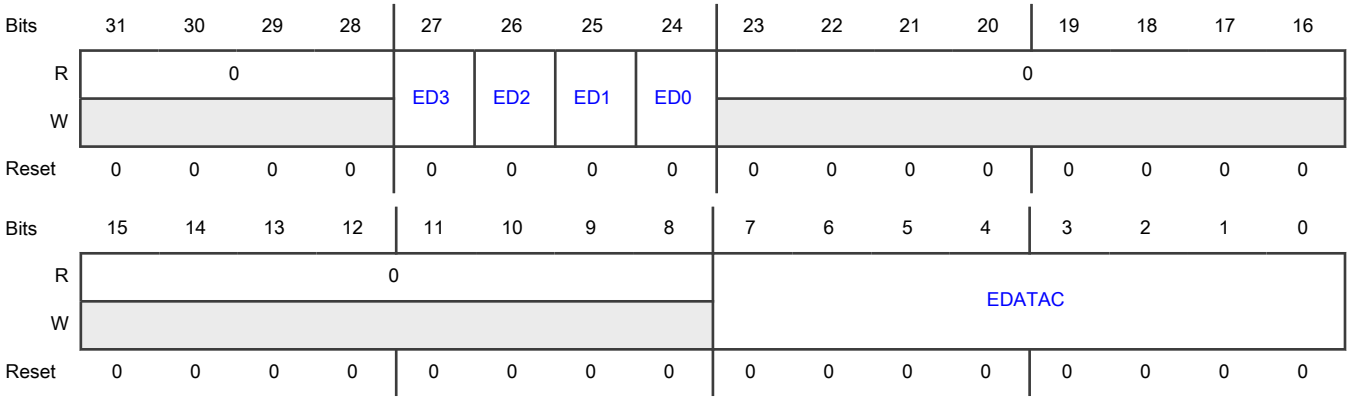
Offset

Register	Offset
UD2	D8h

Function

Provides a means to enable data to be substituted during UTest procedures.

Diagram



Fields

Field	Function
31-28 —	Reserved
27 ED3	ECC Logic Check Double Word 3 Enables checks of ECC logic by allowing data bits to be input into the ECC logic and then read out by doing array reads or array integrity checks. If this field's value becomes 1, the data replacement occurs on DW3 on the page selected in ADR.
26	ECC Logic Check Double Word 2

Table continues on the next page...

Table continued from the previous page...

Field	Function
ED2	Enables checks of ECC logic by allowing data bits to be input into the ECC logic and then read out by doing array reads or array integrity checks. If this field's value becomes 1, the data replacement occurs on DW2 on the page selected in ADR.
25 ED1	ECC Logic Check Double Word 1 Enables checks of ECC logic by allowing data bits to be input into the ECC logic and then read out by doing array reads or array integrity checks. If this field's value becomes 1, the data replacement occurs on DW1 on the page selected in ADR.
24 ED0	ECC Logic Check Double Word 0 Enables checks of ECC logic by allowing data bits to be input into the ECC logic and then read out by doing array reads or array integrity checks. If this field's value becomes 1, the data replacement occurs on DW0 on the page selected in ADR.
23-8 —	Reserved
7-0 EDATAC	ECC Data Check Bits [7:0] Enables checks of ECC logic by allowing data bits to be input into the ECC logic and then read out by doing array reads or array integrity checks. This field corresponds to the eight array bits representing the check bits of all double words in the page selected in ADR.

21.7.1.19 UTest Data 3 (UD3)

Offset

Register	Offset
UD3	DCh

Function

Provides a means to enable data to be substituted during UTest procedures.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	EDDATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	EDDATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 EDDATA	EDC After ECC Data [31:0] Enables checks of EDC after ECC logic by allowing data bits to be input into the EDC after ECC logic and then read out by doing array reads or array integrity checks. This field corresponds to the 32 array bits representing word 0 of all double words in the page selected in ADR.

21.7.1.20 UTest Data 4 (UD4)

Offset

Register	Offset
UD4	E0h

Function

Provides a means to enable data to be substituted during UTest procedures.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	EDDATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	EDDATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 EDDATA	EDC After ECC Data [63:31] Enables checks of EDC after ECC logic by allowing data bits to be input into the EDC after ECC logic and then read out by doing array reads or array integrity checks. This field corresponds to the 32 array bits representing word 1 of all double words in the page selected in ADR.

21.7.1.21 UTest Data 5 (UD5)

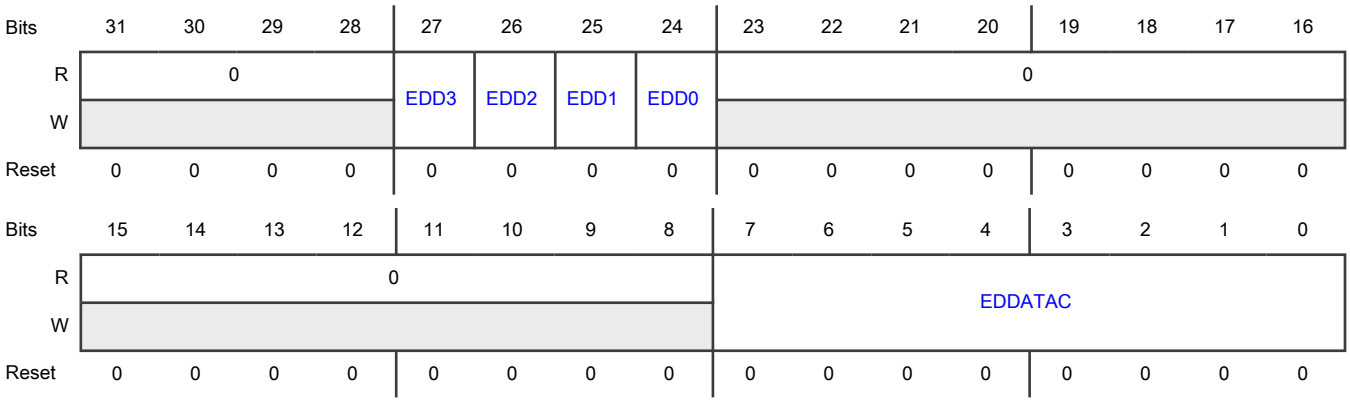
Offset

Register	Offset
UD5	E4h

Function

Provides a means to enable data to be substituted during UTest procedures.

Diagram



Fields

Field	Function
31-28 —	Reserved
27 EDD3	EDC After ECC Logic Check Double Word 3 Enables checks of EDC after ECC logic by allowing data bits to be input into the EDC after ECC logic and then read out by doing array reads or array integrity checks. If this field's value becomes 1, the data replacement occurs on DW3 on the page selected in ADR.

Table continues on the next page...

Table continued from the previous page...

Field	Function
26 EDD2	EDC after ECC Logic Check Double Word 2 Enables checks of EDC after ECC logic by allowing data bits to be input into the EDC after ECC logic and then read out by doing array reads or array integrity checks. If this field's value becomes 1, the data replacement occurs on DW2 on the page selected in ADR.
25 EDD1	EDC After ECC Logic Check Double Word 1 Enables checks of EDC after ECC logic by allowing data bits to be input into the EDC after ECC logic and then read out by doing array reads or array integrity checks. If this field's value becomes 1, the data replacement occurs on DW1 on the page selected in ADR.
24 EDD0	EDC After ECC Logic Check Double Word 0 Enables checks of EDC after ECC logic by allowing data bits to be input into the EDC after ECC logic and then read out by doing array reads or array integrity checks. If this field's value becomes 1, the data replacement occurs on DW0 on the page selected in ADR.
23-8 —	Reserved
7-0 EDDATAC	EDC After ECC Data Check Bits [7:0] Enables checks of EDC after ECC logic by allowing data bits to be input into the EDC after ECC logic and then read out by doing array reads or array integrity checks. This field corresponds to the eight array bits representing the check bits of all double words in the page selected in ADR.

21.7.1.22 UTest Address 0 (UA0)

Offset

Register	Offset
UA0	E8h

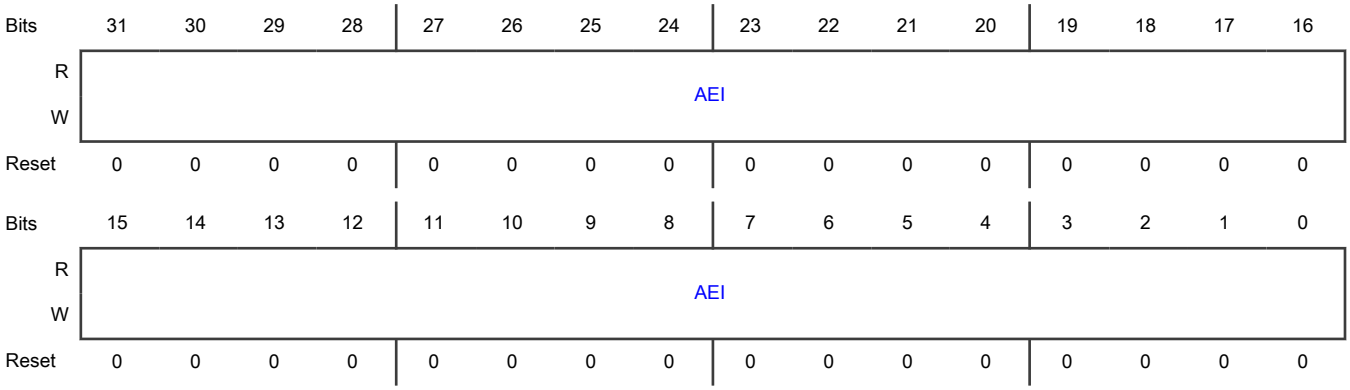
Function

Provides a means to enable address to be inverted during UTest procedures.

NOTE

- UTest Address 0 (UA0) and UTest Address 1 (UA1) combine to represent a 52-bit wide field, which matches the width of the address internally to the flash memory that is used for the address encode comparison.
- Writing 1 to any one of the fields in UA0 or UA1 leads to flagging an AEE error when causing the error injection.
- When the value of a field within the range of bits 18:3 is 1, the corresponding address signal to the flash memory controller or BIU is inverted.

Diagram



Fields

Field	Function
31-0 AEI	Address Encode Invert [31:0] Enables checks of address encode logic by allowing address bits to be inverted into the address encode compare logic, forcing a miscompare. Performing array reads on the page selected in ADR allows the address encode invert(s) to be active. Additionally, in the event of an address encode error during normal user mode operation, the page address provided to the PFC is inverted when an address encode error is encountered. AEI[18:3], which reflects the page address, can be used with the Address Encode Logic Check to control this inversion feature (aligning with ADR). Any bit in AEI[18:3] written to a 1 also enables the inversion of the corresponding signal of that address, allowing checks to be performed on the comparison logic that exists in the PFC. **NOTE** Multiple bits in AEI[18:3] may be written to a 1, which causes these bits to miscompare on the address encode comparison logic, as well as causing multiple bits to have their inversion enabled as part of the address provided to the BIU.

21.7.1.23 UTest Address 1 (UA1)

Offset

Register	Offset
UA1	ECh

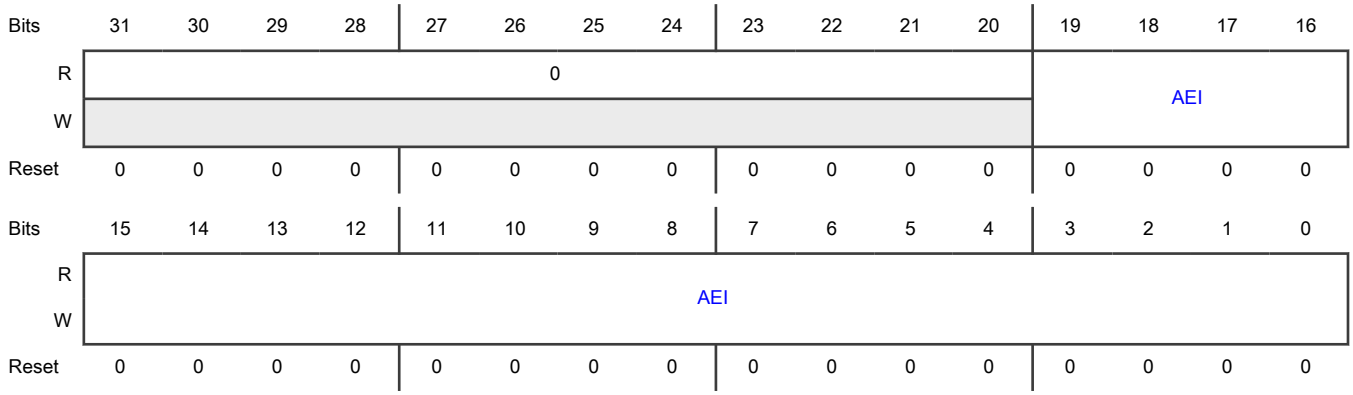
Function

Provides a means to enable address to be inverted during UTest procedures.

NOTE

- UTest Address 0 (UA0) and UTest Address 1 (UA1) combine to represent a 52-bit wide field, which matches the width of the address internally to the flash that is used for the address encode comparison.
- Writing 1 to any one of the fields in UA0 or UA1 leads to flagging an AEE error when causing the error injection.

Diagram



Fields

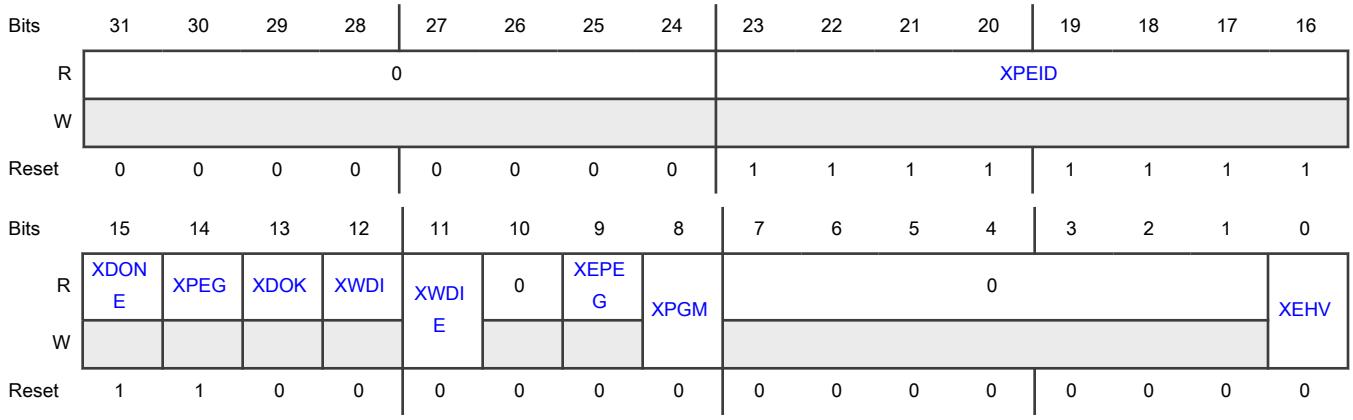
Field	Function
31-20 —	Reserved
19-0 AEI	Address Encode Invert [51:32] Enables checks of address encode logic by allowing address bits to be inverted into the address encode compare logic, forcing a miscompare. Performing array reads on the page selected in ADR allows the address encode invert(s) to be active.

21.7.1.24 Express Module Configuration (XMCR)

Offset

Register	Offset
XMCR	F0h

Diagram



Fields

Field	Function
31-24 —	Reserved
23-16 XPEID	Express Program Master/Domain ID This field shows the ID of the master that has started the express program sequence. The ID is latched when the sequence has started (writing of the XPEADR register). It is set back to the reset value (set to all '1') when the operation is completed. See the chip-specific information for a list of Master IDs.
15 XDONE	Express State Machine Status XDONE indicates whether the embedded flash memory is performing an express program operation. XDONE is set to a 1 on completion of the express program operation. 0b - Executing an express program operation 1b - Not executing an express program operation
14 XPEG	Express Program Good The XPEG field indicates the completion status of the last flash memory express program operation. The value of XPEG is updated automatically after the program operations. XPEG will be 0 if the program operation failed. See MCRS[PEG] for more information, as the features of that bit apply to XPEG as well. 0b - Program operation failed 1b - Program operation successful
13 XDOK	Express Data OK This field is a status signal from the flash to indicate that it is OK to load Data into the DATA registers. It will assert in response to a XPEADR load (once ongoing operations are aborted, if required). XDOK will clear once the Express Program operation is complete (XPGM written to 0). 0b - Flash memory not ready to accept writes to the DATA registers 1b - Writes to DATA registers allowed
12 XWDI	Express Watch Dog Interrupt XWDI is a status register to indicate that the Watchdog Timer for Express Program had expired. XWDI is status only, and will be automatically cleared once the operation that caused the watchdog timeout is terminated or clean up from the operation is completed (clearing of XPGM bit). 0b - Normal Operation, Watchdog Timer has not expired. 1b - Express Program Watchdog Timer has expired.
11 XWDIE	Express Watch Dog Interrupt Enable XWDIE provides a mechanism to trigger an interrupt request upon the assertion of the XWDI bit on the express interface. If XWDIE is asserted, when XWDI asserts an interrupt from the flash will trigger to the system. The interrupt from the flash will mirror exactly the XWDI bit. XWDIE does not affect the register bit XWDI. Writing (and reading) of this bit will be restricted to the master that updated the XPEADR register as captured in the XPEID register at the start of the Express Program sequence.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Express watchdog interrupt disabled 1b - Express watchdog interrupt enabled
10 —	Reserved
9 XEPEG	Express ECC Enabled Program Good XEPEG is a qualifier to XPEG to indicate if a passing program required ECC Enabled verifies to pass. In the event of a failed operation (XPEG=0), XEPEG will always remain 0. The value of XEPEG is updated automatically after the program operations. See MCRS[EPEG] for more information, as the features of that bit apply to XEPEG as well. 0b - Program operation did not require ECC-enabled verifies 1b - Program operation required ECC-enabled verifies to pass
8 XPGM	Express Program XPGM is used as part of the setup for an express program operation. A 0 to 1 transition of XPGM after program interlock write(s) is part of the setup for an express program sequence. A 1 to 0 transition of XPGM ends the express program sequence. XPGM can be set at any time if UT0[UTE] is low, and after the interlock write is completed (XPEADR written, XDOK asserted, and DATAn register(s) written). XPGM may be cleared when XEHV is low. Writing (and reading) of this bit will be restricted to the master that updated the XPEADR register as captured in the XPEID register at the start of the Program sequence. 0b - Flash memory not executing an express program sequence 1b - Flash memory executing an express program sequence
7-1 —	Reserved
0 XEHV	Express Enable High Voltage The XEHV field enables the embedded flash memory for a high voltage program/erase operation. XEHV is cleared on reset. XEHV must be set after DATA writes to start an express program operation and XPGM is set. This field may be set when XPGM = 1, XDOK = 1, and XWDI = 0. XEHV may not be written to a 1 after it is cleared as part of an express program cleanup, until after the next XPEADR write. Express Program operations may not be aborted. XEHV may only be cleared once XDONE is set to 1. Writing (and reading) of this bit will be restricted to the master that updated the XPEADR register as captured in the XPEID register at the start of the Program sequence. 0b - Flash memory is not enabled to perform an express high voltage operation. 1b - Flash memory is enabled to perform an express high voltage operation.

21.7.1.25 Express Program Address (XPEADR)

Offset

Register	Offset
XPEADR	F4h

Function

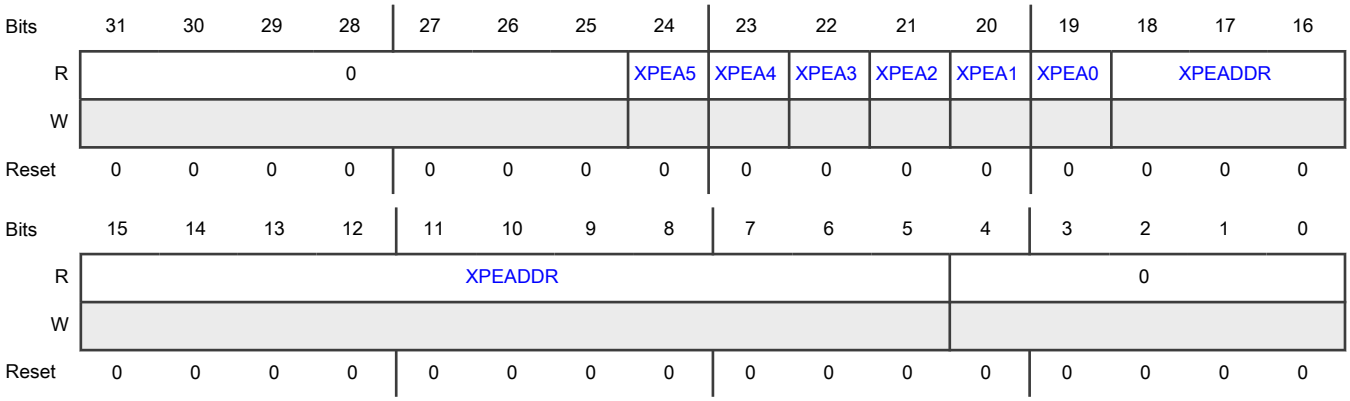
The Express Program Address register (XPEADR) is used to provide the address to be programmed. The express program address register is read only in user mode (user mode writes occur with writes done to the Platform Flash Controller). The address is identified using a combination of bits that identify the memory region and offset address.

NOTE

- The address given is an offset from the base address of the address space.
- The block numbering scheme that corresponds to the offset address is based on the flash memory's *internal* addressing scheme, which may be different than the chip's system addressing scheme.

Understanding the mapping between the flash address, which is specified relative to the flash module's internal addressing scheme, and the location within the chip's system memory map requires a clear understanding of the flash memory layout. See the chip-specific information for a detailed explanation of the chip's flash memory layout and how to map system addresses to the flash module's internal addressing.

Diagram



Fields

Field	Function
31-25 —	Reserved Reserved; reset to 0
24 XPEA5	Express Program Address Region 5 Qualifies the address field (XPEADDR) to the region. If XPEA5 = 1, the XPEADDR field maps to that region. See the description of the XPEADDR field for more information. If this region is not present, the field is locked to 0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Address accessed is not from region 5 1b - Address accessed is from region 5
23 XPEA4	Express Program Address Region 4 Qualifies the address field (XPEADDR) to the region. If XPEA4 = 1, the XPEADDR field maps to that region. See the description of the XPEADDR field for more information. If this region is not present, the field is locked to 0. 0b - Address accessed is not from region 4 1b - Address accessed is from region 4
22 XPEA3	Express Program Address Region 3 Qualifies the address field (XPEADDR) to the region. If XPEA3 = 1, the XPEADDR field maps to that region. See the description of the XPEADDR field for more information. If this region is not present, the field is locked to 0. 0b - Address accessed is not from region 3 1b - Address accessed is from region 3
21 XPEA2	Express Program Address Region 2 Qualifies the address field (XPEADDR) to the region. If XPEA2 = 1, the XPEADDR field maps to that region. See the description of the XPEADDR field for more information. If this region is not present, the field is locked to 0. 0b - Address accessed is not from region 2 1b - Address accessed is from region 2
20 XPEA1	Express Program Address Region 1 Qualifies the address field (XPEADDR) to the region. If XPEA1 = 1, the XPEADDR field maps to that region. See the description of the XPEADDR field for more information. If this region is not present, the field is locked to 0. 0b - Address accessed is not from region 1 1b - Address accessed is from region 1
19 XPEA0	Express Program Address Region 0 Qualifies the address field (XPEADDR) to the region. If XPEA0 = 1, the PEADDR field maps to that region. See the description of the XPEADDR field for more information. If this region is not present, the field is locked to 0. 0b - Address accessed is not from region 0 1b - Address accessed is from region 0
18-5	Express Program Address

Table continues on the next page...

Table continued from the previous page...

Field	Function
XPEADDR	The XPEADDR register provides offset address location to be programmed. This address is always a quad page address that selects 1024 bits. The XPEADDR register is read only in user mode and represents the flash physical address status. XPEADDR may be updated once (and only once) per Express Program event. XPEADDR writes initiate an Express Program request. XPEADDR may be updated four times between reset events. Upon completing the fourth Express Program request (clearing of XPGM register), the XPEADDR register will return to all zeros, and remain locked from future XPEADDR updates and thus future Express Program requests will be ignored until the next reset. Once XPEADDR is updated, it will be locked for the full express program operation, until XMCR[XPGM] is cleared at the end of the operation. Writing of XPEADDR has the effect of blocking other registers used for program and erase to be written (PEADR/MCR registers as well as APEADR/APDATA/AMCR registers) until the express program operation is completed. Upon completion of an Express Program operation (XMCR[XPGM] written to 0) the XPEADR registers will return to an all zeros state. When UT0[UTE] is set or XMCR[XEHV] is set, XPEADDR bits will not be writeable. Once XPEADDR is updated, it will have read restrictions based on the PEID master that did the original write in the Platform Flash Controller. XPEADDR[18:5] is an offset from a base address of 0h for each block region. The XPEA0, XPEA1, XPEA2, XPEA3, XPEA4 and XPEA5 qualify the block size region the XPEADDR field.
4-0 —	Reserved Reserved; reset to 0

21.7.1.26 Program Data (DATA0 - DATA31)

Offset

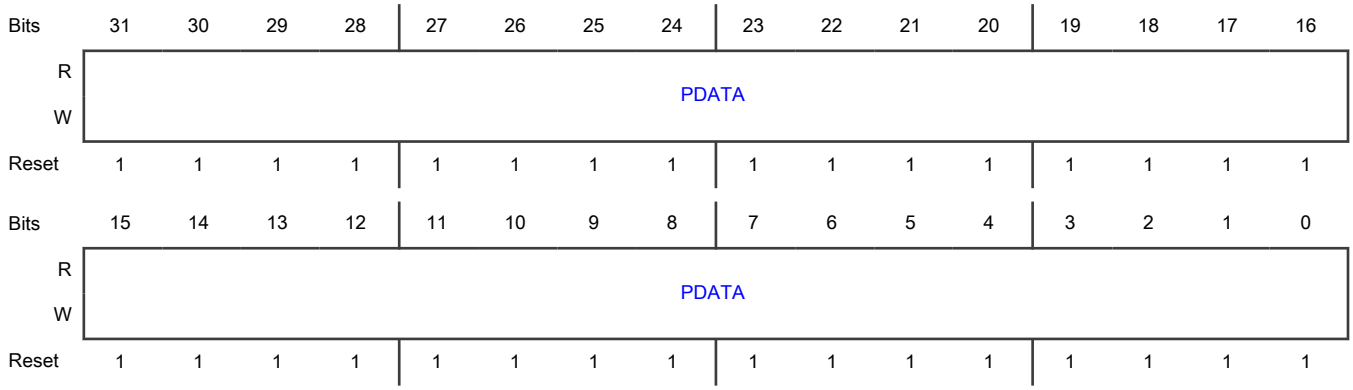
For a = 0 to 31:

Register	Offset
DATAa	100h + (a × 4h)

Function

Enables data to be provided for program (or express program). This set of 32 data registers enables 1024 bits of data to be programmed.

Diagram



Fields

Field	Function
31-0 PDATA	Program Data PDATA registers will reset to all ones. PDATA is not writable before PEADR is written or updated to start a program, express program, erase or UTest operation, and will read a value of all ones. Once PEADR is written, all PDATA registers will be made available for writing. Writing (and reading) of the PDATA registers will be restricted to the master that updated the PEADR register as captured in the PEID register. **NOTE** PDATA restrictions related to PEID will be disabled if UT0[UTE] is set. **NOTE** For Express Program a similar behavior exists on the PDATA registers. Once XPEADR is written, and XDOK comes back as a 1, all PDATA registers will switch to all 1's (program safe state) and be made available for writing (and reading) to the master's ID that matches XPEID captured during the XPEADR update. In the event of an interrupted operation, XPEID permissions will override PEID permissions until the express program operation is completed (XPGM written to a 0). Upon completion of an operation (Program (MCR[PGM] written to 0 and MCRS[PES] equals 0), Erase (MCR[ERS] written to 0 and MCRS[PES] equals 0), Express Program (XMCR[XPGM] written to 0), Array Integrity (UT0[AIE] written to 0), User Margin Read(UT0[AIE] written to 0)) or Program and Erase Sequence clear (MCRS[PES] written to 0) the PDATA registers will return to an all ones state. When UT0[AIE] is set, PDATA bits will not be writeable. When MCRS[PES] is set, PDATA bits will not be writeable except when an express program request occurs and XMCR[XDOK] is set. Also, when MCR[PGM] or MCR[ERS] are set, PDATA bits will not be writeable except when an express program request occurs and XMCR[XDOK] is set. When an express program operation is in process, PDATA bits will not be writeable once XMCR[XEHV] is set until XMCR[XPGM] is cleared. Only PDATA Bits that are written will receive program or express program pulses.

21.8 Glossary

Abort	Premature end of a mode, sequence, state or operation. An abort may leave the embedded flash memory in a state in which it contains indeterminate or corrupted data.
BIU	Bus interface unit(Contains all system-level customization required to make the embedded flash memory part of an SoC)
Block	Subdivision of the Flash Module containing NVM memory bits. Block sizes range from 256KB to 2MB. Each block is made up of 8KB sectors.
Double word	Two words, or 64 bits.
ECC	Error correction code(Internally used to correct single bit errors, or detect double errors within a 64-bit double word.)
ECC segment	64 bit double word, representing the size of data used to calculate ECC information.
EDC	Error detection code
FC	Flash memory core
FMU	Flash Management Unit - Logic that enables program, erase, read, and other flash functions.
Interlock Write	A write to the PEADR register followed by a write to any DATA register performed while initiating a program, erase, array integrity, or user margin read sequence. Interlock in cleared once the operation is finished by clearing the appropriate control bit, causing PEADR register to clear.
MCU	Microcontroller unit
Module	Flash memory instance that contains flash block(s), FMU, and required analog support structures.
NVM	Nonvolatile memory
OTP	One time programmable
PFC	Platform flash memory controller(Contains all system-level customization required to make the embedded flash memory part of an SoC)
Page	8 words of data (256 bits). When doing reads to the embedded flash memory, this is the width of data read. Up to 4 pages can be programmed at a time.
Partition	Subdivision of the Flash Module. Used for read-while-write operation, where addresses in the partition being written may not be read. Partitions is always equal to 1 block, and contains multiple sectors.
RWW	Read-while-write (operation)
Sector	Subdivision of the Flash Block that is independently erasable. Sector Size is always 8 KB.
Super sector	Subdivision of the flash block that includes a group of sectors. Super Sector Size is always 64 KB, and consists of 8 sectors.
SoC	System-on-chip
UTest NVM sector	8 KB sector outside the main address space
Word	32 bits

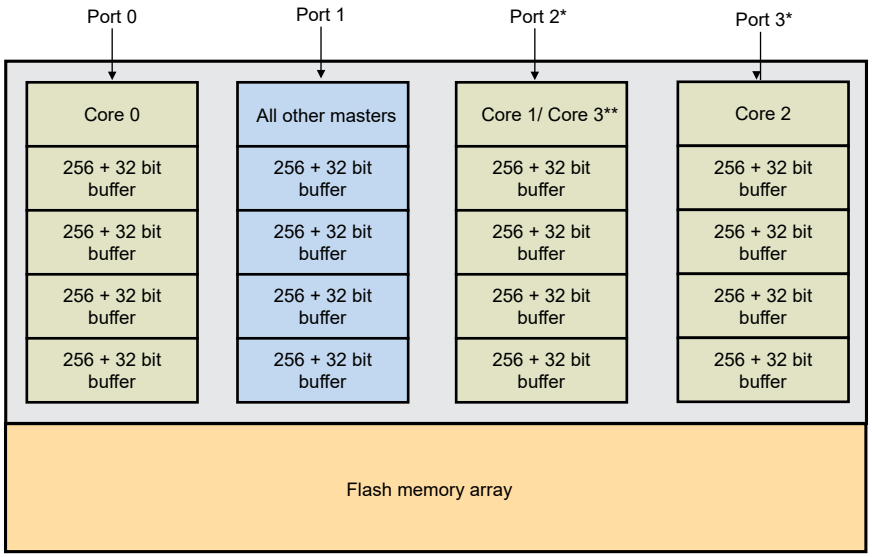
Chapter 22
Flash Memory Controller (PFLASH)

22.1 Chip-specific PFLASH information

22.1.1 Flash memory architecture

The flash memory on the chip consists of a flash memory controller and a flash memory array module. The flash memory controller provides flash–memory configuration and control functions and manages the interface between the flash memory array and the chip's crossbar switch.

This chip implements upto four 64-bit AHB buses.



* This port does not exist on S32K311 and S32K312
** This core is only available in S32K388

Figure 52. Flash memory architecture

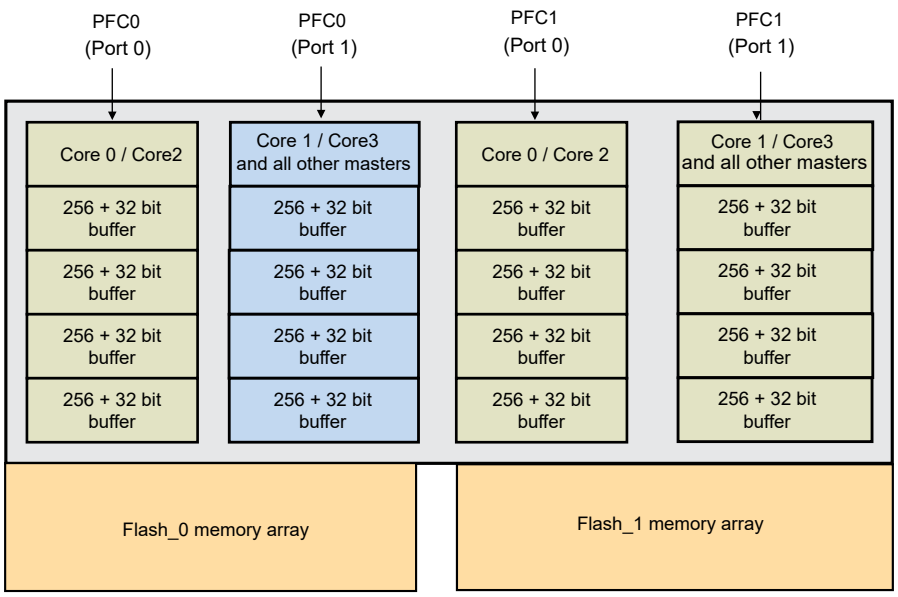


Figure 53. Flash memory architecture (S32K389)

22.1.2 Flash memory controller

The flash memory controller:

- Acts as an interface between the system bus and flash memory array.
- Is a triple-ported controller with a dedicated line buffering per port and per master ID. This enables you to use the line buffers more efficiently because various masters have dedicated buffers that are not compromised when other masters perform read operations.

Also, by having separate ports, you can have separate connections for each CPU instruction bus and a single port for all data accesses as shown in Figure 52.

In general, the flash memory controller registers affect the global flash memory behavior (for example, read buffering and access control).

NOTE

In S32K311, S32K312, and S32K342:

- PFLASH block 3 and block 4 are not present, so both the sector and super sector registers are not available.
- For PFLASH block 2, the super sector registers are not available.

S32K389 supports the following flash operations:

- Single read, per controller
- Dual read, per controller
- Parallel single read, two controllers
- Parallel dual read, two controllers

NOTE

Only parallel dual-read is supported. Quad-read in a single controller, parallel quad-read is no supported.

22.1.2.1 Dual Flash memory controller architecture (for S32K389)

As shown in Figure 54, the device implements two flash controllers with two separate ports.

- PFC0: 8MB C-Flash (4x blocks of 2MB) + 256KB D-Flash
- PFC1: 4MB C-Flash (4x blocks of 1MB)

NOTE

See the “Flash memory read timing parameters” specification in the product data-sheet for the required RWSC (read wait-state control) settings according to the Flash frequency of operation (for each flash).

Natively, both flash controllers map their access requests to addresses starting from “0x40_0000” (see “PHYSICAL” purple box in the left side of the Figure 54). However, the device implements a FAR “Flash Address Remapping” logic for translating the addresses from a continuous memory map (see “USER” green box in the right side of the Figure 54) to the corresponding physical ones.

In other words, the overall system memory map remains continuous, from the user perspective, and each time a system master requests certain address, it is internally remapped to the one that the controller can natively understand.

NOTE

See, table “Flash block configuration (S32K389)” in "Embedded Flash Memory (c40asf)" chapter for more details about the physical address range of each flash.

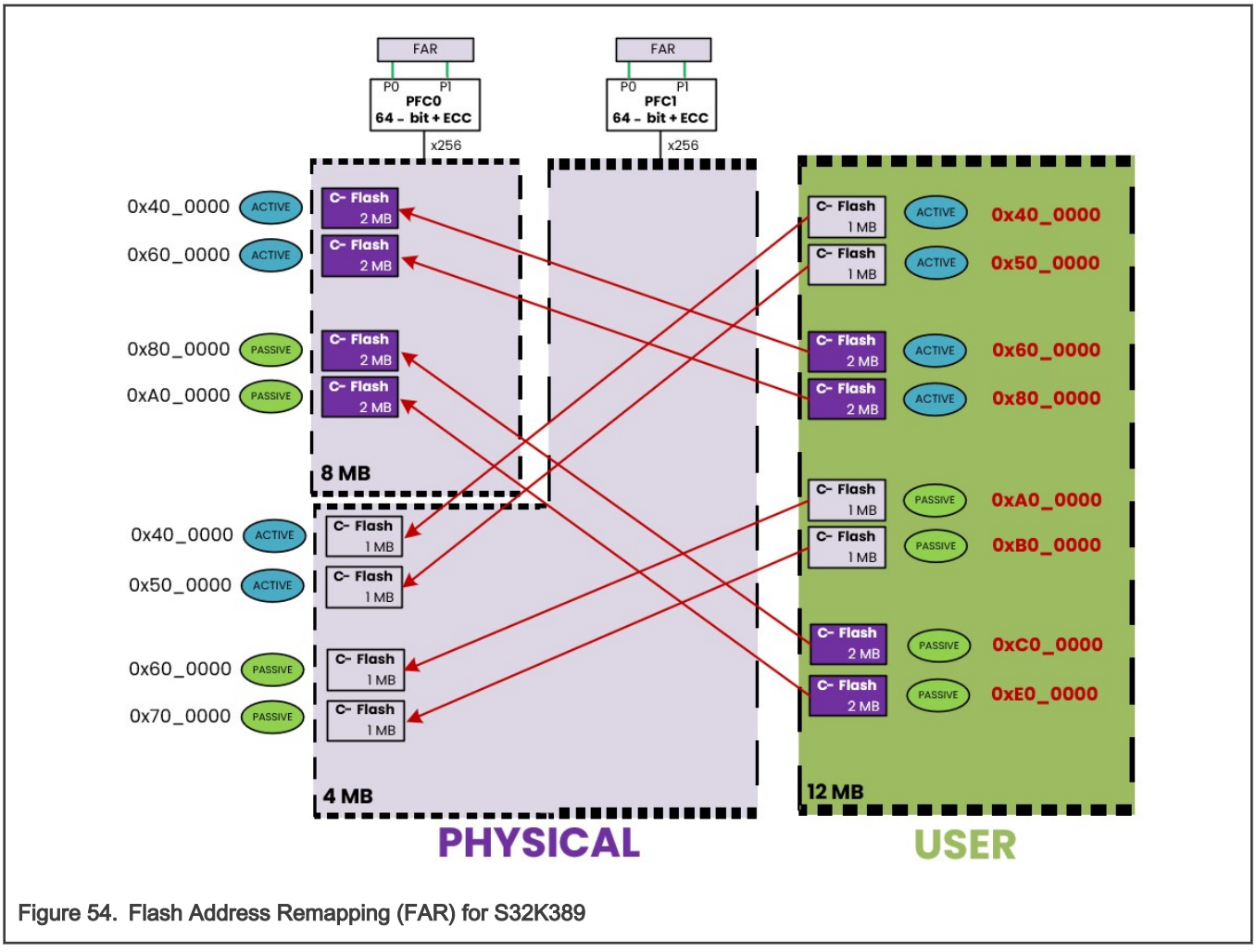


Figure 54. Flash Address Remapping (FAR) for S32K389

22.1.3 ECC error handling on data flash block (for S32K389)

As explained in Dual Flash memory controller architecture (for S32K389), each time a system master requests certain address, it is internally remapped to the one that the controller can natively understand.

In the scenario, where ECC error occurs, the flash controllers detect and send two parameters to the ERM to be latched.

1. Type: single-bit or multi-bit error
2. Error address (for both code and data blocks).

However, the reporting of the second one (error address) is based on the physical address (not the system logic address). Hence, in order to correctly interpret this information, the application software needs to perform this translation. See section "ECC error address remapping (S32K389)" in the "Error Reporting Module (ERM)" chapter.

22.1.4 Platform flash configuration registers (PFCR*n*)

The table below defines the PFCR*n* fields that the chip uses.

Table 111. Platform flash configuration registers (PFCR*n*)

Master	Buffer enable field	Prefetch enable field
Cortex–M7_0	P0_CBFEN, P0_DBFEN	P0_CPFEN, P0_DPFEN
eDMA + HSE_B + EMAC[1] + GMAC_0[2] + GMAC_1	P1_CBFEN, P1_DBFEN	P1_CPFEN, P1_DPFEN
Cortex–M7_1 + Cortex-M7_3[3]	P2_CBFEN, P2_DBFEN	P2_CPFEN, P2_DPFEN
Cortex–M7_2	P3_CBFEN, P3_DBFEN	P3_CPFEN, P3_DPFEN
S32K389: Cortex-M7_0, Cortex-M7_2	PF0_P0_CBFEN, PF0_P0_DBFEN, PF1_P0_CBFEN, PF1_P0_DBFEN	PF0_P0_CPFEN, PF0_P0_DPFEN,PF1_P0_CPFEN, PF1_P0_DPFEN
S32K389: Cortex-M7_1, Cortex_M7_3, eDMA, HSE,GMAC_0, GMAC_1	PF0_P1_CBFEN, PF0_P1_DBFEN, PF1_P1_CBFEN, PF1_P1_DBFEN	PF0_P1_CPFEN, PF0_P1_DPFEN,PF1_P1_CPFEN, PF1_P1_DPFEN

1. S32K358, S32K356, S32K388, S32K389, S32K312, S32K311, and S32K310 do not support EMAC.
2. Only available in S32K358, S32K356, and S32K388.
3. Only available in S32K388.

22.1.5 Platform flash access protection register (PFAPR)

This table defines the PFAPR[M*n*AP] fields that the chip uses. This chip does not use the other master access protection fields, but those fields are readable and writable.

Table 112. Platform flash access protection register (PFAPR)

Master number	Master name	Access protection field
0	Cortex–M7_0 AHBM	M0AP
1	Cortex–M7_1 AHBM	M1AP
2	eDMA	M2AP
3	HSE	M3AP
4	EMAC/GMAC_0	M4AP
5	Cortex–M7_3 AHBM	M5AP
6	uSDHC/ GMAC_1	M6AP
7	Cortex–M7_2 AHBM	M7AP

Table continues on the next page...

Table 112. Platform flash access protection register (PFAPR) (continued)

Master number	Master name	Access protection field
8	ACE_ACCEL RESULT	M8AP
9	ACE_ACCEL FEED	M9AP

22.2 Overview

PFLASH acts as an interface between the system bus (AHB-Lite 2.v6) and flash memory array.

PFLASH supports three 64-bit AHB buses and a 256-bit read data interface from each flash memory array. The slave port assignments and buffer organization are organized to offer maximum performance of code execution in a multicore architecture. The buffer mechanism serves to deliver flash memory read data with zero-wait state response on lines that reside in cache. AHB requests that miss the prefetch cache generate the needed flash memory array access and are forwarded to the AHB upon completion. Each cache entry is 256 bits, matching the flash memory array page size and providing 512 bytes of high-speed local storage.

22.2.1 Block diagrams

The following figure provides a block diagram showing PFLASH and the attached flash memory array.

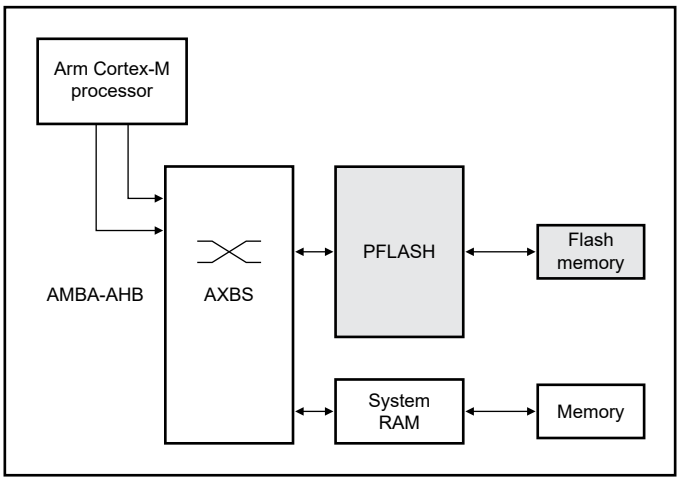


Figure 55. Platform-centric simple block diagram with PFLASH

22.2.2 Features

- Four 64-bit AHB interface ports (p0, p1, p2, p3) allowing simultaneous access to dedicated prefetch mini-cache per slave port
- 256-bit read data bus and 64-bit write data bus
- Configurable read buffering and line prefetching support via a mini-cache, plus a prefetch controller for each AHB port to provide single-cycle buffer hit read response
- Configurable access control based on read/write and AHB master ID attributes
- Support for reporting single-bit and multi-bit flash memory ECC events on a 64-bit doubleword boundary

22.3 Functional description

As shown in Figure 55, PFLASH interfaces between:

- The AHB system bus port
- The flash memory array

For accesses targeting flash memory, the PFLASH generates as inputs to the flash memory array:

- Read and write enables
- Block selects
- Array address
- Write size
- Write data

PFLASH captures read data from the flash memory array and drives it onto the AHB system bus. Up to four pages of data (256-bit page size) may be buffered in each prefetch buffer for AHB Port0, Port1, and Port2. and Port 3. Lines may be prefetched in advance of being requested, allowing single-cycle (zero AHB wait-states) read data responses on buffer hits.

Access protections may be applied on a per-master basis for both reads and writes to support security and privilege mechanisms.

22.3.1 Read transactions

On an incoming AHB read request, a mini-cache lookup and access privilege evaluation are performed during the AHB address phase. If a buffer hit occurs, the requested data is retrieved from the previously loaded prefetch buffer entry and returned on the system bus with a zero wait-state response. If a buffer miss occurs, a flash memory access is initiated.

Read accesses are terminated under control of the appropriate wait state settings. Access timing can be varied to account for the operating conditions of the chip (for example, frequency, voltage, temperature, and so on) by appropriately setting the read wait state field in flash memory.

22.3.2 Write transactions

An interlock write on a program or erase sequence is initiated by first writing to Platform Flash Memory Program Erase Address Logical (PFCPGM_PEADR_L), Platform Flash Memory Express Program Erase Address Logical (PFCPGM_XPEADR_L)(see the Flash Memory chapter for write sequence details).

22.3.3 Access protections

22.3.3.1 PFAPR

PFLASH provides programmable, configurable access protections for read cycles on a per-master basis in PFAPR[M*n*AP]. This field restricts read access on a per-master basis. This functionality is described in Platform Flash Memory Access Protection (PFAPR). Detection of a protection violation based on PFAPR settings results in an error response from PFLASH on the AHB transfer.

22.3.4 Error termination

PFLASH can invoke a system bus error termination in the following scenarios:

- Access privilege violation (see Access protections for details)
- Attempted access by an AHB master to a reserved region in the flash memory map
- Multi-bit ECC error detection on AHB read, and PFCR4[DMEEE] = 0 (for data flash memory, it is further qualified by PFCR4[DERR_SUP] = 0)

22.3.5 Line read buffers and prefetch operation

The PFLASH AHB ports of the a mini-cache. PFLASH uses the buffers for both prefetch and normal demand fetches. Also, the buffers are shared for code and data fetches, and can be controlled independently for code and data from control registers.

Prefetch triggering is controllable on a per-port basis. A PFLASH read access may trigger a prefetch to the next sequential line of array data on the cycle following the request. The access address is incremented by 32 bytes, and a subsequent flash memory access is initiated. A flash memory array prefetch is initiated if the data is not already resident in a line read buffer. Prefetched data is always loaded into the least-recently-used buffer.

For Port0, Port1, and Port2 there are four line buffer entries in their respective prefetch mini-caches that follow a fully associative, least-recently-used replacement policy. Port 3 configuration is the same as other ports.

For prefetching to occur, you must set the following configuration fields, where *n* corresponds to the port number and *m* corresponds to *C* (code) or *D* (data):

PFCRn[Pn_mBFEN] = 1 and PFCRn[Pn_mPFEN] = 1

22.3.6 Array integrity considerations

During an array integrity sequence, the flash memory array ignores any incoming read requests. When a flash memory array integrity check is in progress, PFLASH terminates all flash memory access requests with an error. More specifically, it aborts the incoming flash memory access requests and terminates the system bus transfer with an error.

22.3.7 Safety considerations

22.3.7.1 Flash memory address generation check

Functional safety coverage of the address path and control within PFLASH rely on a feedback path between PFLASH and flash memory. Remember that on a requested access to flash memory, PFLASH must decode the system AHB bus signals to generate the corresponding flash memory interface signals to invoke a flash memory lookup. In addition to providing the requested read data, the flash memory also provides output sidebands reflecting the encoded address and block selects used to perform the actual row lookup.

PFLASH uses this sideband information to verify the expected transaction. If a mismatch is detected, indicating a failure in the address generation or control logic within PFLASH or the transmission path between PFLASH and the flash memory array, then the event is forwarded to the chip fault collection module and the corresponding buffer is invalidated.

22.3.8 ECC error handling on data flash block

When PFCR4[DERR_SUP] is enabled, ECC errors on data flash blocks are handled specially.

If there is a noncorrectable error detection, a fixed, illegal opcode value is returned to the requesting master along with the associated ECC checkbits as determined by the requesting address.

For noncorrectable error detection, PFLASH returns a value of 1555_1555h to the requesting master.

This is mainly used for EEPROM emulation applications.

22.3.9 Read cycles—buffer miss

On an incoming AHB read request, a mini-cache lookup and access privilege evaluation are performed during the AHB address phase. If a buffer miss occurs, a flash memory access is initiated.

If the flash memory access was the direct result of an AHB transaction, the corresponding page buffer is loaded and marked as the most-recently-used. If the flash memory access was the result of a speculative prefetch to the next sequential line, it is loaded into the least-recently-used buffer. The status of this buffer is not changed to most-recently-used until a subsequent buffer hit occurs as a result of an AHB read request.

22.3.10 Read cycles—buffer hit

PFLASH allows single-cycle read responses to the AHB when the requested read access was previously loaded into one of the page buffers. In these cases of a buffer hit, read data is returned on the system bus with a zero wait-state response.

22.3.11 Flash memory error response operation

The flash memory array may signal an error response to terminate a requested access because of improper sequencing during program/erase operations and improper sequencing during array integrity testing. When an error response is received, PFLASH does not update or validate a page read buffer. An error response may be signaled on a read or interlock write operation. For more information on the specifics related to signaling of flash memory errors, including flash memory ECC events, array integrity testing, and read-while-write events, see the flash memory chapter.

22.3.12 Clocking

This module has no clocking considerations.

22.3.13 Interrupts

This module has no interrupts.

22.4 External signals

This module has no external signals.

22.5 PFLASH0 register descriptions

PFLASH provides an IPS programming model mapped to a standard 16 KB on-platform peripheral slot. The programming model consists of flash memory access configuration.

You can reference the programming model only by using a 32-bit (word) access. References that are attempted using different access sizes, or to undefined (reserved) addresses, or in User mode generate an IPS error termination. PFLASH allows access to the programming model by all system bus masters.

Write to read only registers don't generate error termination.

You can only access the programming model in Supervisor mode, except *PEADR* registers which can be accessed in Supervisor or User mode.

Attempted updates to the programming model when PFLASH is in the middle of an operation results in non-deterministic behavior. You must architect software to avoid this scenario. The recommended flow for multicore devices is:

1. Start only one core.
2. Execute initialization code until it is complete.
3. Start the remaining cores.

If you need to reconfigure the flash memory, code execution must be temporarily moved to system RAM.

22.5.1 PFLASH0 memory map

PFLASH0 base address: 4026_8000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Platform Flash Memory Configuration 0 (PFCR0)	32	RW	0000_0003h
4h	Platform Flash Memory Configuration 1 (PFCR1)	32	RW	0000_0003h
8h	Platform Flash Memory Configuration 2 (PFCR2)	32	RW	0000_0003h
Ch	Platform Flash Memory Configuration 3 (PFCR3)	32	RW	0000_0003h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
10h	Platform Flash Memory Configuration 4 (PFCR4)	32	RW	0000_0000h
14h	Platform Flash Memory Access Protection (PFAPR)	32	RW	FFFF_FFFFh
300h	Platform Flash Memory Program Erase Address Logical (PFCPGM_PEADR_L)	32	RW	0000_0000h
304h	Platform Flash Memory Program Erase Address Physical (PFCPGM_PEADR_P)	32	R	0000_0000h
308h	Platform Flash Memory Express Program Erase Address Logical (PFCPGM_XPEADR_L)	32	RW	0000_0000h
30Ch	Platform Flash Memory Express Program Erase Address Physical (PFCPGM_XPEADR_P)	32	R	0000_0000h
340h - 350h	Block n Sector Program Erase Lock (PFCBLK0_SPELOCK - PFCBLK4_SPELOCK)	32	RW	FFFF_FFFFh
358h	Block UTEST Sector Program Erase Lock (PFCBLKU_SPELOCK)	32	RW	0000_0001h
35Ch - 368h	Block n Super Sector Program Erase Lock (PFCBLK0_SSPELOCK - PFCBLK3_SSPELOCK)	32	RW	0FFF_FFFFh
380h - 390h	Block n Set Sector Lock (PFCBLK0_SETSLOCK - PFCBLK4_SETSLOCK)	32	RW	0000_0000h
398h	Block UTEST Set Sector Lock (PFCBLKU_SETSLOCK)	32	RW	0000_0000h
39Ch - 3A8h	Block n Set Super Sector Lock (PFCBLK0_SSETSLOCK - PFCBLK3_SSETSLOCK)	32	RW	0000_0000h
3C0h - 45Ch	Block a Lock Master Sector b (PFCBLK0_LOCKMASTER_S0 - PFCBLK4_LOCKMASTER_S7)	32	R	FFFF_FFFFh
480h	Block UTEST Lock Master Sector (PFCBLKU_LOCKMASTER_S)	32	R	0000_00FFh
484h - 4F0h	Block m Lock Master Super Sector n (PFCBLK0_LOCKMASTER_SS0 - PFCBLK3_LOCKMASTER_SS6)	32	R	FFFF_FFFFh

22.5.2 Platform Flash Memory Configuration 0 (PFCR0)

Offset

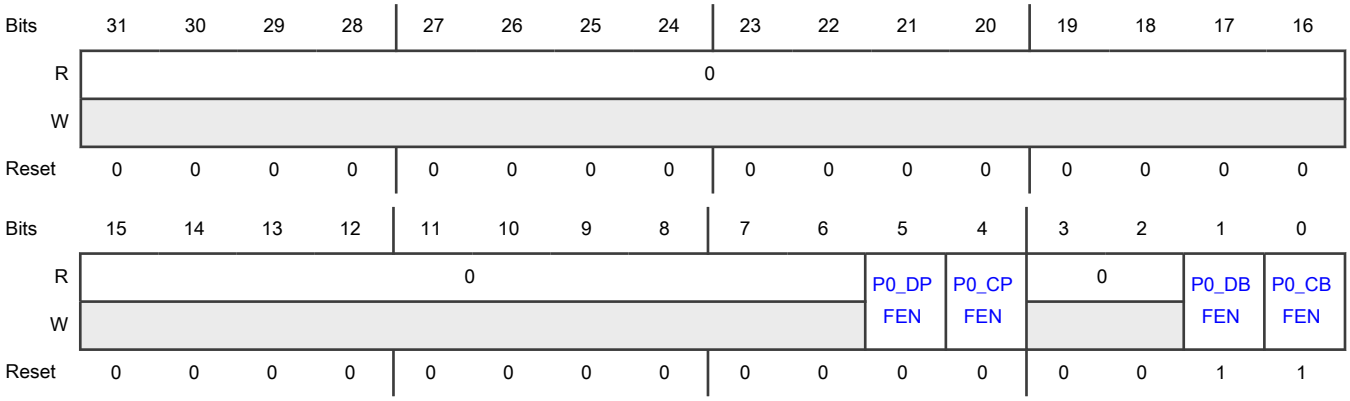
Register	Offset
PFCR0	0h

Function

Specifies the operation of PFLASH Port0.

See the chip-specific PFLASH information for details about the actual masters available on the chip.

Diagram



Fields

Field	Function
31-6 —	Reserved
5 P0_DPFEN	Port0 Data Prefetch Enable Enables or disables data prefetching initiated by a read access on Port0. Prefetching can only be enabled if the buffers are enabled by writing 1 to DBFEN. Hardware reset returns this field to 0. 0b - Disable 1b - Enable
4 P0_CPFEN	Port0 Code Prefetch Enable Enables or disables code prefetching initiated by a read access on Port0. Prefetching can only be enabled if the buffers are enabled by writing 1 to CBFEN. Hardware reset returns this field to 0. 0b - Disable 1b - Enable
3-2 —	Reserved
1 P0_DBFEN	Port0 PFLASH Line Read Data Buffers Enable Enables or disables line read data buffer hits. It is also used to invalidate the buffers. If this field is 0, the line read buffers are disabled from satisfying read requests, and all buffer valid bits are set to 0. If this field is enabled, the line read buffers are enabled to satisfy read requests on hits. Buffer valid bits may be set when the buffers are successfully filled. 0b - Disable 1b - Enable
0 P0_CBFEN	Port0 PFLASH Line Read Code Buffers Enable Enables or disables line read code buffer hits. It is also used to invalidate the buffers.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	If disabled, the line read buffers are disabled from satisfying read requests, and all buffer valid bits are set to 0. If enabled, the line read buffers are enabled to satisfy read requests on hits. Buffer valid bits may be set when the buffers are successfully filled. 0b - Disable 1b - Enable

22.5.3 Platform Flash Memory Configuration 1 (PFCR1)

Offset

Register	Offset
PFCR1	4h

Function

Specifies the operation of PFLASH Port1.

See the chip-specific PFLASH information for details about the actual masters available on the chip.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0										P1_DPFEN	P1_CPFEN	0		P1_DBFEN	P1_CBFEN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Fields

Field	Function
31-6 —	Reserved
5 P1_DPFEN	Port1 Data Prefetch Enable Enables or disables data prefetching initiated by a read access on Port1. Prefetching can only be enabled if the buffers are enabled by writing 1 to DBFEN. Hardware reset returns this field to 0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disable 1b - Enable
4 P1_CPFEN	Port1 Code Prefetch Enable Enables or disables code prefetching initiated by a read access on Port1. Prefetching can only be enabled if the buffers are enabled by writing 1 to CBFEN. Hardware reset returns this field to 0. 0b - Disable 1b - Enable
3-2 —	Reserved
1 P1_DBFEN	Port1 PFLASH Line Read Data Buffers Enable Enables or disables line read data buffer hits. It is also used to invalidate the buffers. If this field is 0, the line read buffers are disabled from satisfying read requests, and all buffer valid bits are set to 0. If this field is enabled, the line read buffers are enabled to satisfy read requests on hits. Buffer valid bits may be set when the buffers are successfully filled. 0b - Disable 1b - Enable
0 P1_CBFEN	Port1 PFLASH Line Read Code Buffers Enable Enables or disables line read code buffer hits. It is also used to invalidate the buffers. If disabled, the line read buffers are disabled from satisfying read requests, and all buffer valid bits are set to 0. If enabled, the line read buffers are enabled to satisfy read requests on hits. Buffer valid bits may be set when the buffers are successfully filled. 0b - Disable 1b - Enable

22.5.4 Platform Flash Memory Configuration 2 (PFCR2)

Offset

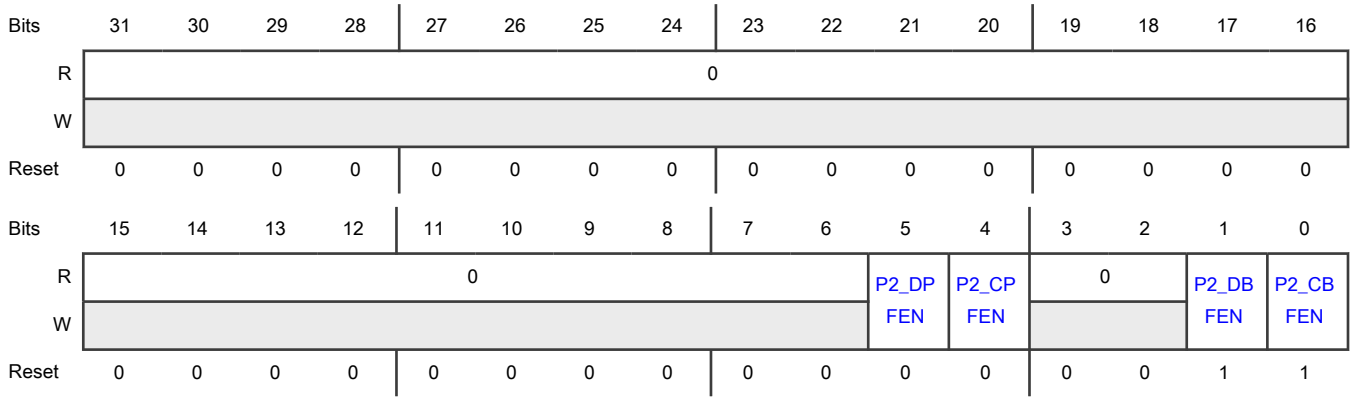
Register	Offset
PFCR2	8h

Function

Specifies the operation of PFLASH Port2.

See the chip-specific PFLASH information for details about the actual masters available on the chip.

Diagram



Fields

Field	Function
31-6 —	Reserved
5 P2_DPFEN	Port2 Data Prefetch Enable Enables or disables data prefetching initiated by a read access on Port2. Prefetching can only be enabled if the buffers are enabled by writing 1 to DBFEN. Hardware reset returns this field to 0. 0b - Disable 1b - Enable
4 P2_CPFEN	Port2 Code Prefetch Enable Enables or disables code prefetching initiated by a read access on Port2. Prefetching can only be enabled if the buffers are enabled by writing 1 to CBFEN. Hardware reset returns this field to 0. 0b - Disable 1b - Enable
3-2 —	Reserved
1 P2_DBFEN	Port2 PFLASH Line Read Data Buffers Enable Enables or disables line read data buffer hits. It is also used to invalidate the buffers. If this field is 0, the line read buffers are disabled from satisfying read requests, and all buffer valid bits are set to 0. If this field is enabled, the line read buffers are enabled to satisfy read requests on hits. Buffer valid bits may be set when the buffers are successfully filled. 0b - Disable 1b - Enable
0 P2_CBFEN	Port2 PFLASH Line Read Code Buffers Enable Enables or disables line read code buffer hits. It is also used to invalidate the buffers.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	If disabled, the line read buffers are disabled from satisfying read requests, and all buffer valid bits are set to 0. If enabled, the line read buffers are enabled to satisfy read requests on hits. Buffer valid bits may be set when the buffers are successfully filled. 0b - Disable 1b - Enable

22.5.5 Platform Flash Memory Configuration 3 (PFCR3)

Offset

Register	Offset
PFCR3	Ch

Function

Specifies the operation of PFLASH Port3.

See the chip-specific PFLASH information for details about the actual masters available on the chip.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0										P3_DPFEN	P3_CPFEN	0		P3_DBFEN	P3_CBFEN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Fields

Field	Function
31-6 —	Reserved
5 P3_DPFEN	Port3 Data Prefetch Enable Enables or disables data prefetching initiated by a read access on Port3. Prefetching can only be enabled if the buffers are enabled by writing 1 to DBFEN. Hardware reset returns this field to 0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disable 1b - Enable
4 P3_CPFEN	Port3 Code Prefetch Enable Enables or disables code prefetching initiated by a read access on Port3. Prefetching can only be enabled if the buffers are enabled by writing 1 to CBFEN. Hardware reset returns this field to 0. 0b - Disable 1b - Enable
3-2 —	Reserved
1 P3_DBFEN	Port3 PFLASH Line Read Data Buffers Enable Enables or disables line read data buffer hits. It is also used to invalidate the buffers. If this field is 0, the line read buffers are disabled from satisfying read requests, and all buffer valid bits are set to 0. If this field is enabled, the line read buffers are enabled to satisfy read requests on hits. Buffer valid bits may be set when the buffers are successfully filled. 0b - Disable 1b - Enable
0 P3_CBFEN	Port3 PFLASH Line Read Code Buffers Enable Enables or disables line read code buffer hits. It is also used to invalidate the buffers. If disabled, the line read buffers are disabled from satisfying read requests, and all buffer valid bits are set to 0. If enabled, the line read buffers are enabled to satisfy read requests on hits. Buffer valid bits may be set when the buffers are successfully filled. 0b - Disable 1b - Enable

22.5.6 Platform Flash Memory Configuration 4 (PFCR4)

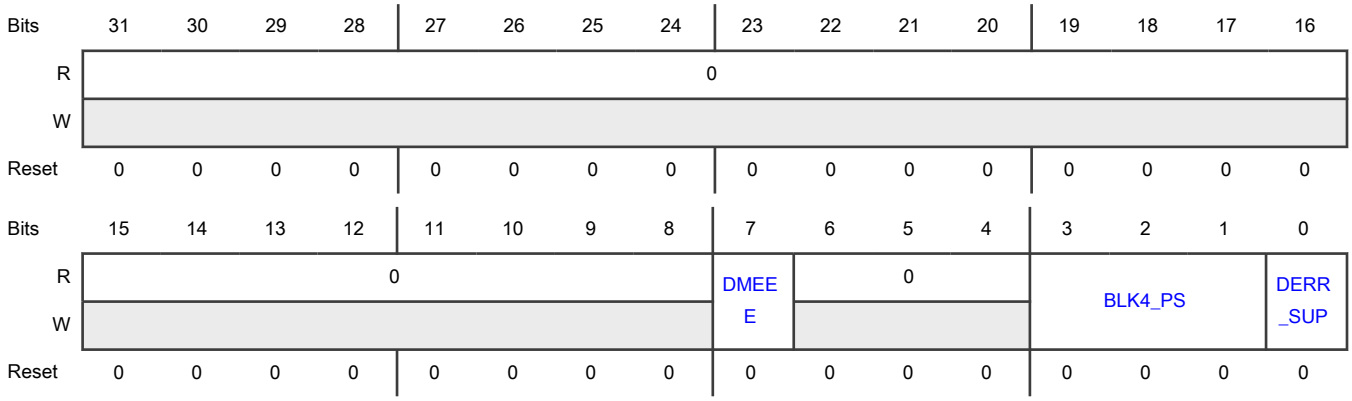
Offset

Register	Offset
PFCR4	10h

Function

Specifies operation of the flash memory controller buffer.

Diagram



Fields

Field	Function
31-8 —	Reserved
7 DMEEE	Disable Multi-Bit ECC Error Exception Enables or disables system bus error response on multi-bit ECC error. Hardware reset returns this field to 0. 0b - Error response sent on system bus for multi-bit ECC error 1b - Error response not sent on system bus for multi-bit ECC error
6-4 —	Reserved
3-1 BLK4_PS	Block 4 Pipe Select Selects the active pipe for flash memory block 4 access. PFLASH has four independent command pipes to issue four parallel read commands to different flash memory blocks. Reads from flash memory block 0–3 are always done through command pipe 0–3, respectively. However, the access to block 4 is not fixed and can be through any of the command pipes. You must only change this field when there is no ongoing block 4 access. A special round-robin arbitration scheme snoops the availability of a command pipe during block 4 access. If any of the command pipes are idle during the first read request to block 4, the ownership of that command pipe gets shared between block 4 and the respective block. If none of the command pipes are idle during a block 4 read request, block 4 gets associated with each of the command pipes in round-robin fashion. When a command pipe acquires ownership of block 4, it keeps that ownership until all the commands to block 4 from all the masters are served. 000b - Block 4 access is always through pipe0 001b - Block 4 access is always through pipe1 010b - Block 4 access is always through pipe2 011b - Block 4 access is always through pipe3

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1xxb - Block 4 access can be through any of the command pipes, based on which command pipe is available for block 4 access
0 DERR_SUP	Data Error Suppression See the Embedded Flash Memory configuration information or system memory map for which flash memory blocks are affected by this field. 0b - Reports ECC events on data flash memory accesses 1b - Single-bit and multi-bit ECC events on data flash memory accesses are suppressed

22.5.7 Platform Flash Memory Access Protection (PFAPR)

Offset

Register	Offset
PFAPR	14h

Function

Controls read accesses to the flash memory array.

See the chip-specific PFLASH information for details about the actual masters available on the chip.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	M0AP		M1AP		M2AP		M3AP		M4AP		M5AP		M6AP		M7AP	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	M8AP		M9AP		M10AP		M11AP		M12AP		M13AP		M14AP		M15AP	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-30: M0AP 29-28: M1AP 27-26: M2AP	Master n Access Protection Controls whether read accesses to the flash memory are allowed based on the master ID of a requesting master. These fields are initialized by hardware reset. The field M4'd3AP is reserved.

Table continues on the next page...

Field	Function
25-24: M3AP 23-22: M4AP 21-20: M5AP 19-18: M6AP 17-16: M7AP 15-14: M8AP 13-12: M9AP 11-10: M10AP 9-8: M11AP 7-6: M12AP 5-4: M13AP 3-2: M14AP 1-0: M15AP	x0b - This master cannot perform any read accesses x1b - This master can perform read accesses

22.5.8 Platform Flash Memory Program Erase Address Logical (PFCPGM_PEADR_L)

Offset

Register	Offset
PFCPGM_PEADR_L	300h

Function

Provides the flash memory address to be programmed, or the location of the sector or block to be erased through main flash memory (pgm/erase) queue. Write access to this register is domain-ID aware.

The respective bus master must have program/erase permission to the flash memory address written to this register. Otherwise a transfer error results. For further information on flash memory address restrictions see the XRDC chapter.

A write to this register is managed via three-cycle access. Before updating the register, you must ensure that no ongoing high-voltage operation is executing through the flash memory main queue.

Unauthorized flash memory address writes result in a transfer error.

Writes to this register during an ongoing high-voltage operation (initiated through the flash memory main queue) or during express program operation are ignored.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PEADR_L															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PEADR_L															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PEADR_L	Program Erase Address Logical Contains the system logical address for flash memory program/erase.

22.5.9 Platform Flash Memory Program Erase Address Physical (PFCPGM_PEADR_P)

Offset

Register	Offset
PFCPGM_PEADR_P	304h

Function

Reflects the flash memory block number and offset address corresponding to Platform Flash Memory Program Erase Address Logical (PFCPGM_PEADR_L). This register has the same format as the PEADR register in the Flash Memory chapter—see it for details.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PEADR_P															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PEADR_P															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PEADR_P	Program Erase Address Physical Contains the flash block select and offset address for flash memory program/erase.

22.5.10 Platform Flash Memory Express Program Erase Address Logical (PFCPGM_XPEADR_L)

Offset

Register	Offset
PFCPGM_XPEADR_L	308h

Function

Provides the flash memory address to be programmed using the flash memory express program feature. Write access to this register is domain-ID aware.

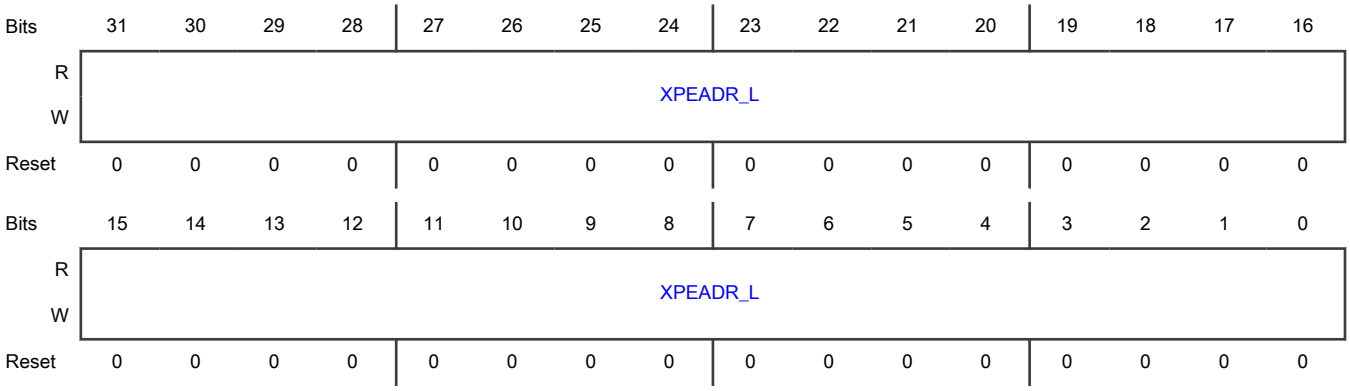
The respective bus master must have program/erase permission to the flash memory address written to this register. Otherwise a transfer error results. See the XRDC chapter for further information on flash memory address restrictions.

A write to this register is managed via three-cycle access. Before updating the register, you must ensure that no ongoing high-voltage operation is executing through the flash memory main queue.

Unauthorized flash memory address writes result in a transfer error.

Writes to this register during an ongoing express program operation are ignored.

Diagram



Fields

Field	Function
31-0 XPEADR_L	Express Program Erase Address Logical Contains the system logical address for express flash program/erase.

22.5.11 Platform Flash Memory Express Program Erase Address Physical (PFCPGM_XPEADR_P)

Offset

Register	Offset
PFCPGM_XPEADR_P	30Ch

Function

Reflects the flash memory block number and offset address corresponding to PFCPGM_XPEADR_L. This register has the same format as the XPEADR register in the Flash Memory chapter—see it for details.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	XPEADR_P															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	XPEADR_P															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 XPEADR_P	Express Program Erase Address Physical Contains the flash memory block select and offset address for flash memory express program/erase.

22.5.12 Block n Sector Program Erase Lock (PFCBLK0_SPELOCK - PFCBLK4_SPELOCK)

Offset

Register	Offset
PFCBLK0_SPELOCK	340h
PFCBLK1_SPELOCK	344h
PFCBLK2_SPELOCK	348h
PFCBLK3_SPELOCK	34Ch
PFCBLK4_SPELOCK	350h

Function

Provides a way to protect sectors from being modified. Sector protection is available on the last 256 KB of every block (for 256 KB blocks, all sectors are available for protection). Each lock bit can be associated with a particular domain ID by writing 1 to the appropriate bit in PFCBLKn_SETSLOCK[SETSLCK]. After the lock is acquired, only a master having the same domain ID can modify the corresponding lock bit. If the corresponding PFCBLKn_SETSLOCK[SETSLCK] bit is not equal to 1, any master can modify the appropriate SLCK bit. If a lock bit is already acquired by a particular domain ID, any effort to modify (1 to 0, or 0 to 1) the lock bit by a master with a different domain ID results in a transfer error.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SLCK															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SLCK															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-0 SLCK	Sector Lock Locks selected sector. If vector bit value = 0, the sector is available for program and erase operations. If vector bit value = 1, the sector is locked and not available for program and erase operations.

22.5.13 Block UTEST Sector Program Erase Lock (PFCBLKU_SPELOCK)

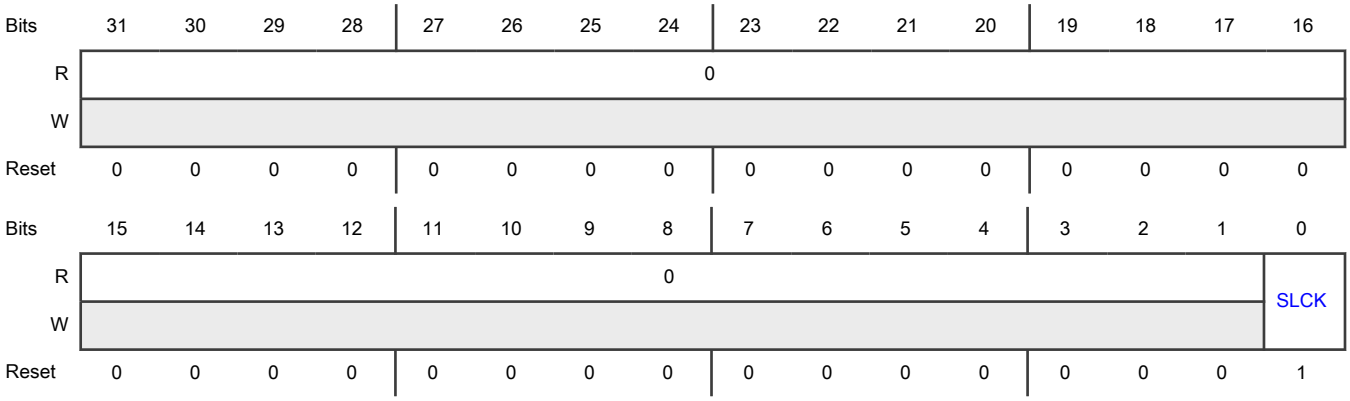
Offset

Register	Offset
PFCBLKU_SPELOCK	358h

Function

Provides a way to protect sectors from being modified. Sector protection is available on the last 256 KB of every block (for 256 KB blocks, all sectors are available for protection). Each lock bit can be associated with a particular domain ID by writing 1 to PFCBLKU_SETSLOCK[SETSLCK]. After the lock is acquired, only a master having the same domain ID can modify the corresponding lock bit. If the corresponding PFCBLKU_SETSLOCK[SETSLCK] bit is not equal to 1, any master can modify the SLCK bit. If a lock bit is already acquired by a particular domain ID, any effort to modify (1 to 0, or 0 to 1) the lock bit by a master with a different domain ID results in a transfer error.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 SLCK	Sector Lock Locks selected sector. If vector bit value = 0, the sector is available for program and erase operations. If vector bit value = 1, the sector is locked and not available for program and erase operations.

22.5.14 Block n Super Sector Program Erase Lock (PFCBLK0_SSPELOCK - PFCBLK3_SSPELOCK)

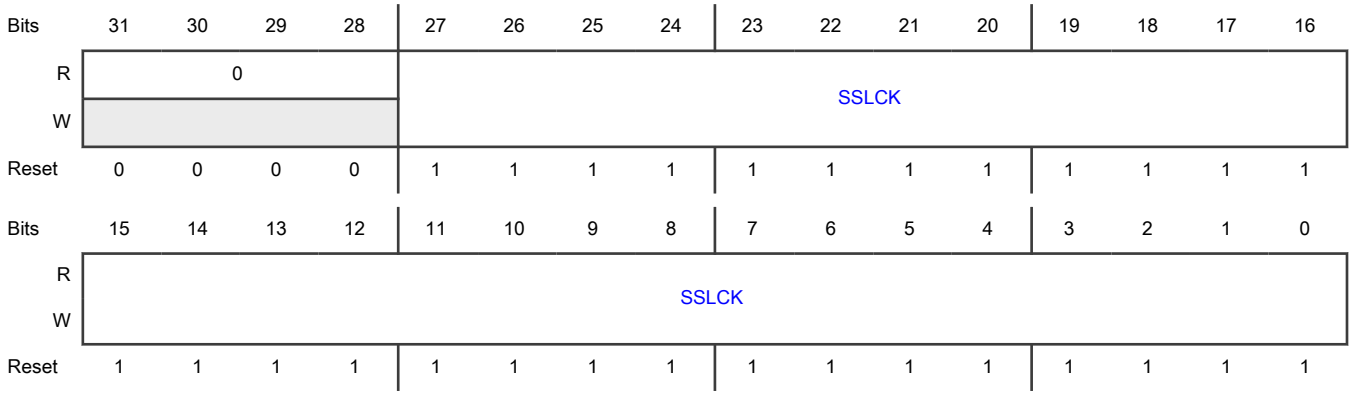
Offset

Register	Offset
PFCBLK0_SSPELOCK	35Ch
PFCBLK1_SSPELOCK	360h
PFCBLK2_SSPELOCK	364h
PFCBLK3_SSPELOCK	368h

Function

Provides a way to protect super sectors from being modified. Super sector protection is available on block space larger than 256 KB. For 256 KB blocks, this register is not applicable. For 512 KB blocks, the first half of the block is protected with super sector granularity. For 1 MB blocks, the first 768 KB is protected with super sector granularity. Each lock bit can be associated with a particular domain ID by writing 1 to the appropriate bit in PFCBLKn_SSETSLOCK[SSETSLCK]. After the lock is acquired, only a master having the same domain ID can modify the corresponding lock bit. If the corresponding PFCBLKn_SSETSLOCK[SSETSLCK] bit is not equal to 1, any master can modify the SSLCK bit. If a lock bit is already acquired by a particular domain ID, any effort to modify (1 to 0, or 0 to 1) the lock bit by a master with a different domain ID results in a transfer error.

Diagram



Fields

Field	Function
31-28 —	Reserved
27-0 SSLCK	Super Sector Lock Locks selected super sector. If vector bit value = 0, the super sector is available for program and erase operations. If vector bit value = 1, the super sector is locked and not available for program and erase operations.

22.5.15 Block n Set Sector Lock (PFCBLK0_SETSLOCK - PFCBLK4_SETSLOCK)

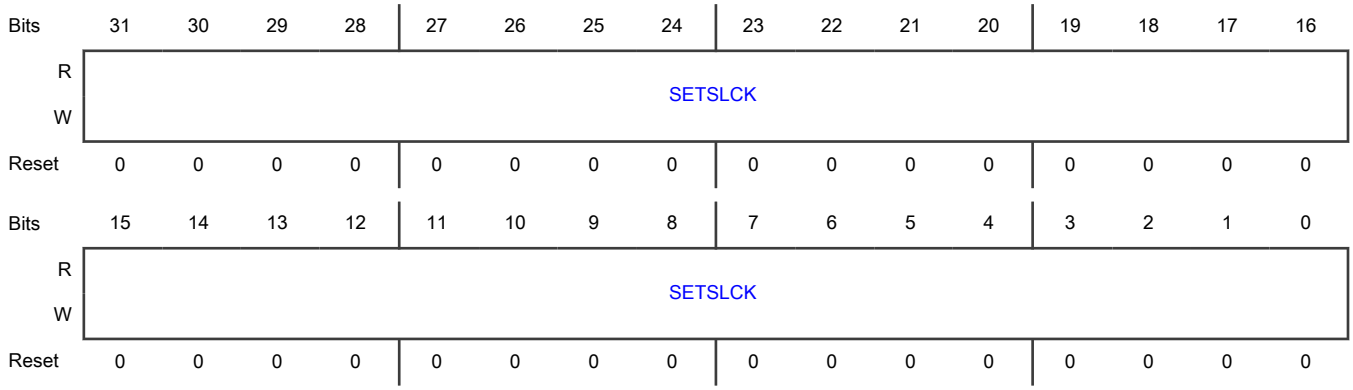
Offset

Register	Offset
PFCBLK0_SETSLOCK	380h
PFCBLK1_SETSLOCK	384h
PFCBLK2_SETSLOCK	388h
PFCBLK3_SETSLOCK	38Ch
PFCBLK4_SETSLOCK	390h

Function

Provides a mechanism for the masters to gain the ownership of the corresponding PFCBLKn_SPELOCK lock bit based on domain id . After it is equal to 1, the bit is returned to 0 at next reset. If any SETSLOCK bit is not equal to 1, the corresponding LOCK bit can be modified by any master. If a bit is already acquired by a particular domain ID, any effort to modify the lock bit by a master with a different domain ID is ignored without transfer error.

Diagram



Fields

Field	Function
31-0 SETSLCK	If the vector bit value = 0, the corresponding lock bit is not owned by any master. If the vector bit value = 1, the lock bit is owned by the masters having the domain ID stored in PFCBLKn_LOCKMASTER_Sm.

22.5.16 Block UTEST Set Sector Lock (PFCBLKU_SETSLOCK)

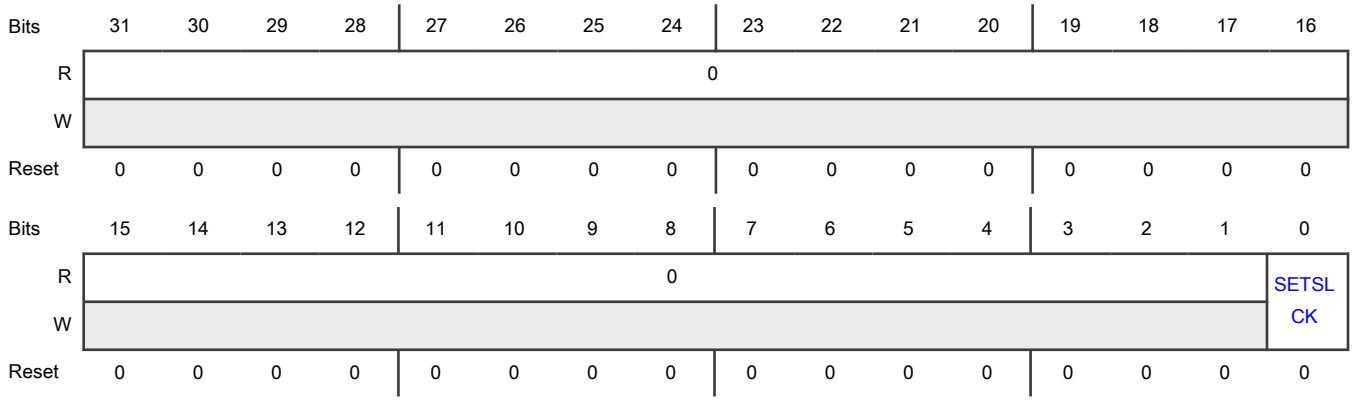
Offset

Register	Offset
PFCBLKU_SETSLOCK	398h

Function

Provides a mechanism for the masters to gain ownership of the corresponding PFCBLKU_SPELOCK lock bit based on domain id. After it is equal to 1, the bit is returned to 0 at next reset. If any SETSLOCK bit is not equal to 1, the corresponding LOCK bit can be modified by any master. If a bit is already acquired by a particular domain ID, any effort to modify the lock bit by a master with a different domain ID is ignored without transfer error.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 SETSLCK	Set Sector Lock Locks selected sector. If vector bit value = 0, the corresponding lock bit is not owned by any master. If vector bit value = 1, the lock bit is owned by the masters having the domain ID stored in PFCBLKn_LOCKMASTER_SSm.

22.5.17 Block n Set Super Sector Lock (PFCBLK0_SSETSLOCK - PFCBLK3_SSETSLOCK)

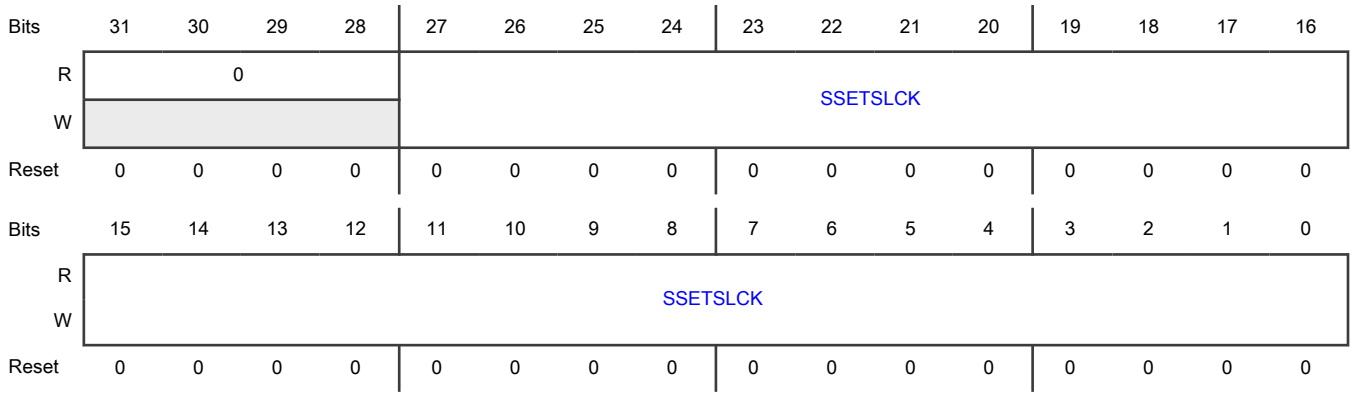
Offset

Register	Offset
PFCBLK0_SSETSLOCK	39Ch
PFCBLK1_SSETSLOCK	3A0h
PFCBLK2_SSETSLOCK	3A4h
PFCBLK3_SSETSLOCK	3A8h

Function

Provides a mechanism for the masters to gain ownership of the corresponding PFCBLKn_SPELOCK lock bit based on domain id. After it is equal to 1, the bit is returned to 0 at next reset. If any SSETSLOCK bit is not equal to 1, the corresponding LOCK bit can be modified by any master. If a bit is already acquired by a particular domain ID, any effort to modify the lock bit by a master with a different domain ID is ignored without transfer error.

Diagram



Fields

Field	Function
31-28 —	Reserved
27-0 SSETSLCK	Set Super Sector Lock Locks selected super sector. If vector bit value = 0, the corresponding lock bit is not owned by any master. If vector bit value = 1, the lock bit is owned by the masters having the domain ID stored in Block a Lock Master Sector b (PFCBLK0_LOCKMASTER_S0 - PFCBLK4_LOCKMASTER_S7).

22.5.18 Block a Lock Master Sector b (PFCBLK0_LOCKMASTER_S0 - PFCBLK4_LOCKMASTER_S7)

Offset

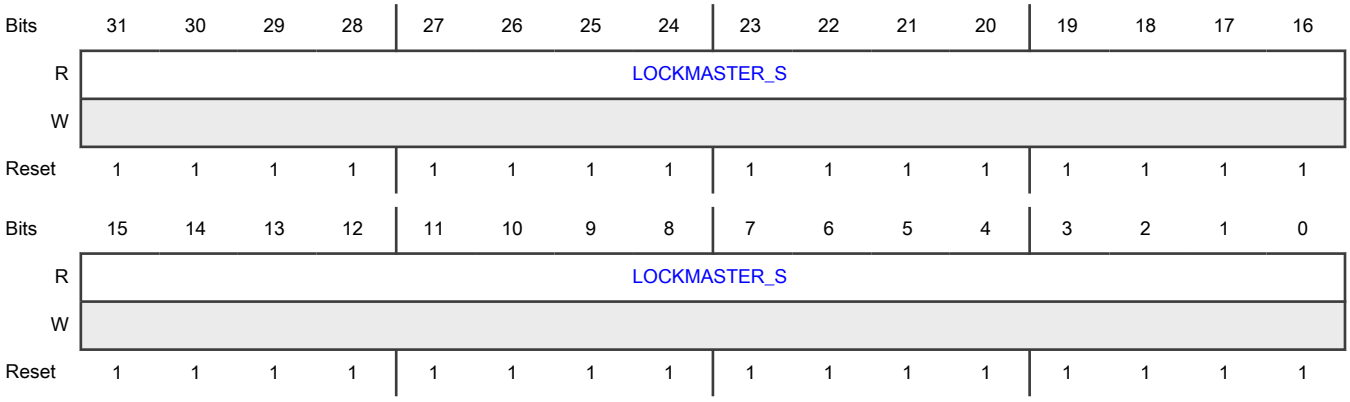
For a = 0 to 4; b = 0 to 7:

Register	Offset
PFCBLKa_LOCKMASTER_Sb	3C0h + (a × 20h) + (b × 4h)

Function

Provides the domain ID information of the master currently acquiring the lock bit. The domain ID is represented by an 8-bit field. This is a read-only register.

Diagram



Fields

Field	Function
31-0	Block a Lock Master Sector b Contains domain ID of the master currently acquiring the lock bit.

Table continues on the next page...

Field	Function
LOCKMASTER_S	PFCBLK0_LOCKMASTER_S0[LOCKMASTER_S[7:0]] holds the domain ID information of PFCBLK0_SPELOCK[0]. PFCBLK0_LOCKMASTER_S0[LOCKMASTER_S[15:8]] holds the domain ID information of PFCBLK0_SPELOCK[1], and so on in incremental order.

22.5.19 Block UTEST Lock Master Sector (PFCBLKU_LOCKMASTER_S)

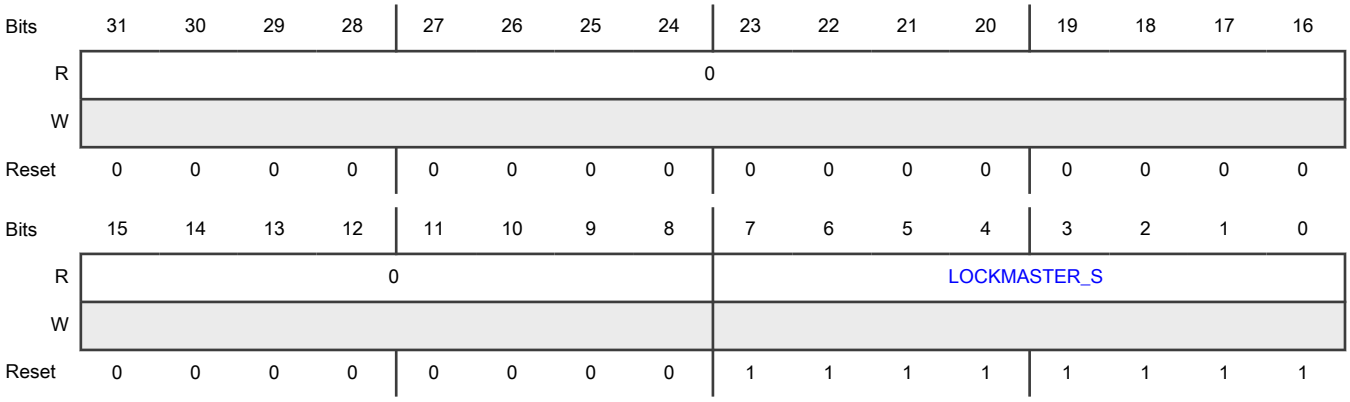
Offset

Register	Offset
PFCBLKU_LOCKMASTER_S	480h

Function

Provides the domain ID information of the master currently acquiring the lock bit. The domain ID is represented by an 8-bit field. This is a read-only register.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 LOCKMASTER_S	Lock Master Sector Contains domain ID of the master currently acquiring the lock bit. PFCBLKU_LOCKMASTER_S[LOCKMASTER_S[7:0]] holds the domain ID information of PFCBLKU_SPELOCK[0].

22.5.20 Block m Lock Master Super Sector n (PFCBLK0_LOCKMASTER_SS0 - PFCBLK3_LOCKMASTER_SS6)

Offset

For a = 0 to 3; b = 0 to 6:

Register	Offset
PFCBLKa_LOCKMASTER_SSb	484h + (a × 1Ch) + (b × 4h)

Function

Provides the domain ID information of the master currently acquiring the lock bit. The domain ID is represented by an 8-bit field. This is a read-only register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LOCKMASTER_SS															
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LOCKMASTER_SS															
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-0 LOCKMASTER_SS	Block a Lock Master Super Sector b Contains domain ID of the master currently acquiring the lock bit. PFCBLK0_LOCKMASTER_SS0[LOCKMASTER_SS[7:0]] holds the domain ID information of PFCBLK0_SSPELOCK[0]. PFCBLK0_LOCKMASTER_SS0[LOCKMASTER_SS[15:8]] holds the domain ID information of PFCBLK0_SSPELOCK[1], and so on in incremental order.

22.6 PFLASH1 register descriptions

PFLASH provides an IPS programming model mapped to a standard 16 KB on-platform peripheral slot. The programming model consists of flash memory access configuration.

You can reference the programming model only by using a 32-bit (word) access. References that are attempted using different access sizes, or to undefined (reserved) addresses, or in User mode generate an IPS error termination. PFLASH allows access to the programming model by all system bus masters.

Write to read only registers don't generate error termination.

You can only access the programming model in Supervisor mode, except *PEADR* registers which can be accessed in Supervisor or User mode.

Attempted updates to the programming model when PFLASH is in the middle of an operation results in non-deterministic behavior. You must architect software to avoid this scenario. The recommended flow for multicore devices is:

1. Start only one core.
2. Execute initialization code until it is complete.
3. Start the remaining cores.

If you need to reconfigure the flash memory, code execution must be temporarily moved to system RAM.

22.6.1 PFLASH1 memory map

PFLASH1 base address: 4006_8000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Platform Flash Memory Configuration 0 (PFCR0)	32	RW	0000_0003h
4h	Platform Flash Memory Configuration 1 (PFCR1)	32	RW	0000_0003h
8h	Platform Flash Memory Configuration 2 (PFCR2)	32	RW	0000_0003h
Ch	Platform Flash Memory Configuration 3 (PFCR3)	32	RW	0000_0003h
10h	Platform Flash Memory Configuration 4 (PFCR4)	32	RW	0000_0000h
14h	Platform Flash Memory Access Protection (PFAPR)	32	RW	FFFF_FFFFh
300h	Platform Flash Memory Program Erase Address Logical (PFCPGM_PEADR_L)	32	RW	0000_0000h
304h	Platform Flash Memory Program Erase Address Physical (PFCPGM_PEADR_P)	32	R	0000_0000h
308h	Platform Flash Memory Express Program Erase Address Logical (PFCPGM_XPEADR_L)	32	RW	0000_0000h
30Ch	Platform Flash Memory Express Program Erase Address Physical (PFCPGM_XPEADR_P)	32	R	0000_0000h
340h - 350h	Block n Sector Program Erase Lock (PFCBLK0_SPELOCK - PFCBLK4_SPELOCK)	32	RW	FFFF_FFFFh
358h	Block UTEST Sector Program Erase Lock (PFCBLKU_SPELOCK)	32	RW	0000_0001h
35Ch - 368h	Block n Super Sector Program Erase Lock (PFCBLK0_SSPELOCK - PFCBLK3_SSPELOCK)	32	RW	0000_0FFFh
380h - 390h	Block n Set Sector Lock (PFCBLK0_SETSLOCK - PFCBLK4_SETSLOCK)	32	RW	0000_0000h
398h	Block UTEST Set Sector Lock (PFCBLKU_SETSLOCK)	32	RW	0000_0000h
39Ch - 3A8h	Block n Set Super Sector Lock (PFCBLK0_SSETSLOCK - PFCBLK3_SSETSLOCK)	32	RW	0000_0000h
3C0h - 45Ch	Block a Lock Master Sector b (PFCBLK0_LOCKMASTER_S0 - PFCBLK4_LOCKMASTER_S7)	32	R	FFFF_FFFFh

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
480h	Block UTEST Lock Master Sector (PFCBLKU_LOCKMASTER_S)	32	R	0000_00FFh
484h	Block m Lock Master Super Sector n (PFCBLK0_LOCKMASTER_SS0)	32	R	FFFF_FFFFh
488h	Block m Lock Master Super Sector n (PFCBLK0_LOCKMASTER_SS1)	32	R	FFFF_FFFFh
48Ch	Block m Lock Master Super Sector n (PFCBLK0_LOCKMASTER_SS2)	32	R	FFFF_FFFFh
494h	Block m Lock Master Super Sector n (PFCBLK1_LOCKMASTER_SS0)	32	R	FFFF_FFFFh
498h	Block m Lock Master Super Sector n (PFCBLK1_LOCKMASTER_SS1)	32	R	FFFF_FFFFh
49Ch	Block m Lock Master Super Sector n (PFCBLK1_LOCKMASTER_SS2)	32	R	FFFF_FFFFh
4A4h	Block m Lock Master Super Sector n (PFCBLK2_LOCKMASTER_SS0)	32	R	FFFF_FFFFh
4A8h	Block m Lock Master Super Sector n (PFCBLK2_LOCKMASTER_SS1)	32	R	FFFF_FFFFh
4ACh	Block m Lock Master Super Sector n (PFCBLK2_LOCKMASTER_SS2)	32	R	FFFF_FFFFh
4B4h	Block m Lock Master Super Sector n (PFCBLK3_LOCKMASTER_SS0)	32	R	FFFF_FFFFh
4B8h	Block m Lock Master Super Sector n (PFCBLK3_LOCKMASTER_SS1)	32	R	FFFF_FFFFh
4BCh	Block m Lock Master Super Sector n (PFCBLK3_LOCKMASTER_SS2)	32	R	FFFF_FFFFh

22.6.2 Platform Flash Memory Configuration 0 (PFCR0)

Offset

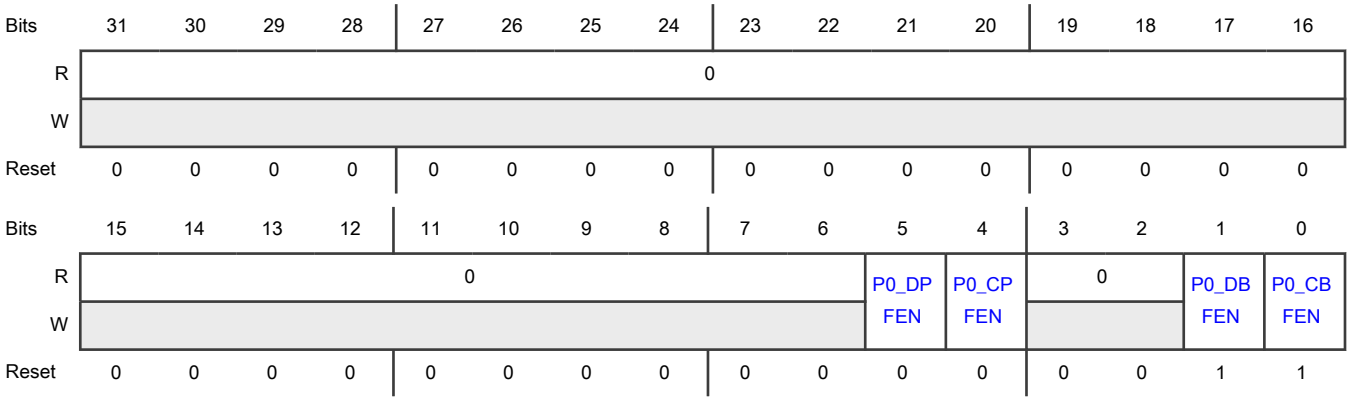
Register	Offset
PFCR0	0h

Function

Specifies the operation of PFLASH Port0.

See the chip-specific PFLASH information for details about the actual masters available on the chip.

Diagram



Fields

Field	Function
31-6 —	Reserved
5 P0_DPFEN	Port0 Data Prefetch Enable Enables or disables data prefetching initiated by a read access on Port0. Prefetching can only be enabled if the buffers are enabled by writing 1 to DBFEN. Hardware reset returns this field to 0. 0b - Disable 1b - Enable
4 P0_CPFEN	Port0 Code Prefetch Enable Enables or disables code prefetching initiated by a read access on Port0. Prefetching can only be enabled if the buffers are enabled by writing 1 to CBFEN. Hardware reset returns this field to 0. 0b - Disable 1b - Enable
3-2 —	Reserved
1 P0_DBFEN	Port0 PFLASH Line Read Data Buffers Enable Enables or disables line read data buffer hits. It is also used to invalidate the buffers. If this field is 0, the line read buffers are disabled from satisfying read requests, and all buffer valid bits are set to 0. If this field is enabled, the line read buffers are enabled to satisfy read requests on hits. Buffer valid bits may be set when the buffers are successfully filled. 0b - Disable 1b - Enable
0 P0_CBFEN	Port0 PFLASH Line Read Code Buffers Enable Enables or disables line read code buffer hits. It is also used to invalidate the buffers.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	If disabled, the line read buffers are disabled from satisfying read requests, and all buffer valid bits are set to 0. If enabled, the line read buffers are enabled to satisfy read requests on hits. Buffer valid bits may be set when the buffers are successfully filled. 0b - Disable 1b - Enable

22.6.3 Platform Flash Memory Configuration 1 (PFCR1)

Offset

Register	Offset
PFCR1	4h

Function

Specifies the operation of PFLASH Port1.

See the chip-specific PFLASH information for details about the actual masters available on the chip.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0										P1_DPFEN	P1_CPFEN	0		P1_DBFEN	P1_CBFEN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Fields

Field	Function
31-6 —	Reserved
5 P1_DPFEN	Port1 Data Prefetch Enable Enables or disables data prefetching initiated by a read access on Port1. Prefetching can only be enabled if the buffers are enabled by writing 1 to DBFEN. Hardware reset returns this field to 0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disable 1b - Enable
4 P1_CPFEN	Port1 Code Prefetch Enable Enables or disables code prefetching initiated by a read access on Port1. Prefetching can only be enabled if the buffers are enabled by writing 1 to CBFEN. Hardware reset returns this field to 0. 0b - Disable 1b - Enable
3-2 —	Reserved
1 P1_DBFEN	Port1 PFLASH Line Read Data Buffers Enable Enables or disables line read data buffer hits. It is also used to invalidate the buffers. If this field is 0, the line read buffers are disabled from satisfying read requests, and all buffer valid bits are set to 0. If this field is enabled, the line read buffers are enabled to satisfy read requests on hits. Buffer valid bits may be set when the buffers are successfully filled. 0b - Disable 1b - Enable
0 P1_CBFEN	Port1 PFLASH Line Read Code Buffers Enable Enables or disables line read code buffer hits. It is also used to invalidate the buffers. If disabled, the line read buffers are disabled from satisfying read requests, and all buffer valid bits are set to 0. If enabled, the line read buffers are enabled to satisfy read requests on hits. Buffer valid bits may be set when the buffers are successfully filled. 0b - Disable 1b - Enable

22.6.4 Platform Flash Memory Configuration 2 (PFCR2)

Offset

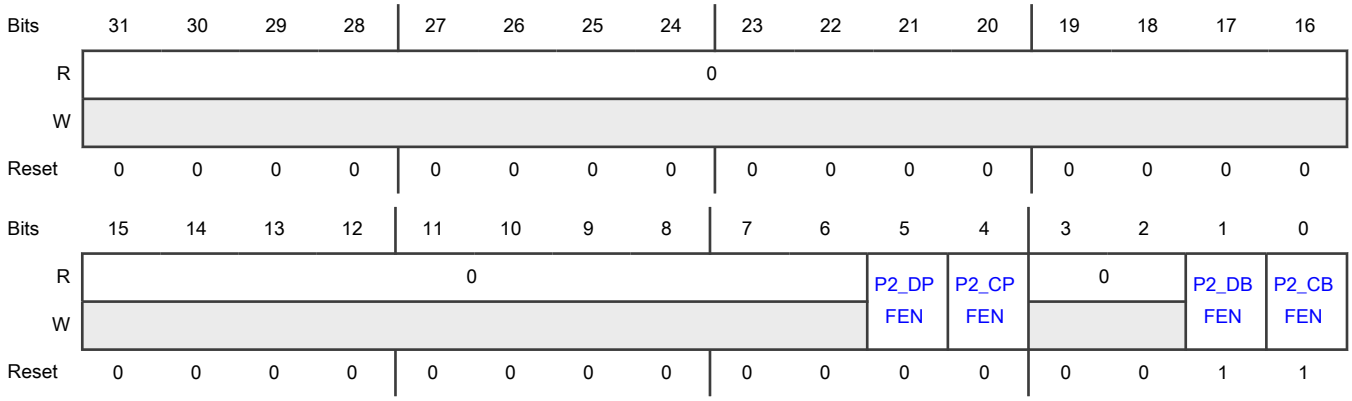
Register	Offset
PFCR2	8h

Function

Specifies the operation of PFLASH Port2.

See the chip-specific PFLASH information for details about the actual masters available on the chip.

Diagram



Fields

Field	Function
31-6 —	Reserved
5 P2_DPFEN	Port2 Data Prefetch Enable Enables or disables data prefetching initiated by a read access on Port2. Prefetching can only be enabled if the buffers are enabled by writing 1 to DBFEN. Hardware reset returns this field to 0. 0b - Disable 1b - Enable
4 P2_CPFEN	Port2 Code Prefetch Enable Enables or disables code prefetching initiated by a read access on Port2. Prefetching can only be enabled if the buffers are enabled by writing 1 to CBFEN. Hardware reset returns this field to 0. 0b - Disable 1b - Enable
3-2 —	Reserved
1 P2_DBFEN	Port2 PFLASH Line Read Data Buffers Enable Enables or disables line read data buffer hits. It is also used to invalidate the buffers. If this field is 0, the line read buffers are disabled from satisfying read requests, and all buffer valid bits are set to 0. If this field is enabled, the line read buffers are enabled to satisfy read requests on hits. Buffer valid bits may be set when the buffers are successfully filled. 0b - Disable 1b - Enable
0 P2_CBFEN	Port2 PFLASH Line Read Code Buffers Enable Enables or disables line read code buffer hits. It is also used to invalidate the buffers.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	If disabled, the line read buffers are disabled from satisfying read requests, and all buffer valid bits are set to 0. If enabled, the line read buffers are enabled to satisfy read requests on hits. Buffer valid bits may be set when the buffers are successfully filled. 0b - Disable 1b - Enable

22.6.5 Platform Flash Memory Configuration 3 (PFCR3)

Offset

Register	Offset
PFCR3	Ch

Function

Specifies the operation of PFLASH Port3.

See the chip-specific PFLASH information for details about the actual masters available on the chip.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0										P3_DPFEN	P3_CPFEN	0		P3_DBFEN	P3_CBFEN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Fields

Field	Function
31-6 —	Reserved
5 P3_DPFEN	Port3 Data Prefetch Enable Enables or disables data prefetching initiated by a read access on Port3. Prefetching can only be enabled if the buffers are enabled by writing 1 to DBFEN. Hardware reset returns this field to 0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disable 1b - Enable
4 P3_CPFEN	Port3 Code Prefetch Enable Enables or disables code prefetching initiated by a read access on Port3. Prefetching can only be enabled if the buffers are enabled by writing 1 to CBFEN. Hardware reset returns this field to 0. 0b - Disable 1b - Enable
3-2 —	Reserved
1 P3_DBFEN	Port3 PFLASH Line Read Data Buffers Enable Enables or disables line read data buffer hits. It is also used to invalidate the buffers. If this field is 0, the line read buffers are disabled from satisfying read requests, and all buffer valid bits are set to 0. If this field is enabled, the line read buffers are enabled to satisfy read requests on hits. Buffer valid bits may be set when the buffers are successfully filled. 0b - Disable 1b - Enable
0 P3_CBFEN	Port3 PFLASH Line Read Code Buffers Enable Enables or disables line read code buffer hits. It is also used to invalidate the buffers. If disabled, the line read buffers are disabled from satisfying read requests, and all buffer valid bits are set to 0. If enabled, the line read buffers are enabled to satisfy read requests on hits. Buffer valid bits may be set when the buffers are successfully filled. 0b - Disable 1b - Enable

22.6.6 Platform Flash Memory Configuration 4 (PFCR4)

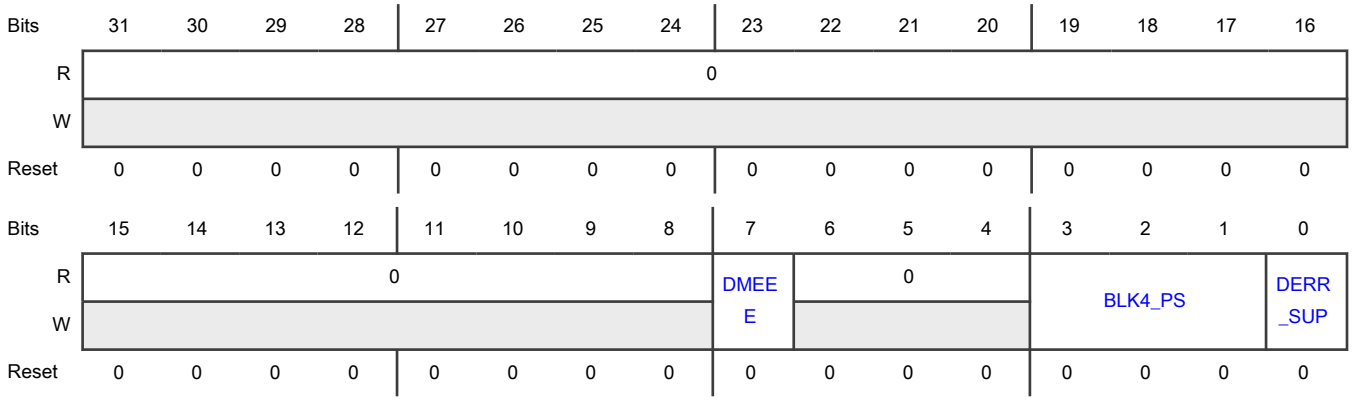
Offset

Register	Offset
PFCR4	10h

Function

Specifies operation of the flash memory controller buffer.

Diagram



Fields

Field	Function
31-8 —	Reserved
7 DMEEE	Disable Multi-Bit ECC Error Exception Enables or disables system bus error response on multi-bit ECC error. Hardware reset returns this field to 0. 0b - Error response sent on system bus for multi-bit ECC error 1b - Error response not sent on system bus for multi-bit ECC error
6-4 —	Reserved
3-1 BLK4_PS	Block 4 Pipe Select Selects the active pipe for flash memory block 4 access. PFLASH has four independent command pipes to issue four parallel read commands to different flash memory blocks. Reads from flash memory block 0–3 are always done through command pipe 0–3, respectively. However, the access to block 4 is not fixed and can be through any of the command pipes. You must only change this field when there is no ongoing block 4 access. A special round-robin arbitration scheme snoops the availability of a command pipe during block 4 access. If any of the command pipes are idle during the first read request to block 4, the ownership of that command pipe gets shared between block 4 and the respective block. If none of the command pipes are idle during a block 4 read request, block 4 gets associated with each of the command pipes in round-robin fashion. When a command pipe acquires ownership of block 4, it keeps that ownership until all the commands to block 4 from all the masters are served. 000b - Block 4 access is always through pipe0 001b - Block 4 access is always through pipe1 010b - Block 4 access is always through pipe2 011b - Block 4 access is always through pipe3

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1xxb - Block 4 access can be through any of the command pipes, based on which command pipe is available for block 4 access
0 DERR_SUP	Data Error Suppression See the Embedded Flash Memory configuration information or system memory map for which flash memory blocks are affected by this field. 0b - Reports ECC events on data flash memory accesses 1b - Single-bit and multi-bit ECC events on data flash memory accesses are suppressed

22.6.7 Platform Flash Memory Access Protection (PFAPR)

Offset

Register	Offset
PFAPR	14h

Function

Controls read accesses to the flash memory array.

See the chip-specific PFLASH information for details about the actual masters available on the chip.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	M0AP		M1AP		M2AP		M3AP		M4AP		M5AP		M6AP		M7AP	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	M8AP		M9AP		M10AP		M11AP		M12AP		M13AP		M14AP		M15AP	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-30: M0AP 29-28: M1AP 27-26: M2AP	Master n Access Protection Controls whether read accesses to the flash memory are allowed based on the master ID of a requesting master. These fields are initialized by hardware reset. The field M4'd3AP is reserved.

Table continues on the next page...

Field	Function
25-24: M3AP 23-22: M4AP 21-20: M5AP 19-18: M6AP 17-16: M7AP 15-14: M8AP 13-12: M9AP 11-10: M10AP 9-8: M11AP 7-6: M12AP 5-4: M13AP 3-2: M14AP 1-0: M15AP	x0b - This master cannot perform any read accesses x1b - This master can perform read accesses

22.6.8 Platform Flash Memory Program Erase Address Logical (PFCPGM_PEADR_L)

Offset

Register	Offset
PFCPGM_PEADR_L	300h

Function

Provides the flash memory address to be programmed, or the location of the sector or block to be erased through main flash memory (pgm/erase) queue. Write access to this register is domain-ID aware.

The respective bus master must have program/erase permission to the flash memory address written to this register. Otherwise a transfer error results. For further information on flash memory address restrictions see the XRDC chapter.

A write to this register is managed via three-cycle access. Before updating the register, you must ensure that no ongoing high-voltage operation is executing through the flash memory main queue.

Unauthorized flash memory address writes result in a transfer error.

Writes to this register during an ongoing high-voltage operation (initiated through the flash memory main queue) or during express program operation are ignored.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PEADR_L															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PEADR_L															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PEADR_L	Program Erase Address Logical Contains the system logical address for flash memory program/erase.

22.6.9 Platform Flash Memory Program Erase Address Physical (PFCPGM_PEADR_P)

Offset

Register	Offset
PFCPGM_PEADR_P	304h

Function

Reflects the flash memory block number and offset address corresponding to Platform Flash Memory Program Erase Address Logical (PFCPGM_PEADR_L). This register has the same format as the PEADR register in the Flash Memory chapter—see it for details.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PEADR_P															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PEADR_P															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PEADR_P	Program Erase Address Physical Contains the flash block select and offset address for flash memory program/erase.

22.6.10 Platform Flash Memory Express Program Erase Address Logical (PFCPGM_XPEADR_L)

Offset

Register	Offset
PFCPGM_XPEADR_L	308h

Function

Provides the flash memory address to be programmed using the flash memory express program feature. Write access to this register is domain-ID aware.

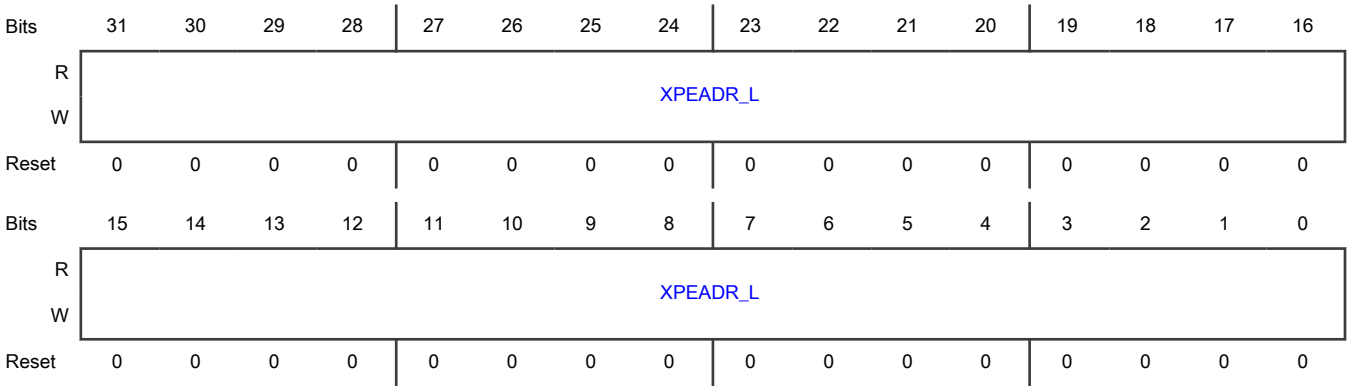
The respective bus master must have program/erase permission to the flash memory address written to this register. Otherwise a transfer error results. See the XRDC chapter for further information on flash memory address restrictions.

A write to this register is managed via three-cycle access. Before updating the register, you must ensure that no ongoing high-voltage operation is executing through the flash memory main queue.

Unauthorized flash memory address writes result in a transfer error.

Writes to this register during an ongoing express program operation are ignored.

Diagram



Fields

Field	Function
31-0 XPEADR_L	Express Program Erase Address Logical Contains the system logical address for express flash program/erase.

22.6.11 Platform Flash Memory Express Program Erase Address Physical (PFCPGM_XPEADR_P)

Offset

Register	Offset
PFCPGM_XPEADR_P	30Ch

Function

Reflects the flash memory block number and offset address corresponding to PFCPGM_XPEADR_L. This register has the same format as the XPEADR register in the Flash Memory chapter—see it for details.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	XPEADR_P															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	XPEADR_P															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 XPEADR_P	Express Program Erase Address Physical Contains the flash memory block select and offset address for flash memory express program/erase.

22.6.12 Block n Sector Program Erase Lock (PFCBLK0_SPELOCK - PFCBLK4_SPELOCK)

Offset

Register	Offset
PFCBLK0_SPELOCK	340h
PFCBLK1_SPELOCK	344h
PFCBLK2_SPELOCK	348h
PFCBLK3_SPELOCK	34Ch
PFCBLK4_SPELOCK	350h

Function

Provides a way to protect sectors from being modified. Sector protection is available on the last 256 KB of every block (for 256 KB blocks, all sectors are available for protection). Each lock bit can be associated with a particular domain ID by writing 1 to the appropriate bit in PFCBLKn_SETSLOCK[SETSLCK]. After the lock is acquired, only a master having the same domain ID can modify the corresponding lock bit. If the corresponding PFCBLKn_SETSLOCK[SETSLCK] bit is not equal to 1, any master can modify the appropriate SLCK bit. If a lock bit is already acquired by a particular domain ID, any effort to modify (1 to 0, or 0 to 1) the lock bit by a master with a different domain ID results in a transfer error.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SLCK															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SLCK															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-0 SLCK	Sector Lock Locks selected sector. If vector bit value = 0, the sector is available for program and erase operations. If vector bit value = 1, the sector is locked and not available for program and erase operations.

22.6.13 Block UTEST Sector Program Erase Lock (PFCBLKU_SPELOCK)

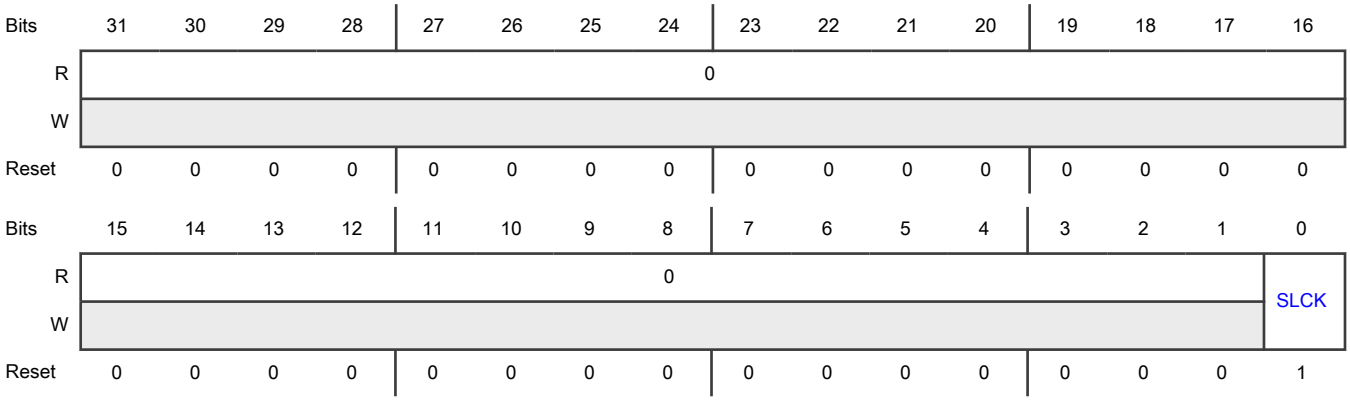
Offset

Register	Offset
PFCBLKU_SPELOCK	358h

Function

Provides a way to protect sectors from being modified. Sector protection is available on the last 256 KB of every block (for 256 KB blocks, all sectors are available for protection). Each lock bit can be associated with a particular domain ID by writing 1 to PFCBLKU_SETSLOCK[SETSLCK]. After the lock is acquired, only a master having the same domain ID can modify the corresponding lock bit. If the corresponding PFCBLKU_SETSLOCK[SETSLCK] bit is not equal to 1, any master can modify the SLCK bit. If a lock bit is already acquired by a particular domain ID, any effort to modify (1 to 0, or 0 to 1) the lock bit by a master with a different domain ID results in a transfer error.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 SLCK	Sector Lock Locks selected sector. If vector bit value = 0, the sector is available for program and erase operations. If vector bit value = 1, the sector is locked and not available for program and erase operations.

22.6.14 Block n Super Sector Program Erase Lock (PFCBLK0_SSPELOCK - PFCBLK3_SSPELOCK)

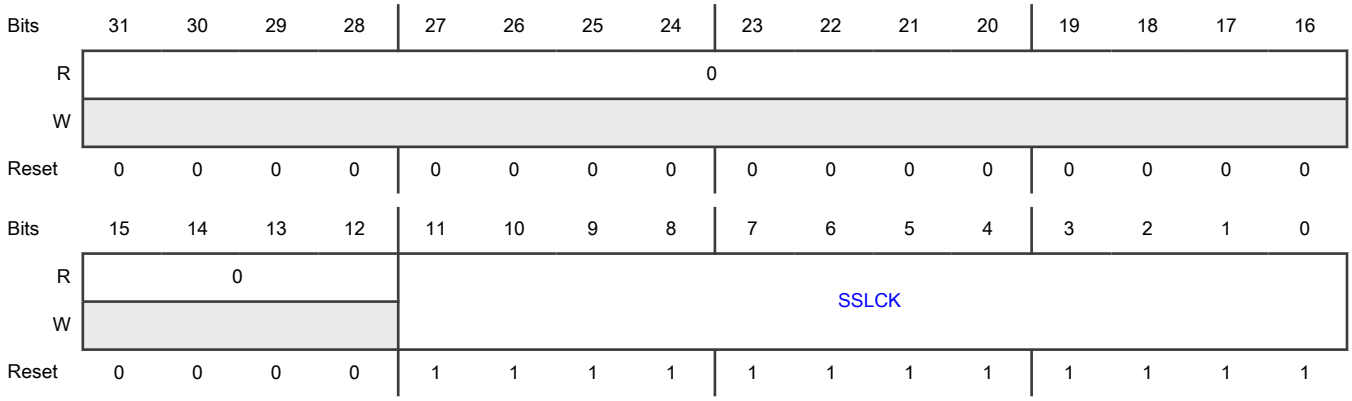
Offset

Register	Offset
PFCBLK0_SSPELOCK	35Ch
PFCBLK1_SSPELOCK	360h
PFCBLK2_SSPELOCK	364h
PFCBLK3_SSPELOCK	368h

Function

Provides a way to protect super sectors from being modified. Super sector protection is available on block space larger than 256 KB. For 256 KB blocks, this register is not applicable. For 512 KB blocks, the first half of the block is protected with super sector granularity. For 1 MB blocks, the first 768 KB is protected with super sector granularity. Each lock bit can be associated with a particular domain ID by writing 1 to the appropriate bit in PFCBLKn_SSETSLOCK[SSETSLCK]. After the lock is acquired, only a master having the same domain ID can modify the corresponding lock bit. If the corresponding PFCBLKn_SSETSLOCK[SSETSLCK] bit is not equal to 1, any master can modify the SSLCK bit. If a lock bit is already acquired by a particular domain ID, any effort to modify (1 to 0, or 0 to 1) the lock bit by a master with a different domain ID results in a transfer error.

Diagram



Fields

Field	Function
31-12 —	Reserved
11-0 SSLCK	Super Sector Lock Locks selected super sector. If vector bit value = 0, the super sector is available for program and erase operations. If vector bit value = 1, the super sector is locked and not available for program and erase operations.

22.6.15 Block n Set Sector Lock (PFCBLK0_SETSLOCK - PFCBLK4_SETSLOCK)

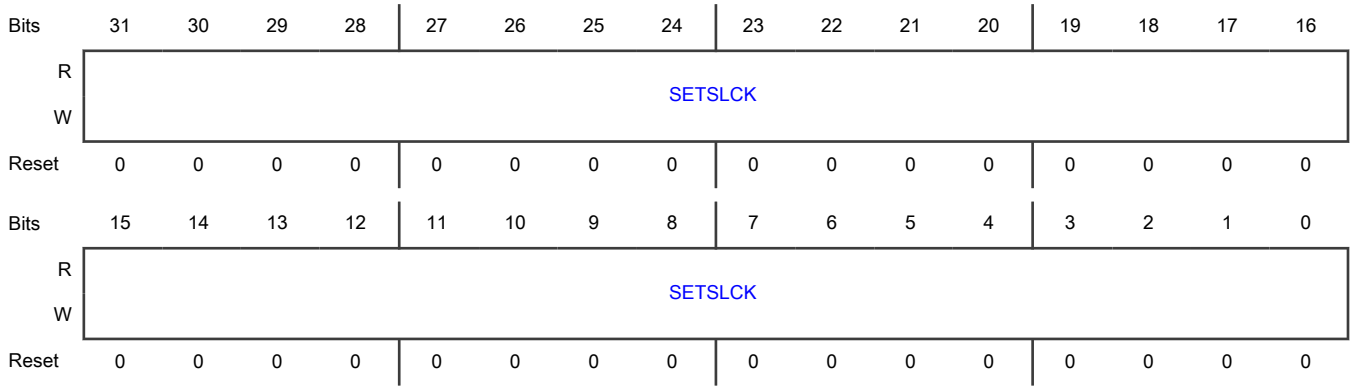
Offset

Register	Offset
PFCBLK0_SETSLOCK	380h
PFCBLK1_SETSLOCK	384h
PFCBLK2_SETSLOCK	388h
PFCBLK3_SETSLOCK	38Ch
PFCBLK4_SETSLOCK	390h

Function

Provides a mechanism for the masters to gain the ownership of the corresponding PFCBLKn_SPELOCK lock bit based on domain id . After it is equal to 1, the bit is returned to 0 at next reset. If any SETSLOCK bit is not equal to 1, the corresponding LOCK bit can be modified by any master. If a bit is already acquired by a particular domain ID, any effort to modify the lock bit by a master with a different domain ID is ignored without transfer error.

Diagram



Fields

Field	Function
31-0 SETSLCK	If the vector bit value = 0, the corresponding lock bit is not owned by any master. If the vector bit value = 1, the lock bit is owned by the masters having the domain ID stored in PFCBLKn_LOCKMASTER_Sm.

22.6.16 Block UTEST Set Sector Lock (PFCBLKU_SETSLOCK)

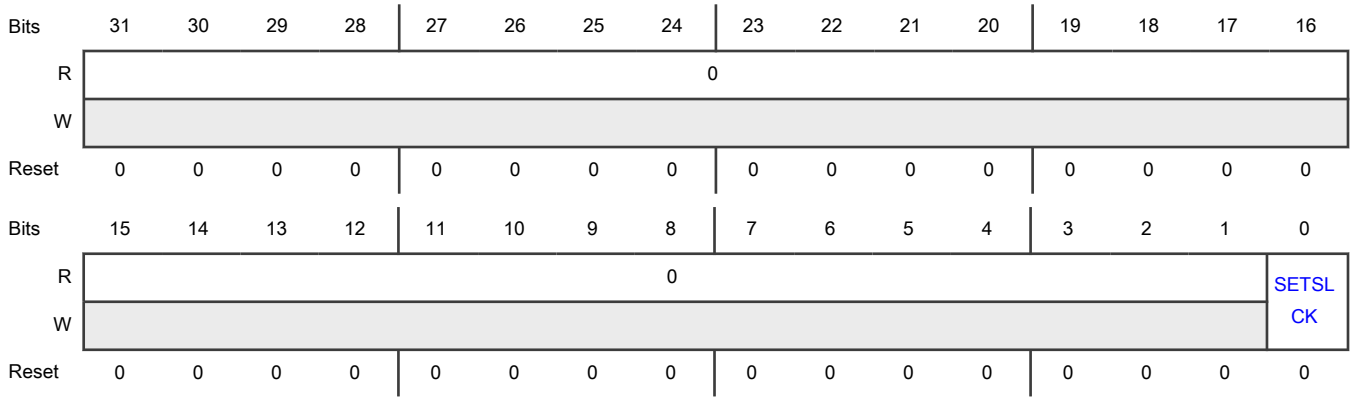
Offset

Register	Offset
PFCBLKU_SETSLOCK	398h

Function

Provides a mechanism for the masters to gain ownership of the corresponding PFCBLKU_SPELOCK lock bit based on domain id. After it is equal to 1, the bit is returned to 0 at next reset. If any SETSLOCK bit is not equal to 1, the corresponding LOCK bit can be modified by any master. If a bit is already acquired by a particular domain ID, any effort to modify the lock bit by a master with a different domain ID is ignored without transfer error.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 SETSLCK	Set Sector Lock Locks selected sector. If vector bit value = 0, the corresponding lock bit is not owned by any master. If vector bit value = 1, the lock bit is owned by the masters having the domain ID stored in PFCBLKn_LOCKMASTER_SSm.

22.6.17 Block n Set Super Sector Lock (PFCBLK0_SSETSLOCK - PFCBLK3_SSETSLOCK)

Offset

Register	Offset
PFCBLK0_SSETSLOCK	39Ch
PFCBLK1_SSETSLOCK	3A0h
PFCBLK2_SSETSLOCK	3A4h
PFCBLK3_SSETSLOCK	3A8h

Function

Provides a mechanism for the masters to gain ownership of the corresponding PFCBLKn_SPELOCK lock bit based on domain id. After it is equal to 1, the bit is returned to 0 at next reset. If any SSETSLOCK bit is not equal to 1, the corresponding LOCK bit can be modified by any master. If a bit is already acquired by a particular domain ID, any effort to modify the lock bit by a master with a different domain ID is ignored without transfer error.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				SSETSLCK											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-12 —	Reserved
11-0 SSETSLCK	Set Super Sector Lock Locks selected super sector. If vector bit value = 0, the corresponding lock bit is not owned by any master. If vector bit value = 1, the lock bit is owned by the masters having the domain ID stored in Block a Lock Master Sector b (PFCBLK0_LOCKMASTER_S0 - PFCBLK4_LOCKMASTER_S7).

22.6.18 Block a Lock Master Sector b (PFCBLK0_LOCKMASTER_S0 - PFCBLK4_LOCKMASTER_S7)

Offset

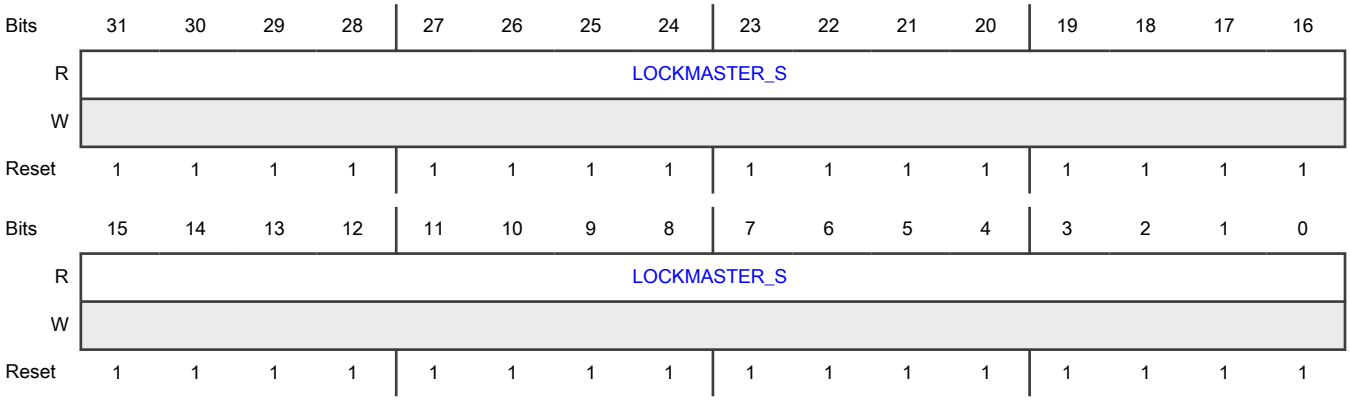
For a = 0 to 4; b = 0 to 7:

Register	Offset
PFCBLKa_LOCKMASTER_Sb	3C0h + (a × 20h) + (b × 4h)

Function

Provides the domain ID information of the master currently acquiring the lock bit. The domain ID is represented by an 8-bit field. This is a read-only register.

Diagram



Fields

Field	Function
31-0	Block a Lock Master Sector b Contains domain ID of the master currently acquiring the lock bit.

Table continues on the next page...

Field	Function
LOCKMASTER_S	PFCBLK0_LOCKMASTER_S0[LOCKMASTER_S[7:0]] holds the domain ID information of PFCBLK0_SPELOCK[0]. PFCBLK0_LOCKMASTER_S0[LOCKMASTER_S[15:8]] holds the domain ID information of PFCBLK0_SPELOCK[1], and so on in incremental order.

22.6.19 Block UTEST Lock Master Sector (PFCBLKU_LOCKMASTER_S)

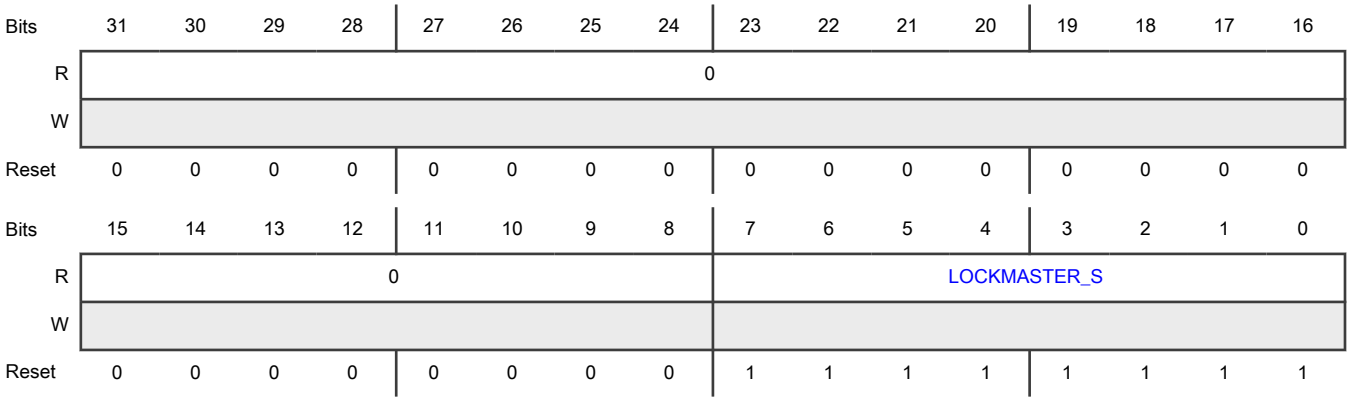
Offset

Register	Offset
PFCBLKU_LOCKMASTER_S	480h

Function

Provides the domain ID information of the master currently acquiring the lock bit. The domain ID is represented by an 8-bit field. This is a read-only register.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 LOCKMASTER_S	Lock Master Sector Contains domain ID of the master currently acquiring the lock bit. PFCBLKU_LOCKMASTER_S[LOCKMASTER_S[7:0]] holds the domain ID information of PFCBLKU_SPELOCK[0].

22.6.20 Block m Lock Master Super Sector n (PFCBLK0_LOCKMASTER_SS0 - PFCBLK3_LOCKMASTER_SS2)

Offset

For a = 0 to 3; b = 0 to 2:

Register	Offset
PFCBLKa_LOCKMASTER_SSb	484h + (a × 10h) + (b × 4h)

Function

Provides the domain ID information of the master currently acquiring the lock bit. The domain ID is represented by an 8-bit field. This is a read-only register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LOCKMASTER_SS															
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LOCKMASTER_SS															
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-0 LOCKMASTER_SS	Block a Lock Master Super Sector b Contains domain ID of the master currently acquiring the lock bit. PFCBLK0_LOCKMASTER_SS0[LOCKMASTER_SS[7:0]] holds the domain ID information of PFCBLK0_SSPELOCK[0]. PFCBLK0_LOCKMASTER_SS0[LOCKMASTER_SS[15:8]] holds the domain ID information of PFCBLK0_SSPELOCK[1], and so on in incremental order.

22.7 Glossary

AHB	Advanced high-performance bus
ECC	Error correcting code
HSE	Hardware security engine
IPS	Internal peripheral system
ID	Identification

Chapter 23
RAM Controller (PRAMC)

23.1 Chip-specific PRAMC information

23.1.1 PRAMC instances and configuration

This chip supports up to three instances of PRAMC.

Table 113. PRAMC instances and configuration

Instance	SRAM array	Applicability
PRAMC_0	SRAM 0	S32K310, S32K311, S32K312, S32K314, S32K322, S32K324, S32K328 S32K338, S32K341, S32K342, S32K344, S32K348, S32K358, S32K356, S32K388, S32K389
PRAMC_1	SRAM 1	S32K322, S32K324, S32K328, S32K338, S32K341, S32K342, S32K344, S32K348, S32K358, S32K356, S32K388, S32K389
PRAMC_2	SRAM 2	S32K338, S32K358, S32K356, S32K388, S32K389, S32K348, S32K328
PRAMC_3	SRAM 3	S32K389

See chapter "Memory and Memory Interfaces" for details on SRAM size.

23.1.2 RAM initialization

After chip power-on reset, you must initialize RAM to a known value using a 64-bit master before 32-bit masters can read or write to RAM. If you do not initialize RAM this way, any attempt to execute a 32-bit read or write access terminates with an uncorrectable ECC error event on this chip.

The size of System RAM for code execution must be at least 64 KB and this size must be identical for all variants.

23.1.3 SRAM0 behavior while XRDC is configured to block access to SRAM0

While XRDC is configured to restrict access to SRAM0, the below behaviors should be considered:

1. In case if XRDC is configured to restrict access to SRAM0 and a read transaction to SRAM0 is performed: To ensure faster SRAM0 accesses, the access is routed to SRAM0 and blocked by PRAMC if XRDC is configured to block access. In such a case, if an error is present on the SRAM0 data, (that is error on data or error injected by EIM), the same is latched by ERM as well. This data is not used by application core since PRAMC gives a bus error in this case.
2. In case if XRDC is configured to restrict access to SRAM0 and a 32-bit write access is performed: Since SRAM is 64-bit wide, so a 32-bit write transaction is performed as a read-modify-write. In the read cycle, again as described in previous scenario, the PRAMC will initiate an error response. In this case also, if there is any error on SRAM0 data bus (i.e., error or error injected via EIM), the same is captured in ERM as well.

NOTE

For any read/write accesses initiated by HSE-B security subsystem to system RAM, the platform must cancel the error injection request on SRAM controller.

23.2 Overview

The RAM controller is an interface between the system bus (AHB-Lite 2.v6) and the integrated system RAM. It converts the protocols between the system bus and the dedicated RAM array interface.

The RAM controller supports one 64-bit AHB interface and an ECC RAM array interface. The AHB port provides a connection to the platform crossbar for direct RAM access from the various crossbar masters.

23.2.1 Block diagram

Shown below in Figure 56 is a simplified block diagram depicting the position of the RAM controller within a typical platform architecture.

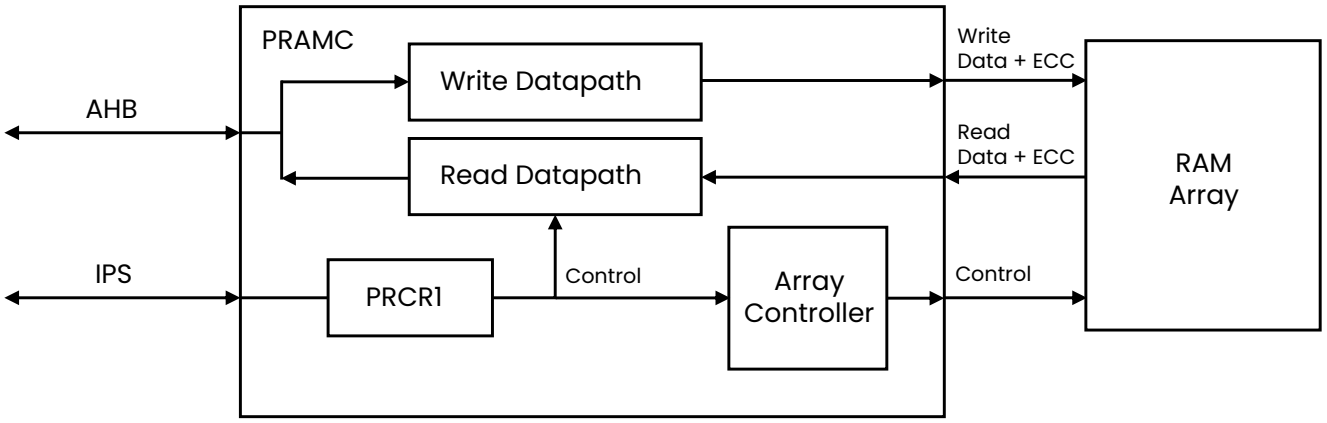


Figure 56. Block diagram

23.2.2 Features

The following list summarizes the key features of the RAM controller:

- System bus supports 64-bit data
- Array interface supports a 64-bit data + 8-bit ECC interface
- Late-write support via 64-bit data + 8-bit ECC storage buffer supports single-cycle write accesses
- Configurable read access timing (zero or one wait state programmable) allowing use in large range of frequency targets
- Read-modify-write operation supports array write size smaller than a double-word
- Hardware EDC after ECC feature to check the ECC logic. Detected errors are connected to the FCCU with configurable error reaction.

23.3 Functional description

This section describes the functions of the RAM controller.

23.3.1 Read and write operations

The RAM controller processes read and write requests to on-chip RAM and provides a register interface for access to performance tuning functions.

23.3.1.1 Reads

Read transfers, of any size, can be configured to complete with a zero or one additional wait state response on the system bus.

23.3.1.2 Optional read wait state

The RAM controller can be optionally programmed to register RAM read data prior to returning it on the system bus. Setting the PRCRx[FT_DIS] field inserts a register in the read data path for use when operating the controller at high frequencies. The state of PRCRx[FT_DIS] field has no effect on writes.

This note is applicable only if the RAM array works on 1/2 system clock. A random, initial access takes two clock cycles (2T) to complete if PRCRx[FT_DIS]=0, and a WRAP4 burst with burst optimization disabled will have an access time of 2-2-2-2. A random, initial access takes three clock cycles (3T) to complete if PRCRx[FT_DIS]=1, and a WRAP4 burst with burst optimization disabled has an access time of 3-2-2-2.

23.3.2 Writes

This section discusses various write operations of the RAM controller.

23.3.2.1 64-bit writes

Aligned 64-bit writes execute in a single AHB data phase cycle, resulting in zero wait states on back-to-back writes of these sizes. If, during the data phase of a write, a read is requested on the AHB, the write is registered in the late-write buffer, enabling the read to take place without a wait state. The valid buffered or late-write data is stored on the next available array address phase.

Back-to-back 64-bit writes execute slightly differently. The first write bypasses the late-write buffer—the write data is stored directly to the array in the same cycle in which it is valid on the AHB.

23.3.2.2 Unaligned writes

The RAM controller is compliant with the AMBA-AHB2.v6 Extensions specification with regard to byte strobes. The size of the transfer is sufficient to cover all bytes being written, and covers more bytes in the case of an unaligned transfer. The address of the transfer is rounded down to the nearest boundary of the size of the transaction.

NOTE

Unaligned writes always generate read-modify-write operations in the RAM controller in order to preserve the validity of the ECC codeword.

23.3.3 Clocking

The following table describes clocks used in this module.

Table 114. Clock sources

Clock name	Description
System clock	The PRAMC runs on a single system clock over the system bus interconnect.

23.3.4 Interrupts

This module has no interrupts.

23.4 External signals

This module has no external signals.

23.5 Initialization/application information

Each memory address must be written to a known value before it is read, to initialize the ECC. This includes reads generated from the read-modify-write operation that occurs when a write transfer of less than 64 bits or an unaligned write is requested. Without first writing an address to a known value, a read from the address will most likely generate an uncorrectable ECC event.

23.6 PRAMC register descriptions

The RAM controller module provides an IPS programming model mapped to a standard on-platform peripheral slot.

NOTE

1. The programming model can only be referenced using a 32-bit (word) access. Attempted references using different access sizes or to undefined (reserved) addresses generate an IPS error termination.
2. Attempted updates to the PRAMC programming model while a PRAMC operation is in progress will result in non-deterministic behavior. Software architect must be such as to avoid this scenario by ensuring that PRAMC configuration changes are made only during system boot or when no master is accessing this RAM.

23.6.1 PRAMC memory map

PRAMC_0 base address: 4026_4000h

PRAMC_1 base address: 4046_4000h

PRAMC_2 base address: 4046_8000h

PRAMC_3 base address: 4058_8000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Platform RAM Configuration 1 (PRCR1)	32	RW	0000_0100h

23.6.2 Platform RAM Configuration 1 (PRCR1)

Offset

Register	Offset
PRCR1	0h

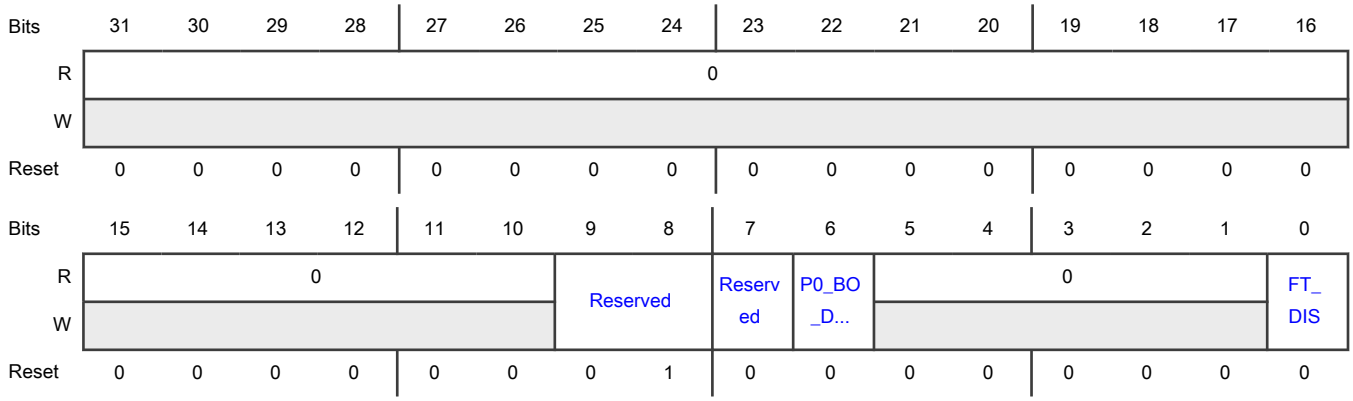
Function

This register is used to specify operation of the RAM controller.

NOTE

This register is accessible only in supervisor mode. Accesses in user mode return a transfer error.

Diagram



Fields

Field	Function
31-10 —	Reserved
9-8 —	Reserved
7 —	Reserved
6 P0_BO_DIS	Port p0 read burst optimization disable 0b - 64-bit WRAP4 read bursts are optimized; the controller returns a 2-1-1-1 response when PRCR1[FT_DIS]=1 1b - 64-bit WRAP4 read bursts are not optimized; the controller returns a 2-2-2-2 response when PRCR1[FT_DIS]=1
5-1 —	Reserved
0 FT_DIS	Flow-through disabled This field defines the AHB response on reads. The state of this field has no impact on the response latency on writes. This bit is cleared by hardware reset. **NOTE** Do not change the FT_DIS bit value while accessing system RAM. Relocate code programming the FT_DIS bit to another memory area, for example, local core memory. 0b - RAM read data is passed directly to the system bus, incurring no additional latency 1b - RAM read data is registered prior to returning on the system bus, incurring one extra cycle of latency

23.7 Glossary

AHB	Arm advanced high-performance bus
DED	Double error detection
DMA	Direct memory access
MUX	Multiplexer
SEC	Single error correction
WRAP4	4-bit wrapping burst

Chapter 24
Clocking

24.1 Introduction

This chapter describes the clocking architecture and includes the following information:

- System clock specifics
- Clock sources
- Clock architecture
- Clock control registers
- Clock monitoring
- Clock gating
- Module clocking

This chapter discusses the clocks generated on the chip. Peripheral-specific protocol clocks are described in the corresponding peripheral chapters.

24.2 Features

- Multiple clock sources supported for clock generation:
  - Fast internal RC oscillator (FIRC)
  - Slow internal RC oscillator (SIRC)
  - Fast external crystal oscillator (FXOSC)
  - Slow external crystal oscillator (SXOSC)

    NOTE

    SXOSC is not available in 100-HDQFP and 48-pin LQFP packages.

  - Phase-locked loop (PLL)
- Frequency-modulated PLL output clock to reduce electromagnetic emissions
- Precise clocks for timers and communication functions
- Glitchless clock switching Clock Generation module (MC_CGM) clock selectors
- System clock progressive clock frequency switching (PCFS)
- Clock monitoring units (CMU_FC, CMU_FM) to check clock integrity
- Core and peripheral clock gating using the Mode Entry module's (MC_ME) partition process configuration registers

24.3 Clocking overview

The S32K3xx clocking architecture consists of multiple:

- Clock sources
- Monitors
- Multiplexers
- Dividers

The blocks in the above bullet list provide the required clocking domains for the different functional blocks.

The sections in the following list show the clocking configuration of the chip:

- Clock source generation: PLL, FXOSC, FIRC, SIRC, and SXOSC
- MC_CGM mux 0 clocks: MC_CGM mux 0 generated clocks (not including EMAC/GMAC clock signals)
- Clockout overview: CLKOUT_STANDBY and CLKOUT_RUN
- Other clocks
- GMAC clocks (S32K388 and S32K389)
- GMAC clocks (S32K328, S32K338, S32K348, S32K356, and S32K358)
- EMAC clocks (S32K344, S32K324, S32K314, S32K322, S32K341 and S32K342)

The figures shown in S32K389 clock system diagram, S32K388 clock system diagram, S32K328, S32K338, S32K348, S32K356, and S32K358 clock system diagram, S32K344, S32K324, S32K314, S32K322, S32K341 and S32K342 clock system diagram, S32K312 clock system diagram, and S32K310 and S32K311 clock system diagram show the overall clock tree for the different chip variants of the S32K3xx family, which are a combination of the sections mentioned in the bullet list above.

24.3.1 Clock source generation

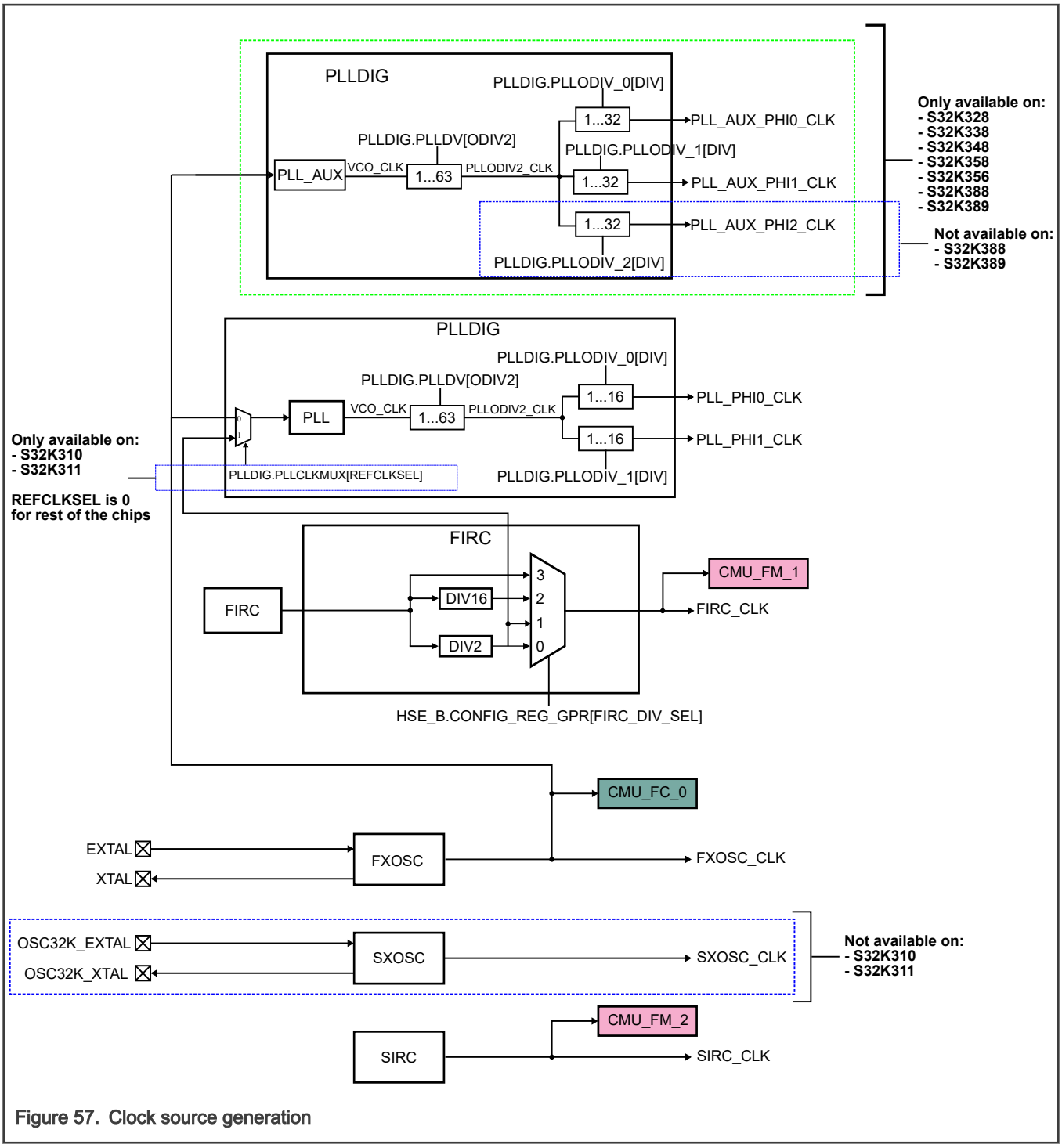


Figure 57. Clock source generation

24.3.2 MC_CGM mux 0 clocks

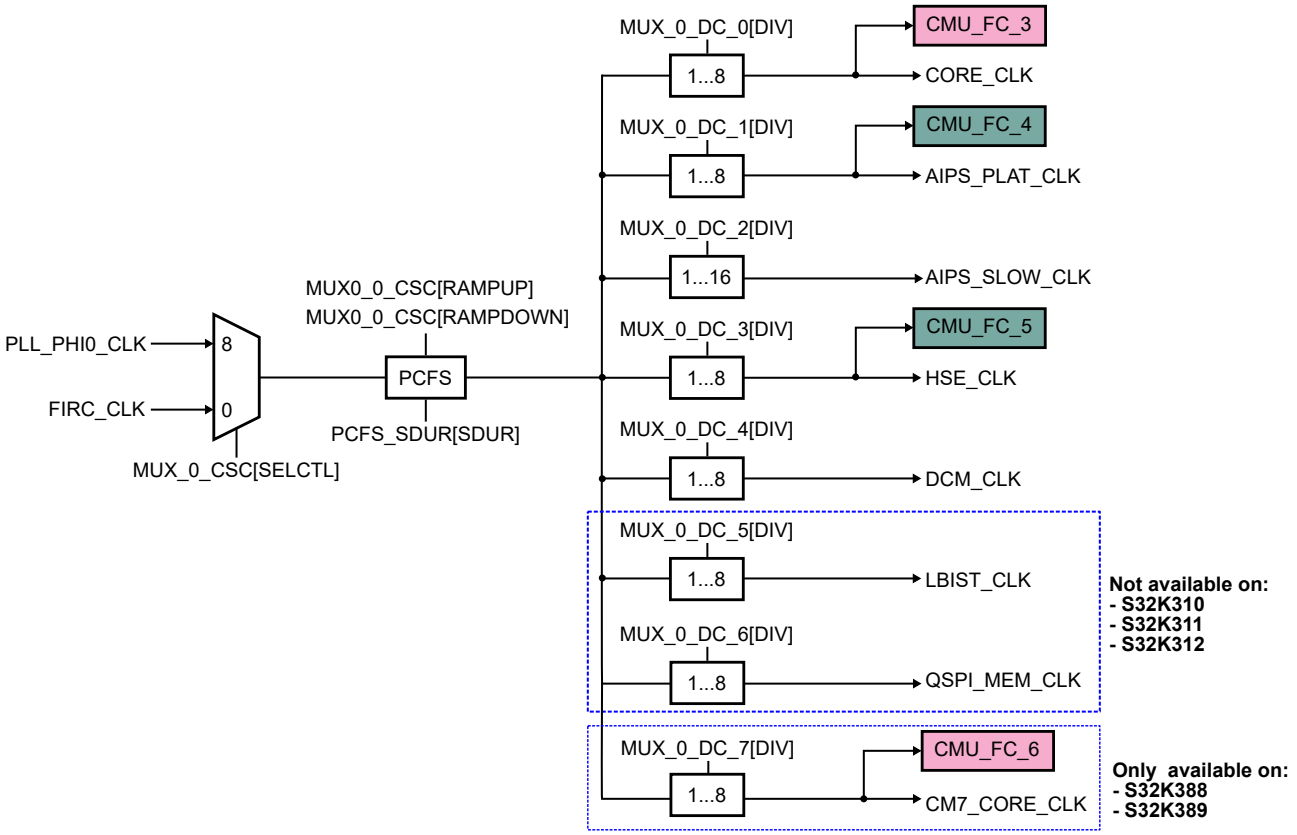


Figure 58. MC_CGM mux 0 clocks

24.3.3 Clockout overview

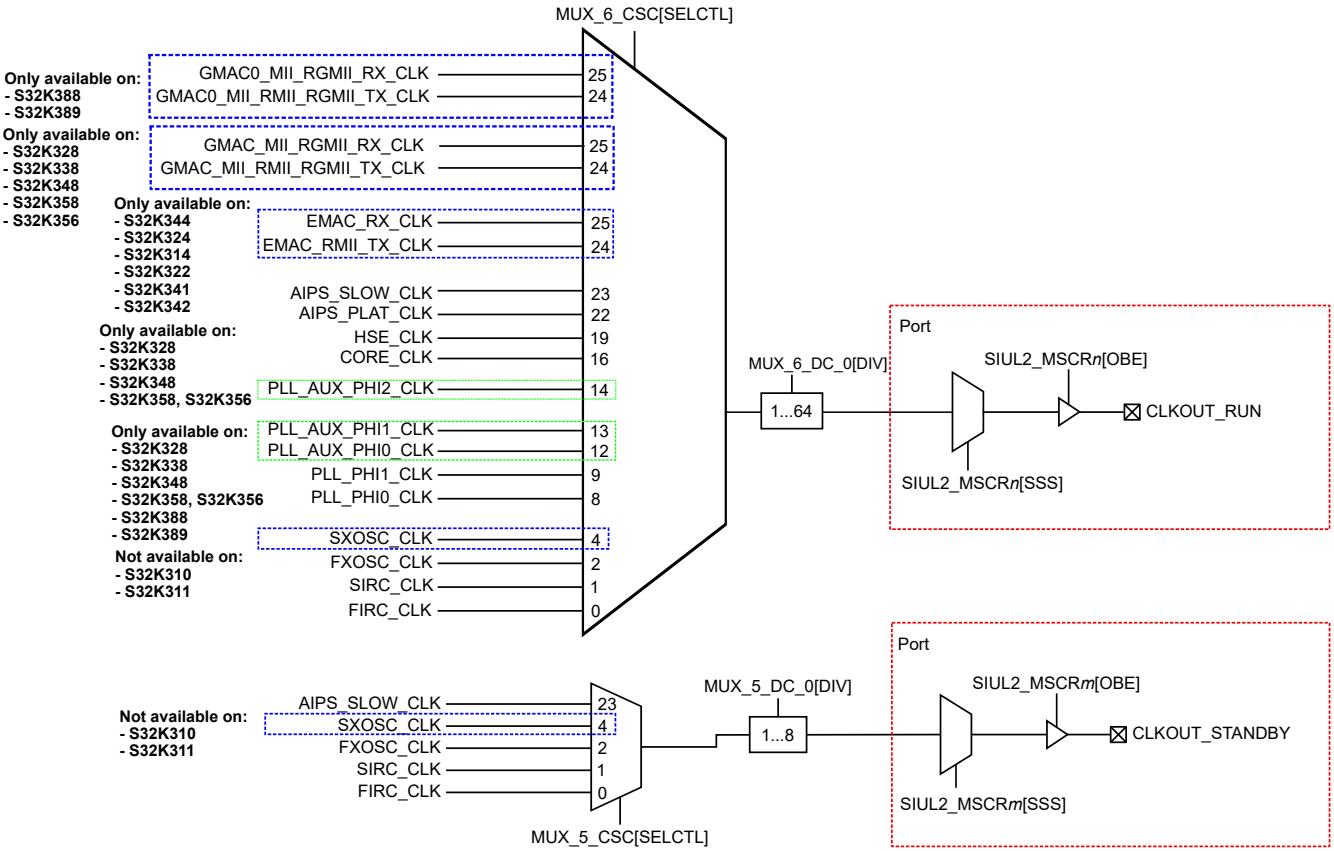


Figure 59. Clockout overview

NOTE

CLKOUT_RUN is not available during Standby mode.

24.3.3.1 SIUL2 options for CLKOUT_RUN

Table 115. SIUL2 options for CLKOUT_RUN

Source	Destination	Port	MSCR*n*	MSCR fields		
				OBE	IBE	SSS
MUX_6_DC_0 divider output	CLKOUT_RUN	PTB5	37	1	0	0101b
		PTD10	106	1	0	0110b
		PTD14	110	1	0	0111b

24.3.3.2 SIUL2 options for CLKOUT_STANDBY

Table 116. SIUL2 options for CLKOUT_STANDBY

Source	Destination	Port	MSCR*m*	MSCR fields		
				OBE	IBE	SSS
MUX_5_DC_0 divider output	CLKOUT_STANDBY	PTA12	12	1	0	0011b
		PTE10	138	1	0	0101b

24.3.4 Other clocks

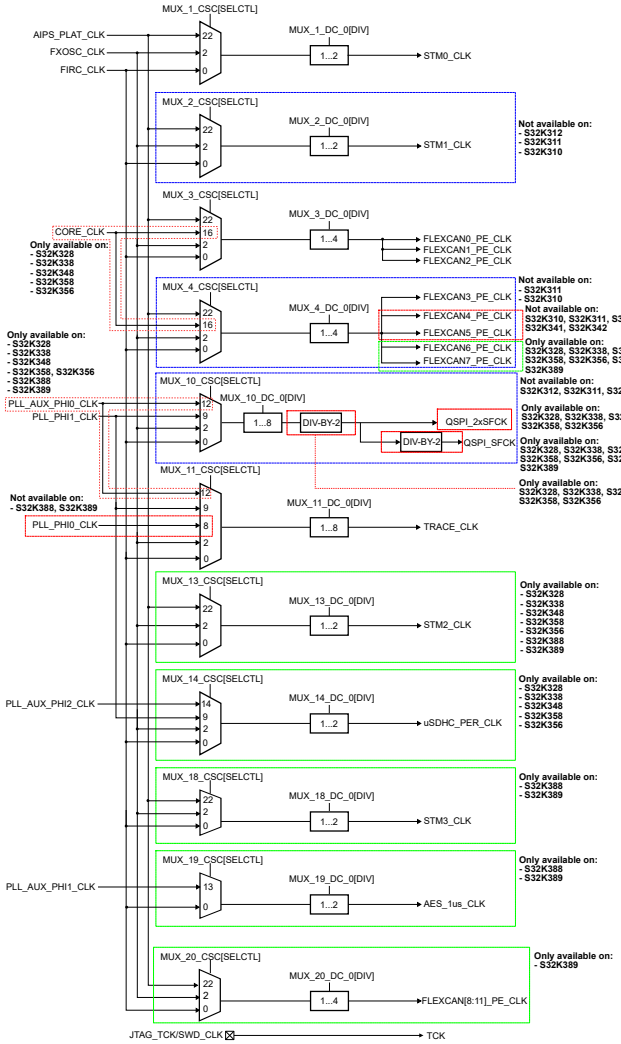


Figure 60. Other clocks

24.3.5 GMAC clocks (S32K388 and S32K389)

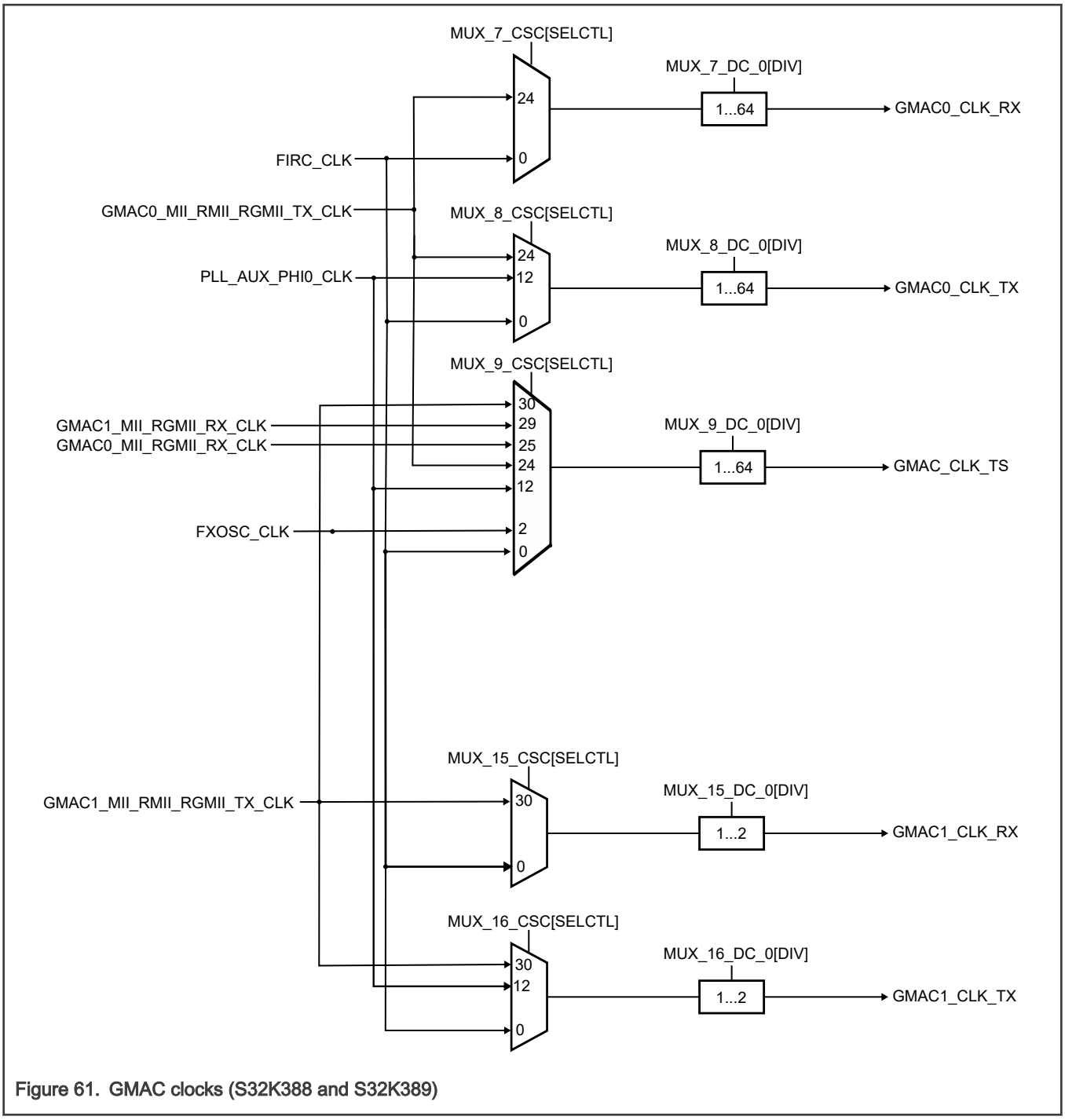


Figure 61. GMAC clocks (S32K388 and S32K389)

24.3.6 GMAC clocks (S32K328, S32K338, S32K348, S32K356, and S32K358)

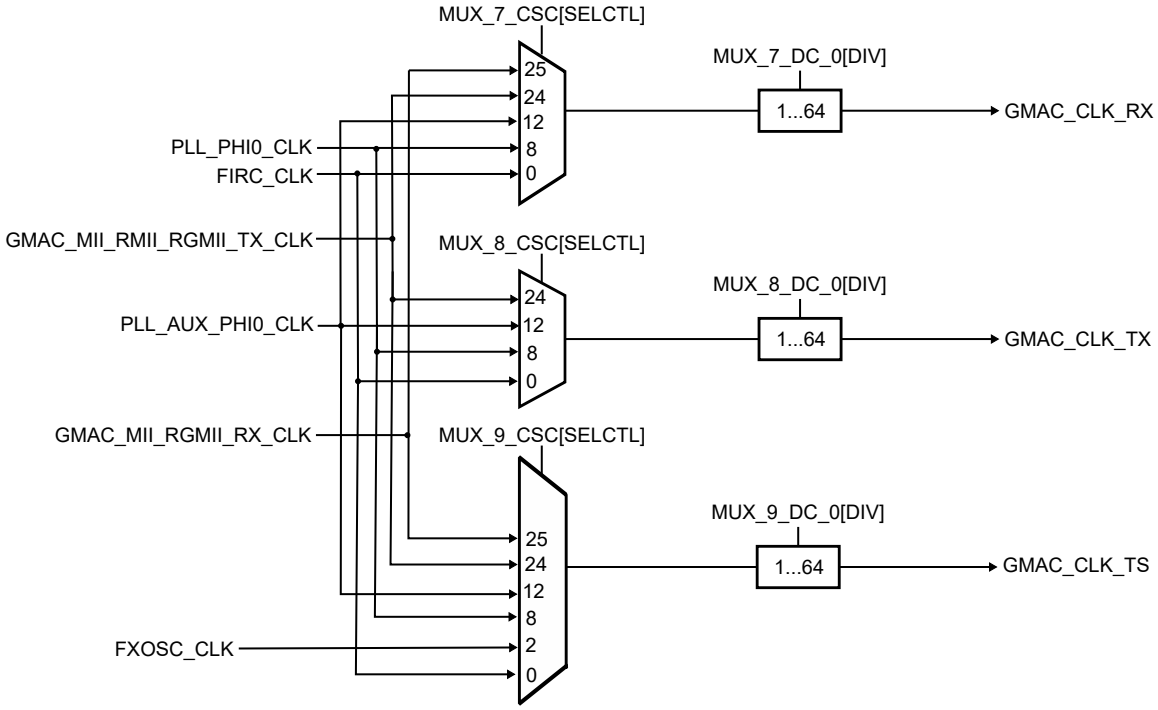


Figure 62. GMAC clocks (S32K328, S32K338, S32K348, S32K356, and S32K358)

24.3.7 EMAC clocks (S32K344, S32K324, S32K314, S32K322, S32K341 and S32K342)

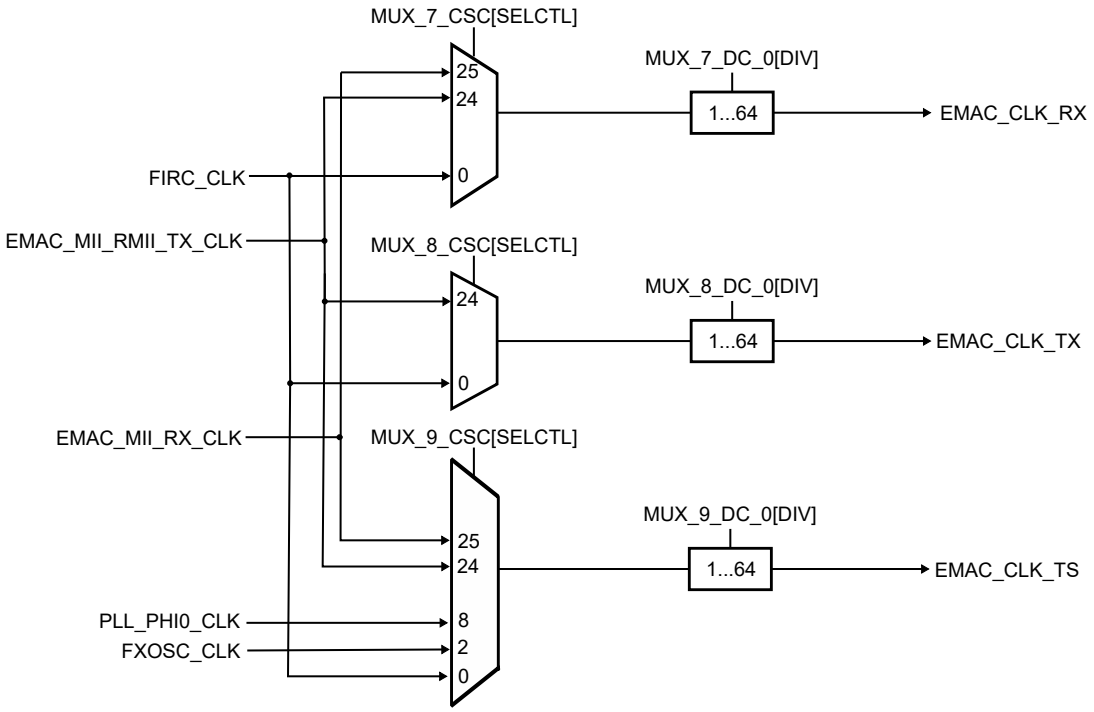


Figure 63. EMAC clocks (S32K344, S32K324, S32K314, S32K322, S32K341 and S32K342)

24.3.8 S32K389 clock system diagram

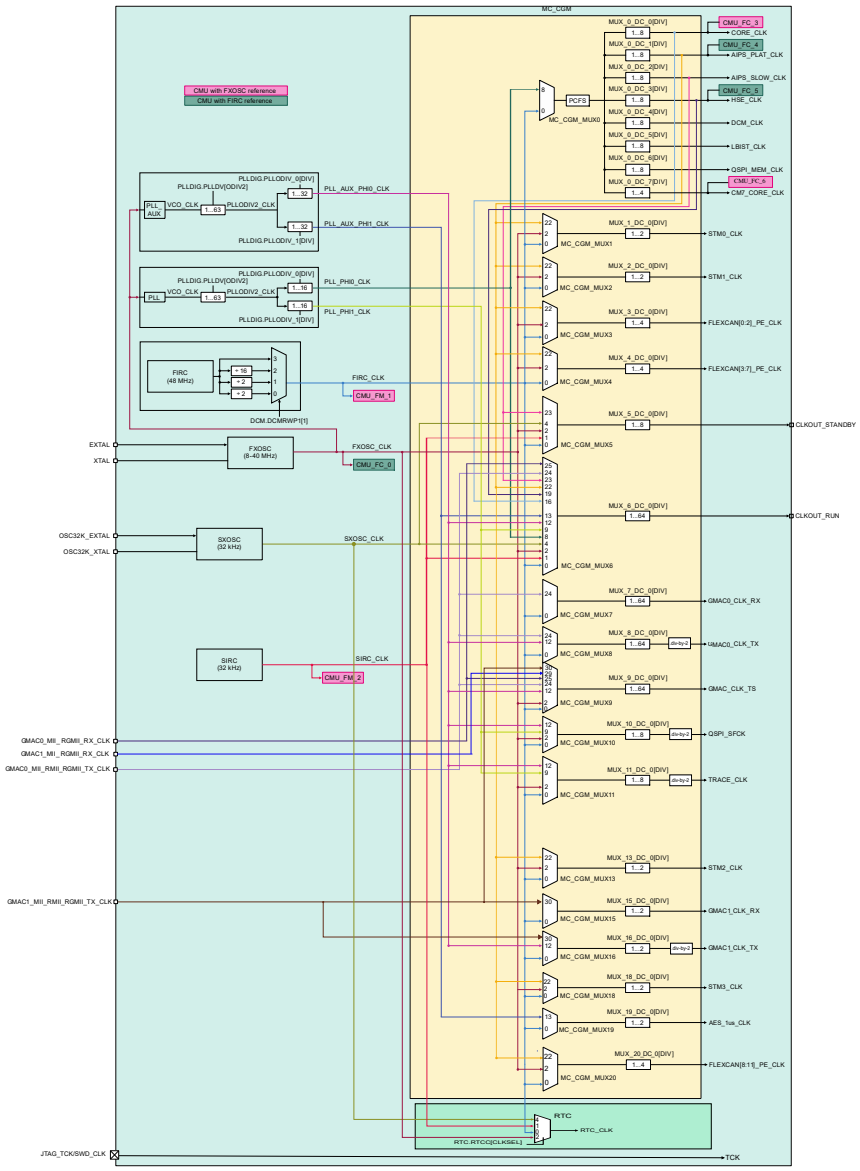


Figure 64. S32K389 clock system diagram

24.3.9 S32K388 clock system diagram

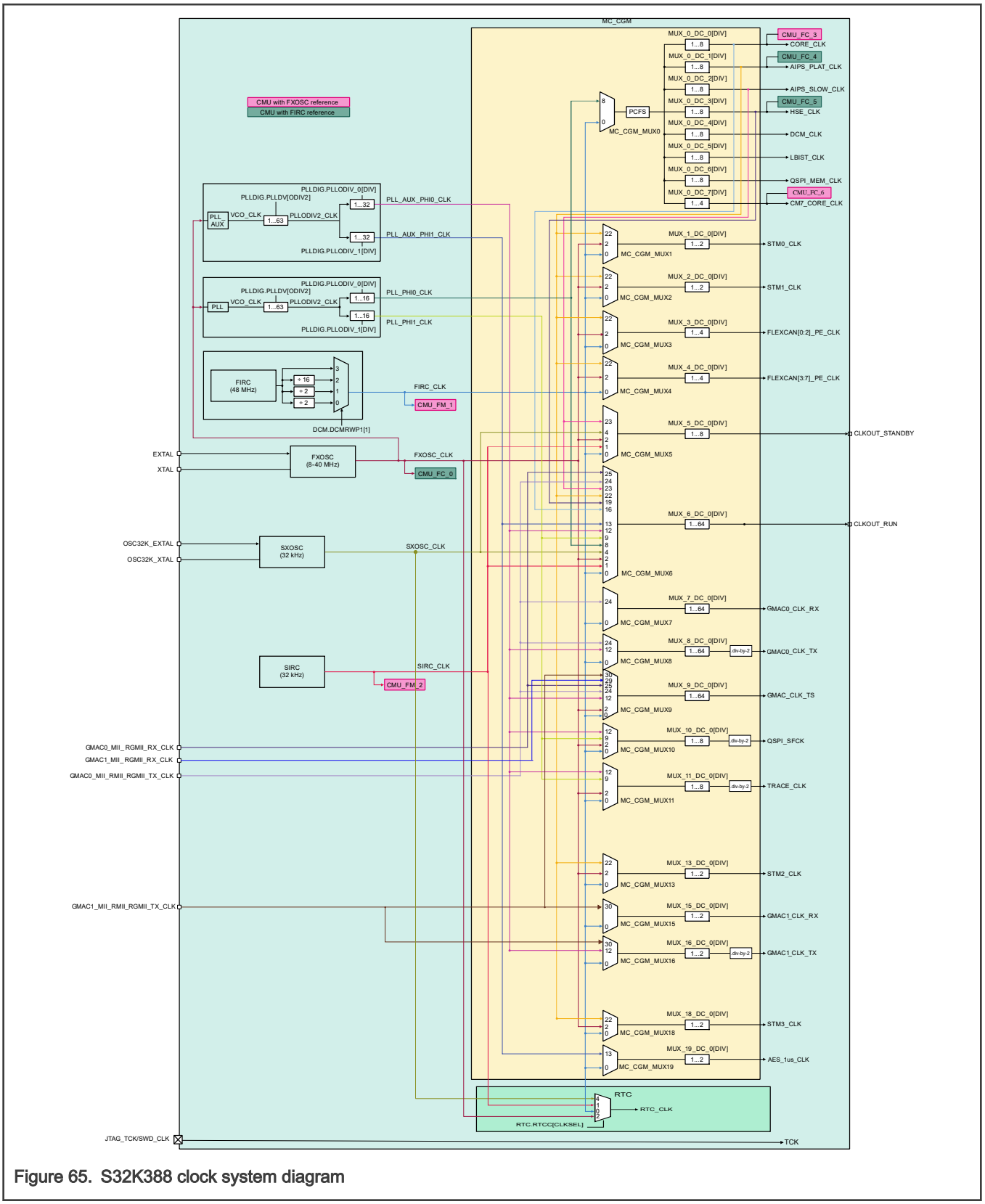


Figure 65. S32K388 clock system diagram

24.3.10 S32K328, S32K338, S32K348, S32K356, and S32K358 clock system diagram

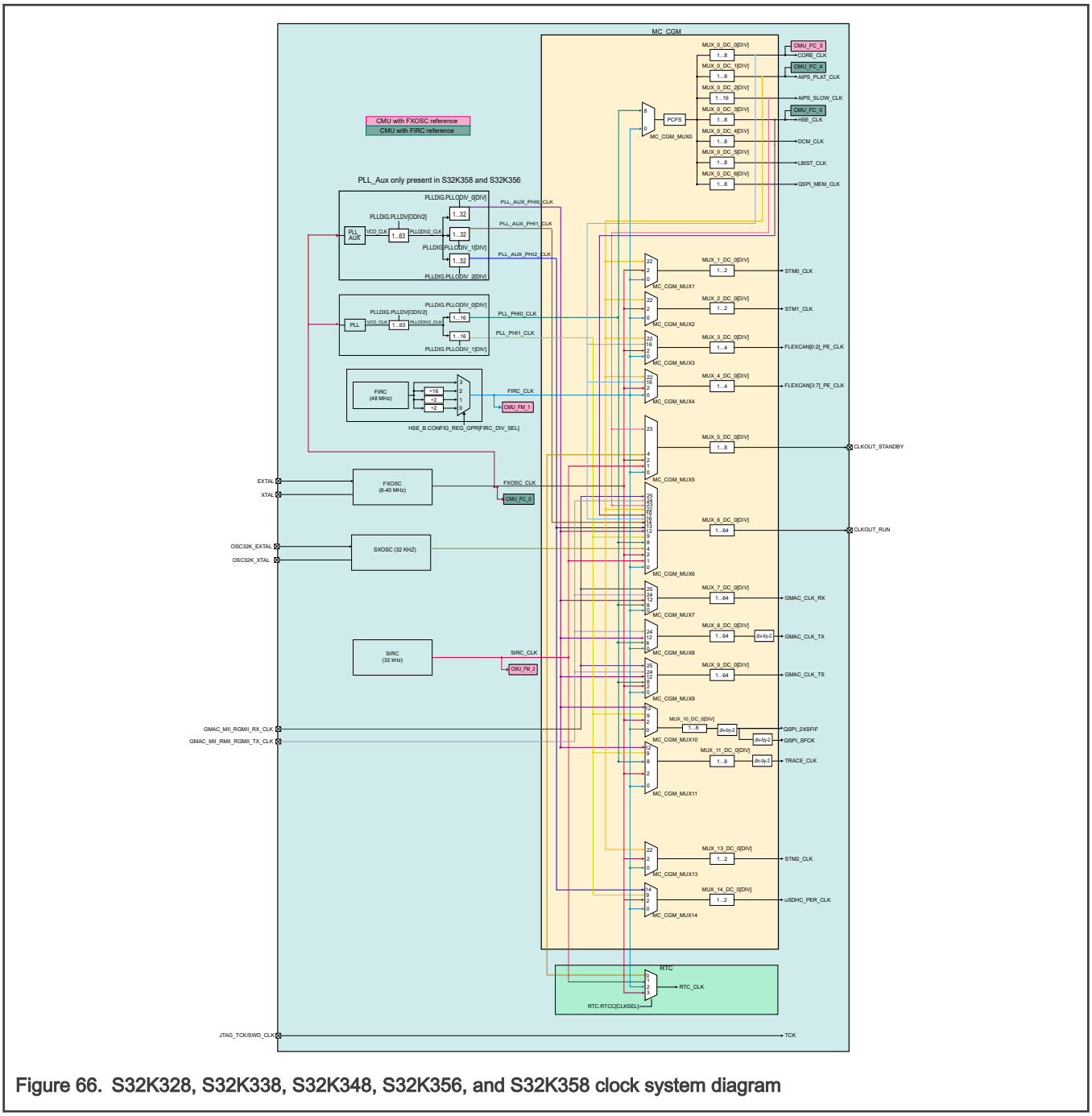


Figure 66. S32K328, S32K338, S32K348, S32K356, and S32K358 clock system diagram

24.3.11 S32K344, S32K324, S32K314, S32K322, S32K341 and S32K342 clock system diagram

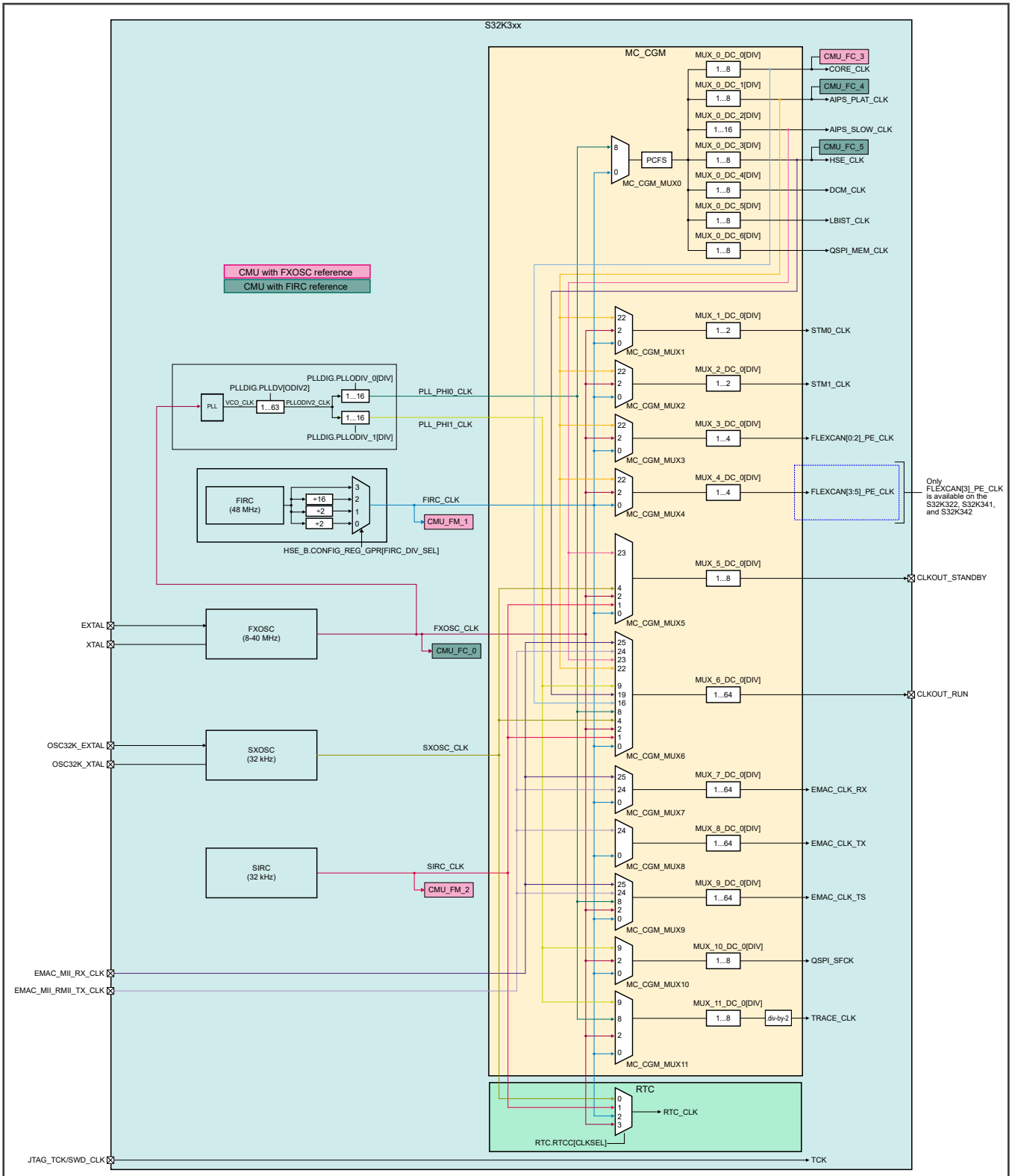


Figure 67. S32K344, S32K324, S32K314, S32K322, S32K341, and S32K342 clock system diagram

24.3.12 S32K312 clock system diagram

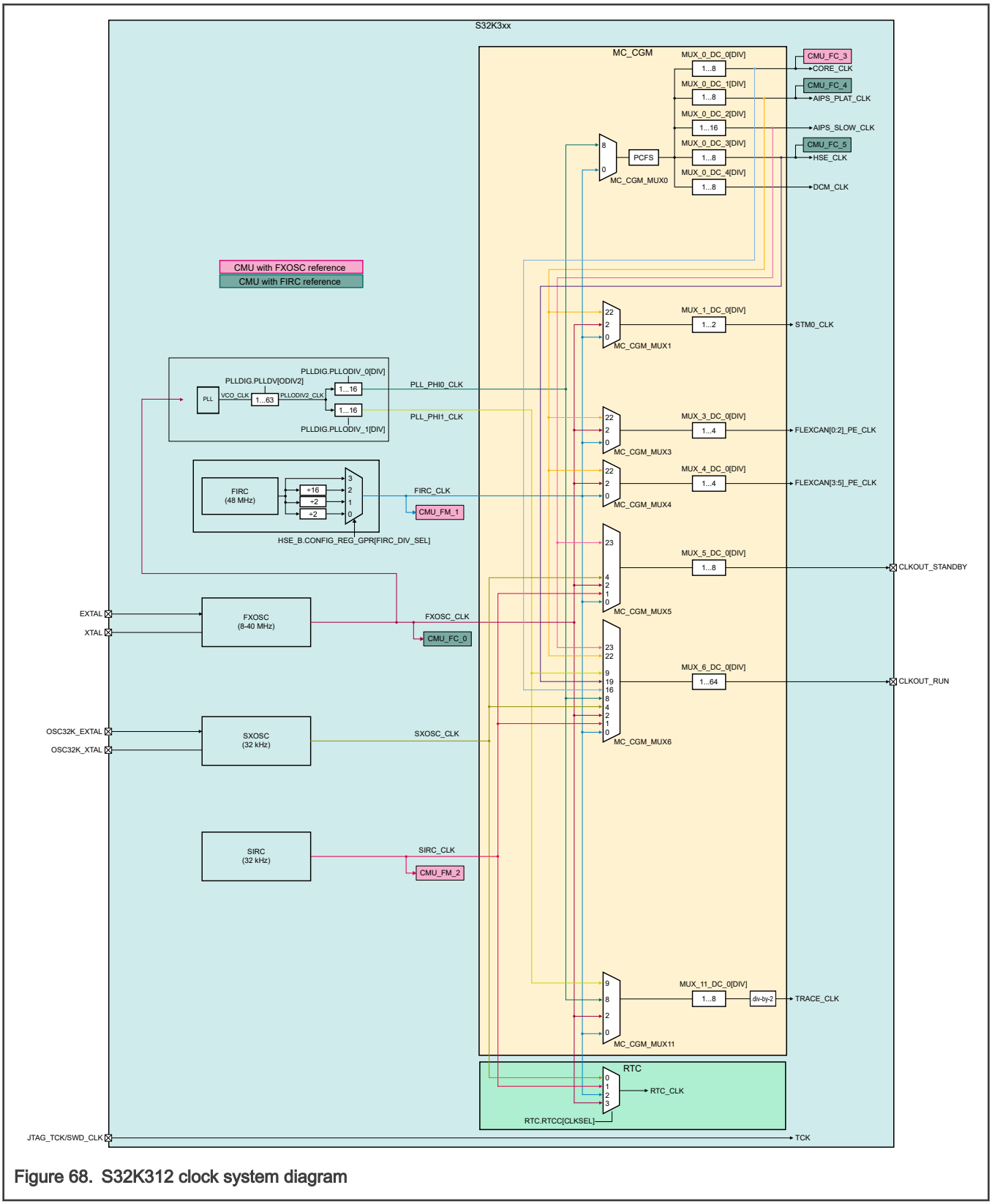


Figure 68. S32K312 clock system diagram

24.3.13 S32K310 and S32K311 clock system diagram

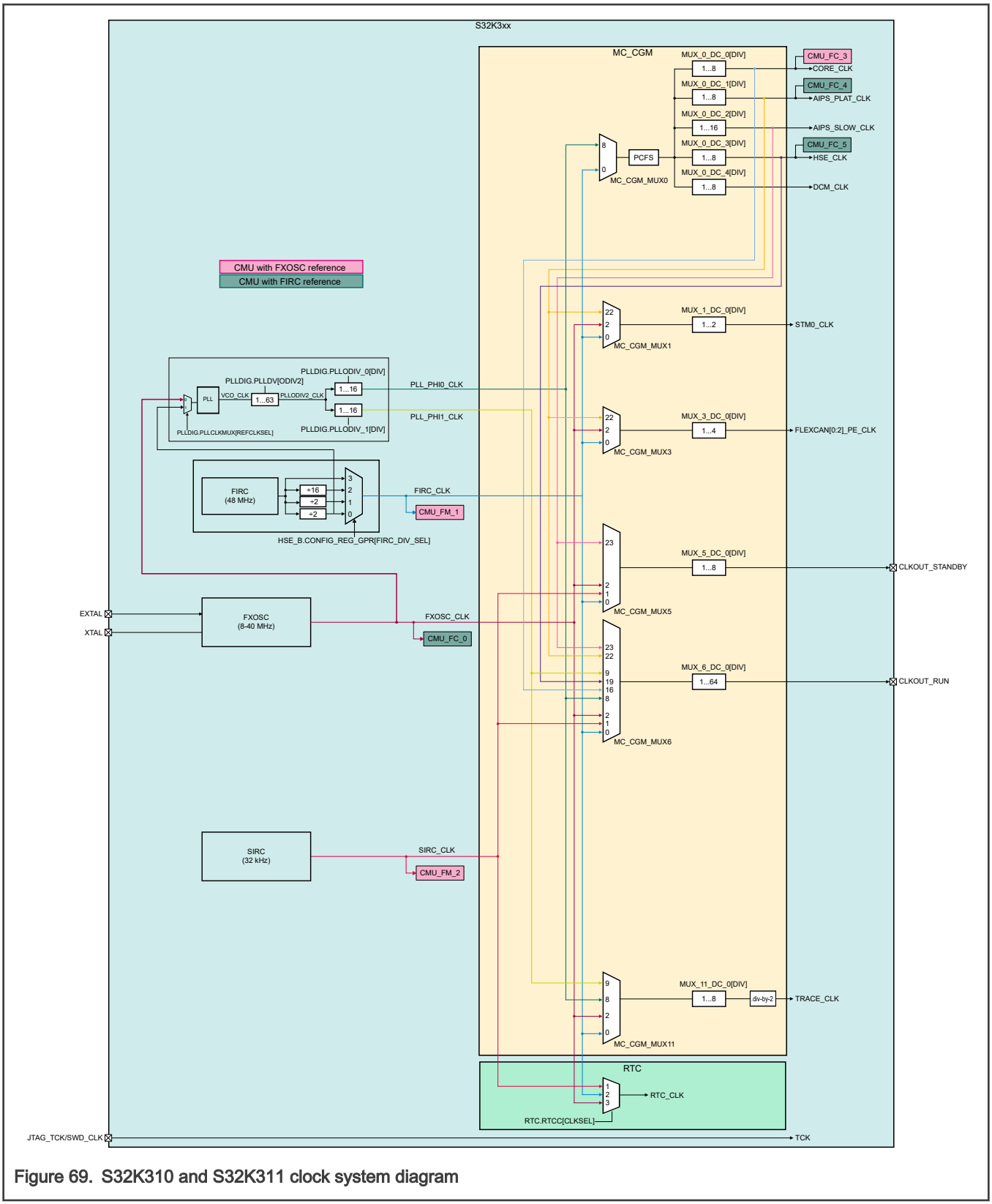


Figure 69. S32K310 and S32K311 clock system diagram

24.4 Clock sources

24.4.1 Introduction

The chip contains the following clock sourcing modules:

- FIRC
  - FIRC_CLK is the default system clock source.
- SIRC
- PLL
- FXOSC
- SXOSC (not available in 100-HDQFP and 48-pin LQFP packages)

The following list shows some of the clock system features:

- All clock sources support application software configurability for enabling or disabling. [3]
- All clock sources, except SXOSC_CLK, are initialized to their default state on functional reset.
- The SXOSC_CLK supports RTC applications across functional reset and is reset on destructive reset.

Only SIRC_CLK and FIRC_CLK are enabled out of reset and are enabled on any functional reset. The other clock sources are disabled on reset.

24.4.1.1 Chip clock sources

Table 117. Chip clock sources

Clock source	Divider	Default state	Reset	Uses
FIRC_CLK (48 MHz)	1, 2, 16	On	POR (enabled on functional and destructive reset) POR assertion - FIRC_CLK disabled asynchronously POR deassertion - FIRC_CLK enabled	• Boot clock • Default system clock source • Safe clock for safety modules FCCU and FOSU • SIUL2 filter clock • MC_RGM clock source
PLL_PHI*n*_CLK (25-480 MHz)	1...16	Off	Functional (disabled on functional reset)	• Optional system clock source • Communication modules (FlexCAN, EMAC/GMAC, QuadSPI, and so on)
PLL_AUX_PHI*n*_CLK (25-500 MHz)	1...32	Off	Functional (disabled on functional reset)	• Communication modules (EMAC/GMAC, uSDHC, SAI, and QuadSPI)
FXOSC_CLK (8-40 MHz)	—	Off	Functional (disabled on functional reset)	• Reference clock source for PLL • Communication modules (FlexCAN, EMAC/GMAC, QuadSPI, and so on)

Table continues on the next page...

[3] FIRC_CLK and SIRC_CLK cannot be disabled during Run mode.

Table 117. Chip clock sources (continued)

Clock source	Divider	Default state	Reset	Uses
SXOSC_CLK[1] (32.768 KHz)	—	Off	Destructive (disabled on destructive reset)	• RTC source for operation across functional reset
SIRC_CLK (32 KHz)	—	On	POR (enabled on functional reset)	• Safe clock along with FIRC_CLK • SWT clock source • POR_WDG source clock

1. See the section "Feature comparison" in this reference manual's "Introduction" chapter for details on this module's availability on your chip variant.

24.4.2 Chip input clocks

Table 118. Chip input clocks

Pin	Description
XTAL	FXOSC crystal pins
EXTAL	
OSC32K_XTAL	SXOSC crystal pins[1]
OSC32K_EXTAL	
EMAC_MII_RMII_TX_CLK	EMAC transmitter clock/EMAC RMII clock[1]
GMAC_MII_RMII_RGMII_TX_CLK	GMAC transmitter clock/GMAC RMII clock[1]
GMAC0_MII_RMII_RGMII_TX_CLK	GMAC0 transmitter clock/GMAC0 RMII clock[1]
GMAC1_MII_RMII_RGMII_TX_CLK	GMAC1 transmitter clock/GMAC1 RMII clock[1]
EMAC_MII_RX_CLK	EMAC receiver clock[1]
GMAC_MII_RGMII_RX_CLK	GMAC receiver clock[1]
GMAC0_MII_RGMII_RX_CLK	GMAC0 receiver clock[1]
GMAC1_MII_RGMII_RX_CLK	GMAC1 receiver clock[1]
JTAG_TCLK/SWD_CLK	JTAG/SWD clock
SAI*n*_MCLK	SAI*n* clock in slave mode[1]
SAI*n*_BCLK	SAI*n* bit clock in slave mode[1]
LPSPI*n*_SCK	LPSPI*n* serial clock in slave mode
LPI2C*n*_SCL	LPI2C*n* clock in slave mode
LPI2C*n*_SCLS	LPI2C*n* secondary clock in slave mode

1. See the section "Feature comparison" in this reference manual's "Introduction" chapter for details on this module's availability on your chip variant.

24.4.3 Chip output clocks

Table 119. Chip output clocks

Pin	Description
CLKOUT_RUN	Available during Run mode, unavailable during Standby mode
CLKOUT_STANDBY	Available during both Run and Standby modes
LPSPI*n*_SCK	LPSPI*n* serial clock in master mode
LPI2C*n*_SCL	LPI2C*n* clock in master mode
LPI2C*n*_SCLS	LPI2C*n* secondary clock in master mode
EMAC_MII_RMII_MDC	EMAC clock for control data transfer to PHY [1]
GMAC_MII_RMII_RGMII_MDC	GMAC clock for control data transfer to PHY[1]
GMAC0_MII_RMII_RGMII_MDC	GMAC0 clock for control data transfer to PHY[1]
GMAC1_MII_RMII_RGMII_MDC	GMAC1 clock for control data transfer to PHY[1]
EMAC_MII_RMII_TX_CLK	EMAC transmit clock[1]
GMAC_MII_RMII_RGMII_TX_CLK	GMAC transmit clock[1]
GMAC0_MII_RMII_RGMII_TX_CLK	GMAC0 transmit clock[1]
GMAC1_MII_RMII_RGMII_TX_CLK	GMAC1 transmit clock[1]
TRACE_ETM_CLKOUT	ETM trace clock[2]
SAI*n*_BCLK	SAI*n* bit clock in master mode[1]
QuadSPI_SCKFA	QuadSPI serial clock for serial flash device[1]
uSDHC_PER_CLK	uSDHC serial clock[1]

1. See the section "Feature comparison" in this reference manual's "Introduction" chapter for details on this module's availability on your chip variant.
2. See the section "Interfaces supported in S32K3 family" in this reference manual's "Debug Subsystem" chapter for details on this module's availability on your chip variant.

24.4.4 Fast internal RC oscillator (FIRC)

The chip has an FIRC with the following features:

- Acts as the system clock source on power-up and after any reset event.
- Acts as the chip's safe clock for safety-relevant applications.
- Is always enabled in Run mode and can be optionally enabled in Standby mode.
- Used as clock source for the following:
  - MC_RGM
  - FCCU and FOSU
  - SIUL2 filters

24.4.4.1 FIRC failure detection

The FIRC_CLK is the safe clock source used as the FCCU and FOSU clock source. The chip supports FIRC_CLK failure detection and recovery by the mechanisms described in the following cases:

- Case 1 - FIRC_CLK not used as system clock and goes out of range:
  - CMU_FM_1 continuously measures the FIRC_CLK clock frequency using FXOSC_CLK as the reference. On each metering window completion, CMU_FM_1 asserts an interrupt (if configured, CMU_FM_1.IER[FMCIE] is 1). You store a configured reference clock count CMU_FM_1.RCCR[REF_CNT] (for example, number of FXOSC_CLK cycles in metering window). Your application software checks the FIRC_CLK clock counts by reading CMU_FM_1.SR[MET_CNT].
  - In the event FIRC_CLK goes out of range, the application software detects the frequency variation after the subsequent metering window by checking CMU_FM_1.SR[MET_CNT] and takes necessary action by either of the recommended options:
    - SBC-driven power cycle: The chip indicates the SBC (through GPIO toggle, QuadSPI communication, and so on, as configured by your software and the connectivity to the SBC) to initiate a power cycle sequence.
    - Application software-driven functional reset: The chip executes a functional reset as configured by your application software.
- Case 2 - FIRC_CLK not used as system clock and fails (becomes stuck):
  - CMU_FM_1 continuously measures the FIRC_CLK frequency with FXOSC_CLK as the reference. Application software must check FIRC_CLK after a defined time limit by reading CMU_FM_1.SR[MET_CNT] and CMU_FM_1.SR[FMTO].
  - If there is an FIRC_CLK failure, the CMU_FM_1 writes a 1 to the timeout status flag CMU_FM_1.SR[FMTO].
  - When CMU_FM_1.SR[FMTO] is 1, application software takes necessary action by an SBC-driven power cycle wherein the chip provides an indication to the SBC (through GPIO toggle, QuadSPI communication, and so on). The SBC then initiates a power cycle sequence.
- Case 3 - FIRC_CLK used as system clock and goes out of range or fails (becomes stuck):
  - CMU_FC_3 continuously monitor the system clock nodes with FXOSC_CLK as reference for FLL or FHH events.
  - CMU_FC_4 and CMU_FC_5 continuously monitor the system clock nodes with FIRC_CLK as reference for FLL or FHH events.
  - In the event of FIRC_CLK failure (when FIRC_CLK acts as the system clock source), these CMUs report the FLL event, acting as a destructive reset source.
  - The system then undergoes the reset sequence, during which the FIRC_CLK is reinitialized.
  - The MC_RGM.DES fields indicate the source of the reset event.
  - In addition, CMU_FM_1 monitors FIRC_CLK. If the application software is not able to service the CMU_FM_1 interrupt before POR_WDG timeout, POR_WDG treats this as a critical FIRC_CLK failure and initiates a POR_WDG recovery. Therefore, you must ensure that, if enabled, the CMU_FM_1 interrupt must be serviced within the POR_WDG timeout duration.

24.4.4.2 FIRC_CLK behavior in Standby mode

FIRC_CLK can be optionally enabled in Standby mode by configuring FIRC.STDBY_ENABLE[STDBY_EN].

When the PMC acknowledges the Standby mode entry, the FIRC_CLK switches from the On state to the Standby mode configuration selected by the FIRC.STDBY_ENABLE[STDBY_EN] configuration. On wakeup from Standby mode, the FIRC_CLK configuration switches from the Standby mode configured state to the On state.

The FIRC at 3 MHz is meant to be used with the Low Speed Run Mode only. If the chip needs to enter Standby mode, then the FIRC can be configured to 48MHz or 24MHz with the FIRC_DIV_SEL as '0b11' or '0b00', or '0b01' before entering Standby mode. After exiting the Standby mode, the FIRC can be changed to 3 MHz if desired.

NOTE

During the functional reset sequence, the analog blocks are loaded with the configured trimmed values. During this step, the FIRC will appear as momentarily disabled, as shown in the "Reset overview" chapter.

24.4.5 SIRC

The chip has an SIRC having the following features:

- Is always enabled in Run mode and can be optionally enabled in Standby mode. Having the SIRC always enabled improves system robustness by ensuring that a clock is always available for various SWTs when reducing the chip power consumption in Standby mode.
- Used as clock source for the following:
  - SWT
  - POR_WDG

24.4.5.1 SIRC failure detection

Like the FIRC_CLK, the SIRC_CLK is a safe clock for the design and it is used as a clock source for SWTs and POR_WDOG and therefore it is important to detect SIRC failure and ensure its recovery. The chip supports SIRC_CLK failure detection and recovery by the mechanism described below.

- Case 1 - SIRC_CLK goes out of range:
  - CMU_FM_2 continuously measures the SIRC_CLK clock frequency with FXOSC_CLK as reference. On each metering window completion, the CMU_FM_2 raises an interrupt. The application software checks the SIRC clock counts by CMU_FM_2.SR[MET_CNT] with respect to the reference clock counts CMU_FM_2.RCCR[REF_CNT].
  - In the event of clock going out of range, the application software detects the frequency variation after the subsequent metering window by checking CMU_FM_2.SR[MET_CNT] and takes necessary action by either of the recommended options:
    1. SBC-driven power cycle. The chip gives an indication to the SBC (through GPIO toggle, QuadSPI communication, and so on.). The SBC then initiates a power cycle sequence.
    2. SW-driven functional reset. The chip executes a functional reset by application software.
- Case 2 - SIRC_CLK fails (becomes stuck):
  - CMU_FM_2 continuously measures the SIRC clock frequency with FXOSC_CLK as reference. Application software needs to check this reference after a predefined time by checking CMU_FM_2.SR[MET_CNT] and CMU_FM_2.SR[FMTO].
  - In the event of SIRC_CLK clock failure, the CMU_FM_2 writes 1 to the timeout status flag field in its status register, namely, CMU_FM_2.SR[FMTO].
  - When CMU_FM_2.SR[FMTO] is 1, the application software takes necessary action by either of the recommended options:
    1. SBC-driven power cycle: The chip gives an indication to the SBC (through GPIO toggle, QuadSPI communication, and so on). The SBC then initiates a power cycle sequence.
    2. Application software-driven functional reset: User application software executes a functional reset.

24.4.5.2 SIRC behavior in Standby mode

SIRC can optionally be enabled in Standby mode by configuring SIRC.MISCELLANEOUS_IN[STANDBY_ENABLE].

When PMC acknowledges the Standby mode entry, the SIRC switches from the On state to the standby configuration selected by SIRC.MISCELLANEOUS_IN[STANDBY_ENABLE] configuration. On wake-up from Standby mode, the SIRC configuration switches back from the standby-configured state to the On state.

NOTE

When the trims are being applied, the SIRC will appear as momentarily disabled similar to FIRC, as shown in section "Signal level reset flow timing".

24.4.6 FXOSC

The chip supports an 8–40 MHz fast crystal oscillator which has following features:

- Acts as the reference for PLL.
- Supports crystal input mode and bypass mode if using an external oscillator.
- Acts as a clock source for communication modules:
  - FlexCAN
  - QuadSPI[4]
  - EMAC (EMAC_CLK_TS)[4]
  - GMAC (GMAC_CLK_TS)[4]

24.4.7 SXOSC

The chip supports a slow crystal oscillator (SXOSC) which has the following features[5]:

- Supports crystal input mode.
- Bypass mode not supported.
- Acts as a clock source for RTC. As SXOSC is not affected by functional reset, it supports RTC operation across functional reset. SXOSC is only reset on destructive reset.

24.4.8 PLL

The chip contains up to two PLL to provide precision clock source with the following features:

- Optional system clock source (in high performance applications)
- System clock source in safety applications
- Can be used as clock source for communication modules, when configured as system clock source:
  - FlexCAN (in SYNC mode operation)
  - EMAC[6]
  - GMAC[6]
  - QuadSPI[6]
  - LPSPI
  - LPI2C
  - FlexIO
  - LPUART
- Supports frequency modulation
- Contains lock status monitoring logic which supports loss-of-lock indication

At power-up, the second PLL (PLL_AUX) will be disabled. It will be up to the application software to enable it when needed. In case of a loss of lock after lock has been acquired, PLL_AUX will generate a loss of lock flag. The default reaction of loss of lock

[4] See the section "Feature comparison" in this reference manual's "Introduction" chapter for details on this module's availability on your chip variant.
[5] See the section "Feature comparison" in this reference manual's "Introduction" chapter for details on this module's availability on your chip variant.
[6] See the section "Feature comparison" in this reference manual's "Introduction" chapter for details on this module's availability on your chip variant.

flag from the Second PLL will be an interrupt. The reaction of a loss of lock from the Second PLL will be optionally configured as a functional reset, controlled by GPR.

The Second PLL supports a clock output frequency of up to 250MHz (nominal), for the FXOSC input frequencies.

NOTE

See "PLL Digital Interface (PLLDIG)" for PLL configuration details. See the *S32K3xx Data Sheet* for PLL specifications.

24.4.8.1 PLL configurations

The PLL output predivider frequency depends on the PLLDIG.PLLDV[RDIV] and PLLDIG.PLLDV[MFI] configurations. The PLL VCO clock can be divided further by configuring PLLDIG.PLLODIV_0[DIV] for PLL_PHI0_CLK and PLLDIG_PLLODIV_1[DIV] for PLL_PHI1_CLK (see the "PLL Digital Interface (PLLDIG)" chapter for configuration details).

24.4.8.1.1 PLL configuration sequence

Before enabling the PLL, you must enable FXOSC_CLK and wait until it is stable. FXOSC.STAT[OSC_STAT] must be monitored to determine the FXOSC_CLK status.

To disable the PLL, the application software must disable PLL first and only then disable FXOSC (if required).

For S32K311 and S32K310, FIRC_DIV2_CLK is also available as a PLL reference.

24.4.9 Chip clock outputs

The chip supports two CLKOUT_*x* pins for allowing viewing of some internal clocks as follows:

- CLKOUT_STANDBY
  - Used for showing clocks available in Run and Standby modes.
- CLKOUT_RUN
  - Used for showing only Run mode clocks.

See the Clockout overview section and the "Clock Generation Module (MC_CGM)" chapter for details on available clocks and configuration.

NOTE

The CLKOUT_STANDBY registers are latched when the chip enters Standby mode and are reset in Standby mode sequence. Therefore, the CLKOUT_STANDBY signal needs to be reconfigured on Standby mode exit.

NOTE

CLKOUT_STANDBY is available on two pads GPIO[12] and GPIO[138] but CLKOUT across functional reset and standby is supported only on GPIO[12] and OBE(output buffer enable) is controlled by DCM GPR bit. Please refer to DCMRWP1[3] bit for detail.

24.5 MC_CGM

The MC_CGM controls the clock functionality of the chip. See the Clock Generation Module (MC_CGM) chapter for details on MC_CGM clocking controls.

24.5.1 MC_CGM clock multiplexer types

In this chip, CLKOUT_RUN, CLKOUT_STANDBY, and TRACE_CLK multiplexers are software-controlled multiplexers. The rest are hardware-controlled multiplexers (see the "Clock Generation Module (MC_CGM)" chapter for details on software and hardware multiplexers).

24.5.2 MC_CGM clock multiplexers for S32K310, S32K311, and S32K312

Table 120. MC_CGM clock multiplexers for S32K310, S32K311, and S32K312

Clock mux	Register description	Source inputs [1]	Register	Divider output
Clock mux 0	Select Control	FIRC_CLK PLL_PHI0_CLK	MUX_0_CSC	—
	Select Status		MUX_0_CSS	
	Divider 0 Control	—	MUX_0_DC_0	CORE_CLK
	Divider 1 Control		MUX_0_DC_1	AIPS_PLAT_CLK
	Divider 2 Control		MUX_0_DC_2	AIPS_SLOW_CLK
	Divider 3 Control		MUX_0_DC_3	HSE_CLK
	Divider 4 Control		MUX_0_DC_4	DCM_CLK
Clock mux 1	Select Control	FIRC_CLK FXOSC_CLK AIPS_PLAT_CLK	MUX_1_CSC	—
	Select Status		MUX_1_CSS	
	Divider 0 Control	—	MUX_1_DC_0	STM0_CLK
Clock mux 3	Select Control	FIRC_CLK FXOSC_CLK AIPS_PLAT_CLK	MUX_3_CSC	—
	Select Status		MUX_3_CSS	
	Divider 0 Control	—	MUX_3_DC_0	FLEXCAN0_PE_CLK FLEXCAN1_PE_CLK FLEXCAN2_PE_CLK
Clock mux 4 [2]	Select Control	FIRC_CLK FXOSC_CLK AIPS_PLAT_CLK	MUX_4_CSC	—
	Select Status		MUX_4_CSS	
	Divider 0 Control	—	MUX_4_DC_0	FLEXCAN3_PE_CLK FLEXCAN4_PE_CLK FLEXCAN5_PE_CLK
Clock mux 5	Select Control	FIRC_CLK SIRC_CLK FXOSC_CLK SXOSC_CLK[3] AIPS_SLOW_CLK	MUX_5_CSC	—
	Select Status		MUX_5_CSS	
	Divider 0 Control	—	MUX_5_DC_0	CLKOUT_STANDBY
Clock mux 6	Select Control	FIRC_CLK SIRC_CLK FXOSC_CLK SXOSC_CLK[3] PLL_PHI0_CLK	MUX_6_CSC	—
	Select Status		MUX_6_CSS	

Table continues on the next page...

Table 120. MC_CGM clock multiplexers for S32K310, S32K311, and S32K312 (continued)

Clock mux	Register description	Source inputs [1]	Register	Divider output
		PLL_PHI1_CLK CORE_CLK HSE_CLK AIPS_PLAT_CLK AIPS_SLOW_CLK		
	Divider 0 Control	—	MUX_6_DC_0	CLKOUT_RUN
Clock mux 11	Select Control	FIRC_CLK FXOSC_CLK PLL_PHI0_CLK PLL_PHI1_CLK	MUX_11_CSC	—
	Select Status		MUX_11_CSS	
	Divider 0 Control	—	MUX_11_DC_0	TRACE_CLK

1. The default clock selected for all clock mux selectors is FIRC_CLK (out of reset).
2. Clock mux 4 and FLEXCAN[3:5]_PE_CLK are not available on S32K310 and S32K311.
3. SXOSC as source for clock mux 5 is not available on S32K310 and S32K311.

24.5.3 MC_CGM clock multiplexers (excluding S32K310, S32K311, and S32K312)

Table 121. MC_CGM clock multiplexers (excluding S32K310, S32K311, and S32K312)

Clock mux	Register description	Source inputs [1]	Register	Divider output
Clock mux 0	Select Control	FIRC_CLK PLL_PHI0_CLK	MUX_0_CSC	—
	Select Status		MUX_0_CSS	
	Divider 0 Control	—	MUX_0_DC_0	CORE_CLK
	Divider 1 Control		MUX_0_DC_1	AIPS_PLAT_CLK
	Divider 2 Control		MUX_0_DC_2	AIPS_SLOW_CLK
	Divider 3 Control		MUX_0_DC_3	HSE_CLK
	Divider 4 Control		MUX_0_DC_4	DCM_CLK
	Divider 5 Control		MUX_0_DC_5	LBIST_CLK
	Divider 6 Control		MUX_0_DC_6	QSPI_MEM_CLK
	Divider 7 Control[2]		MUX_0_DC_7	CM7_CORE_CLK
Clock mux 1	Select Control	FIRC_CLK FXOSC_CLK AIPS_PLAT_CLK	MUX_1_CSC	—
	Select Status		MUX_1_CSS	
	Divider 0 Control	—	MUX_1_DC_0	STM0_CLK
Clock mux 2	Select Control	FIRC_CLK FXOSC_CLK AIPS_PLAT_CLK	MUX_2_CSC	—
	Select Status		MUX_2_CSS	
	Divider 0 Control	—	MUX_2_DC_0	STM1_CLK

Table continues on the next page...

Table 121. MC_CGM clock multiplexers (excluding S32K310, S32K311, and S32K312) (continued)

Clock mux	Register description	Source inputs [1]	Register	Divider output
Clock mux 3	Select Control	FIRC_CLK FXOSC_CLK AIPS_PLAT_CLK CORE_CLK[3]	MUX_3_CSC	—
	Select Status		MUX_3_CSS	
	Divider 0 Control	—	MUX_3_DC_0	FLEXCAN0_PE_CLK FLEXCAN1_PE_CLK FLEXCAN2_PE_CLK
Clock mux 4	Select Control	FIRC_CLK FXOSC_CLK AIPS_PLAT_CLK CORE_CLK[3]	MUX_4_CSC	—
	Select Status		MUX_4_CSS	
	Divider 0 Control	—	MUX_4_DC_0	FLEXCAN3_PE_CLK FLEXCAN4_PE_CLK[4] FLEXCAN5_PE_CLK[4] FLEXCAN6_PE_CLK[6] FLEXCAN7_PE_CLK[6]
Clock mux 5	Select Control	FIRC_CLK SIRC_CLK FXOSC_CLK SXOSC_CLK AIPS_SLOW_CLK	MUX_5_CSC	—
	Select Status		MUX_5_CSS	
	Divider 0 Control	—	MUX_5_DC_0	CLKOUT_STANDBY
Clock mux 6	Select Control	FIRC_CLK SIRC_CLK FXOSC_CLK SXOSC_CLK PLL_PHI0_CLK PLL_PHI1_CLK PLL_AUX_PHI0_CLK[6] PLL_AUX_PHI1_CLK[6] PLL_AUX_PHI2_CLK[3] CORE_CLK HSE_CLK AIPS_PLAT_CLK AIPS_SLOW_CLK GMAC_MII_RMII_RGMII_TX_CLK[3] GMAC_MII_RGMII_RX_CLK[3] GMAC0_MII_RMII_RGMII_TX_CLK[2] GMAC0_MII_RGMII_RX_	MUX_6_CSC	—
	Select Status		MUX_6_CSS	

Table continues on the next page...

Table 121. MC_CGM clock multiplexers (excluding S32K310, S32K311, and S32K312) (continued)

Clock mux	Register description	Source inputs [1]	Register	Divider output
		CLK[2] EMAC_MII_RMII_TX_CLK[5] EMAC_MII_RX_CLK[5]		
	Divider 0 Control	—	MUX_6_DC_0	CLKOUT_RUN
Clock mux 7[5]	Select Control	FIRC_CLK EMAC_MII_RMII_TX_CLK EMAC_MII_RX_CLK	MUX_7_CSC	—
	Select Status		MUX_7_CSS	
	Divider 0 Control	—	MUX_7_DC_0	EMAC_CLK_RX
Clock mux 7[6]	Select Control	FIRC_CLK PLL_PHI0_CLK[3] PLL_AUX_PHI0_CLK[3] GMAC0_MII_RMII_RGMII_TX_CLK[2] GMAC_MII_RGMII_RX_CLK[3] GMAC_MII_RMII_RGMII_TX_CLK[3]	MUX_7_CSC	—
	Select Status		MUX_7_CSS	
	Divider 0 Control	—	MUX_7_DC_0	GMAC0_CLK_RX[2] GMAC_CLK_RX[3]
Clock mux 8[5]	Select Control	FIRC_CLK EMAC_MII_RMII_TX_CLK	MUX_8_CSC	—
	Select Status		MUX_8_CSS	
	Divider 0 Control	—	MUX_8_DC_0	EMAC_CLK_TX
Clock mux 8[6]	Select Control	FIRC_CLK PLL_PHI0_CLK[3] PLL_AUX_PHI0_CLK[6] GMAC0_MII_RMII_RGMII_TX_CLK[2] GMAC_MII_RGMII_RX_CLK[3] GMAC_MII_RMII_RGMII_TX_CLK[3]	MUX_8_CSC	—
	Select Status		MUX_8_CSS	
	Divider 0 Control	—	MUX_8_DC_0	GMAC_CLK_TX[3] GMAC0_CLK_TX[2]

Table continues on the next page...

Table 121. MC_CGM clock multiplexers (excluding S32K310, S32K311, and S32K312) (continued)

Clock mux	Register description	Source inputs [1]	Register	Divider output
Clock mux 9[5]	Select Control	FIRC_CLK FXOSC_CLK PLL_PHI0_CLK EMAC_MII_RMII_TX_CLK EMAC_MII_RX_CLK	MUX_9_CSC	—
	Select Status		MUX_9_CSS	
	Divider 0 Control	—	MUX_9_DC_0	EMAC_CLK_TS
Clock mux 9[6]	Select Control	FIRC_CLK FXOSC_CLK PLL_PHI0_CLK[3] GMAC_MII_RMII_RGMII_TX_CLK[3] GMAC_MII_RGMII_RX_CLK[3] PLL_AUX_PHI0_CLK GMAC0_MII_RMII_RGMII_TX_CLK[2] GMAC0_MII_RGMII_RX_CLK[2] GMAC1_MII_RMII_RGMII_TX_CLK[2] GMAC1_MII_RGMII_RX_CLK[2]	MUX_9_CSC	—
	Select Status		MUX_9_CSS	
	Divider 0 Control	—	MUX_9_DC_0	GMAC_CLK_TS
Clock mux 10	Select Control	FIRC_CLK FXOSC_CLK PLL_PHI1_CLK PLL_AUX_PHI0_CLK[6]	MUX_10_CSC	—
	Select Status		MUX_10_CSS	
	Divider 0 Control	—	MUX_10_DC_0	QSPI_SFCK QSPI_2XSFIF[3]
Clock mux 11	Select Control	FIRC_CLK FXOSC_CLK PLL_PHI0_CLK PLL_PHI1_CLK PLL_AUX_PHI0_CLK[6]	MUX_11_CSC	—
	Select Status		MUX_11_CSS	
	Divider 0 Control	—	MUX_11_DC_0	TRACE_CLK
Clock mux 12	Reserved			
Clock mux 13[6]	Select Control	AIPS_PLAT_CLK FXOSC_CLK FIRC_CLK	MUX_13_CSC	—
	Select Status		MUX_13_CSS	

Table continues on the next page...

Table 121. MC_CGM clock multiplexers (excluding S32K310, S32K311, and S32K312) (continued)

Clock mux	Register description	Source inputs [1]	Register	Divider output
	Divider 0 Control	—	MUX_13_DC_0	STM2_CLK
Clock mux 14[3]	Select Control	FIRC_CLK FXOSC_CLK PLL_PHI1_CLK PLL_AUX_PHI2_CLK	MUX_14_CSC	—
	Select Status		MUX_14_CSS	
	Divider 0 Control	—	MUX_14_DC_0	uSDHC_PER_CLK
Clock mux 15[2]	Select Control	GMAC1_MII_RMII_RGMII_TX_CLK FIRC_CLK	MUX_15_CSC	—
	Select Status		MUX_15_CSS	
	Divider 0 Control	—	MUX_15_DC_0	GMAC1_CLK_RX
Clock mux 16[2]	Select Control	GMAC1_MII_RMII_RGMII_TX_CLK FIRC_CLK	MUX_16_CSC	—
	Select Status		MUX_16_CSS	
	Divider 0 Control	—	MUX_16_DC_0	GMAC1_CLK_TX
Clock mux 17	Reserved			
Clock mux 18[2]	Select Control	AIPS_PLAT_CLK FIRC_CLK FXOSC_CLK	MUX_18_CSC	—
	Select Status		MUX_18_CSS	
	Divider 0 Control	—	MUX_18_DC_0	STM3_CLK
Clock mux 19[2]	Select Control	PLL_AUX_PHI1_CLK FIRC_CLK	MUX_19_CSC	—
	Select Status		MUX_19_CSS	
	Divider 0 Control	—	MUX_19_DC_0	AES_1us_CLK
Clock mux 20[7]	Select Control	AIPS_PLAT_CLK FIRC_CLK FXOSC_CLK	MUX_20_CSC	—
	Select Status		MUX_20_CSS	
	Divider 0 Control	—	MUX_20_DC_0	FLEXCAN[8:11]_PE_CLK

1. The default clock selected for all clock mux selectors is FIRC_CLK (out of reset).
2. Available only in S32K388 and S32K389.
3. For S32K358, S32K356, S32K348, S32K338, and S32K328 only.
4. FLEXCAN[4:5]_PE_CLK are not available on the S32K342, S32K322, and S32K341.
5. For S32K314, S32K322, S32K324, S32K341, S32K342, and S32K344 only.
6. For S32K388 and S32K389 only.
7. Available only in S32K389.

24.5.4 MC_CGM clock sources mapping

Table 122. MC_CGM clock sources mapping

Clock selector index [1]	MC_CGM clock source	Clock source
0	clk_src_0	FIRC_CLK
1	clk_src_1	SIRC_CLK
2	clk_src_2	FXOSC_CLK
3	Reserved	Reserved
4	clk_src_4	SXOSC_CLK [2]
5–7	Reserved	Reserved
8	clk_src_8	PLL_PHI0_CLK
9	clk_src_9	PLL_PHI1_CLK
10–11	Reserved	Reserved
12	clk_src_12	PLL_AUX_PHI0_CLK[3]
13	clk_src_13	PLL_AUX_PHI1_CLK[3]
14	clk_src_14	PLL_AUX_PHI2_CLK[4]
15	Reserved	Reserved
16	clk_src_16	CORE_CLK
17–18	Reserved	Reserved
19	clk_src_19	HSE_CLK
20–21	Reserved	Reserved
22	clk_src_22	AIPS_PLAT_CLK
23	clk_src_23	AIPS_SLOW_CLK
24	clk_src_24	EMAC_MII_RMII_TX_CLK[5] GMAC_MII_RMII_RGMII_TX_CLK[4] GMAC0_MII_RMII_RGMII_TX_CLK[6]
25	clk_src_25	EMAC_MII_RX_CLK [5] GMAC_MII_RGMII_RX_CLK[4] GMAC0_MII_RGMII_RX_CLK[6]
26–28	Reserved	Reserved
29	clk_src_29	GMAC1_MII_RGMII_RX_CLK[6]
30	clk_src_30	GMAC1_MII_RMII_RGMII_TX_CLK[6]
31–50	Reserved	Reserved

1. All clock selector indexes not shown are reserved.
2. SXOSC_CLK is not available on S32K310 and S32K311.
3. Applicable for S32K388, S32K389, S32K358, S32K356, S32K348, S32K338, and S32K328 only.
4. Applicable for S32K358, S32K356, S32K348, S32K338, and S32K328 only.
5. Applicable for S32K344, S32K324, S32K314, S32K322, S32K341, and S32K342.
6. Applicable for S32K388 and S32K389 only.

NOTE

Peripheral input clock source switching must not occur while peripheral is working.

24.6 Peripheral clocking

The module clocking diagrams for the peripherals are shown in the following subsections (see Peripheral clock gating for peripheral clock gating possibilities).

24.6.1 Module clocking

The following sections show how the chip modules use MODULE_CLK and REG_INTF_CLK to control their functionality.

24.6.1.1 Communication modules

Figure 70 shows the REG_INTF_CLK and MODULE_CLK connections, and Table 123 shows the REG_INTF_CLK and MODULE_CLK signals used by these modules. Any module diagram that does not explicitly show a REG_INTF_CLK uses the same source for REG_INTF_CLK as used by MODULE_CLK.

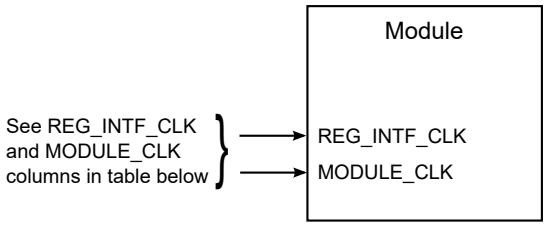


Figure 70. Communication module clocks

Table 123. Communication module clocking

Module	MODULE_CLK
FlexCAN	See FlexCAN*n* clocking.
LPI2C	See LPI2C*n* clocking.
GMAC	See GMAC clocking.
EMAC	See EMAC clocking.
LPSPI	See LPSPI*n* clocking.
LPUART	See LPUART*n* clocking
FlexIO	See FlexIO clocking.
QuadSPI	See QuadSPI clocking.
SAI	See SAI*n* clocking.
uSDHC	See uSDHC clocking

24.6.1.1.1 FlexCAN*n* clocking

The figure below shows the FlexCAN*n* clocking.

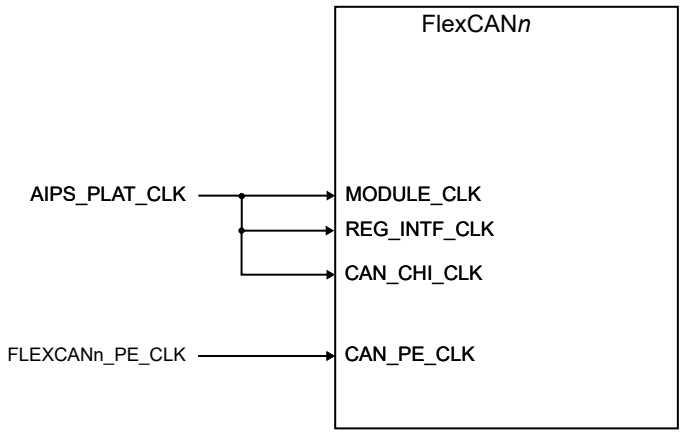


Figure 71. FlexCAN*n* clocking

FlexCAN has the following unique clocks:

- CAN_CHI_CLK—FlexCAN controller host interface clock
- CAN_PE_CLK—FlexCAN protocol engine clock

For all S32K3xx devices, the clock multiplexer to the CAN_FD protocol engine (PE) clock has an option to use the FXOSC_CLK (see Table 120 and Table 121). With a 40 MHz crystal source, FlexCAN supports up to an 8 Mbps data rate. With a 16 MHz crystal source, 3.2 Mbps is achievable. For baud rate calculations, see the section "Protocol timing" in the "CAN (FlexCAN)" chapter.

For some devices, the clock multiplexer to the CAN_FD protocol engine (PE) clock has an option to use the CORE_CLK. See Table 120 and Table 121. When the CORE_CLK is running at 240 MHz, the PLL derived clock to the CAN_FD protocol engine (PE) clock will be at 120 MHz (using the option to divide by 2), at 80 MHz (using the option to divide by 3), or at 60 MHz (using the option to divide by 4).

NOTE

- See the section "Feature comparison" in this reference manual's "Introduction" chapter for details on this module's availability on your chip variant.
- For MC_CGM input sources for different variants, see clocking diagram in 'Clocking Overview' section in 'Clocking' chapter

24.6.1.1.1.1 FlexCAN timestamp implementation

The following figure shows the FlexCAN timestamping implementation. The related table shows the timestamp sources and corresponding clock nodes.

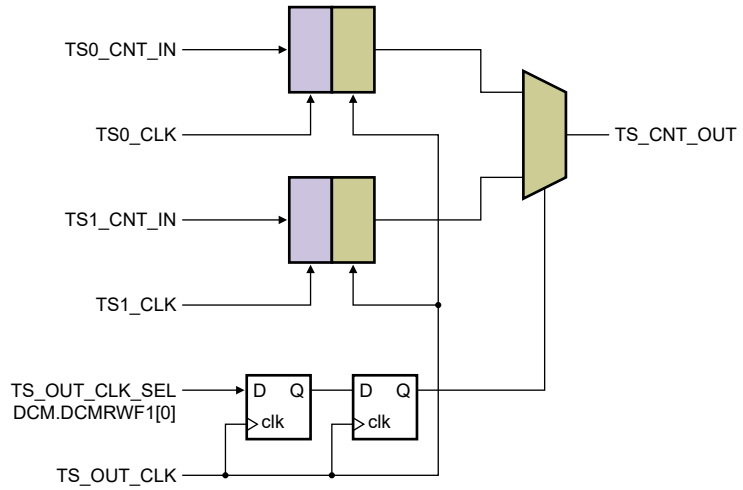


Figure 72. FlexCAN timestamping

Table 124. Timestamp sources and clock nodes

Timestamp	Module	TS clock domain
TS0_CLK	EMAC/GMAC	EMAC_CLK_TS/GMAC_CLK_TS
TS1_CLK	STM0	AIPS_PLAT_CLK
TS_OUT_CLK	FlexCAN*n*	EMAC_CLK_TS/GMAC_CLK_TS

NOTE

- See section "Feature comparison" in this reference manual's "Introduction" chapter for details on this module's availability on your chip variant.
- The timestamp clock (TS_OUT_CLK, EMAC_TS_CLK/GMAC_CLK_TS) must be greater than or equal to FLEXCAN*n*_PE_CLK. When using STM0 as the timestamp source, the FlexCAN timestamp clock (TS_OUT_CLK, EMAC_TS_CLK/GMAC_CLK_TS) must be greater than or equal to STM0_CLK.

24.6.1.1.2 LPI2C*n* clocking

The following figure shows LPI2C*n* clocking, and the related table shows the LPI2C SIUL2 configuration.

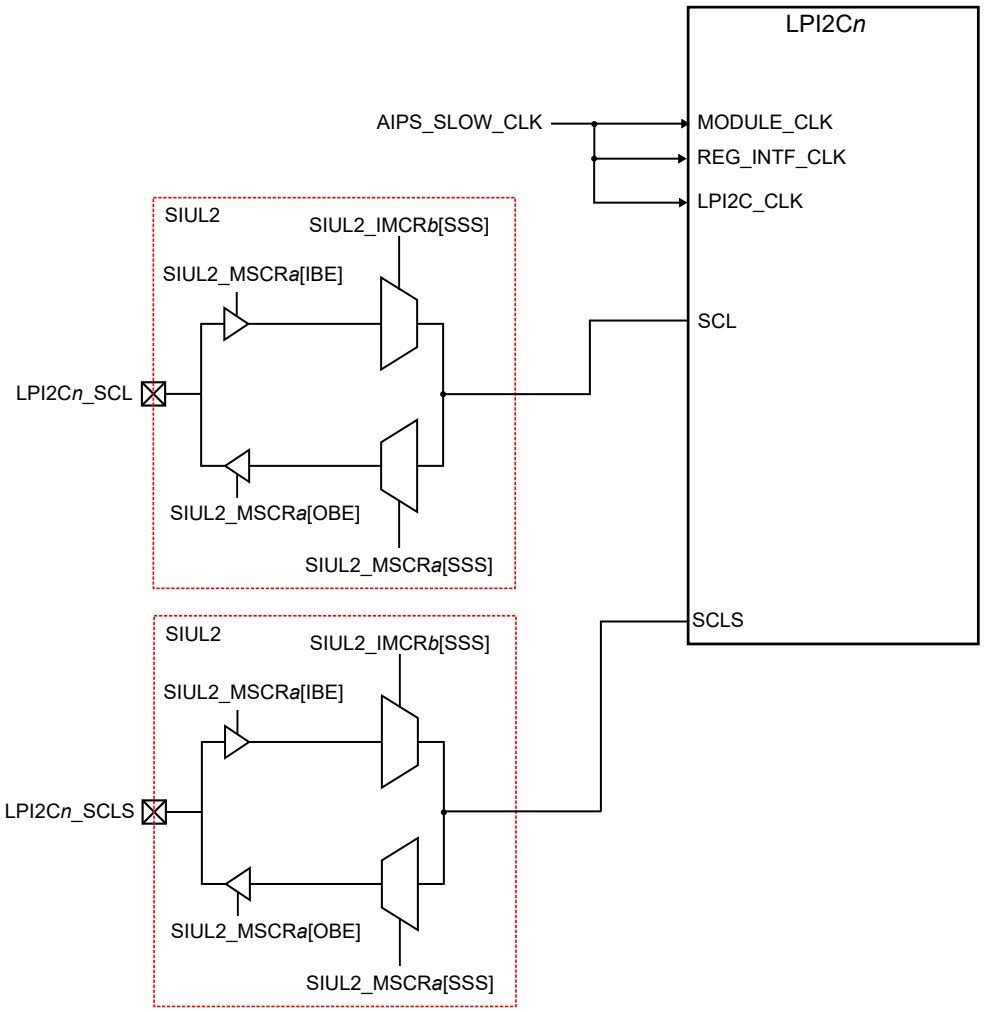


Figure 73. LPI2C*n* clocking

NOTE

- See the section "Feature comparison" in this reference manual's "Introduction" chapter for details on this module's availability on your chip variant.
- For MC_CGM input sources for different variants, see clocking diagram in 'Clocking Overview' section in 'Clocking' chapter

NOTE

See the IOMUX file for your chip variant, attached to this document for details on the ports that support this function.

24.6.1.1.3 GMAC clocking

Clocking details of GMAC for different modes is described in this section.

NOTE

See the "Device Configuration Module General-Purpose Registers (DCM_GPR)" chapter for details on GMAC configuration.

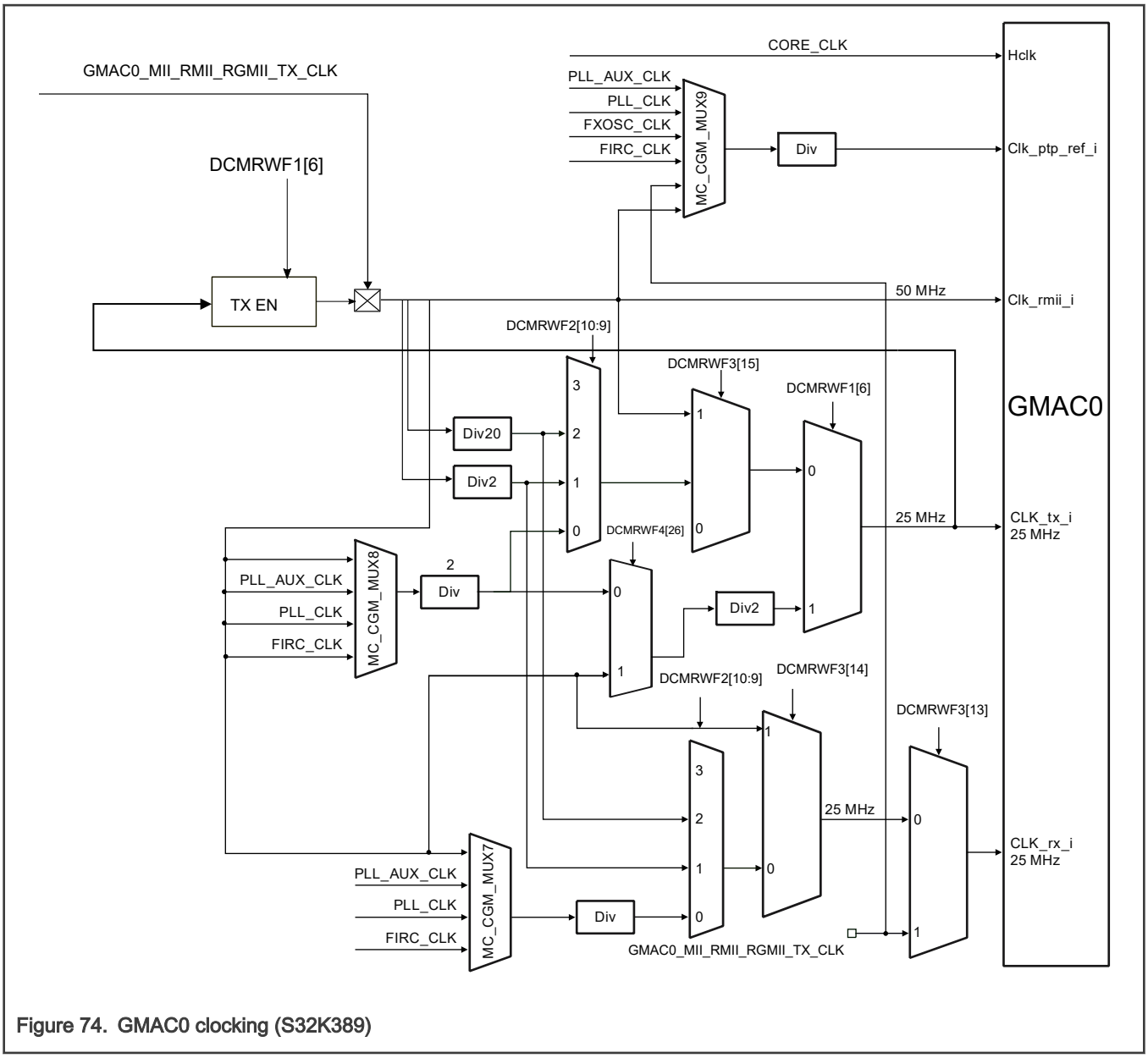


Figure 74. GMAC0 clocking (S32K389)

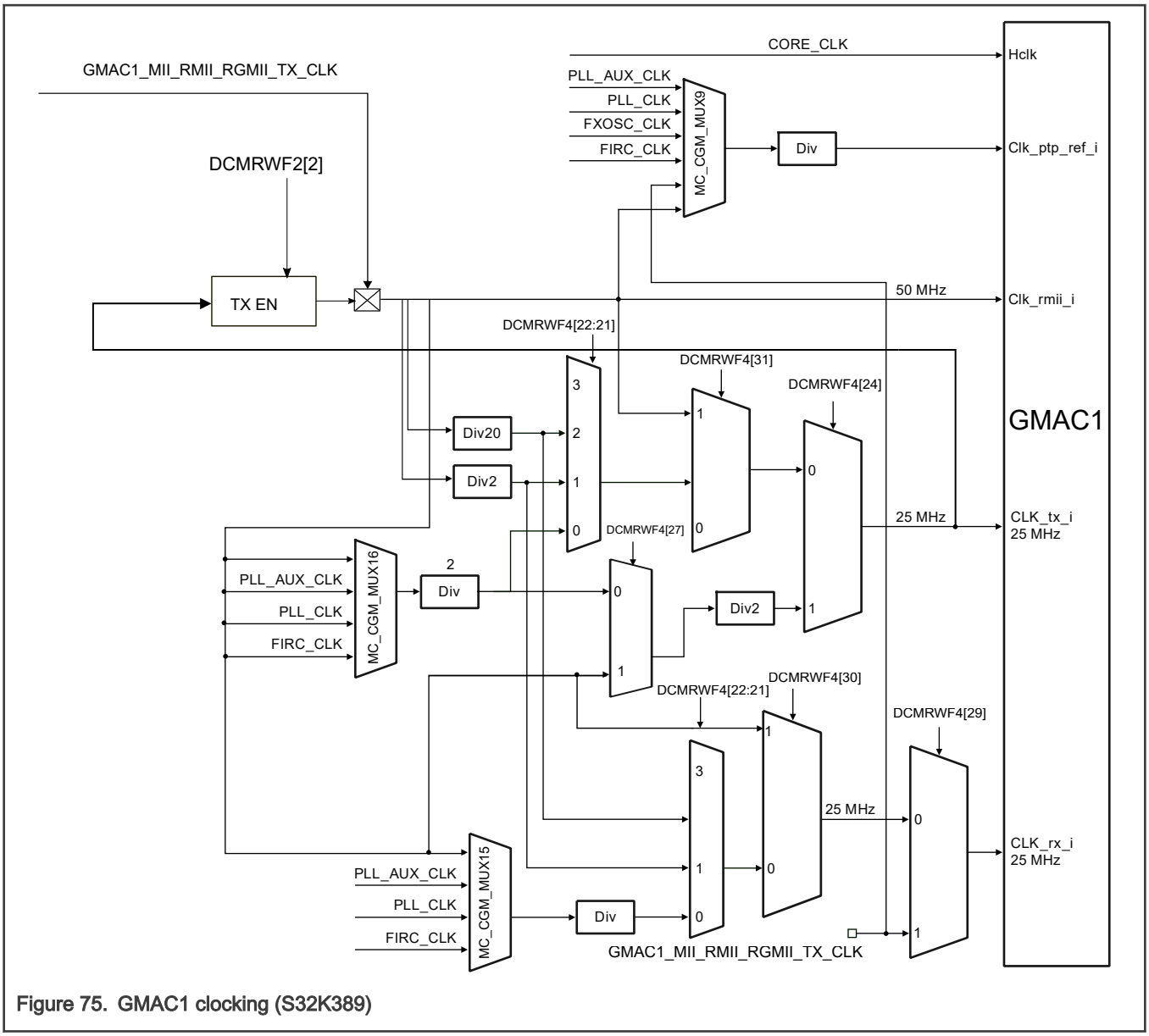


Figure 75. GMAC1 clocking (S32K389)

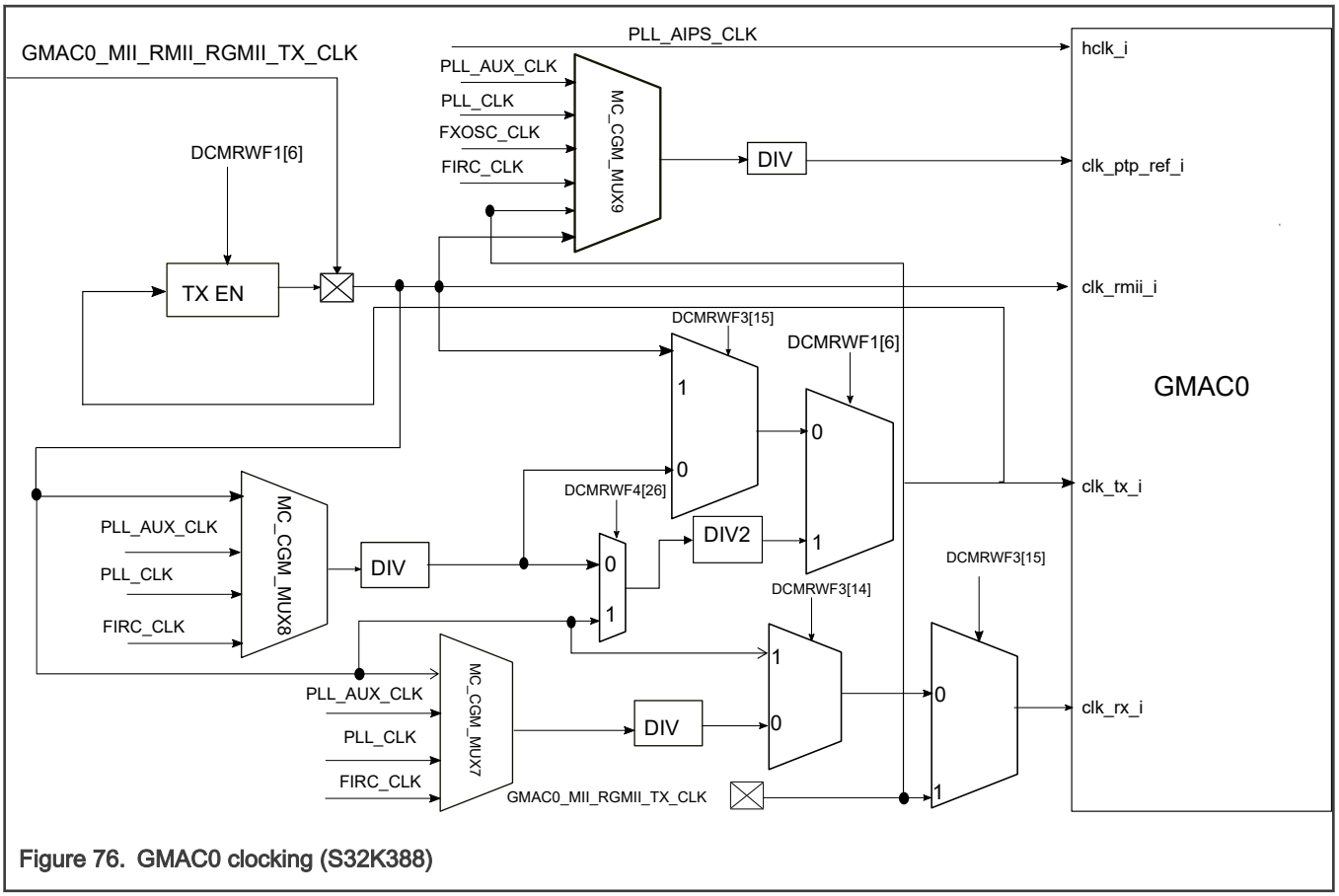


Figure 76. GMAC0 clocking (S32K388)

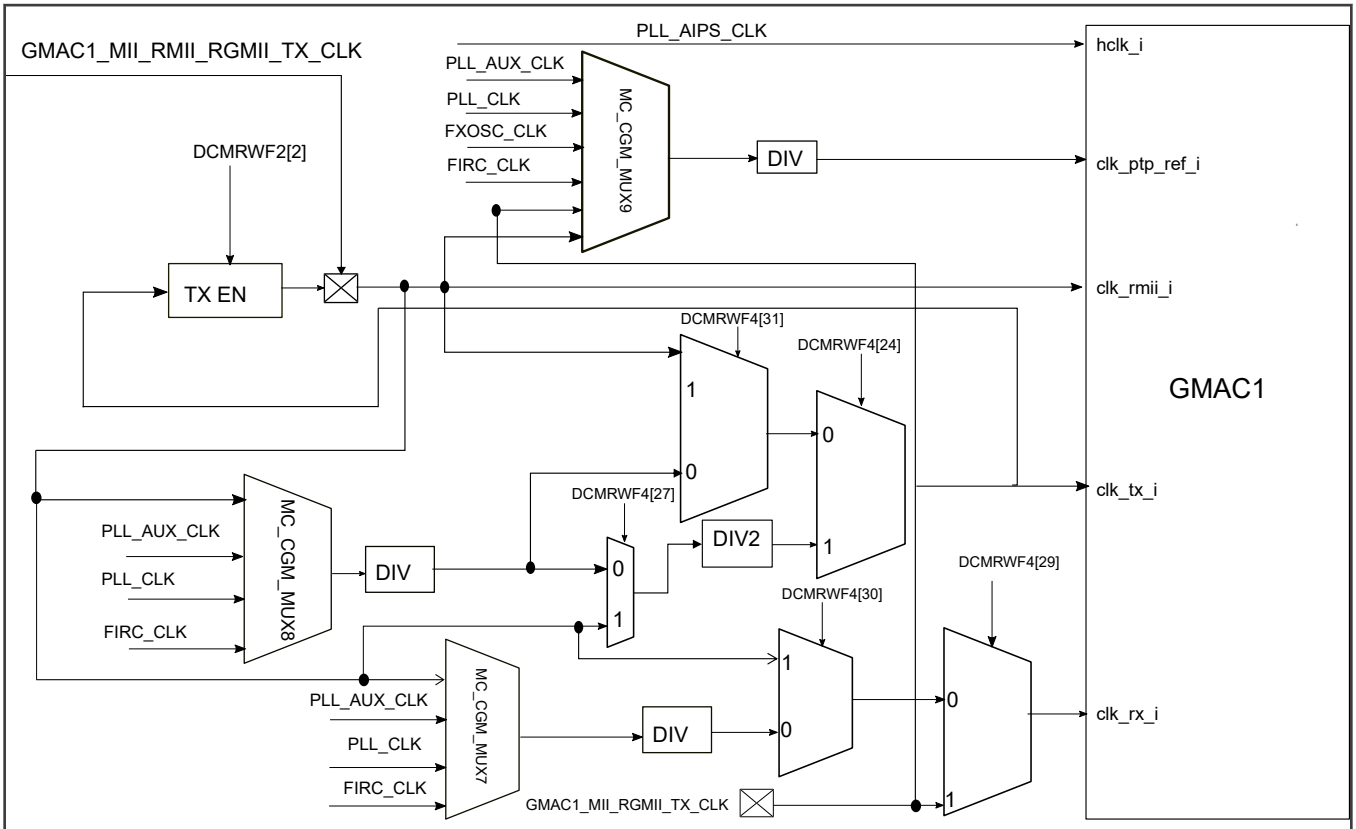


Figure 77. GMAC1 clocking (S32K388)

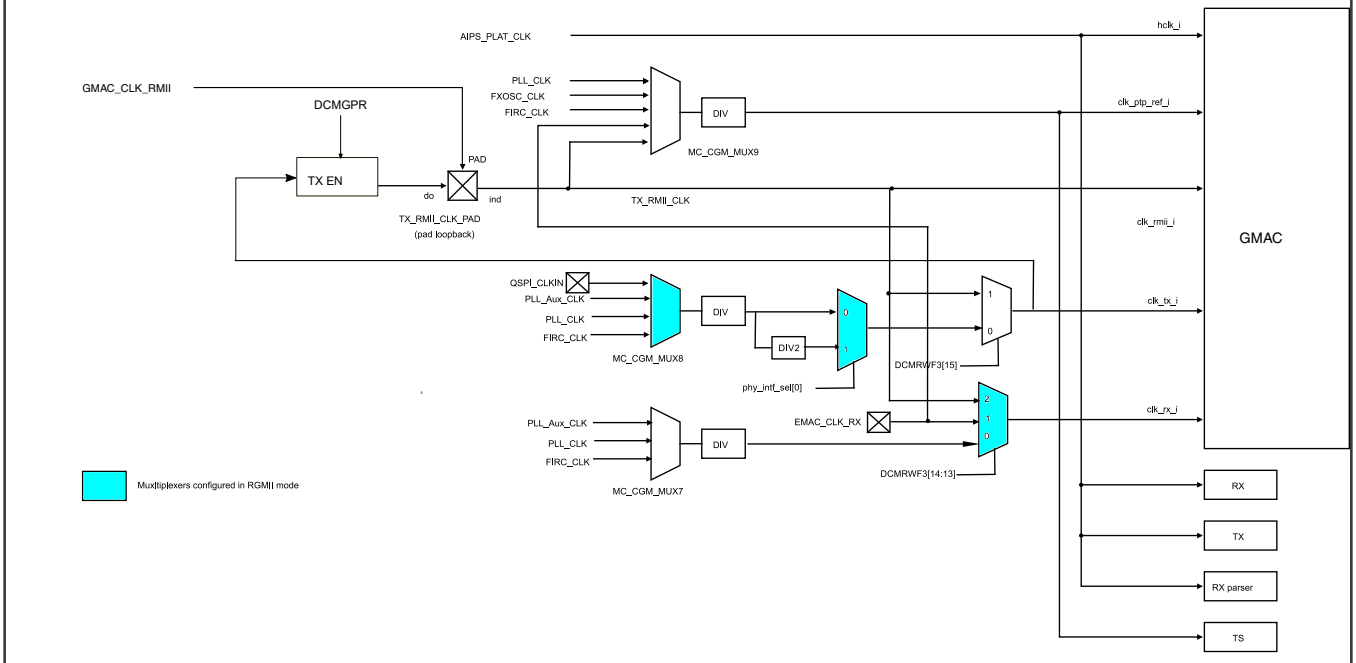


Figure 78. GMAC clocking (S32K358, S32K356, S32K348, S32K338, and S32K328)

Table 125. MII clock configuration (for S32K358 and S32K356)

Source clock	Destination clock	Port	SIUL2					
			MSCRa	MSCR fields			IMCRb	IMCR[SSS]
				OBE	IBE	SSS		
GMAC_MII_RMII_TX_CLK	GMAC_CLK_TX	PTC0	64	0	1	X	808	0100b
	GMAC_CLK_RX							
	GMAC_CLK_TX	PTD6	102	0	1	X	808	0010b
	GMAC_CLK_RX							
	GMAC_CLK_TX	PTD11	107	0	1	X	808	0001b
	GMAC_CLK_RX							
	GMAC_CLK_TX	PTD12	108	0	1	X	808	0011b
	GMAC_CLK_RX							
GMAC_MII_RMII_TX_CLK	GMAC_CLK_TS	PTC0	64	0	1	X	808	0100b
		PTD6	102	0	1	X	808	0010b
		PTD11	107	0	1	X	808	0001b
		PTD12	108	0	1	X	808	0011b
GMAC_MII_RX_CLK	GMAC_CLK_TS	PTC0	64	0	1	X	812	0100b
		PTC1	65	0	1	X	812	0011b
		PTD5	101	0	1	X	812	0010b
		PTD10	106	0	1	X	812	0001b

Table 126. RGMII clock configuration (for S32K358 and S32K356)

Source clock	Destination clock	Port	SIUL2					
			MSCRa	MSCR fields			IMCRb	IMCR[SSS]
				OBE	IBE	SSS		
GMAC_MII_RMII_RGMII_TX_CLK	GMAC_CLK_TX	PTB3	35	0	1	X	808	0101b
	GMAC_CLK_RX							

Table continues on the next page...

Table 126. RGMII clock configuration (for S32K358 and S32K356) (continued)

Source clock	Destination clock	Port	SIUL2					
			MSCRa	MSCR fields			IMCRb	IMCR[SSS]
				OBE	IBE	SSS		
	GMAC_CLK_TX	PTC19	83	0	1	X	808	0110b
	GMAC_CLK_RX							
GMAC_MII_RMII_RGMII_TX_CLK	GMAC_CLK_TS	PTB3	35	0	1	X	808	0101b
		PTC19	83	0	1	X	808	0110b
GMAC_MII_RGMII_RX_CLK	GMAC_CLK_TS	PTB22	54	0	1	X	812	0111b
		PTB26	58	0	1	X	812	0101b
		PTC16	80	0	1	X	812	0110b

Table 127. GMAC_0 MII clock configuration (for S32K388/S32K389)

Source Clock	Destination clock	Port	SIUL2					
			MSCRa	MSCR fields			IMCRb	IMCR[SSS]
				OBE	IBE	SSS		
GMAC0_MII_RMII_TX_CLK	GMAC0_CLK_TX	PTC0	64	0	1	X	808	0111b
	GMAC0_CLK_RX							
	GMAC0_CLK_TX	PTD6	102	0	1	X	808	0010b
	GMAC0_CLK_RX							
	GMAC0_CLK_TX	PTD11	107	0	1	X	808	0001b
	GMAC0_CLK_RX							
	GMAC0_CLK_TX	PTD12	108	0	1	X	808	0011b
	GMAC0_CLK_RX							
GMAC0_MII_RMII_TX_CLK	GMAC_CLK_TS	PTC0	64	0	1	X	808	0111b
		PTD6	102	0	1	X	808	0010b
		PTD11	107	0	1	X	808	0001b

Table continues on the next page...

Table 127. GMAC_0 MII clock configuration (for S32K388/S32K389) (continued)

Source Clock	Destination clock	Port	SIUL2					
			MSCRa	MSCR fields			IMCRb	IMCR[SSS]
				OBE	IBE	SSS		
		PTD12	108	0	1	X	808	0011b
GMAC0_MII_RX_CLK	GMAC_CLK_TS	PTB26	58	0	1	X	812	0101b
		PTC1	65	0	1	X	812	0011b
		PTD5	101	0	1	X	812	0010b
		PTD10	106	0	1	X	812	0001b

Table 128. GMAC_0 RGMII clock configuration (for S32K388/S32K389)

Source clock	Destination clock	Port	SIUL2					
			MSCRa	MSCR fields			IMCRb	IMCR[SSS]
				OBE	IBE	SSS		
GMAC0_MII_RMII_RGMII_TX_CLK	GMAC0_CLK_TX	PTB3	35	0	1	X	808	0101b
	GMAC0_CLK_RX							
	GMAC0_CLK_TX	PTC1	65	0	1	X	808	0111b
	GMAC0_CLK_RX							
	GMAC0_CLK_TX	PTC19	83	0	1	X	808	0110b
	GMAC0_CLK_RX							
GMAC0_MII_RMII_RGMII_TX_CLK	GMAC_CLK_TS	PTB3	35	0	1	X	808	0101b
		PTC1	65	0	1	X	808	0111b
		PTC19	83	0	1	X	808	0110b
GMAC0_MII_RGMII_RX_CLK	GMAC_CLK_TS	PTB22	54	0	1	X	812	0111b
		PTC16	80	0	1	X	812	0110b

Table 129. GMAC_1 RGMII clock configuration (for S32K388/S32K389)

Source Clock	Destination clock	Port	SIUL2					
			MSCRa	MSCR fields			IMCRb	IMCR[SSS]
				OBE	IBE	SSS		
GMAC1_MII_RMII_RGMII_TX_CLK	GMAC1_CLK_TX	PTB3	35	0	1	X	975	0010b
	GMAC1_CLK_RX							
	GMAC1_CLK_TX	PTC1	65	0	1	X	975	0001b
	GMAC1_CLK_RX							
GMAC1_MII_RMII_RGMII_TX_CLK	GMAC_CLK_TS	PTB3	35	0	1	X	975	0010b
		PTC1	65	0	1	X	975	0001b
GMAC1_MII_RGMII_RX_CLK	GMAC_CLK_TS	PTD10	106	0	1	X	962	0001b

24.6.1.1.4 EMAC clocking

Clocking details of EMAC for different modes is described in this section.

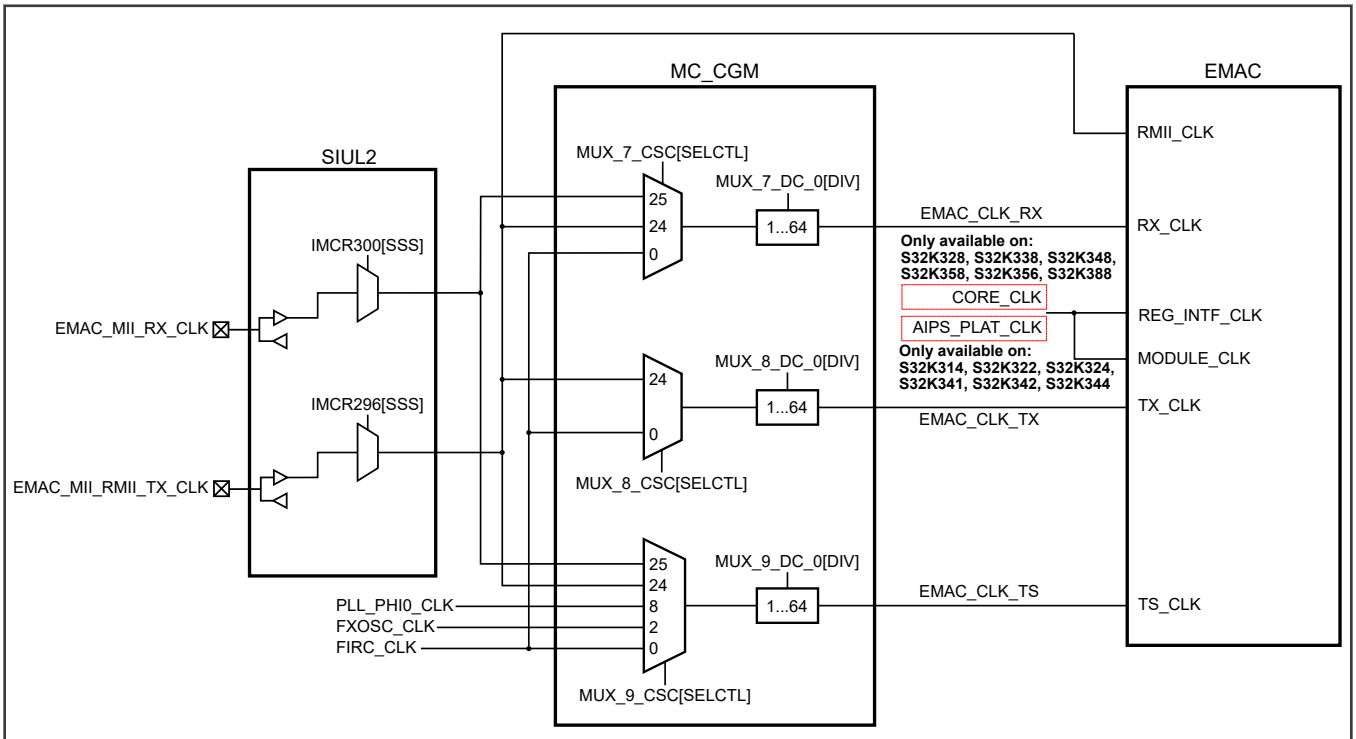


Figure 79. EMAC clocking

NOTE

EMAC operates only in Clock options A, B, A+, and A++, since the module clock becomes lower than the protocol clock (RMII/MII clocks) in other modes.

NOTE

- See the section "Feature comparison" in this reference manual's "Introduction" chapter for details on this module's availability on your chip variant.
- For MC_CGM input sources for different variants, see clocking diagram in 'Clocking Overview' section in 'Clocking' chapter

PLL_AUX should be used for the GMAC TX and TS clocks to support 1 Gbps Ethernet operation.

PLL_AUX should operate in integer mode to meet the RGMII clock accuracy requirements.

24.6.1.1.4.1 EMAC RMII clocking

The following table shows the EMAC RMII clocking, and the related table shows the SIUL2 clock signal configuration for RMII.

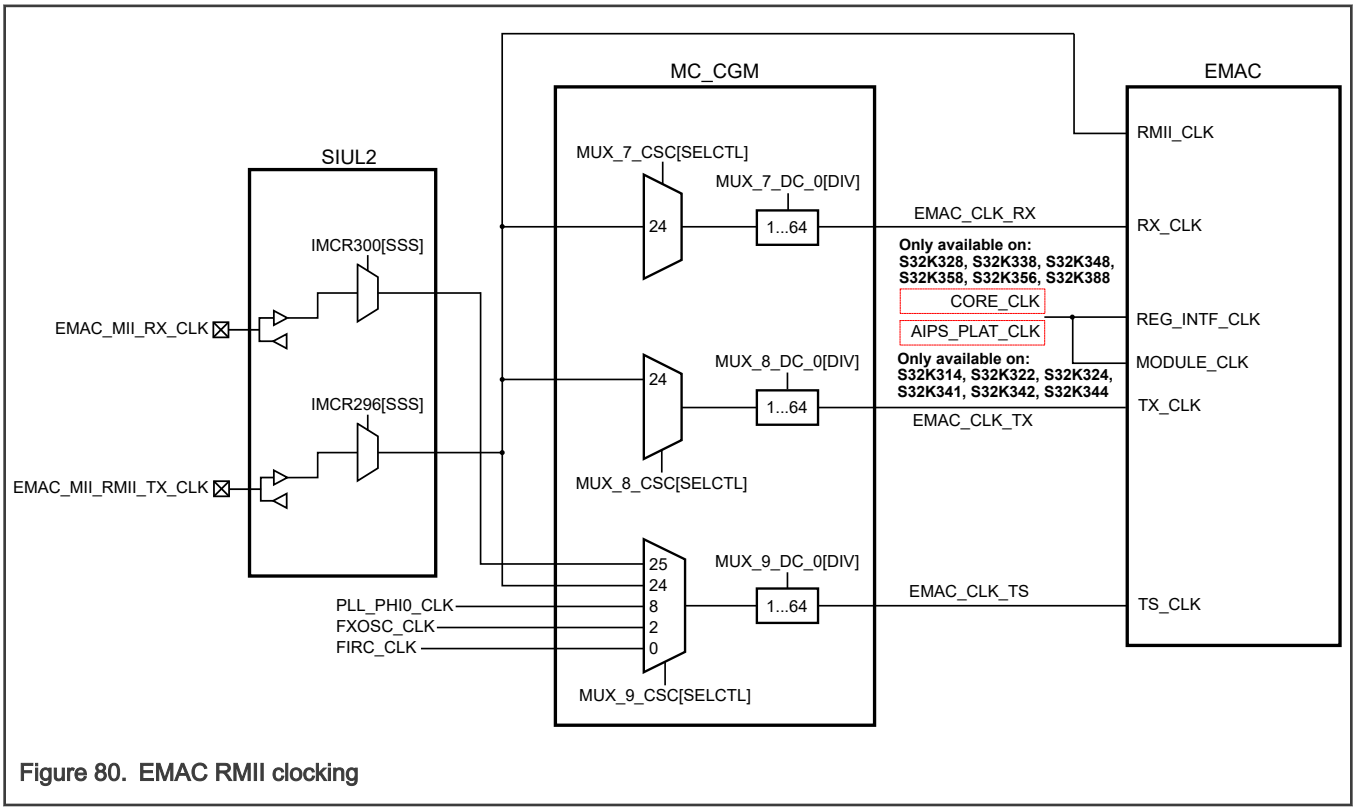


Figure 80. EMAC RMII clocking

NOTE

- See the IOMUX file for your chip variant, attached to this document for details on the ports that support this function.
- The value of CGM divider depends on mode of working. For example, if 25 MHz mode is selected, then the divider value is 2, if 2.5 MHz mode is selected then the divider value is 20.

24.6.1.1.4.2 EMAC MII clocking

The following figure shows the EMAC MII clocking, and the related table shows the SIUL2 clock signal configuration for MII.

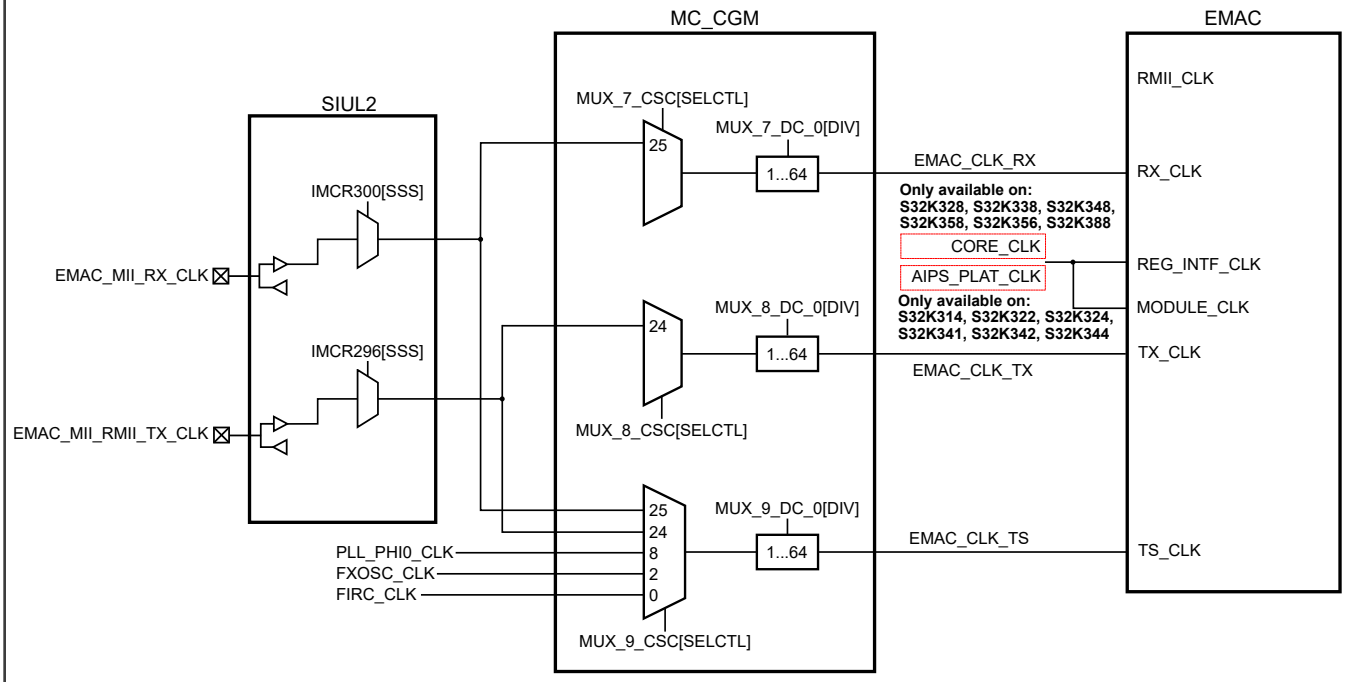
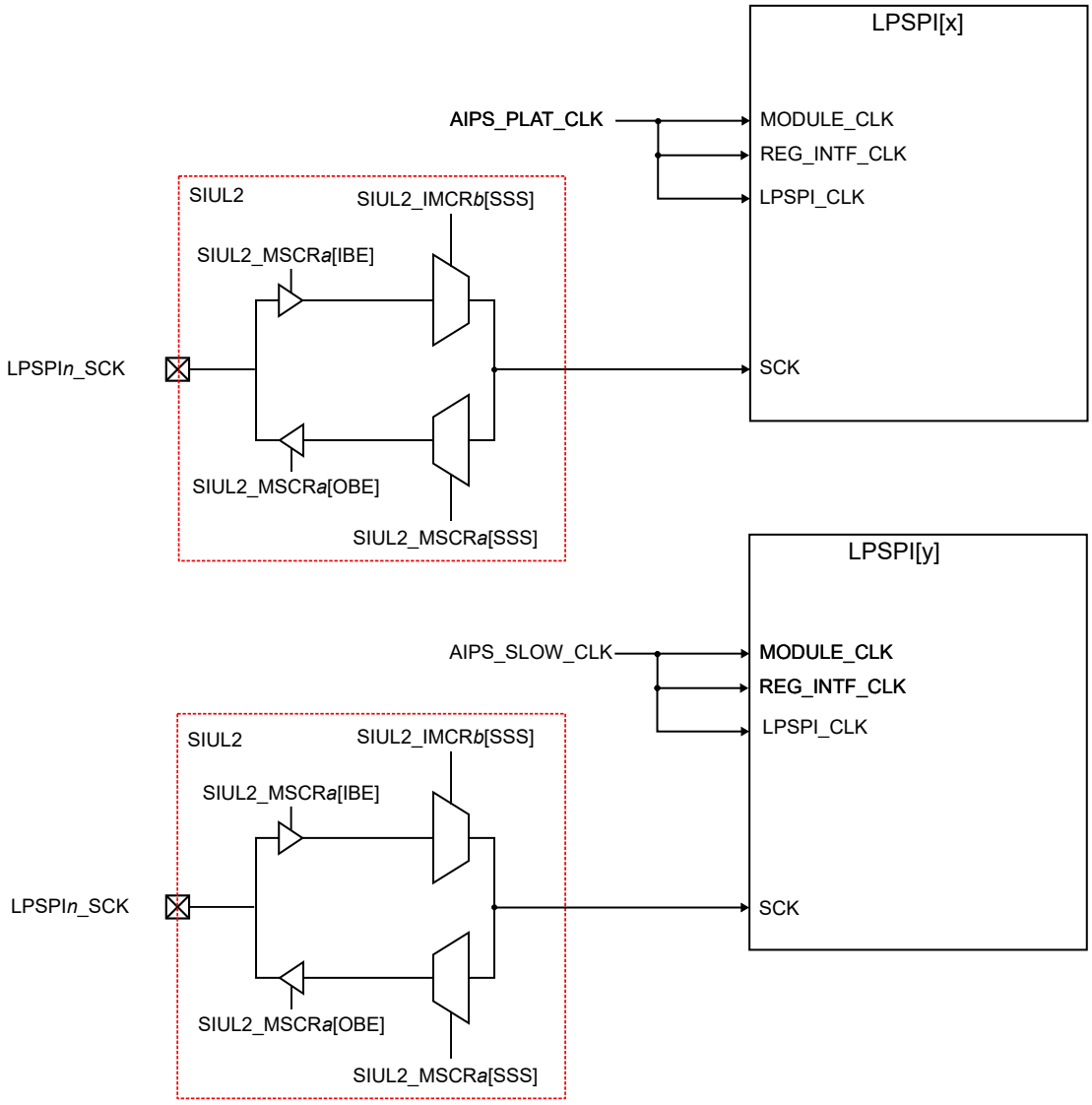


Figure 81. EMAC MII clocking

NOTE

- See the IOMUX file for your chip variant, attached to this document for details on the ports that support this function.
- The value of CGM divider depends on mode of working. For example, if 25 MHz mode is selected, then the divider value is 2, if 2.5 MHz mode is selected then the divider value is 20.

24.6.1.1.5 LPSPI*n* clocking



Note:
For all variants except S32K389 x=0 and y= 1...5
For S32K389 x=0...5

Figure 82. LPSPI*n* clocking

NOTE

- See the section "Feature comparison" in this reference manual's "Introduction" chapter for details on this module's availability on your chip variant.
- For MC_CGM input sources for different variants, see clocking diagram in 'Clocking Overview' section in 'Clocking' chapter

24.6.1.1.6 LPUART*n* clocking

The following figure shows the LPUART*n* clocking configuration, and the related table shows LPUART use case baud rates.

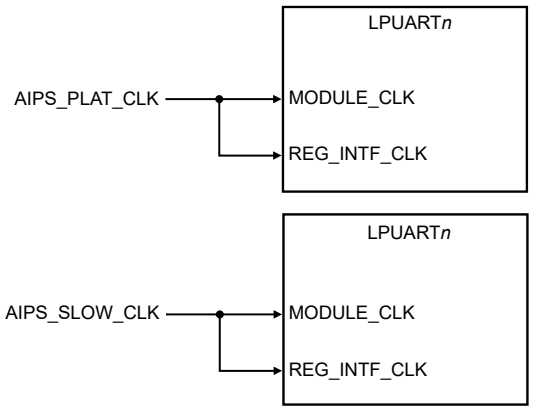


Figure 83. LPUART*n* clocking

See the following table to get correct configuration for each instance in your chip variant.

Table 130. LPUART*n* instance clocking

Variant	LPUART*n* instance	Description
For S32K344, S32K324, and S32K314	16	LPUART [0] and [8] is clocked by AIPS_PLAT_CLK LPUART [1:7] and [9:15] is clocked by AIPS_SLOW_CLK
For S32K312	8	LPUART [0] is clocked by AIPS_PLAT_CLK LPUART [1:7] is clocked by AIPS_SLOW_CLK
For S32K42, S32K341, S32K322, S32K311, and S32K310	4	LPUART [0] and [1] is clocked by AIPS_PLAT_CLK LPUART [2:3] is clocked by AIPS_SLOW_CLK
For S32K358, S32K356, S32K348, S32K338, and S32K328	16	LPUART [0], [1], and [8] is clocked by AIPS_PLAT_CLK LPUART [2:7] and [9:15] is clocked by AIPS_SLOW_CLK
For S32K388 and S32K389	16	LPUART [0:15] is clocked by AIPS_PLAT_CLK

NOTE

See the section "Feature comparison" in this reference manual's "Introduction" chapter for details on this module's availability on your chip variant.

Table 131. LPUART baud rate calculation

Required baud rate (bps)	LPUART_CLK (MHz)	OSR	SBR[12:0]	Calculated baud rate (bps) [1]
8192	40	4	976	8196
8192	80	15	610	8196
8192	48	15	366	8196
115200	48	15	26	115384
115200	40	7	43	116279
19200	80	4	833	19207
19200	40	7	260	19230
38400	40	7	130	38461

1. MODULE_CLK ÷ (LPUART.BAUD[SBR] × (LPUART.BAUD[OSR] + 1))

24.6.1.1.7 FlexIO clocking

The following figure shows the FlexIO clocking interface. The related two tables show the FlexIO baud rate use cases.

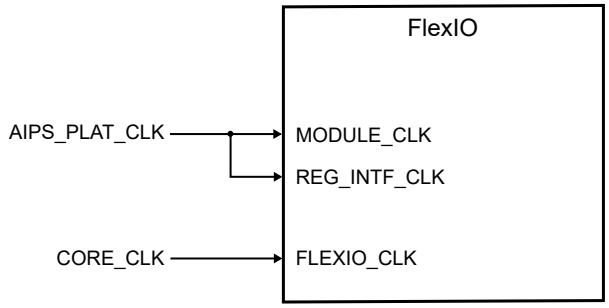


Figure 84. FlexIO clocking

NOTE

See the section "Feature comparison" in this reference manual's "Introduction" chapter for details on this module's availability on your chip variant.

The following tables describes an example how to calculate baud rate for a given frequency. For actual frequency used in your chip, see System clock frequency limitations.

Table 132. FlexIO baud rate calculation (FlexIO.TIMCFG*n*[TIMDEC] = 101b)

FLEXIO_CLK (CORE_CLK)	Required baud rate	TIMCMP*n*[CMP]		Theoretical baud rate [1]	Bit duration	Observed baud rate
		Hex	Decimal			
88 MHz	9600	0010h	16	10110.29	101.33 µs	9868
88 MHz	19200	0007h	7	21484.37	47.44 µs	21079
88 MHz	57600	0001h	1	85937.50	11.66 µs	85763
88 MHz	115200	0000h	0	171875.00	5.88 µs	170068

1. Theoretical baud rate = Frequency ÷ (256 × 2 × (TIMCMP*n*[CMP] + 1))

Table 133. FlexIO baud rate calculation (FlexIO.TIMCFG*n*[TIMDEC] = 100b)

FLEXIO_CLK (CORE_CLK)	Required baud rate	TIMCMP*n*[CMP]		Theoretical baud rate[1]	Bit duration	Observed baud rate
		Hex	Decimal			
88 MHz	19200	8Eh	142	19230.77	51.88 µs	19275
88 MHz	57600	2Eh	46	58510.64	16.89 µs	59206
88 MHz	115200	16h	22	119565.21	8.44 µs	118483

1. Theoretical baud rate = Frequency ÷ (16 × 2 × (TIMCMP*n*[CMP] + 1))

24.6.1.1.8 QuadSPI clocking

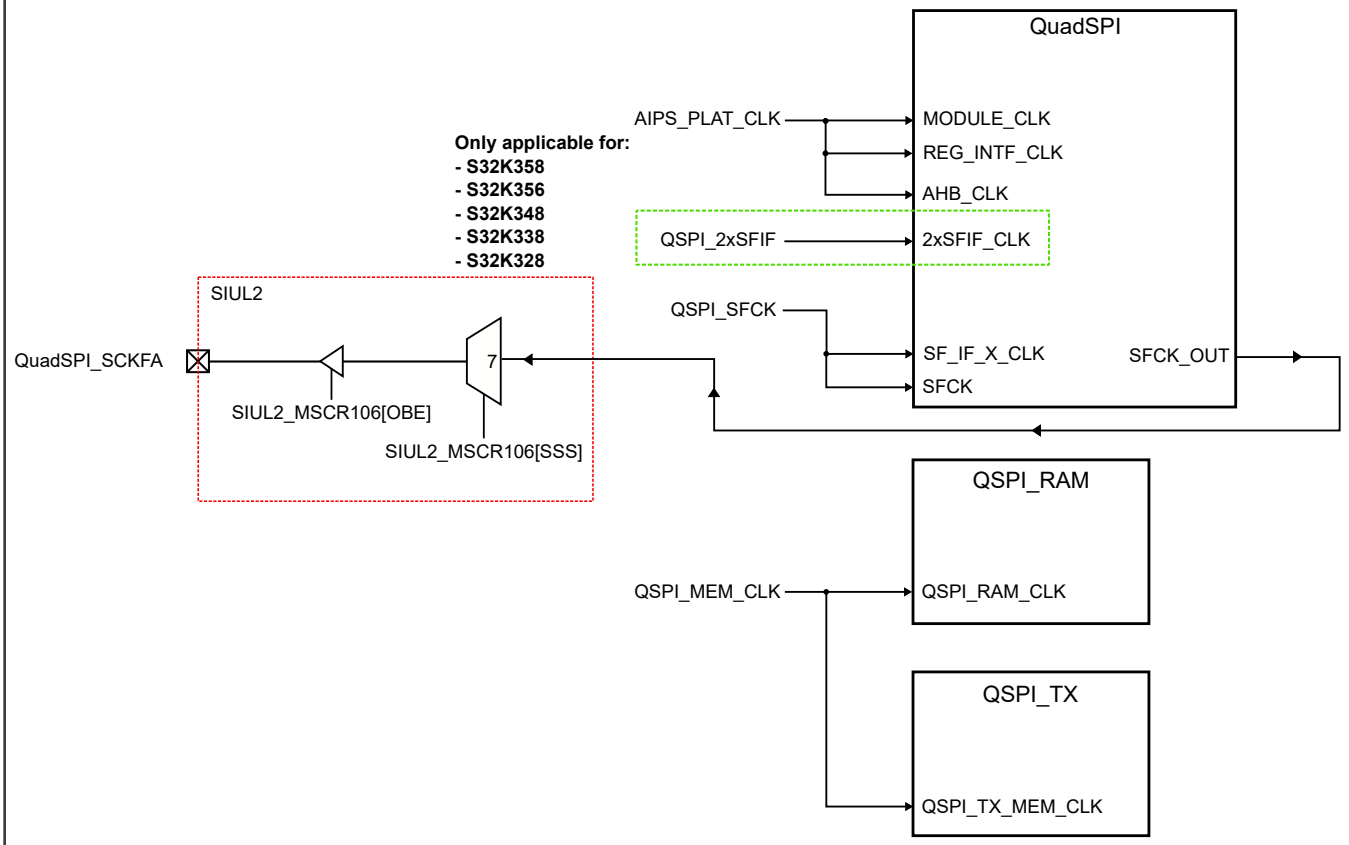


Figure 85. QuadSPI clocking

NOTE

- See the section "Feature comparison" in this reference manual's "Introduction" chapter for details on this module's availability on your chip variant.
- For MC_CGM input sources for different variants, see clocking diagram in 'Clocking Overview' section in 'Clocking' chapter

For S32K358, S32K356, S32K348, S32K338, and S32K328, QSPI_SFCK will be generated by a fixed 1:2 divider from the QSPI_2xSFIF.

NOTE

See System clocking configurations for details.

24.6.1.1.9 SAI*n* clocking

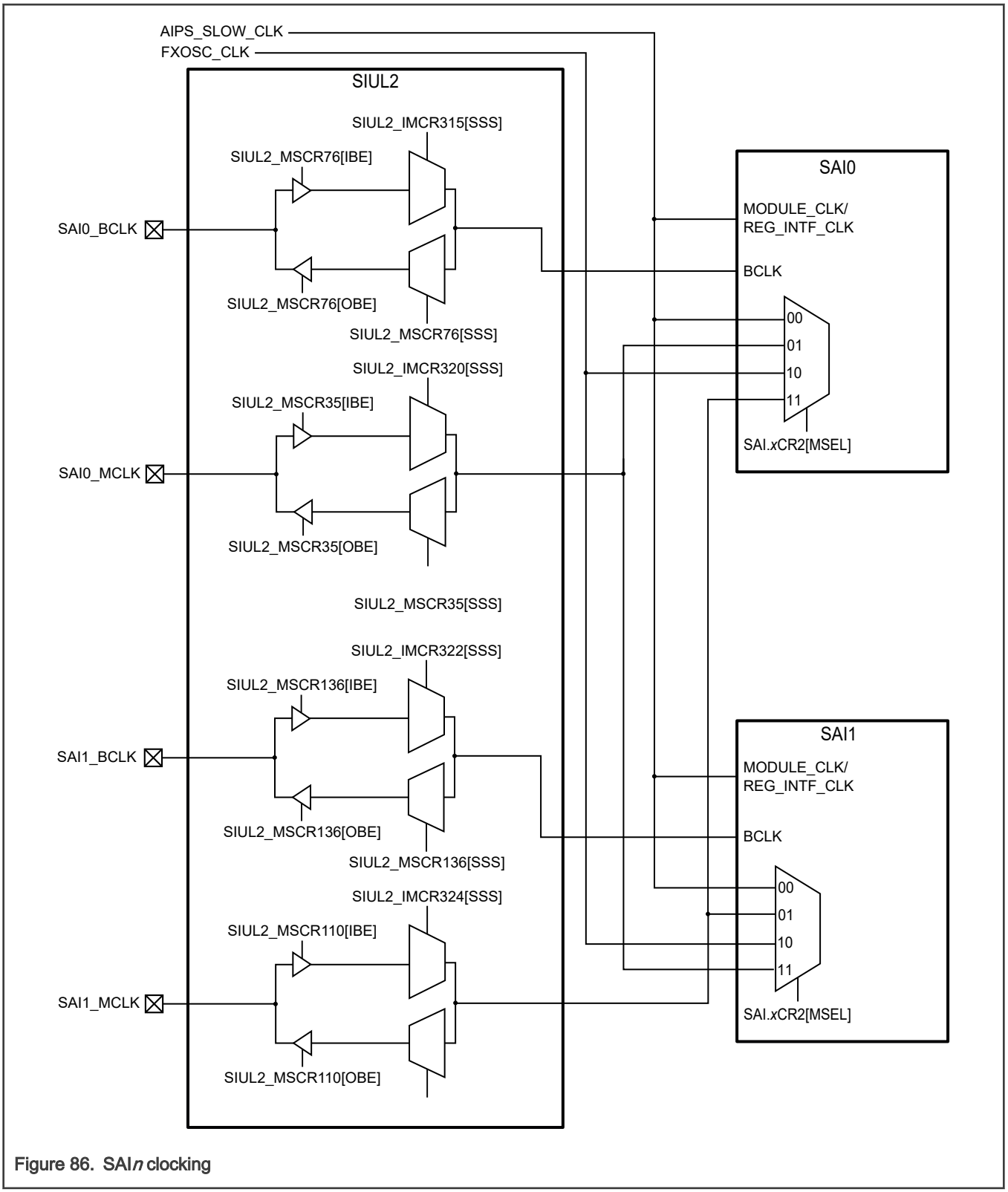


Figure 86. SAI*n* clocking

NOTE

- See the section "Feature comparison" in this reference manual's "Introduction" chapter for details on this module's availability on your chip variant.
- For MC_CGM input sources for different variants, see clocking diagram in 'Clocking Overview' section in 'Clocking' chapter

NOTE

See the IOMUX file for your chip variant, attached to this document for details on the ports that support this function.

Internally generated MCLK is not supported on the S32K3xx chip family.

The Second PLL will be a clock source for the SAI Master Clock.

24.6.1.1.10 uSDHC clocking

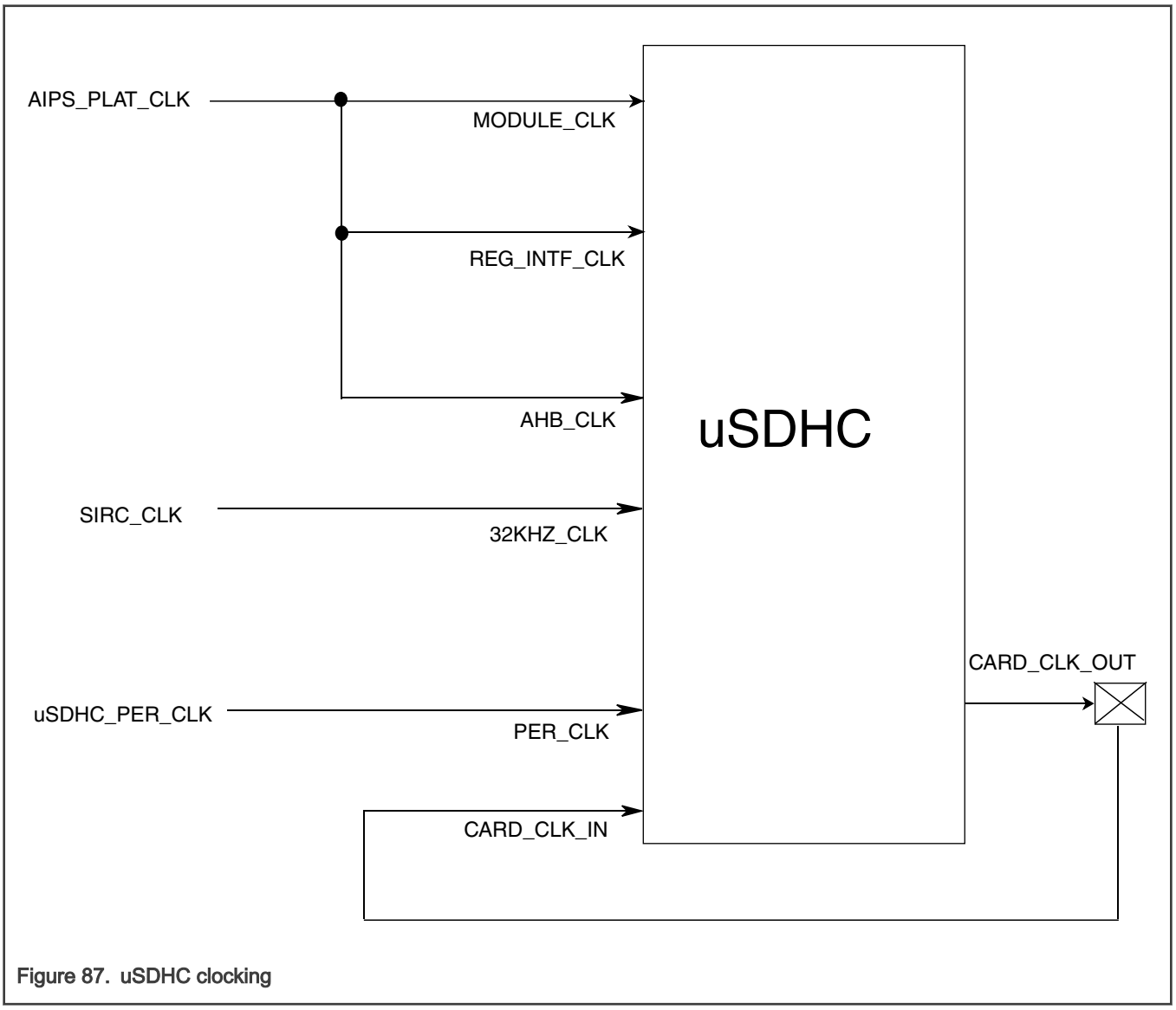


Figure 87. uSDHC clocking

Clock multiplexer provides the clock for the uSDHC module with the following options:

1. FIRC_CLK
2. FXOSC_CLK

3. PLL_PHI1_CLK
4. PLL_AUX_PHI2_CLK

NOTE

- See section "Feature comparison" in this reference manual's "Introduction" chapter for details on this module's availability on your chip variant.
- As MC_CGM_MUX14 have max input frequecny limited to 240 MHz, so while sourcing uSDHC_PER_CLK from PLL_PHI1_CLK, QSPI can run only on 60 Mhz max frequency.

24.6.1.2 System modules

Figure 88 shows the REG_INTF_CLK and MODULE_CLK connections, and Table 134 shows the REG_INTF_CLK and MODULE_CLK signals used by these modules. Any module diagram that does not explicitly show a REG_INTF_CLK uses the same source for REG_INTF_CLK as used by MODULE_CLK.

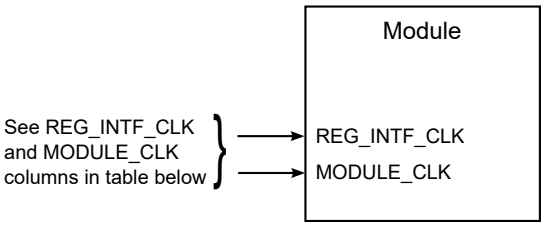


Figure 88. System module clocks

Table 134. System module clocking

Module	MODULE_CLK	REG_INTF_CLK
MSCM	AIPS_PLAT_CLK	AIPS_PLAT_CLK
MCM	AIPS_SLOW_CLK	AIPS_SLOW_CLK
SIUL2	See SIUL2 clocking.	
VIRT_WRAPPER	AIPS_SLOW_CLK	AIPS_SLOW_CLK
AXBS	CORE_CLK	AIPS_PLAT_CLK
DMAMUX	CORE_CLK	CORE_CLK
eDMA	CORE_CLK	AIPS_PLAT_CLK
INTM	AIPS_PLAT_CLK	AIPS_PLAT_CLK
SEMA42	AIPS_PLAT_CLK	AIPS_PLAT_CLK
XBIC	CORE_CLK	AIPS_PLAT_CLK
XRDC	CORE_CLK	AIPS_PLAT_CLK

24.6.1.2.1 SIUL2 clocking

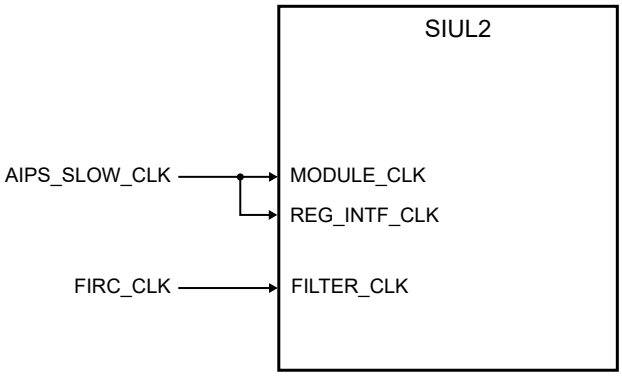


Figure 89. SIUL2 clocking

24.6.1.3 Clocking modules

Figure 90 shows the REG_INTF_CLK and MODULE_CLK connections, and Table 135 shows the REG_INTF_CLK and MODULE_CLK signals used by these modules. Any module diagram that does not explicitly show a REG_INTF_CLK uses the same source for REG_INTF_CLK as used by MODULE_CLK.

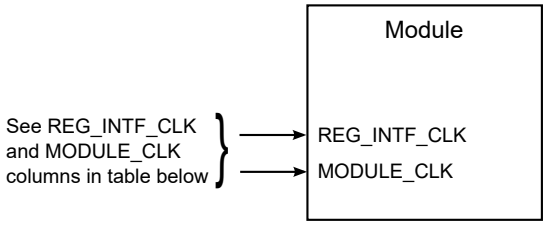


Figure 90. Clocking module clocks

Table 135. Clocking module clocking

Module	MODULE_CLK	REG_INTF_CLK
FXOSC	See FXOSC clocking.	
SXOSC	See SXOSC clocking.	
SIRC	See SIRC clocking.	
FIRC	See FIRC clocking.	
PLLDIG	See PLLDIG clocking.	
MC_CGM	—	AIPS_SLOW_CLK

24.6.1.3.1 FIRC clocking

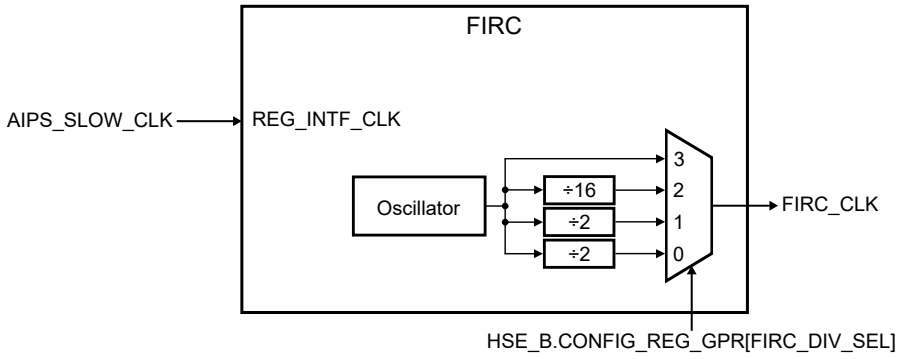


Figure 91. FIRC clocking

24.6.1.3.2 SIRC clocking

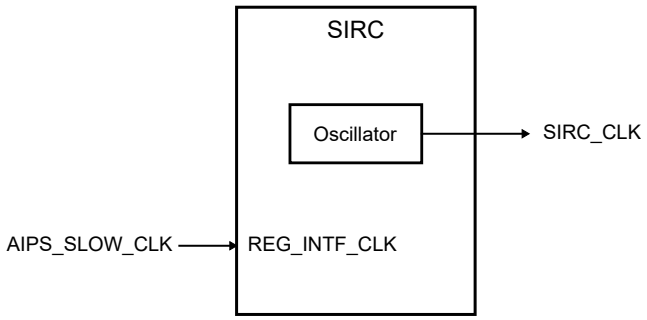


Figure 92. SIRC clocking

24.6.1.3.3 FXOSC clocking

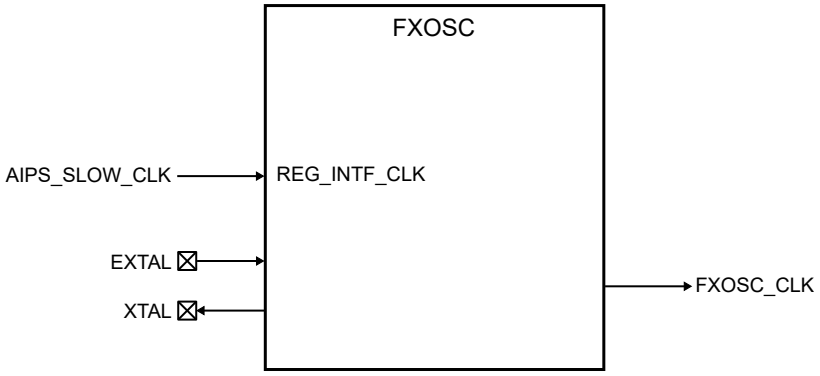


Figure 93. FXOSC clocking

24.6.1.3.4 SXOSC clocking

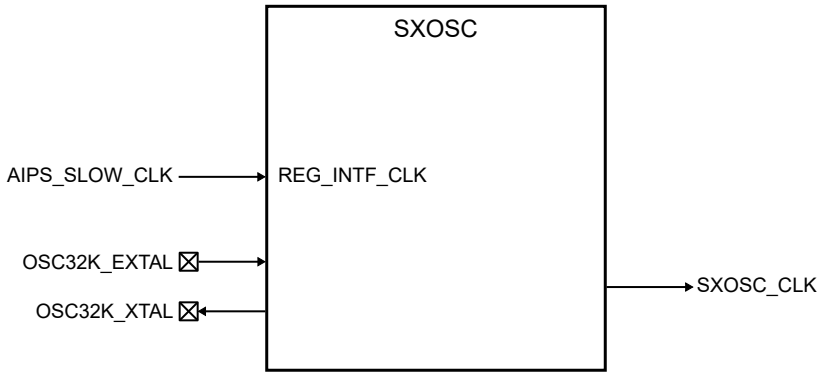


Figure 94. SXOSC clocking

24.6.1.3.5 PLLDIG clocking

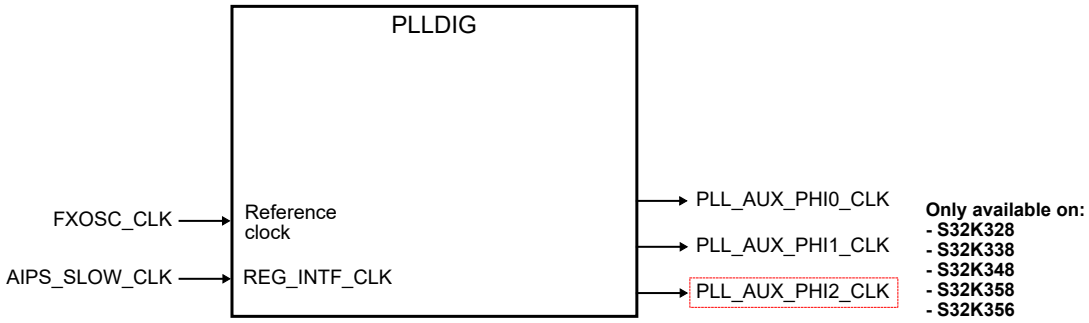


Figure 95. PLLDIG clocking

24.6.1.4 Reset modules

Figure 96 shows the REG_INTF_CLK and MODULE_CLK connections, and Table 136 shows the REG_INTF_CLK and MODULE_CLK signals used by these modules. Any module diagram that does not explicitly show a REG_INTF_CLK uses the same source for REG_INTF_CLK as used by MODULE_CLK.

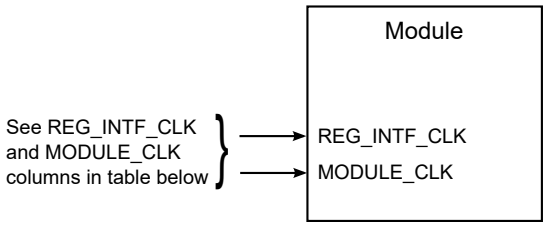


Figure 96. Reset module clocks

Table 136. Reset module clocking

Module	MODULE_CLK	REG_INTF_CLK
MC_RGM	FIRC_CLK	FIRC_CLK

24.6.1.5 Security modules

Figure 97 shows the REG_INTF_CLK and MODULE_CLK connections, and Table 137 shows the REG_INTF_CLK and MODULE_CLK signals used by these modules. Any module diagram that does not explicitly show a REG_INTF_CLK uses the same source for REG_INTF_CLK as used by MODULE_CLK.

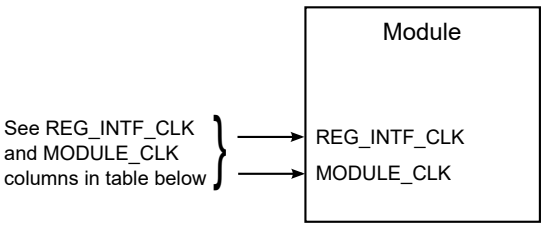


Figure 97. Security module clocks

Table 137. Security module clocking

Module	MODULE_CLK	REG_INTF_CLK
HSE_B	See HSE_B clocking .	
MU	AIPS_SLOW_CLK	AIPS_SLOW_CLK
DCM	DCM_CLK	DCM_CLK
AES_ACCEL	See ACE_ACCEL clocking.	

24.6.1.5.1 HSE_B clocking

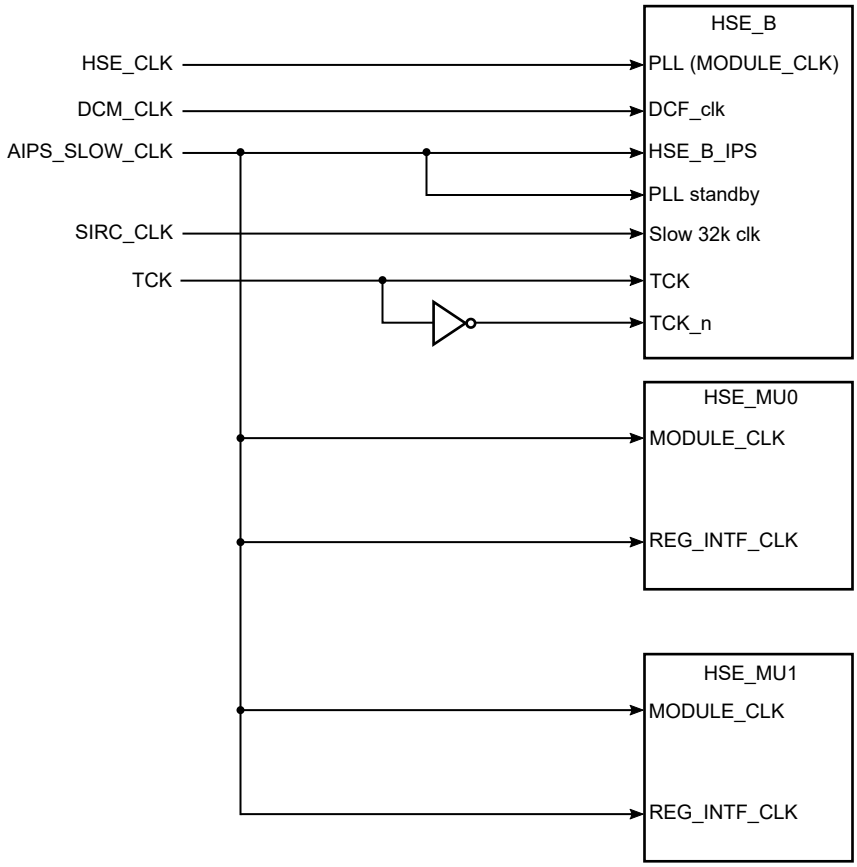


Figure 98. HSE_B clocking

NOTE

The clock frequency relationship between TCK and HSE_CLK clocks for HSE_B must be a minimum ratio of 1:1.5. For example, if HSE_CLK equals 80 MHz, then TCK must be less than or equal to 53 MHz (80 MHz ÷ 1.5).

24.6.1.5.2 ACE_ACCEL clocking

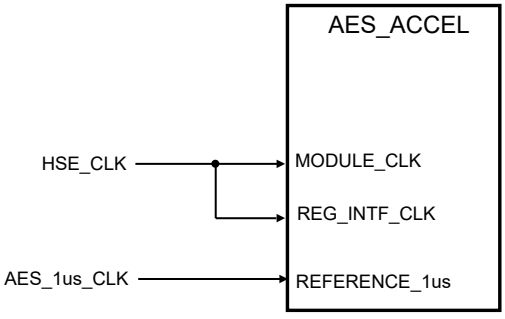


Figure 99. ACE_ACCEL clocking

24.6.1.6 Power-management modules

Figure 100 shows the REG_INTF_CLK and MODULE_CLK connections, and Table 138 shows the REG_INTF_CLK and MODULE_CLK signals used by these modules. Any module diagram that does not explicitly show a REG_INTF_CLK uses the same source for REG_INTF_CLK as used by MODULE_CLK.

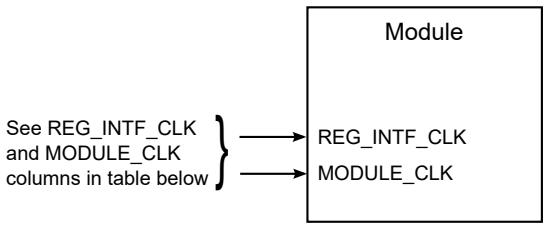


Figure 100. Power-management module clocks

Table 138. Power-management module clocking

Module	MODULE_CLK	REG_INTF_CLK
PMC	AIPS_SLOW_CLK	AIPS_SLOW_CLK
MC_ME	AIPS_SLOW_CLK	AIPS_SLOW_CLK
MC_PCU	FIRC_CLK	FIRC_CLK
WKPU	AIPS_SLOW_CLK	AIPS_SLOW_CLK

24.6.1.7 Safety modules

Figure 101 shows the REG_INTF_CLK and MODULE_CLK connections, and Table 139 shows the REG_INTF_CLK and MODULE_CLK signals used by these modules. Any module diagram that does not explicitly show a REG_INTF_CLK uses the same source for REG_INTF_CLK as used by MODULE_CLK.

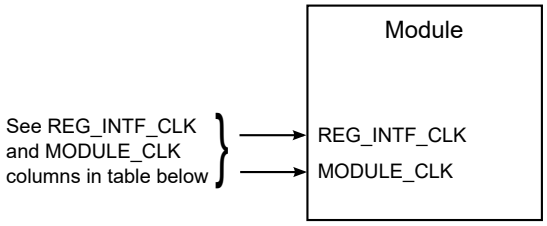


Figure 101. Safety module clocks

Table 139. Safety module clocking

Module	MODULE_CLK	REG_INTF_CLK
EIM	AIPS_PLAT_CLK	AIPS_PLAT_CLK
ERM	See ERM clocking.	
FCCU	See FCCU clocking.	
STCU2	See STCU2 clocking.	

Table continues on the next page...

Table 139. Safety module clocking (continued)

Module	MODULE_CLK	REG_INTF_CLK
REG_PROT	AIPS_SLOW_CLK	AIPS_SLOW_CLK
CMU_FC	AIPS_SLOW_CLK	AIPS_SLOW_CLK
CMU_FM	AIPS_SLOW_CLK	AIPS_SLOW_CLK
CRC	AIPS_PLAT_CLK	AIPS_PLAT_CLK

24.6.1.7.1 FCCU clocking

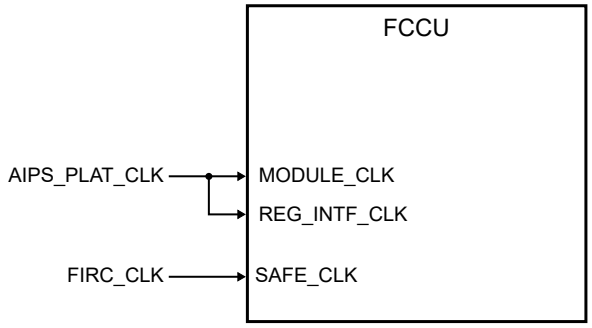


Figure 102. FCCU clocking

24.6.1.7.2 STCU2 clocking

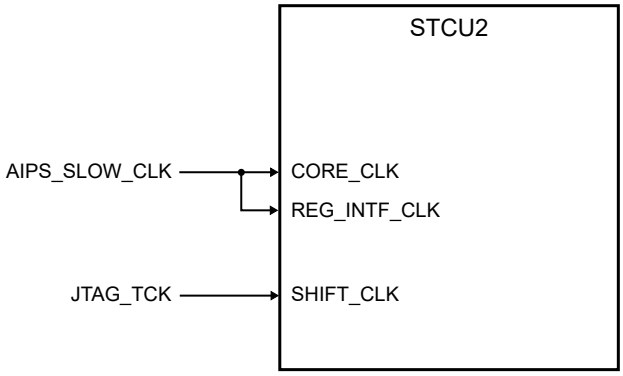


Figure 103. STCU2 clocking

24.6.1.7.3 ERM clocking

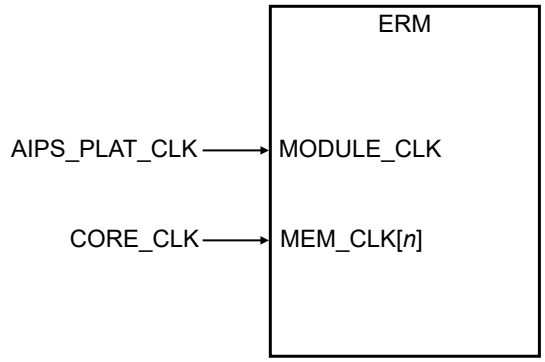


Figure 104. ERM clocking

NOTE

MEM_CLK[20:23] are not used. Source clock for MEM_CLK[0:19] is CORE_CLK.

24.6.1.8 ADC and motor control modules

Figure 105 shows the REG_INTF_CLK and MODULE_CLK connections, and Table 140 shows the REG_INTF_CLK and MODULE_CLK signals used by these modules. Any module diagram that does not explicitly show a REG_INTF_CLK uses the same source for REG_INTF_CLK as used by MODULE_CLK.

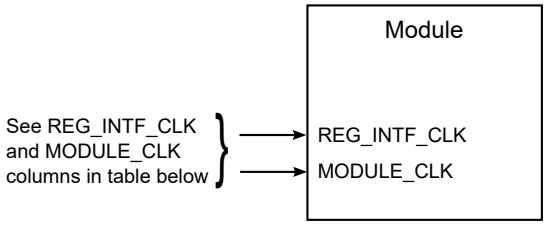


Figure 105. Motor control module clocks

Table 140. Motor control module clocking

Module	MODULE_CLK	REG_INTF_CLK
ADC	See ADC*n* clocking.	
LCU	CORE_CLK	
eMIOS	See eMIOS*n* clocking.	
BCTU	See BCTU clocking.	
TRGMUX	AIPS_SLOW_CLK	
TSPC	AIPS_SLOW_CLK	

24.6.1.8.1 ADC*n* clocking

The following figure shows ADC*n* clocking configuration.

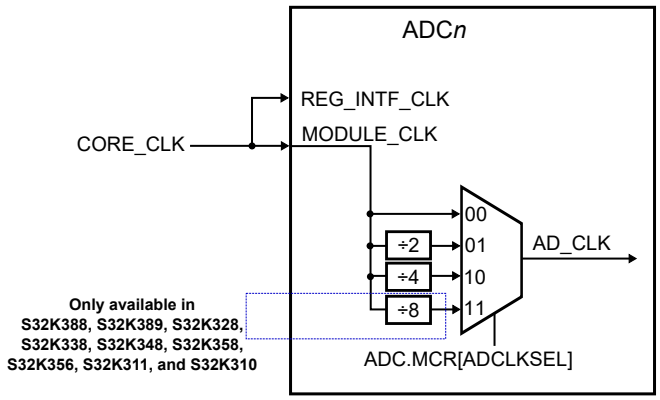


Figure 106. ADC*n* clocking

NOTE

See the section "Feature comparison" in this reference manual's "Introduction" chapter for details on this module's availability on your chip variant.

The prescaler can be bypassed when using FIRC_CLK as the source (see MC_CGM mux 0 clocks for FIRC_CLK use details). The prescaler must be controlled such that the AD_CLK frequency is less than or equal to 120 MHz for S32K388/S32K389/S32K358/S32K356/S32K348/S32K338/S32K328/S32K312/S32K311/S32K310. For other S32K3xx products maximum frequency supported is 80 Mhz.

The minimum operating speed of AD_CLK is 6 MHz using the following configuration:

1. Use FIRC_CLK (48 MHz) as clock source (MC_CGM.MUX_0_CSC[SELCTL] equal to 0000b).
2. Divide FIRC_CLK by 2 for CORE_CLK speed (MC_CGM.MUX_0_DC_0[DIV] equal to 1 (FIRC_CLK divide by 2 = 24 MHz)).
3. Write ADC.MCR[ADCCLKSEL] equal to 10b to divide the FIRC_CLK further by 4 (AD_CLK = 6 MHz).

However, at this lower speed, the ADC*n* results will be degraded.

24.6.1.8.2 eMIOS*n* clocking

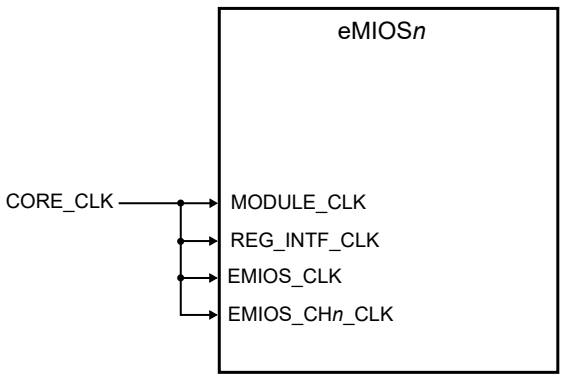


Figure 107. eMIOS*n* clocking

NOTE

See the section "Feature comparison" in this reference manual's "Introduction" chapter for details on this module's availability on your chip variant.

24.6.1.8.3 BCTU clocking

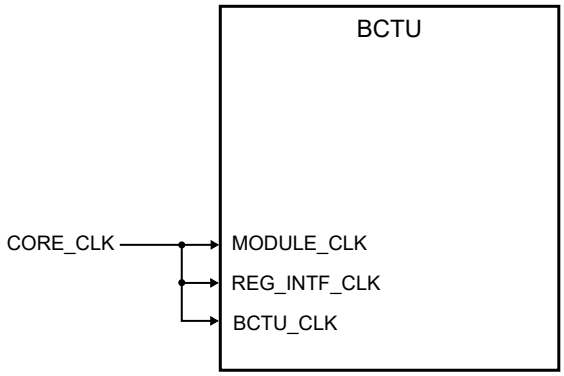


Figure 108. BCTU clocking

24.6.1.9 Timer modules

Figure 109 shows the REG_INTF_CLK and MODULE_CLK connections, and Table 141 shows the REG_INTF_CLK and MODULE_CLK signals used by these modules. Any module diagram that does not explicitly show a REG_INTF_CLK uses the same source for REG_INTF_CLK as used by MODULE_CLK.

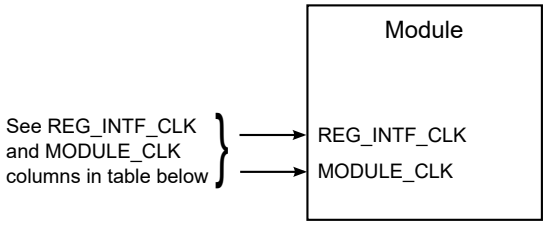


Figure 109. Timer module clocks

Table 141. Timer module clocking

Module	MODULE_CLK	REG_INTF_CLK
PIT	See PIT*n* clocking.	
SWT	See SWT*n* clocking.	
STM*n*	STM*n*_CLK	STM*n*_CLK
RTC	See RTC clocking.	

24.6.1.9.1 PIT*n* clocking

The following figure shows the PIT*n* clocking configuration. The related tables show the use case configuration.

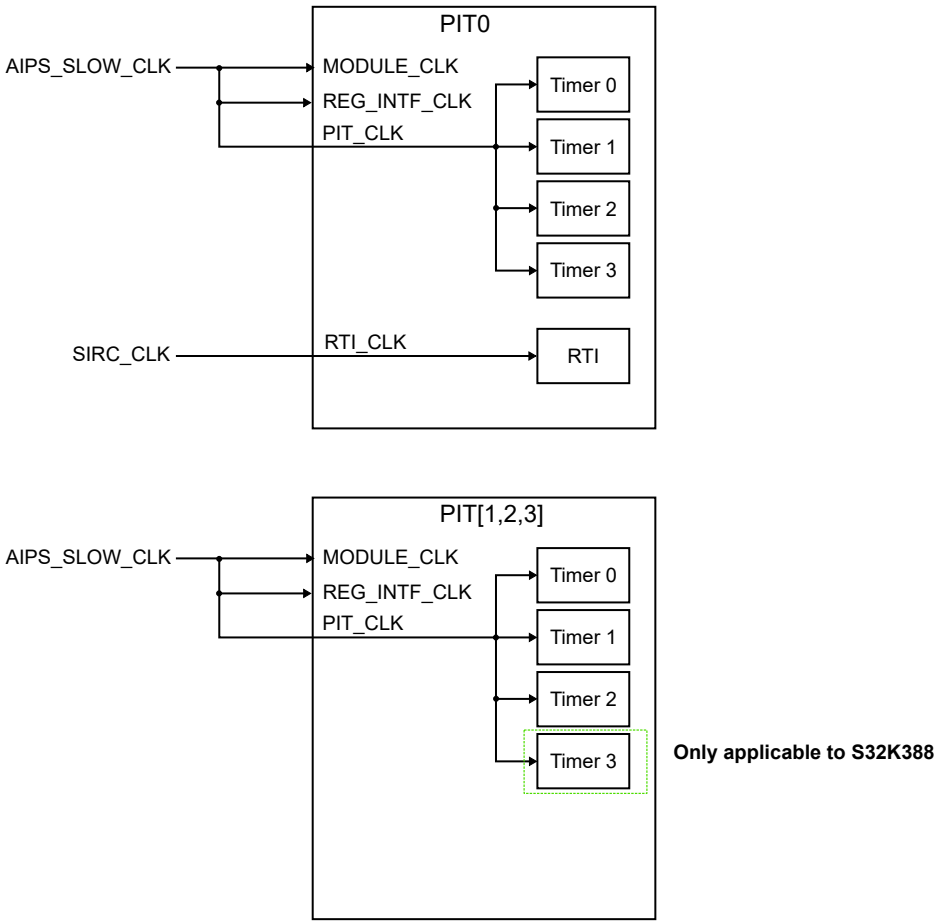


Figure 110. PIT*n* clocking

NOTE

See the section "Feature comparison" in this reference manual's "Introduction" chapter for details on this module's availability on your chip variant.

Table 142. PIT0 modes of operation

MC_ME.PRTN0_COFB1_CLKEN[REQ44]	PIT.MCR[MDIS]	PIT.MCR[MDIS_RTI]	Mode	Application
0	X	X	PIT clock gated (minimum power)	Module clock gated and unused / Standby mode, PIT and RTI unused
1	0	0	Both PIT and RTI enabled	Run mode
1	0	1	PIT running, RTI disabled	Run mode with only PIT active
1	1	0	PIT disabled, RTI enabled	Standby mode with RTI enabled
1	1	1	Both PIT and RTI disabled	Standby mode with RTI disabled

Table 143. PIT[1,2,3] modes of operation

MC_ME.PRTN*n*_COFB1_CLKEN[REQ[45,63,64]] [1]	PIT.MCR[MDIS]	Mode	Application
0	X	PIT clock gated (minimum power)	Module clock gated and unused, Standby mode
1	0	PIT enabled	Run mode
1	0	PIT running	Run mode with PIT active
1	1	PIT disabled	Standby mode

1. PIT1, PIT2, and PIT3 MC_ME partition registers used are:
   - PIT1 - MC_ME.PRTN0_COFB1_CLKEN[REQ45]
   - PIT2 - MC_ME.PRTN1_COFB1_CLKEN[REQ63]
   - PIT3 - MC_ME.PRTN1_COFB1_CLKEN[REQ64]

24.6.1.9.2 SWT*n* clocking

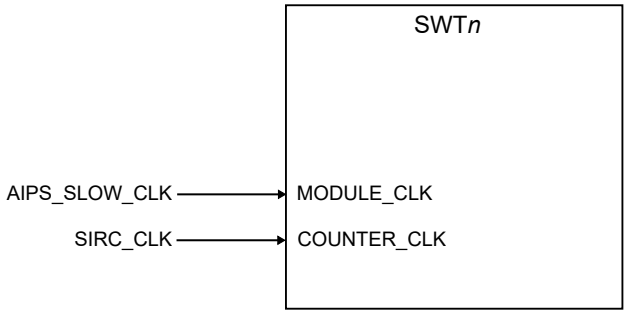


Figure 111. SWT*n* clocking

NOTE

- See the section "Feature comparison" in this reference manual's "Introduction" chapter for details on this module's availability on your chip variant.
- For MC_CGM input sources for different variants, see clocking diagram in 'Clocking Overview' section in 'Clocking' chapter

24.6.1.9.3 RTC clocking

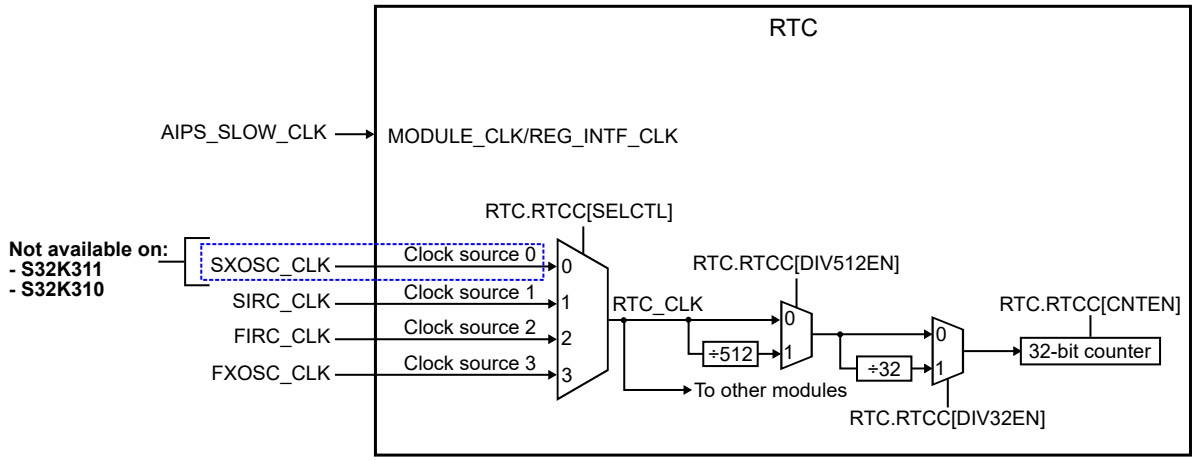


Figure 112. RTC clocking

NOTE

The RTC is available in Standby mode. Although bus clock is gated, the RTC can run on FIRC_CLK, SIRC_CLK, FXOSC_CLK, or SXOSC_CLK.

24.6.1.10 Debug modules

Figure 113 shows the REG_INTF_CLK and MODULE_CLK connections, and Table 144 shows the REG_INTF_CLK and MODULE_CLK signals used by these modules. Any module diagram that does not explicitly show a REG_INTF_CLK uses the same source for REG_INTF_CLK as used by MODULE_CLK.

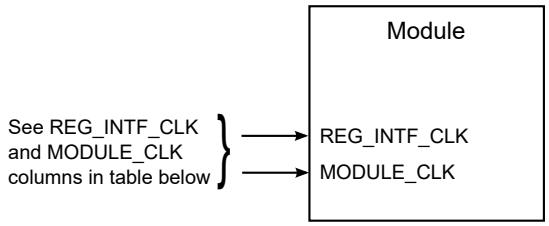


Figure 113. Debug module clocks

Table 144. Debug module clocking

Module	MODULE_CLK	REG_INTF_CLK
JTAGC	See JTAGC clocking.	
JDC	See JDC clocking.	

24.6.1.10.1 JTAGC clocking

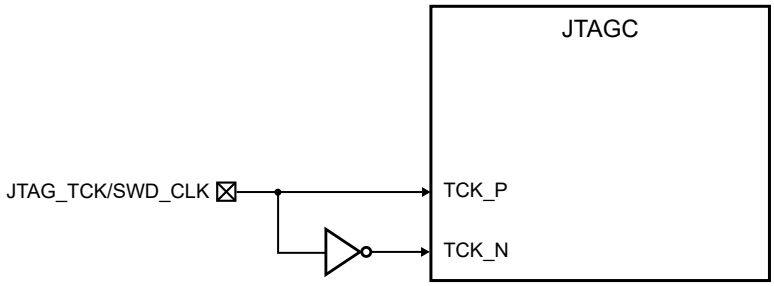


Figure 114. JTAGC clocking

24.6.1.10.2 JDC clocking

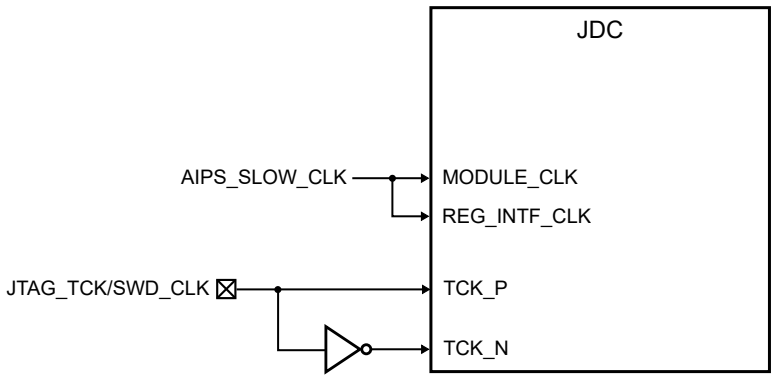


Figure 115. JDC clocking

24.6.1.11 Analog modules

Figure 116 shows the REG_INTF_CLK and MODULE_CLK connections, and Table 145 shows the REG_INTF_CLK and MODULE_CLK signals used by these modules. Any module diagram that does not explicitly show a REG_INTF_CLK uses the same source for REG_INTF_CLK as used by MODULE_CLK.

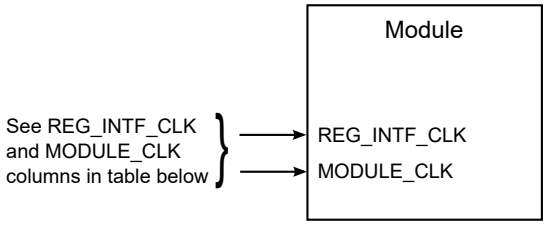


Figure 116. Analog module clocks

Table 145. Analog module clocking

Module	MODULE_CLK	REG_INTF_CLK
LPCMP	See LPCMP*n* clocking.	
Temperature Sensor	AIPS_SLOW_CLK	AIPS_SLOW_CLK

24.6.1.11.1 LPCMP*n* clocking

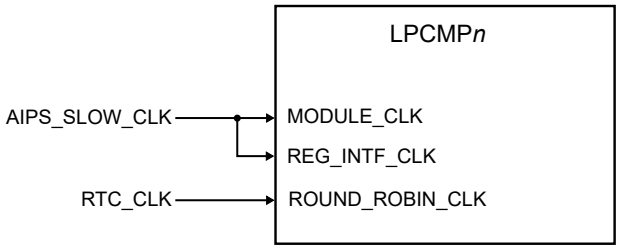


Figure 117. LPCMP*n* clocking

NOTE

See the section "Feature comparison" in this reference manual's "Introduction" chapter for details on this module's availability on your chip variant.

24.6.1.12 Memory modules

Figure 118 shows the REG_INTF_CLK and MODULE_CLK connections, and Table 146 shows the REG_INTF_CLK and MODULE_CLK signals used by these modules. Any module diagram that does not explicitly show a REG_INTF_CLK uses the same source for REG_INTF_CLK as used by MODULE_CLK.

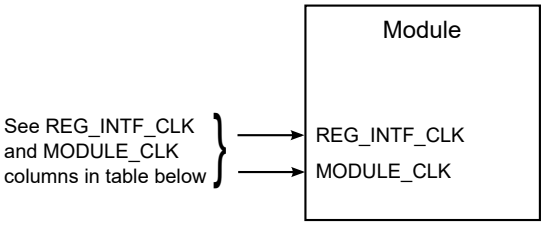


Figure 118. Memory module clocks

Table 146. Memory module clocking

Module	MODULE_CLK	REG_INTF_CLK
PFLASH/ FLASH	CORE_CLK	AIPS_SLOW_CLK
PRAM/ SRAM	CORE_CLK	AIPS_SLOW_CLK

24.6.2 Peripheral data rates

Table 147. Peripheral data rates

Peripheral	Maximum data rate	
	S32K322, S32K342, S32K341, S32K314, S32K324, S32K344, S32K328, S32K338, S32K348, S32K358, S32K356, S32K388, and S32K389	S32K310, S32K311, and S32K312
ADC	See the *S32K3xx Data Sheet* for details.	See the *S32K3xx Data Sheet* for details.
eMIOS[1]	Able to shift PWM edge by 1 ÷ (240 MHz) = 4.17 ns[12].	Able to shift PWM edge by 1 ÷ (120 MHz) = 8.33 ns.

Table continues on the next page...

Table 147. Peripheral data rates (continued)

Peripheral	Maximum data rate	
	S32K322, S32K342, S32K341, S32K314, S32K324, S32K344, S32K328, S32K338, S32K348, S32K358, S32K356, S32K388, and S32K389	S32K310, S32K311, and S32K312
	Able to shift PWM edge by 1 ÷ (160 MHz) = 6.25 ns[2].	
BCTU[3]	BCTU to generate triggers at 160 MHz.	BCTU to generate triggers at 120 MHz.
LCU	Same domain as EMIOS and BCTU.	Same domain as EMIOS and BCTU.
QuadSPI	• Flash memory interface: SDR 120 MHz [4] • Flash memory interface: SDR 125 MHz[2] • DDR and hyperflash not supported[12]	-
EMAC	See the section "Description" in the "Ethernet Media Access Controller (EMAC)" chapter for details.	See the section "Description" in the "Ethernet Media Access Controller (EMAC)" chapter for details.
FlexCAN[5]	8 Mbps	8 Mbps
LPI2C[6]	400 Kbps in fast mode.	400 Kbps in fast mode.
LPSPI[7]	For all variants except S32K389: • LPSPI0 is to have a high clock rate of 20 Mbps • LPSPI1–LPSPI5 can be 10 Mbps For S32K389: • LPSPI0–LPSPI5 can be 20 Mbps	• LPSPI0 is to have a high clock rate of 15 Mbps • LPSPI1–LPSPI3 can be 7.5 Mbps in case of 50:50 percent duty cycle • LPSPI1–LPSPI3 can be 10 Mbps in case of 33:66 or 66:33 percent duty cycle
SAI0/SAI1 (I2S)[8,9]	• Bit rate = 12.288 MHz (12.288 Mbps—bit clock frequency governs the bit rate) • Master clock = 24.576 MHz • SAI0 and SAI1 operate asynchronously to each other	-
LPUART[10]	See the section "Baud rate generation" in the "Low Power Universal Asynchronous Receiver/ Transmitter (LPUART)" chapter and Table 131 for details	See the section "Baud rate generation" in the "Low Power Universal Asynchronous Receiver/ Transmitter (LPUART)" chapter and Table 131 for details
FlexIO[11]	The different protocol data rates supported by FlexIO are listed below. For master mode, max baud rate is FLEXIO_CLK ÷ 4. For slave mode, max baud rate is FLEXIO_CLK ÷ 6. The baud rate is controlled by TIMCMP (lower 8 bits in 8-bit mode and 16 bits in 16-bit mode). • UART: 19200 bps • I2C: 400 Kbps • SPI: 10 Mbps • I2S: 12.288 Mbps	The different protocol data rates supported by FlexIO are listed below. For master mode, max baud rate is FLEXIO_CLK ÷ 4. For slave mode, max baud rate is FLEXIO_CLK ÷ 6. The baud rate is controlled by TIMCMP (lower 8 bits in 8-bit mode and 16 bits in 16-bit mode). • UART: 19200 bps • I2C: 400 Kbps • SPI: 7.5 Mbps • I2S: 12.288 Mbps

Table continues on the next page...

Table 147. Peripheral data rates (continued)

Peripheral	Maximum data rate	
	S32K322, S32K342, S32K341, S32K314, S32K324, S32K344, S32K328, S32K338, S32K348, S32K358, S32K356, S32K388, and S32K389	S32K310, S32K311, and S32K312
Trace	• Fast-speed pins: 120 MHz[4] • Fast-speed pins: 125 MHz[2] • Standard plus-speed pins: 25 MHz	• SWO trace
uSDHC[12]	• eMMC mode: 800 Mbps • SD mode: Maximum of 200 Mbps	-

1. See section "Global Clock Prescaler Submodule (GCP)" in the "Enhanced Modular IO Subsystem (eMIOS)" chapter for details.
2. Only applicable for S32K388/S32K389.
3. See section "Triggers" in the "Enhanced Modular IO Subsystem (eMIOS)" chapter.
4. Only applicable for S32K322, S32K341, S32K342, S32K314, S32K324, S32K344, S32K328, S32K338, S32K348, S32K356, and S32K358.
5. See the section "Protocol timing" in the "CAN (FlexCAN)" chapter for data rate calculation details.
6. See the section "Clocks" in the "Low Power Inter-Integrated Circuit (LPI2C)" chapter for LPI2C_CLK frequency details.
7. See the section "Clocks" in the "Low Power Serial Peripheral Interface (LPSPI)" chapter.
8. SAI is not present in S32K310,S32K311, and S32K12.
9. See the section "SAI clocking" in the "Synchronous Audio Interface (SAI)" chapter and SAI*n* clocking for details.
10. At least one pair of LPUART instances (LPUART0 and LPUART1 for S32K328, S32K338, S32K348, S32K356, and S32K358) support up to 12 Mbps.
11. See the section "Application Information" and "Chip-specific FlexIO information" in the "Flexible I/O (FlexIO)" chapter and FlexIO clocking for baud configuration details.
12. Only applicable for S32K328, S32K338, S32K348, S32K356, and S32K358.

24.6.3 Core and peripheral clock control

The chip provides provisions for core and peripheral clock gating. The next sections describe the details on clock gating possibilities and controls (see "Power Management Controller (PMC)" and "Mode Entry Module (MC_ME)" for details).

24.6.3.1 Clock gating

Application core clocks are gated by individual MC_ME core clock enable bits. Additionally, application cores can be clock gated by executing WFI (see the "Mode Entry Module (MC_ME)" chapter for details).

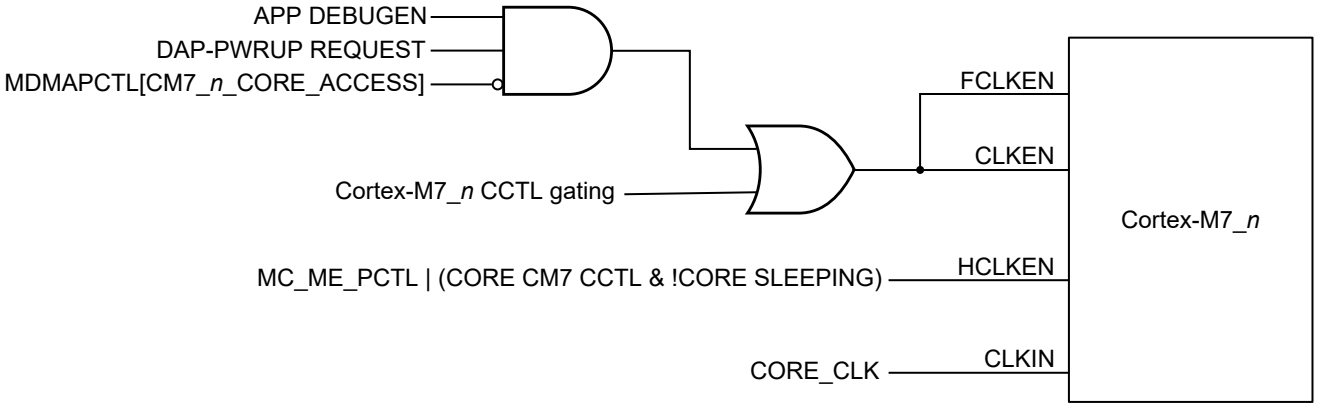


Figure 119. Cortex-M7 core clock gating

To support core debug across functional reset, challenge response done (application debug enable) gating is done so that the debugger can access core debug logic. When the debugger completes its programming with core debug logic, it must program the MDM_AP DAP control bit to shift the control of CLKIN and FCLK to CCTL.

See the section "MDM_AP register descriptions" in the "Debug Subsystem" chapter and the "Memory Map" chapter for details.

There are two cases in which other masters can access the TCM of each core:

- When the TCM is used as system memory—HCLK will always remain on if TCM PCTL is 1.
- In applications where TCM is not used as system memory, TCM PCTL is written to 0. TCM will then function as the core's memory and HCLK will be gated on WFI.

Ensure that the TCM AHBS interface is not accessed with HCLK disabled. In such case, the read/write transactions do not result in any error response.

24.6.3.2 Peripheral clock gating

See the tables in section "Core and peripheral clock control" for the chip partitions, plus peripheral initialization and shutdown details.

24.7 Clocking details

24.7.1 System clock frequency limitations

Table 148. System clock frequency limitations (For S32K388/S32K389)

System clock node	System clock divider	Maximum frequency allowed for S32K388/S32K389	Remarks
CM7_CORE_CLK	MC_CGM.MUX_0_DC_7[DIV]	• 320 MHz	This is the frequency for all cores in the S32K388/S32K389. CM7_CORE_CLK is always greater than or equal to CORE_CLK.
CORE_CLK	MC_CGM.MUX_0_DC_0[DIV]	• 160 MHz	For S32K388/S32K389, CORE_CLK is the frequency used for AXBS interfaces and fast peripherals.

Table continues on the next page...

Table 148. System clock frequency limitations (For S32K388/S32K389) (continued)

System clock node	System clock divider	Maximum frequency allowed for S32K388/S32K389	Remarks
			It does not correspond to the frequency of the Cortex-M7 cores. For CM7_CORE_CLK frequencies > 160 MHz, CORE_CLK is always half of CM7_CORE_CLK (2:1 relation).
AIPS_PLAT_CLK	MC_CGM.MUX_0_DC_1[DIV]	• 80 MHz	AIPS_PLAT_CLK is always less than or equal to CORE_CLK.
AIPS_SLOW_CLK	MC_CGM.MUX_0_DC_2[DIV]	• 40 MHz	AIPS_SLOW_CLK is always less than or equal to AIPS_PLAT_CLK.
HSE_CLK	MC_CGM.MUX_0_DC_3[DIV]	• 160 MHz	HSE_CLK
DCM_CLK	MC_CGM.MUX_0_DC_4[DIV]	• 48 MHz	DCM_CLK
LBIST_CLK	MC_CGM.MUX_0_DC_5[DIV]	• 48 MHz	LBIST clock
QSPI_MEM_CLK	MC_CGM.MUX_0_DC_6[DIV]	• 160 MHz	QSPI_MEM_CLK is always equal to CORE_CLK.

Table 149. System clock frequency limitations (For S32K358, S32K356, S32K348, S32K338, and S32K328)

System clock node	System clock divider	Maximum frequency allowed	Remarks
CORE_CLK	MC_CGM.MUX_0_DC_0[DIV]	• 240 MHz	CORE_CLK[1] is always greater than or equal to AIPS_PLAT_CLK.
AIPS_PLAT_CLK	MC_CGM.MUX_0_DC_1[DIV]	• 120 MHz	AIPS_PLAT_CLK is always less than or equal to CORE_CLK.
AIPS_SLOW_CLK	MC_CGM.MUX_0_DC_2[DIV]	• 60 MHz	AIPS_SLOW_CLK is always less than or equal to AIPS_PLAT_CLK.
HSE_CLK	MC_CGM.MUX_0_DC_3[DIV]	• 120 MHz	When CORE_CLK is equal to or less than 120 MHz, HSE_CLK can be equal to CORE_CLK. When CORE_CLK is higher than 120 MHz, HSE_CLK must be half of the CORE_CLK.
DCM_CLK	MC_CGM.MUX_0_DC_4[DIV]	• 60 MHz	DCM_CLK
LBIST_CLK	MC_CGM.MUX_0_DC_5[DIV]	• 60 MHz	LBIST clock
QSPI_MEM_CLK	MC_CGM.MUX_0_DC_6[DIV]	• 240 MHz	QSPI_MEM_CLK is always equal to CORE_CLK except in 1:1 mode (see Option F - Operation in 1:1 mode with CORE_CLK and AIPS_PLAT_CLK at same speed (For all chips except S32K388/S32K389)).

1. CORE_CLK is the frequency used for the CM7 cores, AXBS interfaces, and fast peripherals.

Table 150. System clock frequency limitations (For S32K344, S32K324, S32K314, S32K342, S32K322, and S32K341)

System clock node	System clock divider	Maximum frequency allowed	Remarks
CORE_CLK	MC_CGM.MUX_0_DC_0[DIV]	• 160 MHz	CORE_CLK[1] is always greater than or equal to AIPS_PLAT_CLK.
AIPS_PLAT_CLK	MC_CGM.MUX_0_DC_1[DIV]	• 80 MHz	AIPS_PLAT_CLK is always less than or equal to CORE_CLK.
AIPS_SLOW_CLK	MC_CGM.MUX_0_DC_2[DIV]	• 40 MHz	AIPS_SLOW_CLK is always less than or equal to AIPS_PLAT_CLK.
HSE_CLK	MC_CGM.MUX_0_DC_3[DIV]	• 120 MHz	When CORE_CLK is equal to or less than 120 MHz, HSE_CLK can be equal to CORE_CLK. When CORE_CLK is higher than 120 MHz, HSE_CLK must be half of the CORE_CLK.
DCM_CLK	MC_CGM.MUX_0_DC_4[DIV]	• 48 MHz	DCM_CLK
LBIST_CLK	MC_CGM.MUX_0_DC_5[DIV]	• 48 MHz	LBIST clock
QSPI_MEM_CLK	MC_CGM.MUX_0_DC_6[DIV]	• 160 MHz	QSPI_MEM_CLK is always equal to CORE_CLK except in 1:1 mode (see Option F - Operation in 1:1 mode with CORE_CLK and AIPS_PLAT_CLK at same speed (For all chips except S32K388/S32K389)).

1. CORE_CLK is the frequency used for the CM7 cores, AXBS interfaces, and fast peripherals.

Table 151. System clock frequency limitations (For S32K312, S32K311, and S32K310)

System clock node	System clock divider	Maximum frequency allowed	Remarks
CORE_CLK	MC_CGM.MUX_0_DC_0[DIV]	• 120 MHz	CORE_CLK[1] is always greater than or equal to AIPS_PLAT_CLK.
AIPS_PLAT_CLK	MC_CGM.MUX_0_DC_1[DIV]	• 80 MHz	AIPS_PLAT_CLK is always less than or equal to CORE_CLK.
AIPS_SLOW_CLK	MC_CGM.MUX_0_DC_2[DIV]	• 40 MHz	AIPS_SLOW_CLK is always less than or equal to AIPS_PLAT_CLK.
HSE_CLK	MC_CGM.MUX_0_DC_3[DIV]	• 120 MHz	When CORE_CLK is equal to or less than 120 MHz, HSE_CLK can be equal to CORE_CLK. When CORE_CLK is higher than 120 MHz, HSE_CLK must be half of the CORE_CLK.
DCM_CLK	MC_CGM.MUX_0_DC_4[DIV]	• 48 MHz	DCM_CLK

1. CORE_CLK is the frequency used for the CM7 cores, AXBS interfaces, and fast peripherals.

NOTE

The chip supports 1:1 clocking mode, whereby the core(s) are clocked at the same frequency as the slave ports (flash memory, PRAM controller, AIPS controller). See Option F - Operation in 1:1 mode with CORE_CLK and AIPS_PLAT_CLK at same speed (For all chips except S32K388/S32K389).

The frequencies in the table above are maximum frequencies for a specific clock. However, any clock frequency selected must adhere to the same clock divider ratios shown in System clocking configurations.

24.7.2 System clocking configurations

The chip supports the clocking modes shown in Table 152. All clock configurations are implemented by appropriate register settings.

Table 152. System clocking configurations

Clocking options	Option A	Option B	Option C	Option D	Option E	Option E2	Option F	Option G
	High-performance mode	Reduced speed mode	Boot (default) Standby configuration (for low dynamic current consumption)	FIRC divider bypassed	Low speed Run mode, clocked by divided FIRC	Very low speed Run mode, clock by divided FIRC	Operation in 1:1 mode with core and AXBS at same speed	PLL providing 48 MHz (test bench use case)
System clock source (SYS_CLK)	PLL_PHI0_CLK	PLL_PHI0_CLK	FIRC_CLK ÷ 2	FIRC_CLK	FIRC_CLK ÷ 2	FIRC_CLK ÷ 16	PLL_PHI0_CLK	PLL_PHI0_CLK
PLL VCO frequency	480 MHz	480 MHz	—	—	—	—	480 MHz	480 MHz
PLL_PHI1_CLK	K344: 240 MHz (VCO ÷ 2) K342: 160 MHz (VCO ÷ 3)	K344: 240 MHz (VCO ÷ 2) K342: 160 MHz (VCO ÷ 3)	—	—	—	—	K344: 240 MHz (VCO/2) K342: 160 MHz (VCO/3)	—
PLL_PHI0_CLK	160 MHz (VCO ÷ 3)	120 MHz (VCO ÷ 4)	—	—	—	—	160 MHz (VCO ÷ 3)	96 MHz (VCO ÷ 5)
CORE_CLK (application cores, AXBS, SRAM, AIPS0, flash memory controller port clock,	160 MHz (SYS_CLK)	120 MHz (SYS_CLK)	24 MHz (FIRC_CLK ÷ 2)	48 MHz (FIRC)	3 MHz ((FIRC ÷ 2) ÷ 8)	187.5 kHz ((FIRC ÷ 16) ÷ 16)	80 MHz (SYS_CLK ÷ 2)	48 MHz (SYS_CLK ÷ 2)

Table continues on the next page...

Table 152. System clocking configurations (continued)

Clocking options	Option A	Option B	Option C	Option D	Option E	Option E2	Option F	Option G
	High-performance mode	Reduced speed mode	Boot (default) Standby configuration (for low dynamic current consumption)	FIRC divider bypassed	Low speed Run mode, clocked by divided FIRC	Very low speed Run mode, clock by divided FIRC	Operation in 1:1 mode with core and AXBS at same speed	PLL providing 48 MHz (test bench use case)
QSPI memory clock, fast-speed peripherals clock)								
QSPI mem clock	160 MHz	120 MHz	—	—	—	—	160 MHz	—
AIPS_PLAT_CLK (medium-speed peripheral clock	80 MHz	60 MHz	24 MHz	48 MHz	3 MHz	187.5 kHz	80 MHz	48 MHz
AIPS_SLOW_CLK (slow-speed peripheral clock)	40 MHz	30 MHz	12 MHz	24 MHz	1.5 MHz	93.75 kHz	40 MHz	24 MHz
DCM/ DCF_CLK	40 MHz	30 MHz	24 MHz	48 MHz	3 MHz	187.5 kHz	40 MHz	48 MHz
HSE_CLK	80 MHz	K34x: 60/120 MHz K31x: 60/120 MHz	24 MHz	48 MHz	3 MHz	187.5 kHz	80 MHz	48 MHz
LBIST_CLK	40 MHz	30 MHz	—	—	—	—	40 MHz	—
QSPI_SFCK	K344: 120 MHz (QSPI only) 80 MHz (QSPI + ENET RMII) K342: 80 MHz	K344: 120 MHz (QSPI only) 80 MHz (QSPI + ENET RMII) K342: 80 MHz	—	—	—	—	K344: 120 MHz (QSPI only) 80 MHz (QSPI + ENET RMII)	—

NOTE

In system clocking configurations where CORE_CLK and AIPS_PLAT_CLK are configured for operation at same frequency, the RAM wait states and flash memory read/write wait cycles need to be configured. See Clock Option B in Gasket Configurations section for RAM wait states and gasket configurations for clocking options.

The PLL should only be enabled if it is used as the system clock source. When the FIRC is used directly as the system clock, the PLL must be disabled.

While enabling PLL, the PMC last mile regulator should be enabled first, by configuring PMC_CONFIG[LMEN] and PMC_CONFIG[LMBCTLEN] (in case of external BJT). The last mile regulator should be disabled after PLL is disabled.

PLL can be locked at minimum 640 MHz and then 8 MHz can be achieved with dividers in series - PLL divider(divide by 16) and CGM divider(divide by 5).

24.7.2.1 Option A++ - Very High Performance mode (CM7_CORE_CLK @ 320 MHz) (For S32K388/S32K389)

This option is only available in Run mode.

Table 153. Option A++ - Very High Performance mode (CM7_CORE_CLK @ 320 MHz) (For S32K388/S32K389)

Clocking options	Clock frequencies [1]
	S32K388/S32K389
PLL VCO frequency	640 MHz
PLLODIV2_CLK (PLLDIG.PLLDV[ODIV2])	640 MHz (0001b)
FIRC_CLK (HSE_B.CONFIG_REG_GPR[FIRC_DIV_SEL])	48 MHz (11b)
PLL_PHI1_CLK-related clocks [2]	
PLL_PHI1_CLK (PLLDIG.PLLODIV_1[DIV])	160 MHz (0011b)
PLL_AUX-related clocks [3]	
PLL_AUX_VCO_CLK	1000 MHz
PLL_AUXODIV2_CLK (PLLDIG.PLLDV[ODIV2])	500 MHz (0010b)
PLL_AUX_PHI0_CLK (PLLDIG.PLLODIV_0[DIV])	250 MHz (0001b)
PLL_AUX_PHI1_CLK (PLLDIG.PLLODIV_1[DIV])	25 MHz (10011b)
QSPI_SFCK (MC_CGM.MUX_10_DC_0[DIV])	125 MHz [4] (0000b)

Table continues on the next page...

Table 153. Option A++ - Very High Performance mode (CM7_CORE_CLK @ 320 MHz) (For S32K388/S32K389) (continued)

Clocking options	Clock frequencies [1]
	S32K388/S32K389
GMAC0_CLK_TX (MC_CGM.MUX_8_DC_0[DIV])	125 MHz [4] (0000b)
GMAC1_CLK_TX (MC_CGM.MUX_16_DC_0[DIV])	125 MHz [4] (0000b)
TRACE_CLK (MC_CGM.MUX_11_DC_0[DIV])	**For fast pads**
	125 MHz [4] (0000b)
	For standard-plus pads
	25 MHz [4] (0100b)
PLL_PHI0_CLK-related clocks [5]	
PLL_PHI0_CLK (PLLDIG.PLLODIV_0[DIV])	320 MHz (0001b)
CM7_CORE_CLK (MC_CGM.MUX_0_DC_7[DIV])	320 MHz (0000b)
CORE_CLK • AXBS • SRAM • Flash memory controller port clock • AIPS0 (high-speed peripheral clock) (MC_CGM.MUX_0_DC_0[DIV])	160 MHz (0001b)
QSPI_MEM_CLK (MC_CGM.MUX_0_DC_6[DIV])	160 MHz (0001b)
AIPS_PLAT_CLK (medium-speed peripheral clock) (MC_CGM.MUX_0_DC_1[DIV])	80 MHz (0011b)
AIPS_SLOW_CLK (slow-speed peripheral clock) (MC_CGM.MUX_0_DC_2[DIV])	40 MHz (0111b)
DCM_CLK (MC_CGM.MUX_0_DC_4[DIV])	40 MHz (0111b)

Table continues on the next page...

Table 153. Option A++ - Very High Performance mode (CM7_CORE_CLK @ 320 MHz) (For S32K388/S32K389) (continued)

Clocking options	Clock frequencies [1]
	S32K388/S32K389
HSE_CLK (MC_CGM.MUX_0_DC_3[DIV])	160 MHz (0001b)
LBIST_CLK (MC_CGM.MUX_0_DC_5[DIV])	40 MHz (0111b)

1. This table is only applicable for S32K388/S32K389.
2. MC_CGM.MUX_10_CSC[SELCTL] and MC_CGM.MUX_11_CSC[SELCTL] must equal 1001b.
3. MC_CGM.MUX_10_CSC[SELCTL] and MC_CGM.MUX_11_CSC[SELCTL] must equal 1100b.
4. The final clock frequency is derived from additional divider stages. See the chip's clock diagrams for details.
5. MC_CGM.MUX_0_CSC[SELCTL] must equal 1000b.

24.7.2.2 Option A+ - Very High Performance mode (CM7_CORE_CLK @ 240 MHz) (For S32K388/S32K389)

This option is only available in Run mode.

Table 154. Option A+ - Very High Performance mode (CM7_CORE_CLK @ 240 MHz) (For S32K388/S32K389)

Clocking options	Clock frequencies
	S32K388/S32K389[1]
PLL VCO frequency	960 MHz
PLLODIV2_CLK (PLLDIG.PLLDV[ODIV2])	480 MHz (0010b)
FIRC_CLK (HSE_B.CONFIG_REG_GPR[FIRC_DIV_SEL])	48 MHz (11b)
PLL_PHI1_CLK-related clocks [2]	
PLL_PHI1_CLK (PLLDIG.PLLODIV_1[DIV])	160 MHz (0010b)
PLL_AUX-related clocks [3]	
PLL_AUX_VCO_CLK	1000 MHz
PLLAUXODIV2_CLK (PLLDIG.PLLDV[ODIV2])	500 MHz (0010b)
PLL_AUX_PHI0_CLK PLLODIV_0[DIV]	250 MHz (0001b)
PLL_AUX_PHI1_CLK PLLODIV_1[DIV]	25 MHz (10011b)

Table continues on the next page...

Table 154. Option A+ - Very High Performance mode (CM7_CORE_CLK @ 240 MHz) (For S32K388/S32K389) (continued)

Clocking options	Clock frequencies
	S32K388/S32K389[1]
QSPI_SFCK (MC_CGM.MUX_10_DC_0[DIV])	125 MHz[4] (0000b)
GMAC0_CLK_TX (MC_CGM.MUX_8_DC_0[DIV])	125 MHz [4] (0000b)
GMAC1_CLK_TX (MC_CGM.MUX_16_DC_0[DIV])	125 MHz [4] (0000b)
TRACE_CLK (MC_CGM.MUX_11_DC_0[DIV])	**For fast pads**
	125 MHz [4] (0000b)
	For standard-plus pads
	25 MHz [4] (0100b)
PLL_PHI0_CLK-related clocks [5]	
PLL_PHI0_CLK (PLLDIG.PLLODIV_0[DIV])	240 MHz (0001b)
CM7_CORE_CLK (MC_CGM.MUX_0_DC_7[DIV])	240 MHz (0000b)
CORE_CLK • AXBS • SRAM • Flash memory controller port clock • AIPS0 (high-speed peripheral clock) (MC_CGM.MUX_0_DC_0[DIV])	120 MHz (0001b)
QSPI_MEM_CLK (MC_CGM.MUX_0_DC_6[DIV])	120 MHz (0001b)
AIPS_PLAT_CLK (medium-speed peripheral clock) (MC_CGM.MUX_0_DC_1[DIV])	60 MHz (0011b)
AIPS_SLOW_CLK (slow-speed peripheral clock) (MC_CGM.MUX_0_DC_2[DIV])	30 MHz (0111b)

Table continues on the next page...

Table 154. Option A+ - Very High Performance mode (CM7_CORE_CLK @ 240 MHz) (For S32K388/S32K389) (continued)

Clocking options	Clock frequencies
	S32K388/S32K389[1]
DCM_CLK (MC_CGM.MUX_0_DC_4[DIV])	30 MHz (0111b)
HSE_CLK (MC_CGM.MUX_0_DC_3[DIV])	120 MHz (0001b)
LBIST_CLK (MC_CGM.MUX_0_DC_5[DIV])	30 MHz (0111b)

1. This table is only applicable for S32K388/S32K389.
2. MC_CGM.MUX_10_CSC[SELCTL] and MC_CGM.MUX_11_CSC[SELCTL] must equal 1001b.
3. MC_CGM.MUX_10_CSC[SELCTL] and MC_CGM.MUX_11_CSC[SELCTL] must equal 1100b.
4. The final clock frequency is derived from additional divider stages. See the chip's clock diagrams for details.
5. MC_CGM.MUX_0_CSC[SELCTL] must equal 1000b.

24.7.2.3 Option A+ - Very High Performance mode (CORE_CLK @ 240 MHz) (For S32K328, S32K338, S32K348, S32K356, and S32K358)

This option is only available in Run mode.

Table 155. Option A+ - High Performance mode (CORE_CLK @ 240 MHz)

Clocking options	Clock frequencies
	S32K358, S32K356, S32K348, S32K338, and S32K328[1]
PLL VCO frequency	960 MHz
PLLODIV2_CLK (PLLDIG.PLLDV[ODIV2])	480 MHz (0010b)
FIRC_CLK (HSE_B.CONFIG_REG_GPR[FIRC_DIV_SEL])	48 MHz (11b)
PLL_PHI1_CLK-related clocks [2]	
PLL_PHI1_CLK (PLLDIG.PLLODIV_1[DIV])	480 MHz (0000b)
QSPI_SFCK (MC_CGM.MUX_10_DC_0[DIV])	120 MHz [3] (0000b)
TRACE_CLK (MC_CGM.MUX_11_DC_0[DIV])	**For fast pads** 120 MHz [3] (0001b)
PLL_AUX-related clocks [4]	

Table continues on the next page...

Table 155. Option A+ - High Performance mode (CORE_CLK @ 240 MHz) (continued)

Clocking options	Clock frequencies	
	S32K358, S32K356, S32K348, S32K338, and S32K328[1]	
PLL_AUX_VCO_CLK	1000 MHz	800 MHz
PLLAUXODIV2_CLK (PLLDIG.PLLDV[ODIV2])	500 MHz (0010b)	400 MHz (0010b)
PLL_AUX_PHI0_CLK-related clocks		
PLL_AUX_PHI0_CLK (PLLDIG.PLLODIV_0[DIV])	500 MHz (0000b)	400 MHz (0000b)
QSPI_SFCK (MC_CGM.MUX_10_DC_0[DIV])	NA	100 MHz [3] (0000b)
GMAC_CLK_TX (MC_CGM.MUX_8_DC_0[DIV])	125 MHz [3] (0001b)	NA
TRACE_CLK (MC_CGM.MUX_11_DC_0[DIV])	**For standard-plus pads** 25 MHz [3] (1001b)	NA
PLL_AUX_PHI1_CLK (PLLDIG.PLLODIV_1[DIV])	25 MHz (10011b)	
PLL_AUX_PHI2_CLK (PLLDIG.PLLODIV_2[DIV])	100 MHz (0100b)	
PLL_PHI0_CLK-related clocks [5]		
PLL_PHI0_CLK (PLLDIG.PLLODIV_0[DIV])	240 MHz (0001b)	
CORE_CLK • Application cores • AXBS • SRAM • Flash memory controller port clock • AIPS0 (high-speed peripheral clock) (MC_CGM.MUX_0_DC_0[DIV])	240 MHz (0000b)	
QSPI_MEM_CLK (MC_CGM.MUX_0_DC_6[DIV])	240 MHz (0000b)	

Table continues on the next page...

Table 155. Option A+ - High Performance mode (CORE_CLK @ 240 MHz) (continued)

Clocking options	Clock frequencies
	S32K358, S32K356, S32K348, S32K338, and S32K328[1]
AIPS_PLAT_CLK (medium-speed peripheral clock) (MC_CGM.MUX_0_DC_1[DIV])	120 MHz (0001b)
AIPS_SLOW_CLK (slow-speed peripheral clock) (MC_CGM.MUX_0_DC_2[DIV])	60 MHz (0011b)
DCM_CLK (MC_CGM.MUX_0_DC_4[DIV])	60 MHz (0011b)
HSE_CLK (MC_CGM.MUX_0_DC_3[DIV])	120 MHz (0001b)
LBIST_CLK (MC_CGM.MUX_0_DC_5[DIV])	60 MHz (0011b)

1. This table is only applicable for S32K328, S32K338, S32K348, S32K356, and S32K358.
2. MC_CGM.MUX_10_CSC[SELCTL] and MC_CGM.MUX_11_CSC[SELCTL] must equal 1001b.
3. The final clock frequency is derived from additional divider stages. See the chip's clock diagrams for details.
4. MC_CGM.MUX_10_CSC[SELCTL] and MC_CGM.MUX_11_CSC[SELCTL] must equal 1100b.
5. MC_CGM.MUX_0_CSC[SELCTL] must equal 1000b.

24.7.2.4 Option A - High Performance mode (CM7_CORE_CLK @ 160 MHz) (For S32K388/S32K389

This option is only available in Run mode.

Table 156. Option A - High Performance mode (CM7_CORE_CLK @ 160 MHz) (For S32K388/S32K389

Clocking options	Clock frequencies
	S32K388/S32K389[1]
PLL VCO frequency	640 MHz
PLLODIV2_CLK (PLLDIG.PLLDV[ODIV2])	640 MHz (0001b)
FIRC_CLK (HSE_B.CONFIG_REG_GPR[FIRC_DIV_SEL])	48 MHz (11b)
PLL_PHI1_CLK-related clocks [2]	
PLL_PHI1_CLK (PLLDIG.PLLODIV_1[DIV])	160 MHz (0011b)
PLL_AUX-related clocks [3]	
PLL_AUX_VCO_CLK	1000 MHz

Table continues on the next page...

Table 156. Option A - High Performance mode (CM7_CORE_CLK @ 160 MHz) (For S32K388/S32K389 (continued)

Clocking options	Clock frequencies
	S32K388/S32K389[1]
PLLAUXODIV2_CLK (PLLDIG.PLLDV[ODIV2])	500 MHz (0010b)
PLL_AUX_PHI0_CLK PLLODIV_0[DIV]	250 MHz (0001b)
PLL_AUX_PHI1_CLK PLLODIV_1[DIV]	25 MHz (10011b)
QSPI_SFCK (MC_CGM.MUX_10_DC_0[DIV])	125 MHz [4] (0000b)
GMAC0_CLK_TX (MC_CGM.MUX_8_DC_0[DIV])	125 MHz [4] (0000b)
GMAC1_CLK_TX (MC_CGM.MUX_16_DC_0[DIV])	125 MHz [4] (0000b)
TRACE_CLK (MC_CGM.MUX_11_DC_0[DIV])	**For fast pads**
	125 MHz [4] (0000b)
	For standard-plus pads
	25 MHz [4] (0100b)
PLL_PHI0_CLK-related clocks [5]	
PLL_PHI0_CLK (PLLDIG.PLLODIV_0[DIV])	320 MHz (0001b)
CM7_CORE_CLK (MC_CGM.MUX_0_DC_7[DIV])	160 MHz (0001b)
CORE_CLK • AXBS • SRAM • Flash memory controller port clock • AIPS0 (high-speed peripheral clock) (MC_CGM.MUX_0_DC_0[DIV])	160 MHz (0001b)
QSPI_MEM_CLK (MC_CGM.MUX_0_DC_6[DIV])	160 MHz (0001b)

Table continues on the next page...

Table 156. Option A - High Performance mode (CM7_CORE_CLK @ 160 MHz) (For S32K388/S32K389 (continued)

Clocking options	Clock frequencies
	S32K388/S32K389[1]
AIPS_PLAT_CLK (medium-speed peripheral clock) (MC_CGM.MUX_0_DC_1[DIV])	80 MHz (0011b)
AIPS_SLOW_CLK (slow-speed peripheral clock) (MC_CGM.MUX_0_DC_2[DIV])	40 MHz (0111b)
DCM_CLK (MC_CGM.MUX_0_DC_4[DIV])	40 MHz (0111b)
HSE_CLK (MC_CGM.MUX_0_DC_3[DIV])	160 MHz (0001b)
LBIST_CLK (MC_CGM.MUX_0_DC_5[DIV])	40 MHz (0111b)

1. This table is only applicable for S32K388/S32K389.
2. MC_CGM.MUX_10_CSC[SELCTL] and MC_CGM.MUX_11_CSC[SELCTL] must equal 1001b.
3. MC_CGM.MUX_10_CSC[SELCTL] and MC_CGM.MUX_11_CSC[SELCTL] must equal 1100b.
4. The final clock frequency is derived from additional divider stages. See the chip's clock diagrams for details.
5. MC_CGM.MUX_0_CSC[SELCTL] must equal 1000b.

24.7.2.5 Option A - High Performance mode (CORE_CLK @ 160 MHz)

This option is only available in Run mode.

Table 157. Option A - High Performance mode (CORE_CLK @ 160 MHz)

Clocking options	Clock frequencies		
	S32K344, S32K324, S32K314, S32K342, S32K341, and S32K322 [1]		S32K328, S32K338, S32K348, S32K356, and S32K358
PLL VCO frequency	960 MHz		960 MHz
PLLODIV2_CLK (PLLDIG.PLLDV[ODIV2])	480 MHz (0010b)		480 MHz (0010b)
FIRC_CLK (HSE_B.CONFIG_REG_GPR[FIRC_DIV_SEL])	48 MHz (11b)		48 MHz (11b)
PLL_PHI1_CLK-related clocks [2]			
PLL_PHI1_CLK (PLLDIG.PLLODIV_1[DIV])	240 MHz (0001b)	160 MHz (0010b)	480 MHz (0000b)
QSPI_SFCK (MC_CGM.MUX_10_DC_0[DIV])	120 MHz (0001b)	80 MHz (0001b)	120 MHz [3] (0000b)

Table continues on the next page...

Table 157. Option A - High Performance mode (CORE_CLK @ 160 MHz) (continued)

Clocking options	Clock frequencies			
	S32K344, S32K324, S32K314, S32K342, S32K341, and S32K322 [1]		S32K328, S32K338, S32K348, S32K356, and S32K358	
TRACE_CLK (MC_CGM.MUX_11_DC_0[DIV])	For fast pads		For fast pads	
	120 MHz (0001b)	80 MHz (0001b)	120 MHz [3] (0001b)	
	For standard-plus pads			
	24 MHz (1001b)	16 MHz (1001b)		
PLL_AUX-related clocks				
PLL_AUX_VCO_CLK	NA		1000 MHz	800 MHz
PLLAUXODIV2_CLK (PLLDIG.PLLDV[ODIV2])	NA		500 MHz (0010b)	400 MHz (0010b)
PLL_AUX_PHI0_CLK-related clocks				
PLL_AUX_PHI0_CLK (PLLDIG.PLLODIV_0[DIV])	NA		500 MHz (0000b)	400 MHz (0000b)
QSPI_SFCK (MC_CGM.MUX_10_DC_0[DIV])	NA		NA	100 MHz [3] (0000b)
GMAC_CLK_TX (MC_CGM.MUX_8_DC_0[DIV])	NA		125 MHz [3] (0001b)	NA
TRACE_CLK (MC_CGM.MUX_11_DC_0[DIV])	NA		25 MHz [3] (1001b)	NA
PLL_AUX_PHI1_CLK (PLLDIG.PLLODIV_1[DIV])	NA		25 MHz (10011b)	
PLL_AUX_PHI2_CLK (PLLDIG.PLLODIV_2[DIV])	NA		100 MHz (0100b)	
PLL_PHI0_CLK-related clocks [4]				
PLL_PHI0_CLK (PLLDIG.PLLODIV_0[DIV])	160 MHz (0010b)		160 MHz (0010b)	
CORE_CLK • Application cores • AXBS	160 MHz (0000b)		160 MHz (0000b)	

Table continues on the next page...

Table 157. Option A - High Performance mode (CORE_CLK @ 160 MHz) (continued)

Clocking options	Clock frequencies	
	S32K344, S32K324, S32K314, S32K342, S32K341, and S32K322 [1]	S32K328, S32K338, S32K348, S32K356, and S32K358
• SRAM • Flash memory controller port clock • AIPS0 (high-speed peripheral clock) (MC_CGM.MUX_0_DC_0[DIV])		
QSPI_MEM_CLK (MC_CGM.MUX_0_DC_6[DIV])	160 MHz (0000b)	160 MHz (0000b)
AIPS_PLAT_CLK (medium-speed peripheral clock) (MC_CGM.MUX_0_DC_1[DIV])	80 MHz (0001b)	80 MHz (0001b)
AIPS_SLOW_CLK (slow-speed peripheral clock) (MC_CGM.MUX_0_DC_2[DIV])	40 MHz (0011b)	40 MHz (0011b)
DCM_CLK (MC_CGM.MUX_0_DC_4[DIV])	40 MHz (0011b)	40 MHz (0011b)
HSE_CLK (MC_CGM.MUX_0_DC_3[DIV])	80 MHz (0001b)	80 MHz (0001b)
LBIST_CLK (MC_CGM.MUX_0_DC_5[DIV])	40 MHz (0011b)	40 MHz (0011b)

1. This table does not apply to S32K310, S32K311, and S32K312.
2. MC_CGM.MUX_10_CSC[SELCTL] and MC_CGM.MUX_11_CSC[SELCTL] must equal 1001b.
3. The final clock frequency is derived from additional divider stages. See the chip's clock diagrams for details.
4. MC_CGM.MUX_0_CSC[SELCTL] must equal 1000b.

24.7.2.6 Option B - Reduced Speed mode (CORE_CLK @ 120 MHz)

This option is only available in Run mode.

Table 158. Option B - Reduced Speed mode (CORE_CLK @ 120 MHz)

Clocking options	Clock frequencies		
	S32K344, S32K324, S32K314, S32K342, S32K341, and S32K322	S32K310, S32K311, and S32K312	S32K328, S32K338, S32K348, S32K356, and S32K358
PLL VCO frequency	960 MHz	960 MHz	960 MHz

Table continues on the next page...

Table 158. Option B - Reduced Speed mode (CORE_CLK @ 120 MHz) (continued)

Clocking options	Clock frequencies			
	S32K344, S32K324, S32K314, S32K342, S32K341, and S32K322		S32K310, S32K311, and S32K312	S32K328, S32K338, S32K348, S32K356, and S32K358
PLLODIV2_CLK (PLLDIG.PLLDV[ODIV2])	480 MHz (0010b)		240 MHz (0100b)	480 MHz (0010b)
FIRC_CLK (HSE_B.CONFIG_REG_GPR[FIRC_DIV_SEL])	48 MHz (11b)			
PLL_PHI1_CLK-related clocks [1], [2]				
PLL_PHI1_CLK (PLLDIG.PLLODIV_1[DIV])	240 MHz (0001b)	160 MHz (0010b)	48 MHz (0100b)	480 MHz (0000b)
QSPI_SFCK (MC_CGM.MUX_10_DC_0[DIV])	120 MHz (001b)	80 MHz (001b)	—	120 MHz[3] (0000b)
TRACE_CLK (MC_CGM.MUX_11_DC_0[DIV])	**For fast pads**		—	**For fast pads**
	120 MHz (001b)	80 MHz (001b)	—	120 MHz[3] (0001b)
	For standard-plus pads		—	
	24 MHz (1001b)	16 MHz (1001b)	—	
PLL_AUX-related clocks[4]				
PLL_AUX_VCO_CLK	NA		1000 MHz	800 MHz
PLLAUXODIV2_CLK (PLLDIG.PLLDV[ODIV2])	NA		500 MHz (0010b)	400 MHz (0010b)
PLL_AUX_PHI0_CLK-related clocks				
PLL_AUX_PHI0_CLK (PLLDIG.PLLODIV_0[DIV])	NA		500 MHz (0000b)	400 MHz (0000b)
QSPI_SFCK (MC_CGM.MUX_10_DC_0[DIV])	NA		NA	100 MHz[3] (0000b)
GMAC_CLK_TX (MC_CGM.MUX_8_DC_0[DIV])	NA		125 MHz [3] (0001b)	NA

Table continues on the next page...

Table 158. Option B - Reduced Speed mode (CORE_CLK @ 120 MHz) (continued)

Clocking options	Clock frequencies			
	S32K344, S32K324, S32K314, S32K342, S32K341, and S32K322	S32K310, S32K311, and S32K312	S32K328, S32K338, S32K348, S32K356, and S32K358	
TRACE_CLK (MC_CGM.MUX_11_DC_0[DIV])	NA		For standard-plus pads 25 MHz[3] (1001b)	NA
PLL_AUX_PHI1_CLK (PLLDIG.PLLODIV_1[DIV])	NA		25 MHz (10011b)	
PLL_AUX_PHI2_CLK (PLLDIG.PLLODIV_2[DIV])	NA		100 MHz (0100b)	
PLL_PHI0_CLK-related clocks [5]				
PLL_PHI0_CLK (PLLDIG.PLLODIV_0[DIV])	120 MHz (011b)	120 MHz (001b)	120 MHz (011b)	
CORE_CLK • Application cores • AXBS • SRAM • Flash memory controller port clock • AIPS0 (high-speed peripheral clock) (MC_CGM.MUX_0_DC_0[DIV])	120 MHz (000b)	120 MHz (000b)	120 MHz (000b)	
QSPI_MEM_CLK (MC_CGM.MUX_0_DC_6[DIV])	120 MHz (000b)	—	120 MHz (000b)	
AIPS_PLAT_CLK (medium-speed peripheral clock) (MC_CGM.MUX_0_DC_1[DIV])	60 MHz (001b)	60 MHz (001b)	60 MHz (001b)	
AIPS_SLOW_CLK (slow-speed peripheral clock) (MC_CGM.MUX_0_DC_2[DIV])	30 MHz (011b)	30 MHz (011b)	30 MHz (011b)	
DCM_CLK (MC_CGM.MUX_0_DC_4[DIV])	30 MHz (011b)	30 MHz (011b)	30 MHz (011b)	

Table continues on the next page...

Table 158. Option B - Reduced Speed mode (CORE_CLK @ 120 MHz) (continued)

Clocking options	Clock frequencies					
	S32K344, S32K324, S32K314, S32K342, S32K341, and S32K322		S32K310, S32K311, and S32K312		S32K328, S32K338, S32K348, S32K356, and S32K358	
HSE_CLK (MC_CGM.MUX_0_DC_3[DIV])	120 MHz[6] (000b)	60 MHz (001b)	120 MHz[6] (000b)	60 MHz (001b)	120 MHz[6] (000b)	60 MHz (001b)
LBIST_CLK (MC_CGM.MUX_0_DC_5[DIV])	30 MHz (011b)		30 MHz (011b)		30 MHz (011b)	

1. MC_CGM.MUX_10_CSC[SELCTL] and MC_CGM.MUX_11_CSC[SELCTL] must equal 1001b.
2. Do not use the combination of different frequencies mentioned below across the 2x2 matrix. The values shown in each cell are valid and must not be clubbed with the values mentioned in any other cells.
3. The final clock frequency is derived from additional divider stages. See the chip's clock diagrams for details.
4. MC_CGM.MUX_10_CSC[SELCTL] and MC_CGM.MUX_11_CSC[SELCTL] must equal 1100b.
5. MC_CGM.MUX_0_CSC[SELCTL] must equal 1000b.
6. When 120 MHz is selected for HSE_CLK while AIPS_SLOW_CLK is 30 MHz, an DCF record for UTEST_MISC DCF client should be added with setting HSE_CLK_MODE_AND_GSKT_CTRL bit field as 2'b10 or 2'b11.

24.7.2.7 Option C - Boot Standby mode (CM7_CORE_CLK @ 24 MHz) (For S32K388/S32K389)

Table 159. Option C - Boot Standby mode (CM7_CORE_CLK @ 24 MHz) (For S32K388/S32K389)

Clocking options	Clock frequencies
	S32K388/S32K389[1]
PLL VCO frequency	-
PLLODIV2_CLK (PLLDIG.PLLDV[ODIV2])	-
FIRC_CLK (HSE_B.CONFIG_REG_GPR[FIRC_DIV_SEL])	24 MHz[2]
PLL_PHI1_CLK-related clocks [3]	
PLL_PHI1_CLK (PLLDIG.PLLODIV_1[DIV])	-
PLL_AUX-related clocks [4]	
PLL_AUX_VCO_CLK	-
PLLAUXODIV2_CLK (PLLDIG.PLLDV[ODIV2])	-
PLL_AUX_PHI0_CLK PLLODIV_0[DIV]	-

Table continues on the next page...

Table 159. Option C - Boot Standby mode (CM7_CORE_CLK @ 24 MHz) (For S32K388/S32K389) (continued)

Clocking options	Clock frequencies
	S32K388/S32K389[1]
PLL_AUX_PHI1_CLK PLLODIV_1[DIV]	-
QSPI_SFCK (MC_CGM.MUX_10_DC_0[DIV])	-
TRACE_CLK (MC_CGM.MUX_11_DC_0[DIV])	-
PLL_PHI0_CLK-related clocks [5]	
PLL_PHI0_CLK (PLLDIG.PLLODIV_0[DIV])	-
CM7_CORE_CLK (MC_CGM.MUX_0_DC_7[DIV])	24 MHz (0000b)
CORE_CLK • AXBS • SRAM • Flash memory controller port clock • AIPS0 (high-speed peripheral clock) (MC_CGM.MUX_0_DC_0[DIV])	24 MHz (0000b)
QSPI_MEM_CLK (MC_CGM.MUX_0_DC_6[DIV])	-
AIPS_PLAT_CLK (medium-speed peripheral clock) (MC_CGM.MUX_0_DC_1[DIV])	24 MHz (0000b)
AIPS_SLOW_CLK (slow-speed peripheral clock) (MC_CGM.MUX_0_DC_2[DIV])	12 MHz (0001b)
DCM_CLK (MC_CGM.MUX_0_DC_4[DIV])	24 MHz (0000b)
HSE_CLK (MC_CGM.MUX_0_DC_3[DIV])	24 MHz (0000b)
LBIST_CLK (MC_CGM.MUX_0_DC_5[DIV])	-

1. This table is only applicable for S32K388/S32K389.
2. The FIRC_DIV_SEL is configured by the sBAF code. It is set to 11b after reset or normal standby exit and FIRC_CLK is 48 MHz. In case of fast standby exit, FIRC_DIV_SEL is 00b and FIRC_CLK is 24 MHz.

3. MC_CGM.MUX_10_CSC[SELCTL] and MC_CGM.MUX_11_CSC[SELCTL] values are don't care since QSPI_SFCK and TRACE_CLK are not used in this use case.
4. MC_CGM.MUX_10_CSC[SELCTL] and MC_CGM.MUX_11_CSC[SELCTL] must equal 1100b.
5. MC_CGM.MUX_0_CSC[SELCTL] must equal 0000b.

24.7.2.8 Option C - Boot Standby mode (CORE_CLK @ 24 MHz)

Table 160. Option C - Boot Standby mode (CORE_CLK @ 24 MHz)

Clocking options	Clock frequencies
	S32K3xx[1]
PLL VCO frequency	—
PLLODIV2_CLK (PLLDIG.PLLDV[ODIV2])	—
FIRC_CLK (HSE_B.CONFIG_REG_GPR[FIRC_DIV_SEL])	24 MHz[2] (00b)
PLL_PHI1_CLK-related clocks [3]	
PLL_PHI1_CLK (PLLDIG.PLLODIV_1[DIV])	—
QSPI_SFCK (MC_CGM.MUX_10_DC_0[DIV])	—
TRACE_CLK (MC_CGM.MUX_11_DC_0[DIV])	—
PLL_PHI0_CLK-related clocks [4]	
PLL_PHI0_CLK (PLLDIG.PLLODIV_0[DIV])	—
LBIST_CLK (MC_CGM.MUX_0_DC_5[DIV])	—
QSPI_MEM_CLK (MC_CGM.MUX_0_DC_6[DIV])	—
CORE_CLK • Application cores • AXBS • SRAM • Flash memory controller port clock • AIPS0 (high-speed peripheral clock) (MC_CGM.MUX_0_DC_0[DIV])	24 MHz (0000b)

Table continues on the next page...

Table 160. Option C - Boot Standby mode (CORE_CLK @ 24 MHz) (continued)

Clocking options	Clock frequencies
	S32K3xx[1]
AIPS_PLAT_CLK (medium-speed peripheral clock) (MC_CGM.MUX_0_DC_1[DIV])	24 MHz (0000b)
AIPS_SLOW_CLK (slow-speed peripheral clock) (MC_CGM.MUX_0_DC_2[DIV])	12 MHz (0001b)
DCM_CLK (MC_CGM.MUX_0_DC_4[DIV])	24 MHz (0000b)
HSE_CLK (MC_CGM.MUX_0_DC_3[DIV])	24 MHz (0000b)

1. This table is applicable for all S32K3xx variants except S32K388/S32K389.
2. The FIRC_DIV_SEL is configured by the sBAF code. It is set to 11b after reset or normal standby exit and FIRC_CLK is 48 MHz. In case of fast standby exit, FIRC_DIV_SEL is 00b and FIRC_CLK is 24 MHz.
3. MC_CGM.MUX_10_CSC[SELCTL] and MC_CGM.MUX_11_CSC[SELCTL] values are don't care since QSPI_SFCK and TRACE_CLK are not used in this use case.
4. MC_CGM.MUX_0_CSC[SELCTL] must equal 0000b.

24.7.2.9 Option D - Low-Speed Run mode (CM7_CORE_CLK @ 48 MHz) (For S32K388/S32K389)

Table 161. Option D - Low-Speed Run mode (CM7_CORE_CLK @ 48 MHz) (For S32K388/S32K389)

Clocking options	Clock frequencies
	S32K388/S32K389[1]
PLL VCO frequency	-
PLLODIV2_CLK (PLLDIG.PLLDV[ODIV2])	-
FIRC_CLK (HSE_B.CONFIG_REG_GPR[FIRC_DIV_SEL])	48 MHz (11b)
PLL_PHI1_CLK-related clocks [2]	
PLL_PHI1_CLK (PLLDIG.PLLODIV_1[DIV])	-
PLL_AUX-related clocks [3]	
PLL_AUX_VCO_CLK	-
PLLAUXODIV2_CLK (PLLDIG.PLLDV[ODIV2])	-
PLL_AUX_PHI0_CLK PLLODIV_0[DIV]	-

Table continues on the next page...

Table 161. Option D - Low-Speed Run mode (CM7_CORE_CLK @ 48 MHz) (For S32K388/S32K389) (continued)

Clocking options	Clock frequencies
	S32K388/S32K389[1]
PLL_AUX_PHI1_CLK PLLODIV_1[DIV]	-
QSPI_SFCK (MC_CGM.MUX_10_DC_0[DIV])	-
TRACE_CLK (MC_CGM.MUX_11_DC_0[DIV])	-
PLL_PHI0_CLK-related clocks [4]	
PLL_PHI0_CLK (PLLDIG.PLLODIV_0[DIV])	-
CM7_CORE_CLK (MC_CGM.MUX_0_DC_7[DIV])	48 MHz (0000b)
CORE_CLK • AXBS • SRAM • Flash memory controller port clock • AIPS0 (high-speed peripheral clock) (MC_CGM.MUX_0_DC_0[DIV])	48 MHz (0000b)
QSPI_MEM_CLK (MC_CGM.MUX_0_DC_6[DIV])	-
AIPS_PLAT_CLK (medium-speed peripheral clock) (MC_CGM.MUX_0_DC_1[DIV])	48 MHz (0000b)
AIPS_SLOW_CLK (slow-speed peripheral clock) (MC_CGM.MUX_0_DC_2[DIV])	24 MHz (0001b)
DCM_CLK (MC_CGM.MUX_0_DC_4[DIV])	48 MHz (0000b)
HSE_CLK (MC_CGM.MUX_0_DC_3[DIV])	48 MHz (0000b)
LBIST_CLK (MC_CGM.MUX_0_DC_5[DIV])	-

1. This table is only applicable for S32K388/S32K389.
2. MC_CGM.MUX_10_CSC[SELCTL] and MC_CGM.MUX_11_CSC[SELCTL] values are don't care since QSPI_SFCK and TRACE_CLK are not used in this use case.
3. MC_CGM.MUX_10_CSC[SELCTL] and MC_CGM.MUX_11_CSC[SELCTL] must equal 1100b.

4. MC_CGM.MUX_0_CSC[SELCTL] must equal 0000b.

24.7.2.10 Option D - Low-Speed Run mode (CORE_CLK @ 48 MHz)

Table 162. Option D - Low-Speed Run mode (CORE_CLK @ 48 MHz)

Clocking options	Clock frequencies
	S32K3xx[1]
PLL VCO frequency	—
PLLODIV2_CLK (PLLDIG.PLLDV[ODIV2])	—
FIRC_CLK (HSE_B.CONFIG_REG_GPR[FIRC_DIV_SEL])	48 MHz (11b)
PLL_PHI1_CLK-related clocks [2]	
PLL_PHI1_CLK (PLLDIG.PLLODIV_1[DIV])	—
QSPI_SFCK (MC_CGM.MUX_10_DC_0[DIV])	—
TRACE_CLK (MC_CGM.MUX_11_DC_0[DIV])	—
PLL_PHI0_CLK-related clocks [3]	
PLL_PHI0_CLK (PLLDIG.PLLODIV_0[DIV])	—
LBIST_CLK (MC_CGM.MUX_0_DC_5[DIV])	—
QSPI_MEM_CLK (MC_CGM.MUX_0_DC_6[DIV])	—
CORE_CLK • Application cores • AXBS • SRAM • Flash memory controller port clock • AIPS0 (high-speed peripheral clock) (MC_CGM.MUX_0_DC_0[DIV])	48 MHz (0000b)
AIPS_PLAT_CLK (medium-speed peripheral clock) (MC_CGM.MUX_0_DC_1[DIV])	48 MHz (0000b)

Table continues on the next page...

Table 162. Option D - Low-Speed Run mode (CORE_CLK @ 48 MHz) (continued)

Clocking options	Clock frequencies
	S32K3xx[1]
AIPS_SLOW_CLK (slow-speed peripheral clock) (MC_CGM.MUX_0_DC_2[DIV])	24 MHz (0001b)
DCM_CLK (MC_CGM.MUX_0_DC_4[DIV])	48 MHz (0000b)
HSE_CLK (MC_CGM.MUX_0_DC_3[DIV])	48 MHz (0000b)

1. This table is applicable for all S32K3xx variants except S32K388/S32K389.
2. MC_CGM.MUX_10_CSC[SELCTL] and MC_CGM.MUX_11_CSC[SELCTL] values are don't care since QSPI_SFCK and TRACE_CLK are not used in this use case.
3. MC_CGM.MUX_0_CSC[SELCTL] must equal 0000b.

24.7.2.11 Option E - Low-Speed Run mode (CORE_CLK @ 3 MHz) (For S32K388/S32K389)

Table 163. Option E - Low-Speed Run mode (CORE_CLK @ 3 MHz) (For S32K388/S32K389)

Clocking options	Clock frequencies
	S32K388/S32K389[1]
PLL VCO frequency	—
PLLODIV2_CLK (PLLDIG.PLLDV[ODIV2])	—
FIRC_CLK (HSE_B.CONFIG_REG_GPR[FIRC_DIV_SEL])	3 MHz[2] (10b)
PLL_PHI1_CLK-related clocks [3]	
PLL_PHI1_CLK (PLLDIG.PLLODIV_1[DIV])	—
QSPI_SFCK (MC_CGM.MUX_10_DC_0[DIV])	—
TRACE_CLK (MC_CGM.MUX_11_DC_0[DIV])	—
PLL_PHI0_CLK-related clocks [4]	
PLL_PHI0_CLK (PLLDIG.PLLODIV_0[DIV])	—
LBIST_CLK (MC_CGM.MUX_0_DC_5[DIV])	—

Table continues on the next page...

Table 163. Option E - Low-Speed Run mode (CORE_CLK @ 3 MHz) (For S32K388/S32K389) (continued)

Clocking options	Clock frequencies
	S32K388/S32K389[1]
QSPI_MEM_CLK (MC_CGM.MUX_0_DC_6[DIV])	—
CM7_CORE_CLK (MC_CGM.MUX_0_DC_7[DIV])	3 MHz (0000b)
CORE_CLK • AXBS • SRAM • Flash memory controller port clock • AIPS0 (high-speed peripheral clock) (MC_CGM.MUX_0_DC_0[DIV])	3 MHz (0000b)
AIPS_PLAT_CLK (medium-speed peripheral clock) (MC_CGM.MUX_0_DC_1[DIV])	3 MHz (0000b)
AIPS_SLOW_CLK (slow-speed peripheral clock) (MC_CGM.MUX_0_DC_2[DIV])	1.5 MHz (0001b)
DCM_CLK (MC_CGM.MUX_0_DC_4[DIV])	3 MHz (0000b)
HSE_CLK (MC_CGM.MUX_0_DC_3[DIV])	3 MHz (0000b)

1. This table is only applicable for S32K388/S32K389.
2. The FIRC_DIV_SEL is configured by the sBAF code. It is set to 11b after reset or normal standby exit and FIRC_CLK is 48 MHz. In case of fast standby exit, FIRC_DIV_SEL is 10b and FIRC_CLK is 3 MHz.
3. MC_CGM.MUX_10_CSC[SELCTL] and MC_CGM.MUX_11_CSC[SELCTL] values are don't care since QSPI_SFCK and TRACE_CLK are not used in this use case.
4. MC_CGM.MUX_0_CSC[SELCTL] must equal 0000b.

NOTE

For FIRC_CLK frequency modes less than 24 MHz, safety modules like the CMU_F*x_n* must be disabled for safety applications, because safety applications are to run on the PLL clocks. The CMU_F*x_n* will cause erroneous FHH events if not disabled.

24.7.2.12 Option E - Low-Speed Run mode (CORE_CLK @ 3 MHz)

Table 164. Option E - Low-Speed Run mode (CORE_CLK @ 3 MHz)

Clocking options	Clock frequencies
	S32K3xx[1]
PLL VCO frequency	—

Table continues on the next page...

Table 164. Option E - Low-Speed Run mode (CORE_CLK @ 3 MHz) (continued)

Clocking options	Clock frequencies
	S32K3xx[1]
PLLODIV2_CLK (PLLDIG.PLLDV[ODIV2])	—
FIRC_CLK (HSE_B.CONFIG_REG_GPR[FIRC_DIV_SEL])	3 MHz[2] (10b)
PLL_PHI1_CLK-related clocks [3]	
PLL_PHI1_CLK (PLLDIG.PLLODIV_1[DIV])	—
QSPI_SFCK (MC_CGM.MUX_10_DC_0[DIV])	—
TRACE_CLK (MC_CGM.MUX_11_DC_0[DIV])	—
PLL_PHI0_CLK-related clocks [4]	
PLL_PHI0_CLK (PLLDIG.PLLODIV_0[DIV])	—
LBIST_CLK (MC_CGM.MUX_0_DC_5[DIV])	—
QSPI_MEM_CLK (MC_CGM.MUX_0_DC_6[DIV])	—
CORE_CLK • Application cores • AXBS • SRAM • Flash memory controller port clock • AIPS0 (high-speed peripheral clock) (MC_CGM.MUX_0_DC_0[DIV])	3 MHz (0000b)
AIPS_PLAT_CLK (medium-speed peripheral clock) (MC_CGM.MUX_0_DC_1[DIV])	3 MHz (0000b)
AIPS_SLOW_CLK (slow-speed peripheral clock) (MC_CGM.MUX_0_DC_2[DIV])	1.5 MHz (0001b)
DCM_CLK (MC_CGM.MUX_0_DC_4[DIV])	3 MHz (0000b)

Table continues on the next page...

Table 164. Option E - Low-Speed Run mode (CORE_CLK @ 3 MHz) (continued)

Clocking options	Clock frequencies
	S32K3xx[1]
HSE_CLK (MC_CGM.MUX_0_DC_3[DIV])	3 MHz (0000b)

1. This table is applicable for all S32K3xx variants except S32K388/S32K389.
2. The FIRC_DIV_SEL is configured by the sBAF code. It is set to 11b after reset or normal standby exit and FIRC_CLK is 48 MHz. In case of fast standby exit, FIRC_DIV_SEL is 10b and FIRC_CLK is 3 MHz.
3. MC_CGM.MUX_10_CSC[SELCTL] and MC_CGM.MUX_11_CSC[SELCTL] values are don't care since QSPI_SFCK and TRACE_CLK are not used in this use case.
4. MC_CGM.MUX_0_CSC[SELCTL] must equal 0000b.

NOTE

For FIRC_CLK frequency modes less than 24 MHz, safety modules like the CMU_F*x_n* must be disabled for safety applications, because safety applications are to run on the PLL clocks. The CMU_F*x_n* will cause erroneous FHH events if not disabled.

24.7.2.13 Option E2 - Very-Low-Speed Run mode (CORE_CLK @ 750 KHz) (For S32K388/S32K389)

Table 165. Option E2 - Very-Low-Speed Run mode (CORE_CLK @ 750 KHz) (For S32K388/S32K389)

Clocking options	Clock frequencies
	S32K3xx[1]
PLL VCO frequency	—
PLLODIV2_CLK (PLLDIG.PLLDV[ODIV2])	—
FIRC_CLK (HSE_B.CONFIG_REG_GPR[FIRC_DIV_SEL])	3 MHz[2] (10b)
PLL_PHI1_CLK-related clocks [3]	
PLL_PHI1_CLK (PLLDIG.PLLODIV_1[DIV])	—
QSPI_SFCK (MC_CGM.MUX_10_DC_0[DIV])	—
TRACE_CLK (MC_CGM.MUX_11_DC_0[DIV])	—
PLL_PHI0_CLK-related clocks [4]	
PLL_PHI0_CLK (PLLDIG.PLLODIV_0[DIV])	—
LBIST_CLK (MC_CGM.MUX_0_DC_5[DIV])	—

Table continues on the next page...

Table 165. Option E2 - Very-Low-Speed Run mode (CORE_CLK @ 750 KHz) (For S32K388/S32K389) (continued)

Clocking options	Clock frequencies
	S32K3xx[1]
QSPI_MEM_CLK (MC_CGM.MUX_0_DC_6[DIV])	—
CM7_CORE_CLK (MC_CGM.MUX_0_DC_7[DIV])	750 KHz (0011b)
CORE_CLK • AXBS • SRAM • Flash memory controller port clock • AIPS0 (high-speed peripheral clock) (MC_CGM.MUX_0_DC_0[DIV])	750 KHz (0011b)
AIPS_PLAT_CLK (medium-speed peripheral clock) (MC_CGM.MUX_0_DC_1[DIV])	750 KHz (0011b)
AIPS_SLOW_CLK (slow-speed peripheral clock) (MC_CGM.MUX_0_DC_2[DIV])	375 KHz (0111b)
DCM_CLK (MC_CGM.MUX_0_DC_4[DIV])	750 KHz (0011b)
HSE_CLK (MC_CGM.MUX_0_DC_3[DIV])	750 KHz (0011b)

1. This table is only applicable for S32K388/S32K389.
2. The FIRC_DIV_SEL is configured by the sBAF code. It is set to 11b after reset or normal standby exit and FIRC_CLK is 48 MHz. In case of fast standby exit, FIRC_DIV_SEL is 10b and FIRC_CLK is 3 MHz.
3. MC_CGM.MUX_10_CSC[SELCTL] and MC_CGM.MUX_11_CSC[SELCTL] values are don't care since QSPI_SFCK and TRACE_CLK are not used in this use case.
4. MC_CGM.MUX_0_CSC[SELCTL] must equal 0000b.

NOTE

For FIRC_CLK frequency modes less than 24 MHz, safety modules like the CMU_F*x_n* must be disabled for safety applications, because safety applications are to run on the PLL clocks. The CMU_F*x_n* will cause erroneous FHH events if not disabled.

24.7.2.14 Option E2 - Very-Low-Speed Run mode (CORE_CLK @ 750 KHz)

Table 166. Option E2 - Very-Low-Speed Run mode (CORE_CLK @ 750 KHz)

Clocking options	Clock frequencies
	S32K3xx[1]
PLL VCO frequency	—

Table continues on the next page...

Table 166. Option E2 - Very-Low-Speed Run mode (CORE_CLK @ 750 KHz) (continued)

Clocking options	Clock frequencies
	S32K3xx[1]
PLLODIV2_CLK (PLLDIG.PLLDV[ODIV2])	—
FIRC_CLK (HSE_B.CONFIG_REG_GPR[FIRC_DIV_SEL])	3 MHz[2] (10b)
PLL_PHI1_CLK-related clocks [3]	
PLL_PHI1_CLK (PLLDIG.PLLODIV_1[DIV])	—
QSPI_SFCK (MC_CGM.MUX_10_DC_0[DIV])	—
TRACE_CLK (MC_CGM.MUX_11_DC_0[DIV])	—
PLL_PHI0_CLK-related clocks [4]	
PLL_PHI0_CLK (PLLDIG.PLLODIV_0[DIV])	—
LBIST_CLK (MC_CGM.MUX_0_DC_5[DIV])	—
QSPI_MEM_CLK (MC_CGM.MUX_0_DC_6[DIV])	—
CORE_CLK • Application cores • AXBS • SRAM • Flash memory controller port clock • AIPS0 (high-speed peripheral clock) (MC_CGM.MUX_0_DC_0[DIV])	750 KHz (0011b)
AIPS_PLAT_CLK (medium-speed peripheral clock) (MC_CGM.MUX_0_DC_1[DIV])	750 KHz (0011b)
AIPS_SLOW_CLK (slow-speed peripheral clock) (MC_CGM.MUX_0_DC_2[DIV])	375 KHz (0111b)
DCM_CLK (MC_CGM.MUX_0_DC_4[DIV])	750 KHz (0011b)

Table continues on the next page...

Table 166. Option E2 - Very-Low-Speed Run mode (CORE_CLK @ 750 KHz) (continued)

Clocking options	Clock frequencies
	S32K3xx[1]
HSE_CLK (MC_CGM.MUX_0_DC_3[DIV])	750 KHz (0011b)

1. This table is applicable for all S32K3xx variants except S32K388/S32K389.
2. The FIRC_DIV_SEL is configured by the sBAF code. It is set to 11b after reset or normal standby exit and FIRC_CLK is 48 MHz. In case of fast standby exit, FIRC_DIV_SEL is 10b and FIRC_CLK is 3 MHz.
3. MC_CGM.MUX_10_CSC[SELCTL] and MC_CGM.MUX_11_CSC[SELCTL] values are don't care since QSPI_SFCK and TRACE_CLK are not used in this use case.
4. MC_CGM.MUX_0_CSC[SELCTL] must equal 0000b.

NOTE

For FIRC_CLK frequency modes less than 24 MHz, safety modules like the CMU_F*x_n* must be disabled for safety applications, because safety applications are to run on the PLL clocks. The CMU_F*x_n* will cause erroneous FHH events if not disabled.

24.7.2.15 Option F - Operation in 1:1 mode with CORE_CLK and AIPS_PLAT_CLK at same speed (For all chips except S32K388/S32K389)

This option is only available in Run mode.

Table 167. Option F - Operation in 1:1 mode with CORE_CLK and AIPS_PLAT_CLK at same speed

Clocking options	Clock frequencies			
	S32K344, S32K324, S32K314, S32K342, S32K341, and S32K322		S32K310, S32K311, and S32K312	S32K328, S32K338, S32K348, S32K356, and S32K358
PLL VCO frequency	960 MHz		960 MHz	960 MHz
PLLODIV2_CLK (PLLDIG.PLLDV[ODIV2])	480 MHz (0010b)		240 MHz (0100b)	480 MHz (0010b)
FIRC_CLK (HSE_B.CONFIG_REG_GPR[FIRC_DIV_SEL])	48 MHz (11b)			
PLL_PHI1_CLK-related clocks [1]				
PLL_PHI1_CLK (PLLDIG.PLLODIV_1[DIV])	240 MHz (0001b)	160 MHz (0010b)	48 MHz (0100b)	480 MHz (0000b)
QSPI_SFCK (MC_CGM.MUX_10_DC_0[DIV])	120 MHz (001b)	80 MHz (001b)	—	120 MHz[2] (0000b)
TRACE_CLK (MC_CGM.MUX_11_DC_0[DIV])	**For fast pads**		—	**For fast pads**
	120 MHz (001b)	80 MHz (001b)	—	120 MHz[2] (0001b)

Table continues on the next page...

Table 167. Option F - Operation in 1:1 mode with CORE_CLK and AIPS_PLAT_CLK at same speed (continued)

Clocking options	Clock frequencies			
	S32K344, S32K324, S32K314, S32K342, S32K341, and S32K322		S32K310, S32K311, and S32K312	S32K328, S32K338, S32K348, S32K356, and S32K358
	For standard-plus pads		—	
	24 MHz (1001b)	16 MHz (1001b)	—	
PLL_AUX-related clocks[3]				
PLL_AUX_VCO_CLK	NA			1000 MHz	800 MHz
PLLAUXODIV2_CLK (PLLDIG.PLLDV[ODIV2])	NA			500 MHz (0000b)	400 MHz (0000b)
PLL_AUX_PHI0_CLK-related clocks				
PLL_AUX_PHI0_CLK (PLLDIG.PLLODIV_0[DIV])	NA			500 MHz (0010b)	400 MHz (0010b)
QSPI_SFCK (MC_CGM.MUX_10_DC_0[DIV])	NA			NA	100 MHz[2] (0000b)
GMAC_CLK_TX (MC_CGM.MUX_8_DC_0[DIV])	NA			125 MHz [2] (0001b)	NA
TRACE_CLK (MC_CGM.MUX_11_DC_0[DIV])	NA			**For standard-plus pads** 25 MHz[2] (1001b)	NA
PLL_AUX_PHI1_CLK (PLLDIG.PLLODIV_1[DIV])	NA			25 MHz (10011b)	
PLL_AUX_PHI2_CLK (PLLDIG.PLLODIV_2[DIV])	NA			100 MHz (0100b)	
PLL_PHI0_CLK-related clocks [4]				
PLL_PHI0_CLK (PLLDIG.PLLODIV_0[DIV])	160 MHz (010b)		80 MHz (010b)	160 MHz (010b)	
CORE_CLK • Application cores • AXBS • SRAM	80 MHz (001b)		80 MHz (000b)	80 MHz (001b)	

Table continues on the next page...

Table 167. Option F - Operation in 1:1 mode with CORE_CLK and AIPS_PLAT_CLK at same speed (continued)

Clocking options	Clock frequencies		
	S32K344, S32K324, S32K314, S32K342, S32K341, and S32K322	S32K310, S32K311, and S32K312	S32K328, S32K338, S32K348, S32K356, and S32K358
• Flash memory controller port clock • AIPS0 (high-speed peripheral clock) (MC_CGM.MUX_0_DC_0[DIV])			
QSPI_MEM_CLK (MC_CGM.MUX_0_DC_6[DIV])	160 MHz (000b)	—	160 MHz (000b)
AIPS_PLAT_CLK (medium-speed peripheral clock) (MC_CGM.MUX_0_DC_1[DIV])	80 MHz (001b)	80 MHz (000b)	80 MHz (001b)
AIPS_SLOW_CLK (slow-speed peripheral clock) (MC_CGM.MUX_0_DC_2[DIV])	40 MHz (011b)	40 MHz (001b)	40 MHz (011b)
DCM_CLK (MC_CGM.MUX_0_DC_4[DIV])	40 MHz (011b)	40 MHz (001b)	40 MHz (011b)
HSE_CLK (MC_CGM.MUX_0_DC_3[DIV])	80 MHz (001b)	80 MHz (000b)	80 MHz (001b)
LBIST_CLK (MC_CGM.MUX_0_DC_5[DIV])	40 MHz (011b)	40 MHz (001b)	40 MHz (011b)

1. MC_CGM.MUX_10_CSC[SELCTL] and MC_CGM.MUX_11_CSC[SELCTL] must equal 1001b.
2. The final clock frequency is derived from additional divider stages. See the chip's clock diagrams for details.
3. MC_CGM.MUX_10_CSC[SELCTL] and MC_CGM.MUX_11_CSC[SELCTL] must equal 1100b.
4. MC_CGM.MUX_0_CSC[SELCTL] must equal 1000b.

24.7.3 Gasket configurations in various clocking modes

Table 168. Gasket configurations in various clocking modes (for S32K388/S32K389)

Gasket configuration	Option A++[1]	Option A+[2]	Option A[3]	Option C (Boot)[4]	Option D[5]	Option E[6]	Option E2[7]
eDMA (S0)	1:1	Bypass / 1:1 (3 priority)	1:1	Bypass			
HSE_B	1:1			Bypass			
AES_ACCEL	1:1			Bypass			
AES_SLAVE	1:1						

Table continues on the next page...

Table 168. Gasket configurations in various clocking modes (for S32K388/S32K389) (continued)

Gasket configuration	Option A++[1]	Option A+[2]	Option A[3]	Option C (Boot)[4]	Option D[5]	Option E[6]	Option E2[7]
AIPS1/2/3	1:1	Bypass/1:1 (1 priority)	1:1	Bypass			
QuadSPI	2:1			Bypass			
PRAMC	WS enabled			WS disabled			
Cortex M7_Core	2:1		1:1				
GMAC	1:1						
BDRAM 64:32	1:2		1:1				

1. See Option A++ - Very High Performance mode (CM7_CORE_CLK @ 320 MHz) (For S32K388/S32K389) for details.
2. See Option A+ - Very High Performance mode (CM7_CORE_CLK @ 240 MHz) (For S32K388/S32K389) for details.
3. See Option A - High Performance mode (CM7_CORE_CLK @ 160 MHz) (For S32K388/S32K389 for details.
4. See Option C - Boot Standby mode (CM7_CORE_CLK @ 24 MHz) (For S32K388/S32K389) for details.
5. See Option D - Low-Speed Run mode (CM7_CORE_CLK @ 48 MHz) (For S32K388/S32K389) for details.
6. See Option E - Low-Speed Run mode (CORE_CLK @ 3 MHz) (For S32K388/S32K389) for details.
7. See Option E2 - Very-Low-Speed Run mode (CORE_CLK @ 750 KHz) (For S32K388/S32K389) for details.

Table 169. Gasket configurations in various clocking modes (for S32K328, S32K338, S32K348, S32K356, and S32K358)

Gasket configuration	Option A+[1]	Option A[2]	Option B[3]	Option C[4]	Option D[5]	Option E[6]	Option E2[7]	Option F[8]
eDMA /STAM (S0, S1)	1:1		Bypass					
HSE_B	1:2		1:1	Bypass				
APIS0	1:1	1:1/Bypass	Bypass					
AIPS1/2	2:1		2:1	Bypass				
QuadSPI	2:1		2:1	Bypass				1:1
PRAM GSKT	1:1		Bypass					
Flash Read Path pipeline	1:1	Bypass						
PRAM/SRAM	WS enabled			WS disabled				
BDRAM 64:32	1:1							
GMAC	1:1							
uSDHC	1:2			1:1				

1. See Option A+ - Very High Performance mode (CORE_CLK @ 240 MHz) (For S32K328, S32K338, S32K348, S32K356, and S32K358) for details.
2. See Option A - High Performance mode (CORE_CLK @ 160 MHz) for details.
3. See Option B - Reduced Speed mode (CORE_CLK @ 120 MHz) for details.
4. See Option C - Boot Standby mode (CORE_CLK @ 24 MHz) for details.
5. See Option D - Low-Speed Run mode (CORE_CLK @ 48 MHz) for details.
6. See Option E - Low-Speed Run mode (CORE_CLK @ 3 MHz) for details.
7. See Option E2 - Very-Low-Speed Run mode (CORE_CLK @ 750 KHz) for details.
8. See Option F - Operation in 1:1 mode with CORE_CLK and AIPS_PLAT_CLK at same speed (For all chips except S32K388/S32K389) for details.

Table 170. Gasket configurations in various clocking modes (for S32K344, S32K324, S32K314, S32K342, S32K341, and S32K322)

Gasket configurations	Option A[1]	Option B[2]	Option C[3]	Option D[4]	Option E[5]	Option E2[6]	Option F[7]
eDMA (S0)	1:1	Bypass					
eDMA (S1)	Bypass						
HSE_B	1:2	1:1, 1:2	Bypass				
AIPS1/AIPS2	2:1		Bypass				
QuadSPI	2:1		Bypass				
PRAM/SRAM	WS enabled		WS disabled				
EMAC 32:64	1:1						
BDRAM 64:32	1:1						

1. See Option A - High Performance mode (CORE_CLK @ 160 MHz) for details.
2. See Option B - Reduced Speed mode (CORE_CLK @ 120 MHz) for details.
3. See Option C - Boot Standby mode (CORE_CLK @ 24 MHz) for details.
4. See Option D - Low-Speed Run mode (CORE_CLK @ 48 MHz) for details.
5. See Option E - Low-Speed Run mode (CORE_CLK @ 3 MHz) for details.
6. See Option E2 - Very-Low-Speed Run mode (CORE_CLK @ 750 KHz) for details.
7. See Option F - Operation in 1:1 mode with CORE_CLK and AIPS_PLAT_CLK at same speed (For all chips except S32K388/S32K389) for details.

Table 171. Gasket configurations in various clocking modes (for S32K311 and S32K312)

Gasket configurations	Option B[1]	Option C[2]	Option D[3]	Option E[4]	Option E2[5]	Option F[6]
HSE_B	1:1, 1:2	Bypass				
AIPS1	2:1	Bypass				
PRAM/SRAM	WS disabled					
BDRAM 64:32 (TCM WS)	1:1					

1. See Option B - Reduced Speed mode (CORE_CLK @ 120 MHz) for details.
2. See Option C - Boot Standby mode (CORE_CLK @ 24 MHz) for details.
3. See Option D - Low-Speed Run mode (CORE_CLK @ 48 MHz) for details.
4. See Option E - Low-Speed Run mode (CORE_CLK @ 3 MHz) for details.
5. See Option E2 - Very-Low-Speed Run mode (CORE_CLK @ 750 KHz) for details.
6. See Option F - Operation in 1:1 mode with CORE_CLK and AIPS_PLAT_CLK at same speed (For all chips except S32K388/S32K389) for details.

24.7.4 Default clock configuration

At reset recovery, the chip runs on the FIRC_CLK as the default configuration as shown in Option C - Boot Standby mode (CORE_CLK @ 24 MHz). Clocking configuration Option_C is the default configuration out of reset with the HSE_B core as the boot core. The Cortex-M7_*n* application core clocks are gated by default. You need to enable the core clocks by the configuring the corresponding core clock enable bits shown in Core Clock Gating.

24.7.5 PCFS

The chip supports software-controllable PCFS for MC_CGM MUX_0 (see section "Progressive Clock Frequency Switching (PCFS)" in the "Clock Generation Module (MC_CGM)" chapter for details). PCFS increases and decreases the frequency in steps, avoiding any overshoots or undershoots. When a functional reset event occurs with PCFS enabled, the PCFS process runs and is then followed by the divider configuration updates.

24.7.6 Updating dividers: crossbar halt handshake

The clock divider update process consists of the crossbar halt handshake sequence (see section "Clock dividers update" in the "Clock Generation Module (MC_CGM)" chapter for the clock divider update process). A divider update asserts a request to the crossbar switch to halt any transaction that is in process. The dividers are updated when the crossbar switch acknowledges the request for halt. The halt request disables the crossbar switch gaskets in the following order:

1. Core gaskets (HSE_B gaskets)
2. Crossbar switch (AXBS)
3. Flash AXBS bridge
4. PRAM/SRAM gasket

Dividers are updated after the gaskets acknowledge the halt request.

24.7.7 Changing system clock configurations

Application software sequence for switching clock configurations from FIRC to PLL or PLL to PLL:

1. Before changing the system clock dividers or before system clock switching, the communication modules working on the system clock should be disabled by the application software to avoid any erroneous communication during the clock transition.
2. The peripherals working on the system clock should be clock-gated used the MC_ME.PRTNx_COFBy_CLKEN configurations. The peripherals working on non-system clocks which are not being switched, can continue to operate.
3. All cores must be clock-gated, except the core being used to control the clock switching.
4. System clock switching can be done by configuration of MC_CGM_MUX_0_CSC[SELCTL].

Application software sequence for switching clock configurations from PLL to FIRC clock switching:

1. Before changing the system clock dividers or before system clock switching, the communication modules working on the system clock should be disabled by the application software to avoid any erroneous communication during the clock transition.
2. The peripherals working on the system clock should be clock-gated used the MC_ME.PRTNx_COFBy_CLKEN configurations. The peripherals working on non-system clocks which are not being switched, can continue to operate.
3. All cores must be clock-gated, except the core being used to control the clock switching.
4. System clock switching can be done by configuration of MC_CGM_MUX_0_CSC[SELCTL].

NOTE

See the Clock divider update section in the MC_CGM chapter, which outlines the procedure for updating the dividers.

NOTE

When enabling PLL, the PMC last mile regulator should be enabled first by configuring PMC_CONFIG[LMEN] and PMC_CONFIG[LMBCTLEN] (if using an external BJT). The last mile regulator must be disabled after PLL is disabled. See the "Power Management" chapter in this reference manual for details on this module's availability on your chip variant.

24.8 Clock monitoring

The chip contains an independent clock monitoring mechanism which signals malfunctions in the clocking system. This chip consists of seven Clock Monitoring Units (CMU_F*x*_[0:6]) for monitoring system clock and clocking module outputs. Figure 120 and Figure 121 show a lower-level view of the clocking monitoring system. Table 172 describes each CMU instance. Each CMU instance provides an independent interrupt or reset indication when the clock signal is out of range or lost. The CMU_FM_*n* provides a timeout indication in case there is a loss of metered clock. Your application software must periodically check the CMU_FM_1 and CMU_FM_2 status within the chip FTTI (as specified in the Safety Manual).

NOTE

You must disable the CMU corresponding to the system clocks if the application changes the system clock source or changes the system clock divider configuration.

You must disable the CMU monitoring a clock source before disabling the clock source, then enable it after enabling the clock source.

The CMUs should be turned ON only after device has moved to PLL source (wherein LMR is ON).

In the below figure, the Mask input in CMU_FC_O comes from MC_RGM which is asserted at time of functional reset entry and prevents false interrupt / reset generation.

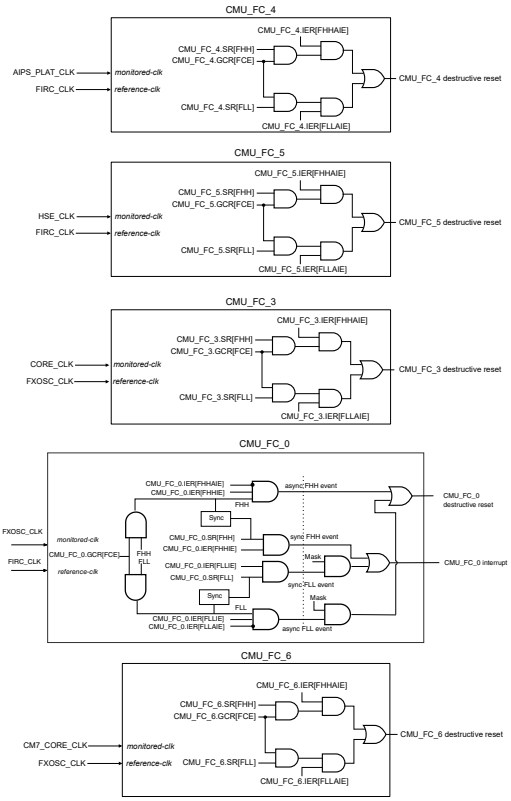


Figure 120. Frequency checking (FC) instances

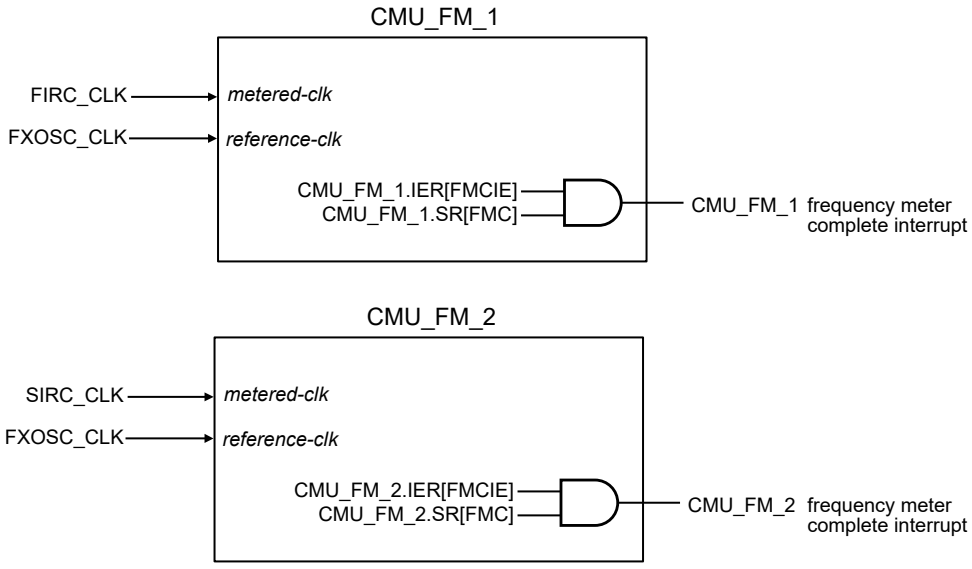


Figure 121. Frequency metering (FM) instances

Table 172. System clock monitors

CMU	Reference clock	Monitored or metered clock	Failure reaction	Monitoring type description
CMU_FC_0	FIRC_CLK	FXOSC_CLK	Destructive reset or interrupt	Precision over and under frequency
CMU_FM_1	FXOSC_CLK	FIRC_CLK	Interrupt	Current frequency measurement periodically triggered by application software
CMU_FM_2	FXOSC_CLK	SIRC_CLK	Interrupt	Current frequency measurement periodically triggered by application software
CMU_FC_3	FXOSC_CLK	CORE_CLK	Destructive reset	Precision over and under frequency
CMU_FC_4	FIRC_CLK	AIPS_PLAT_CLK	Destructive reset	Precision over and under frequency
CMU_FC_5	FIRC_CLK	HSE_CLK	Destructive reset	Precision over and under frequency
CMU_FC_6	FXOSC_CLK	CM7_CORE_CLK	Destructive reset	Precision over and under frequency

NOTE

See the clock system diagrams in the "Clocking Overview" section for details on the monitored clocks availability on your chip variant.

24.9 Glossary

MODULE_CLK Module operating clock

REG_INTF_CLK Module register interface clock used for register read and write

PCFS	Progressive Clock Frequency Switching (see section "Progressive Clock Frequency Switching (PCFS)" for details)
POR	Power On Reset
SBC	System Basis Chip (see NXP SBC portfolio)
FLL	Frequency lower than low frequency reference
FHH	Frequency higher than high frequency reference
FTTI	Fault Tolerance Time Interval

Chapter 25
Clock Generation Module (MC_CGM)

25.1 Chip-specific MC_CGM information

25.1.1 Associated content references

See the Clocking chapter for details pertaining to these:

- Chip clocking
- MC_CGM clock source mapping (see the "MC_CGM clock sources mapping" section for this)
- MC_CGM clock multiplexers (see the "MC_CGM clock multiplexers" section for this)

NOTE

Clock sources listed in the MUX_n_CSC[SELCTL] bit field are defined based on S32K388 and S32K389. For other variants, see the clock system diagrams in section “Clocking overview” in the “Clocking ” chapter.

CAUTION

MC_CGM clock multiplexer configurations to non-supported and reserved clock sources are prohibited and can result in chip malfunctioning.

25.1.2 Clock Mux 7 select control register MUX_7_CSC[SELCTL] description (S32K328, S32K338, S32K348, S32K356, and S32K358)

Table 173. MUX_7_CSC[SELCTL] description

Bit field	Description
28-24 SELCTL	Clock source selection control Selects the source clock for clock mux 7. The reserved values are not displayed. 0_0000b - FIRC_CLK 0_1000b - PLL_PHI0_CLK 0_1100b - PLL_AUX_PHI0_CLK 1_1000b - GMAC_MII_RMII_RGMII_TX_CLK 1_1001b - GMAC_MII_RGMII_RX_CLK

25.1.3 Clock Mux 7 select control register MUX_7_CSC[SELCTL] description (S32K344, S32K324, S32K314, S32K322, S32K341 and S32K342)

Table 174. MUX_7_CSC[SELCTL] description

Bit field	Description
28-24 SELCTL	Clock source selection control Selects the source clock for clock mux 7. The reserved values are not displayed. 0_0000b - FIRC_CLK 1_1000b - EMAC_MII_RMII_TX_CLK 1_1001b - EMAC_MII_RX_CLK

25.2 Introduction

The clock generation module (MC_CGM) is used to set up the configurable clock domains used by various chip blocks as per the application needs. It includes the clock multiplexers that allow software to select the desired clock sources for these domains. This is managed by the MC_CGM to ensure that the changing of the clock selection from one source to another occurs in a glitch-less fashion. In addition, the MC_CGM includes the clock dividers that can be configured by software.

See following figure for the MC_CGM block diagram:

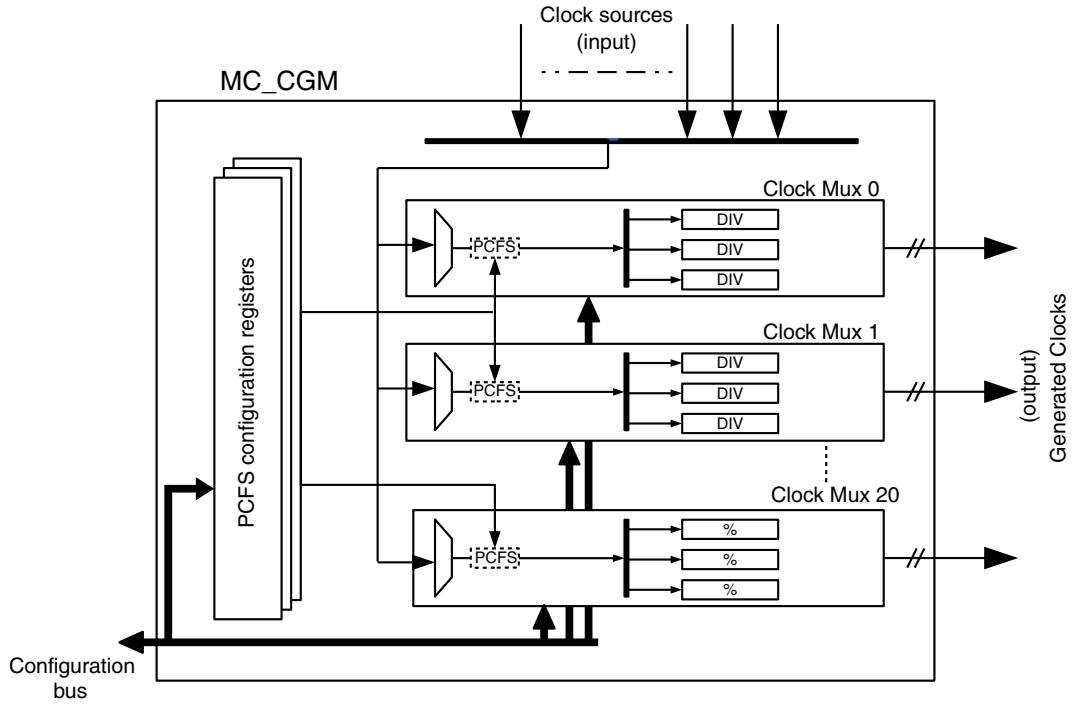


Figure 122. MC_CGM block diagram

NOTE

The block diagram is generic and does not necessarily reflect any specific MC_CGM implementation.

25.3 Features

MC_CGM includes the following features:

- Implements software configurable clock multiplexers for selecting from various clock sources
- Provides hardware-controlled multiplexers that guarantee glitchless transition, while the software-controlled multiplexers need a software sequence to ensure such a transition
- Provides software configurable automatic PCFS on certain clocks to minimize the impact of a sudden power consumption change through a gentle ramp-down and -up of the clock frequency when switching clock sources
- Implements software configurable clock dividers

25.4 Functional description

25.4.1 Clock selection multiplexer

Each of the clock multiplexers inside the MC_CGM either implements a fully hardware-controlled clock multiplexer or a software-controlled clock multiplexer. The following sections describe the two variants of the clock multiplexer.

25.4.1.1 Hardware-controlled clock multiplexer

In hardware-based clock multiplexing, the underlying assumption is that under some conditions, error or reset states, software may not be active. Therefore, the clock switching is fully hardware based and is glitchless. To facilitate clock switching requests with software, the MUX_n_CSC and MUX_n_CSS registers implement request and status for the clock multiplexer, the rest is managed in hardware. Using these registers, the software can monitor the state of the hardware-based clock multiplexer and also make clock switching requests. It is recommended that a new clock switch request is given only when there are no pending/ongoing clock switching requests. However, a switch to the safe clock, that is, FIRC, can be performed at any instance of time. A switch to the safe clock is always completed. Software should ensure that while making a software-based switch to safe clock, the register configuration clock should be available, at least for completing the write to the MUX_n_CSC register. This means that the clock should be running for the register write to complete. Hardware clock multiplexer also supports hardware-based switch to safe clock, which is requested externally to MC_CGM (for example, by MC_RGM). For a hardware-based switch to safe clock, it is not required to have a register configuration clock for MC_CGM.

NOTE

- "Switch to safe clock" from software is requested by programming the MUX_n_CSC[SAFE_SW] bit field only and not by combining the MUX_n_CSC[CLK_SW] and MUX_n_CSC[SELCTL] bit fields of the MUX_n_CSC register. Writes to other fields are ignored when requesting switch for safe clock.
- After the switch to the safe clock requested by the MC_RGM has completed, the MC_RGM also requests the clock dividers to switch to their default values. This hardware-triggered divider update can be monitored in the same manner as for software-configured updates, and software should use it to ensure that the configuration update has completed before reconfiguring the clocks.
- Write accesses to the MUX_n_CSC register with clock select pointing to the "reserved" input clock source are aborted with bus transfer errors.

See Figure 123 that shows the flowchart representation for the sequence of steps to be followed when making a clock switch request to hardware-controlled clock multiplexer. Switch to safe clock can be requested at any instance of time, and for clarity reasons, it is not shown in Figure 123.

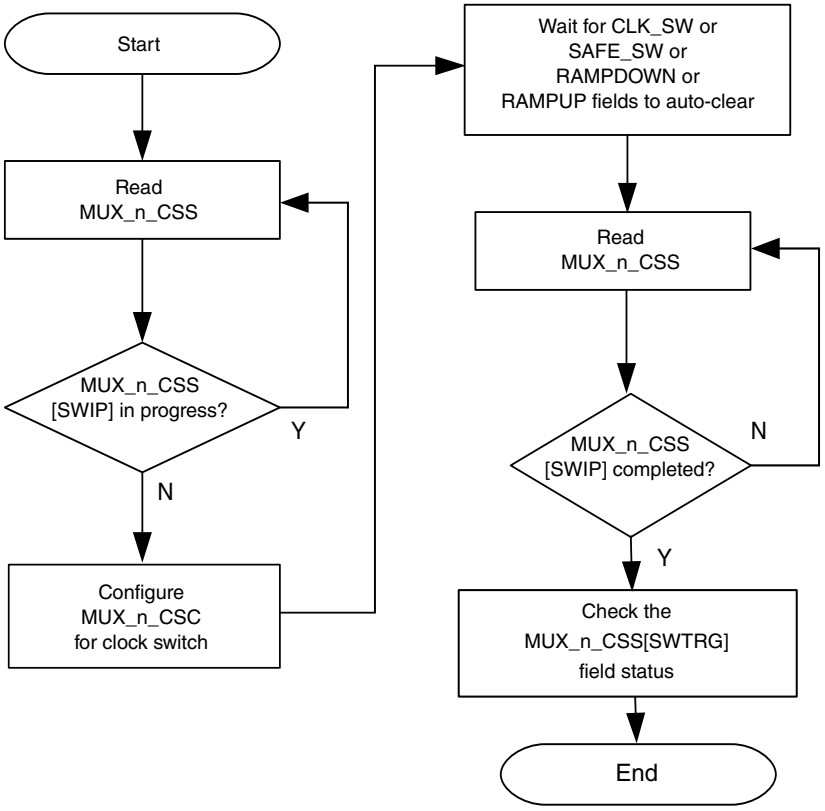


Figure 123. Flow for clock switch request on hardware-controlled clock multiplexer

NOTE

- A switch to the safe clock command always leads to a ramp-down from the currently selected clock and then a switch to "safe clock", except when there is an ongoing clock switch requested by the software without ramp-up and ramp-down. A safe clock switch request when there is an ongoing clock switch without ramp-up or ramp-down results in a switch to the safe clock without performing a ramp-down (either by MC_RGM or provided using the MUX_n_CSC register) does not perform a ramp-down before switching to "safe clock".
- The above flowchart steps can be preceded by points 1 and 3 below:
  1. Disable the divider.
  2. Switch clocks through hardware multiplexer.
  3. Enable and configure the dividers (atomic write instruction).

25.4.1.2 Software-controlled clock multiplexer

In software-based clock multiplexing, the underlying assumption is that the software is always available and there are no error or reset conditions in the chip. This implies that a glitchless switch between input clock sources of MC_CGM MUX can be achieved by following a sequence of steps in software. The software-based clock multiplexer implements a clock gate at the output of the clock mux. The software can gate the selected clock of MC_CGM MUX using a synchronous/graceful clock gate bit (that is, MUX_n_CSC[CG]) or a forced clock gate bit (that is, MUX_n_CSC[FCG]). The hardware does not guarantee that any glitches will escape when using forced clock gating. When a forced clock gate bit is written to 1, the internal clock gate forcefully gates the selected clock to logic-0, therefore, to avoid clock glitches, it should be ensured that the selected clock source is not running. See Figure 124 that shows the flowchart representation for the sequence of steps to be followed when making a clock switch request to software-controlled clock multiplexer. No switch to safe clock is supported in software-controlled clock multiplexer.

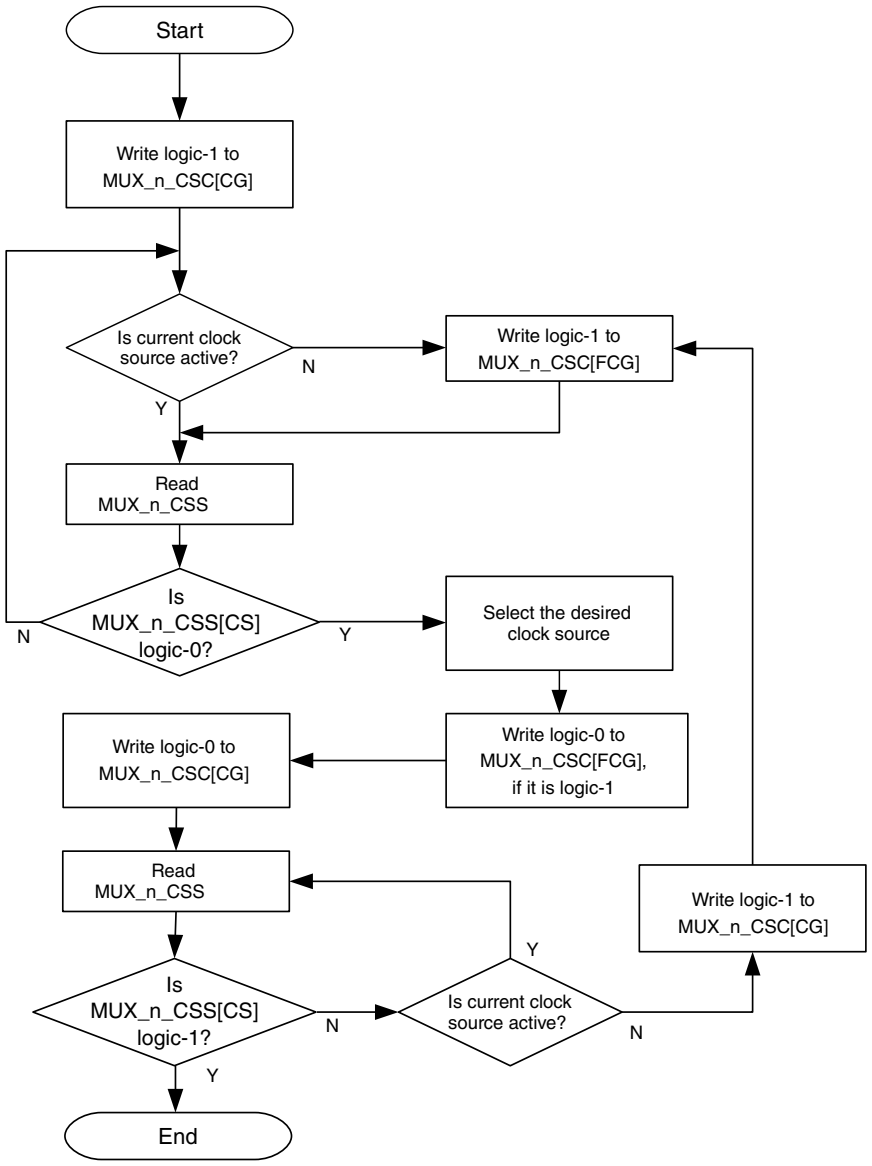


Figure 124. Flow for clock switch request on software controlled clock multiplexer

NOTE

- Ensure that before using the force gate bit, any IP or other logic using the clock of MC_CGM MUX is in the inactive state or a clock glitch resulting from usage of forced gating does not effect the IP (that is, it is clock gated after the MC_CGM).
- In Figure 124, the clock source to be selected should be active at the time of clock switch, else the MUX_n_CSS[CS] field will remain set to logic-0. In case the clock source to be selected becomes inactive (that is, loss of clock, and so on) during the switching operation, the switch to another clock source can be initiated by writing both the MUX_n_CSC[CG] and MUX_n_CSC[FCG] fields to logic-1.
- Writing a 'reserved' value for the MUX_n_CSC[SELCTL] field may result in an unpredictable clock at the output of the clock multiplexer.

NOTE

The above flowchart steps can be preceded by points 1 and 3 below:

1. Disable the divider.
2. Switch clocks through software multiplexer.
3. Enable and configure the dividers (atomic write instruction).

25.4.2 PCFS

MC_CGM implements PCFS when changing clock source at an MC_CGM clock mux. The PCFS is only implemented in a hardware-controlled clock multiplexer and not in a software-controlled clock multiplexer. It allows a gradual load change for a power/voltage supply unit by employing a gradual frequency changeover from one to another. The frequency changeover is achieved by clock division of input clock source with a sequence of division values, where frequency ramp down and ramp up is achieved when the sequence of division values are ascending and descending in nature, respectively. As the clock division (fractional) is implemented in digital logic, therefore, it is a coarse-level clock division rather than being an accurate-level division, implying that the duty cycle of the progressively divided clock can vary with time.

The PCFS feature is utilized when a drain current to frequency relationship is known, that is, for a given drain current what is the maximum allowed change in frequency (f_{chg}). The f_{chg} is the first input parameter for PCFS and other parameters for PCFS are specified or calculated in relation to FIRC.

The PCFS hardware generates the clock division factors based on certain values that are programmed into the PCFS configuration register. The following pseudo codes represent the generation of clock division value sequence (d_i).

```
/* ramp down division values (d_n) with k steps*/
      delta_1 = RATE/1000;
      delta_2 = RATE/1000;
      d_0     = 1.0;
      for i=1 to k-1 do
        d_i = d_i-1 + delta_1;
        delta_1 = delta_1 + delta_2;
      endfor
```

```
/* ramp up division values (d_n) with k steps*/
      delta_1 = RATE*k/1000;
      delta_2 = RATE/1000;
      d_0     = 1.0 + RATE*k*(k+1)/2;
      for i=1 to k-1 do
        d_i = d_i-1 - delta_1;
        delta_1 = delta_1 - delta_2;
    endfor
```

As the generation of clock division values is not a closed-bounded function, calculating RATE for a given f_{chg} is an iterative process. Find a value of RATE that produces a clock division sequence, which when used does not lead to a frequency change greater than f_{chg}. See Table 175 that tabulates some of the RATE values against a_{max}, where

$$a_{max}=f_{chg}/F_i$$

where, F_i is the frequency of ith input clock source of the clock mux.

Table 175. PCFS RATE values

a_{max}	PCFS rate
0.005	12
0.01	48

Table continues on the next page...

Table 175. PCFS RATE values (continued)

a_{max}	PCFS rate
0.15	112
0.20	184

The last clock division factor in case of ramp-down or the first clock in case of ramp-up (following a ramp-down procedure), should be such that clock switch from any input clock source to another should be termed as safe, indicating that load changes have sustainable power effects. This frequency level is referred to as "safe frequency", equivalent to frequency of FIRC referred to as "safe clock" with frequency "safe clock frequency" (f_{safe}). The last clock division factor in the sequence of clock division factors happens after "k" steps. The factor k (=steps) is calculated by using the following formula:

```
k = ceil(0.5 + sqrt(0.25 - (2000 * (1 -(Fi/fsafe)) / RATE)))
```

Using the above formula, all the PCFS register configuration values can be calculated for a given frequency of a clock source:

```
PCFS_DIVEi.DIV = (Fi/fsafe)*1000-1;
        PCFS_DIVCi.INIT = RATE * k;
        PCFS_DIVCi.RATE = RATE;
        PCFS_DIVSi.DIV  = 999 + (RATE * k * (k+1)/2);
```

See Figure 125 that shows a graphical representation of the change in frequency, which happens during PCFS ramp-down and ramp-up.

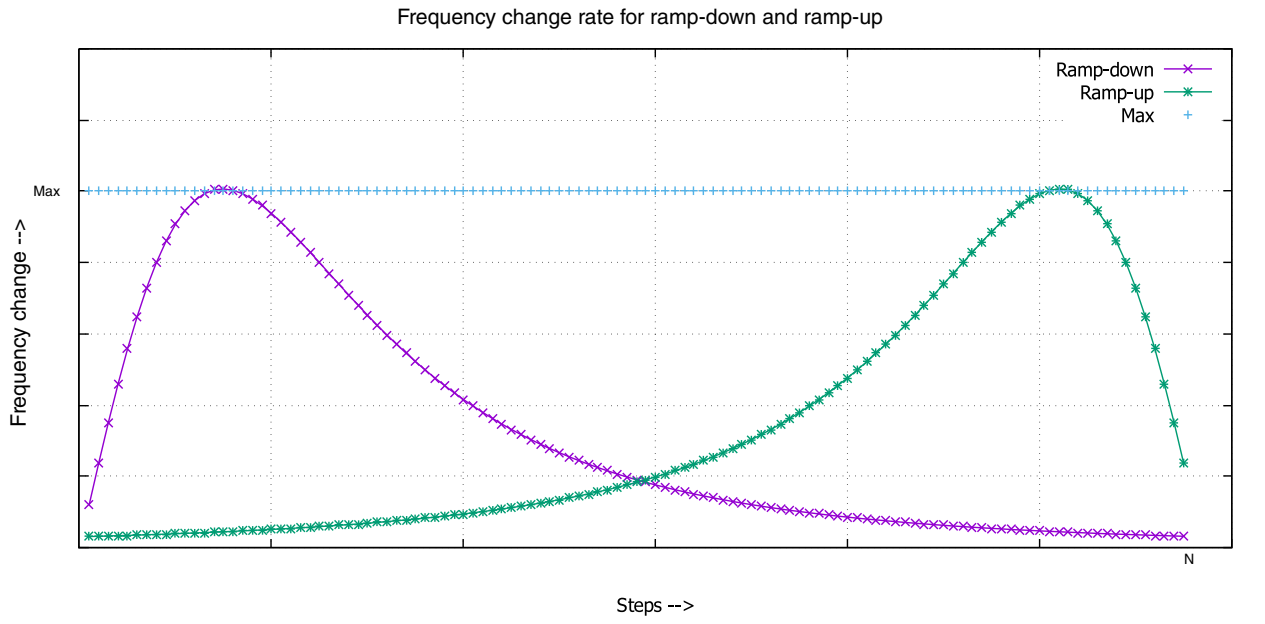


Figure 125. PCFS steps vs frequency changes

For any given clock source, if its frequency is less than that of FIRC, then its corresponding registers should be programmed to default values, where the default values are such that PCFS divider start and PCFS divider end values are same and equal to divide-by-1. The default values ensure that no progressive clock division is done when a clock switch request is given to switch from or to that source.

NOTE

Calculate the minimum frequency during the PCFS RAMPDOWN and RAMPUP operations by using the formula:

```
(FIRC/((PCFS_DIVE+1)/1000)) MHz
```

25.4.2.1 PCFS control

The PCFS operation is configured by a set of configuration registers. One set pertains to the calculation of the clock division factors, which are PCFS_DIVC, PCFS_DIVE, PCFS_DIVS, and PCFS_SDUR, while the registers MUX_n_CSC and MUX_n_CSS implement trigger and status of the PCFS operation. The clock division factors are expected to be programmed before any other MC_CGM operation is initiated and remain unchanged. All the registers corresponding to the clock division factors should be programmed with FIRC as the configuration clock and before doing any clock switch or PCFS operation on any of the MC_CGM mux. It should be noted that the default values of the register corresponding to clock division factors are such that only the clock division factor is calculated by hardware that is divide-by-1. The PCFS operation is always triggered when the safe clock request is generated except when there is an on-going clock switch without ramp-up or ramp-down. Therefore, the software needs to ensure that the PCFS configuration is complete and correct.

While configuring MUX_n_CSC, only valid PCFS and clock switch requests should be provided. PCFS or clock switch requests should only be provided if the PCFS operation is in the idle state. If there is an ongoing PCFS operation, it is recommended not to provide any new PCFS triggers (except switch to safe clock) until the ongoing operation is completed. Switch to safe clock via hardware or by register configuration can be provided at any instance of time and is always completed. Valid combinations of PCFS and clock switches triggers are listed in Table 176. All the PCFS commands should be atomic in nature, which means a single register write should provide complete PCFS sequence to be executed that is ramp-down, clock switch, and ramp-up.

Table 176. Valid PCFS and clock switch requests

PCFS operation state	Command
Idle	Ramp-down, clock switch, and ramp-up
Idle	Clock switch only (without ramp-up or ramp-down)

When a switch to safe clock is provided by writing to MUX_n_CSC, then writes to other register fields are ignored.

25.4.2.2 Clock source power-up and selection

This section provides guidelines for powering up a given clock source and selecting it at the MC_CGM clock multiplexer.

Following is the power-up procedure for a clock source:

1. Configure the parameter, if any.
2. Configure the power-up (or power-down) control field.
3. Wait for the power-up status indication.

After a power-up indication, the clock source can be selected to provide output clock at an MC_CGM clock multiplexer. A clock-monitoring setup can also be activated on the powered-up clock source.

See Figure 126 that shows a flow chart representation of this sequence.

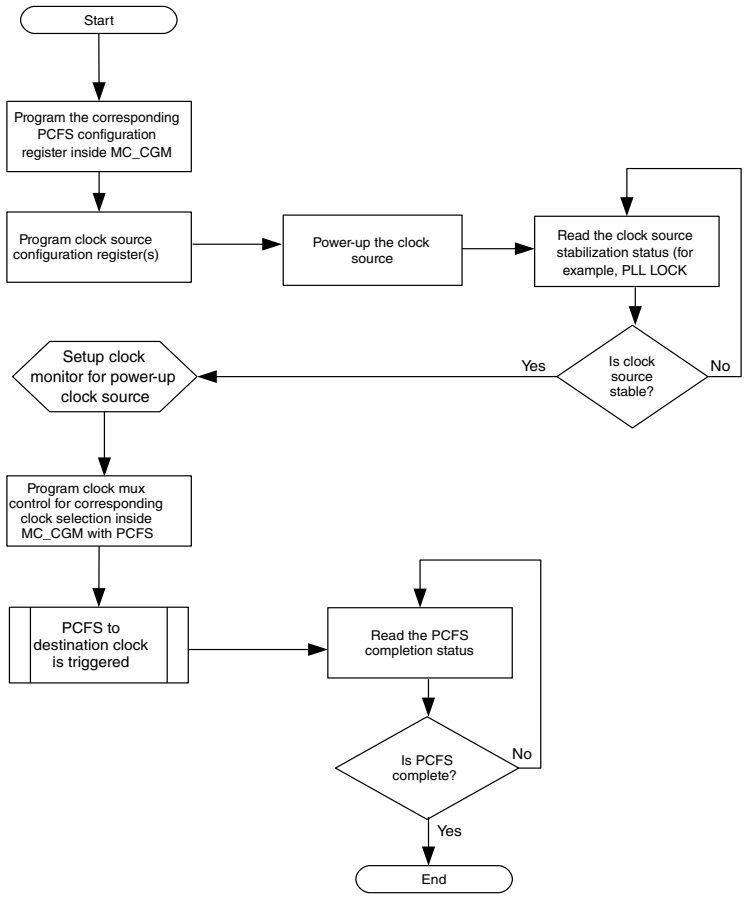


Figure 126. Clock source power-up and selection procedure

25.4.2.3 Clock source power-down and deselection

This section provides guidelines for powering down a given clock source and deselecting it at MC_CGM clock multiplexer.

Following is the power-down procedure for a clock source:

1. Deselect the clock source at all MC_CGM clock multiplexers.
2. Configure the power-up (or power-down) control field.
3. Wait for the power-down status indication.

See Figure 127 that shows a flow chart representation of this sequence.

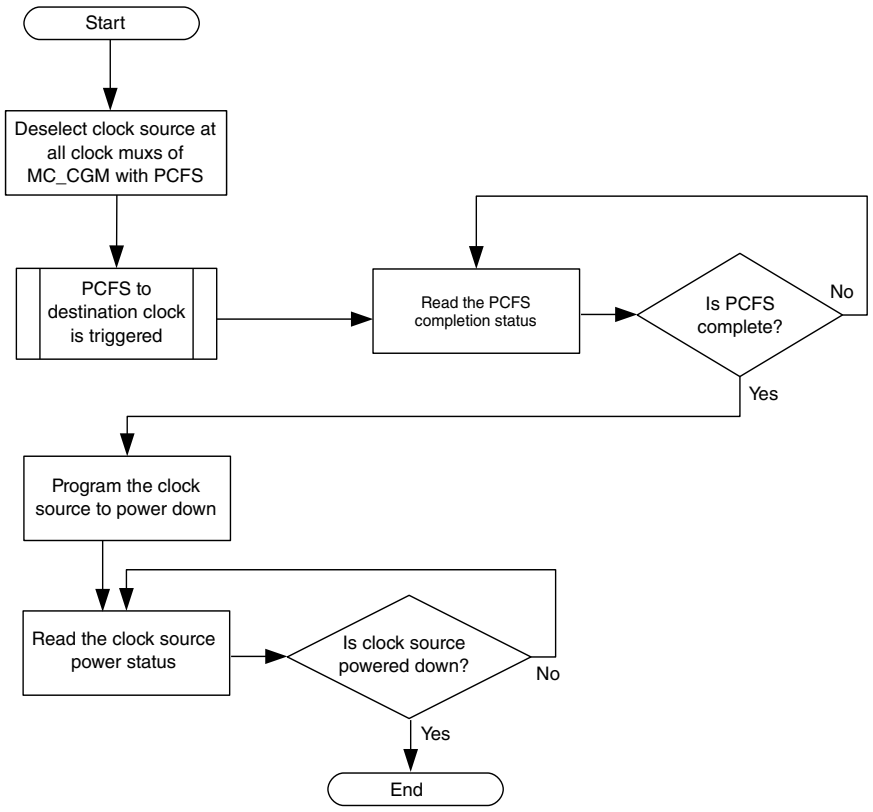


Figure 127. Clock source power-down and clock source deselection procedure

25.4.2.4 Clock switch with load change

This section provides guidelines to switch between two high-frequency clock sources along with load changes in the system. Load change is referred to switching ON or OFF of logic/peripherals in the system, which has an effect of significant capacitance changes on the chip. This triggers the voltage regulation for the chip.

When a large number of peripherals or digital logic is enabled or disabled, it is recommended that this step should be performed at a low frequency. Independent of whether a clock switch is required, this criteria needs to be met. When a clock switch is required at two high frequencies, the recommended sequence is as follows:

1. Intermediately switch to FIRC.
2. Change load (that is, enable/disable peripherals).
3. Switch to the target clock.

See the following figure that shows a flow chart representation of this sequence.

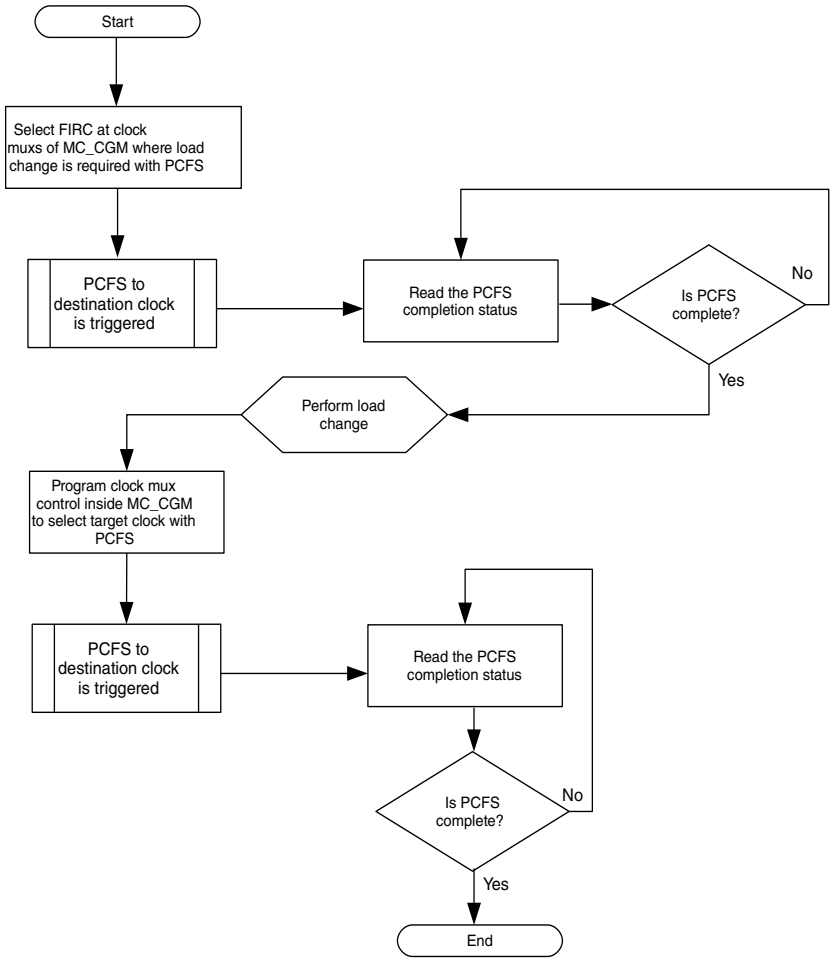


Figure 128. Clock switch with load change procedure

25.4.3 Clock dividers

Clock dividers are used for the generation of a divided clock that is used for running IPs or peripherals. The MC_CGM provides the following built-in dividers at each clock mux:

- 50% clock dividers

Each divider can be controlled by the Divider Enable (DE) bit and the Division Value (DIV) field. If a divider has its DE bit set to logic-0 in the respective configuration register, then that divider is disabled and the output divided clock is held to logic-0. If the DE bit is logic-1, the divider is enabled and provides a divided clock according to the value set in the DIV field.

25.4.3.1 50% clock divider

50% duty cycle dividers generate a real divided clock. The division factor is always an integer but is not restricted to even numbers. The rising edge of the divided clock is always synchronous to the rising edge of the divider source clock, but the falling edge is synchronous to the rising edge or the falling edge depending on whether the division factor is even or odd, respectively. If the input clock has a duty cycle of 50%, the divided output clock maintains the same 50% duty cycle. See Figure 129 that shows the 50% clock divider operation and its associated signals.

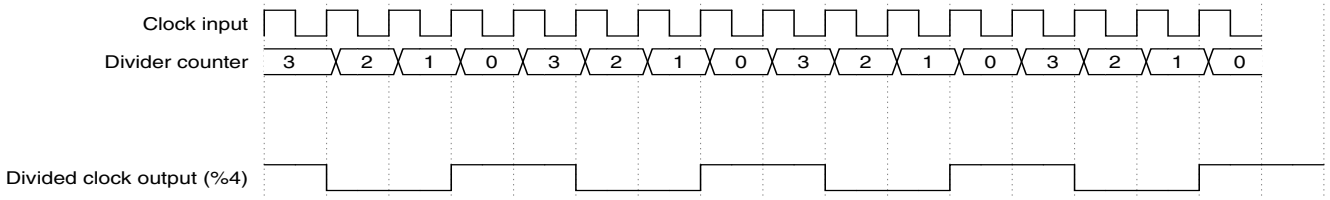


Figure 129. 50% clock divider programmed for divide by 4 (DIV=3)

A 50% divided clock can be considered asynchronous (not edge aligned) to other divided clocks from dividers at the same clock mux. 50% clock dividers are implemented without an active closed loop, and are expected not to get stuck if the input clock glitches for a single cycle.

25.4.3.2 Clock dividers update

To update the division value or the divider enable, the software should follow the procedure as shown in Figure 130. These updates happen only after the current division cycle has elapsed. However, if the phase of the clock divider is updated, the update happens independently from the state of the division cycle. Any update to the clock divider fields does not result in clock glitch either at the divided clock output or the phase-divided clock output.

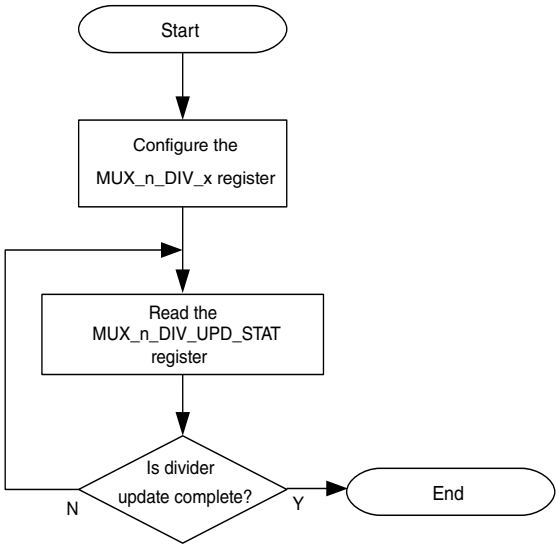


Figure 130. Flow for clock divider programming update

Following is the procedure for updating the dividers using the common trigger update:

1. Configure the MUX_n_DIV_TRIG_CTRL register.
2. Wait for the update to finish (until MUX_n_DIV_UPD_STAT is 0).
3. Update the clock dividers (only 50%) per the divider update procedure.
4. After the divider update is finished, perform a write operation on the MUX_n_DIV_TRIG register.
5. Wait for the update to finish, that is, until MUX_n_DIV_UPD_STAT is 0. During this period, the following process takes place:
   - Halt handshake is initiated if configured in step 1.
   - Clock dividers is updated only when AXBS is halted (that is, halt acknowledgment is received by MC_CGM). It is initiated, else the dividers are updated at alignment.
   - After the clock dividers are updated, MUX_n_DIV_UPD_STAT is asserted to 0.

- When the bit fields MUX_x_DIV_TRIG_CTRL[TCTL] and MUX_x_DIV_TRIG_CTRL[HHEN] are set to 1, then any write operation on trigger register will assert (MUX_n_DIV_UPD_STAT). Once the dividers are updated and aligned (MUX_n_DIV_UPD_STAT) will be deasserted.

NOTE

- The MUX_x_DIV_TRIG_CTRL[HHEN] bit should only be set when MUX_x_DIV_TRIG_CTRL[TCTL] is set, otherwise it may lead to misalignment of the dividers.
- In case of divider initialization by MC_RGM, a halt handshake protocol is initiated if the corresponding register bit is set and the clock dividers are initialized after the halt handshake protocol completion.

6. This completes the divider update.
7. When multiple writes to the dividers of same clock MUX is made without waiting for the previous update status signal to finish may lead to misalignment of the dividers.

For aligned dividers, the LCM of the division values programmed in the dividers of respective clock mux should be less than 100.

NOTE

Performing multiple writes to the divider without waiting for the earlier update to complete can lead to misalignment of the dividers.

Recommended software sequence for ensuring no undivided output at MC_CGM:

1. Reset is de-asserted
2. MC_RGM goes to IDLE
3. Enable the clock dividers of MC_CGM to provide FIRC clock so that reset of fixed dividers can be lifted.
4. Program the MC_CGM as per use case division values.
5. Switch the clock of MC_CGM Mux to desired one, and run the system.

25.5 MC_CGM register descriptions

MC_CGM implements a set of clock multiplexers that share PCFS configuration registers. MC_CGM registers have the following properties:

- All registers are 32-bit wide.
- Only 32-bit read and write accesses are supported.
- Read/write accesses of less than 32 bits terminate with an error.
- Writes to read-only register fields in writable registers are ignored and do not provide an error response.
- Writes to read-only registers are aborted with an error response.

25.5.1 MC_CGM memory map

MC_CGM base address: 402D_8000h

Offset	Register	Width (In bits)	Access	Reset value
0h	PCFS Step Duration (PCFS_SDUR)	32	RW	0000_0000h
58h	PCFS Divider Change 8 Register (PCFS_DIVC8)	32	RW	0000_0000h
5Ch	PCFS Divider End 8 Register (PCFS_DIVE8)	32	RW	0000_03E7h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
60h	PCFS Divider Start 8 Register (PCFS_DIVS8)	32	RW	0000_03E7h
300h	Clock Mux 0 Select Control Register (MUX_0_CSC)	32	RW	0000_0000h
304h	Clock Mux 0 Select Status Register (MUX_0_CSS)	32	R	0008_0000h
308h	Clock Mux 0 Divider 0 Control Register (MUX_0_DC_0)	32	RW	8000_0000h
30Ch	Clock Mux 0 Divider 1 Control Register (MUX_0_DC_1)	32	RW	8000_0000h
310h	Clock Mux 0 Divider 2 Control Register (MUX_0_DC_2)	32	RW	8001_0000h
314h	Clock Mux 0 Divider 3 Control Register (MUX_0_DC_3)	32	RW	8000_0000h
318h	Clock Mux 0 Divider 4 Control Register (MUX_0_DC_4)	32	RW	8000_0000h
31Ch	Clock Mux 0 Divider 5 Control Register (MUX_0_DC_5)	32	RW	8007_0000h
320h	Clock Mux 0 Divider 6 Control Register (MUX_0_DC_6)	32	RW	8000_0000h
324h	Clock Mux 0 Divider 7 Control Register (MUX_0_DC_7)	32	RW	8000_0000h
334h	Clock Mux 0 Divider Trigger Control Register (MUX_0_DIV_TRIG_CTRL)	32	RW	0000_0000h
338h	Clock Mux 0 Divider Trigger Register (MUX_0_DIV_TRIG)	32	RW	0000_0000h
33Ch	Clock Mux 0 Divider Update Status Register (MUX_0_DIV_UPD_STAT)	32	R	0000_0000h
340h	Clock Mux 1 Select Control Register (MUX_1_CSC)	32	RW	0000_0000h
344h	Clock Mux 1 Select Status Register (MUX_1_CSS)	32	R	0008_0000h
348h	Clock Mux 1 Divider 0 Control Register (MUX_1_DC_0)	32	RW	0000_0000h
37Ch	Clock Mux 1 Divider Update Status Register (MUX_1_DIV_UPD_STAT)	32	R	0000_0000h
380h	Clock Mux 2 Select Control Register (MUX_2_CSC)	32	RW	0000_0000h
384h	Clock Mux 2 Select Status Register (MUX_2_CSS)	32	R	0008_0000h
388h	Clock Mux 2 Divider 0 Control Register (MUX_2_DC_0)	32	RW	0000_0000h
3BCh	Clock Mux 2 Divider Update Status Register (MUX_2_DIV_UPD_STAT)	32	R	0000_0000h
3C0h	Clock Mux 3 Select Control Register (MUX_3_CSC)	32	RW	0000_0000h
3C4h	Clock Mux 3 Select Status Register (MUX_3_CSS)	32	R	0008_0000h
3C8h	Clock Mux 3 Divider 0 Control Register (MUX_3_DC_0)	32	RW	0000_0000h
3FCh	Clock Mux 3 Divider Update Status Register (MUX_3_DIV_UPD_STAT)	32	R	0000_0000h
400h	Clock Mux 4 Select Control Register (MUX_4_CSC)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
404h	Clock Mux 4 Select Status Register (MUX_4_CSS)	32	R	0008_0000h
408h	Clock Mux 4 Divider 0 Control Register (MUX_4_DC_0)	32	RW	0000_0000h
43Ch	Clock Mux 4 Divider Update Status Register (MUX_4_DIV_UPD_STAT)	32	R	0000_0000h
440h	Clock Mux 5 Select Control Register (MUX_5_CSC)	32	RW	0000_0000h
444h	Clock Mux 5 Select Status Register (MUX_5_CSS)	32	R	0002_0000h
448h	Clock Mux 5 Divider 0 Control Register (MUX_5_DC_0)	32	RW	8001_0000h
47Ch	Clock Mux 5 Divider Update Status Register (MUX_5_DIV_UPD_STAT)	32	R	0000_0000h
480h	Clock Mux 6 Select Control Register (MUX_6_CSC)	32	RW	0000_0000h
484h	Clock Mux 6 Select Status Register (MUX_6_CSS)	32	R	0002_0000h
488h	Clock Mux 6 Divider 0 Control Register (MUX_6_DC_0)	32	RW	8001_0000h
4BCh	Clock Mux 6 Divider Update Status Register (MUX_6_DIV_UPD_STAT)	32	R	0000_0000h
4C0h	Clock Mux 7 Select Control Register (MUX_7_CSC)	32	RW	0000_0000h
4C4h	Clock Mux 7 Select Status Register (MUX_7_CSS)	32	R	0008_0000h
4C8h	Clock Mux 7 Divider 0 Control Register (MUX_7_DC_0)	32	RW	0000_0000h
4FCh	Clock Mux 7 Divider Update Status Register (MUX_7_DIV_UPD_STAT)	32	R	0000_0000h
500h	Clock Mux 8 Select Control Register (MUX_8_CSC)	32	RW	0000_0000h
504h	Clock Mux 8 Select Status Register (MUX_8_CSS)	32	R	0008_0000h
508h	Clock Mux 8 Divider 0 Control Register (MUX_8_DC_0)	32	RW	0000_0000h
53Ch	Clock Mux 8 Divider Update Status Register (MUX_8_DIV_UPD_STAT)	32	R	0000_0000h
540h	Clock Mux 9 Select Control Register (MUX_9_CSC)	32	RW	0000_0000h
544h	Clock Mux 9 Select Status Register (MUX_9_CSS)	32	R	0008_0000h
548h	Clock Mux 9 Divider 0 Control Register (MUX_9_DC_0)	32	RW	0000_0000h
57Ch	Clock Mux 9 Divider Update Status Register (MUX_9_DIV_UPD_STAT)	32	R	0000_0000h
580h	Clock Mux 10 Select Control Register (MUX_10_CSC)	32	RW	0000_0000h
584h	Clock Mux 10 Select Status Register (MUX_10_CSS)	32	R	0008_0000h
588h	Clock Mux 10 Divider 0 Control Register (MUX_10_DC_0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
5BCh	Clock Mux 10 Divider Update Status Register (MUX_10_DIV_UPD_STAT)	32	R	0000_0000h
5C0h	Clock Mux 11 Select Control Register (MUX_11_CSC)	32	RW	0000_0000h
5C4h	Clock Mux 11 Select Status Register (MUX_11_CSS)	32	R	0002_0000h
5C8h	Clock Mux 11 Divider 0 Control Register (MUX_11_DC_0)	32	RW	8000_0000h
5FCh	Clock Mux 11 Divider Update Status Register (MUX_11_DIV_UPD_STAT)	32	R	0000_0000h
640h	Clock Mux 13 Select Control Register (MUX_13_CSC)	32	RW	0000_0000h
644h	Clock Mux 13 Select Status Register (MUX_13_CSS)	32	R	0008_0000h
648h	Clock Mux 13 Divider 0 Control Register (MUX_13_DC_0)	32	RW	0000_0000h
67Ch	Clock Mux 13 Divider Update Status Register (MUX_13_DIV_UPD_STAT)	32	R	0000_0000h
6C0h	Clock Mux 15 Select Control Register (MUX_15_CSC)	32	RW	0000_0000h
6C4h	Clock Mux 15 Select Status Register (MUX_15_CSS)	32	R	0008_0000h
6C8h	Clock Mux 15 Divider 0 Control Register (MUX_15_DC_0)	32	RW	0000_0000h
6FCh	Clock Mux 15 Divider Update Status Register (MUX_15_DIV_UPD_STAT)	32	R	0000_0000h
700h	Clock Mux 16 Select Control Register (MUX_16_CSC)	32	RW	0000_0000h
704h	Clock Mux 16 Select Status Register (MUX_16_CSS)	32	R	0008_0000h
708h	Clock Mux 16 Divider 0 Control Register (MUX_16_DC_0)	32	RW	0000_0000h
73Ch	Clock Mux 16 Divider Update Status Register (MUX_16_DIV_UPD_STAT)	32	R	0000_0000h
780h	Clock Mux 18 Select Control Register (MUX_18_CSC)	32	RW	0000_0000h
784h	Clock Mux 18 Select Status Register (MUX_18_CSS)	32	R	0008_0000h
788h	Clock Mux 18 Divider 0 Control Register (MUX_18_DC_0)	32	RW	0001_0000h
7BCh	Clock Mux 18 Divider Update Status Register (MUX_18_DIV_UPD_STAT)	32	R	0000_0000h
7C0h	Clock Mux 19 Select Control Register (MUX_19_CSC)	32	RW	0000_0000h
7C4h	Clock Mux 19 Select Status Register (MUX_19_CSS)	32	R	0008_0000h
7C8h	Clock Mux 19 Divider 0 Control Register (MUX_19_DC_0)	32	RW	0000_0000h
7FCh	Clock Mux 19 Divider Update Status Register (MUX_19_DIV_UPD_STAT)	32	R	0000_0000h
800h	Clock Mux 20 Select Control Register (MUX_20_CSC)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
804h	Clock Mux 20 Select Status Register (MUX_20_CSS)	32	R	0008_0000h
808h	Clock Mux 20 Divider 0 Control Register (MUX_20_DC_0)	32	RW	0000_0000h
83Ch	Clock Mux 20 Divider Update Status Register (MUX_20_DIV_UPD_STAT)	32	R	0000_0000h

25.5.2 PCFS Step Duration (PCFS_SDUR)

Offset

Register	Offset
PCFS_SDUR	0h

Function

This register specifies the step duration of each PCFS step. The value provided in this register specifies the PCFS step duration in terms of the number of cycles of FIRC.

This register is reset only by a destructive reset. For details, see PCFS.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SDUR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	This field is reserved and reads return zeros.
15-0 SDUR	Step duration Count value of the step duration

25.5.3 PCFS Divider Change 8 Register (PCFS_DIVC8)

Offset

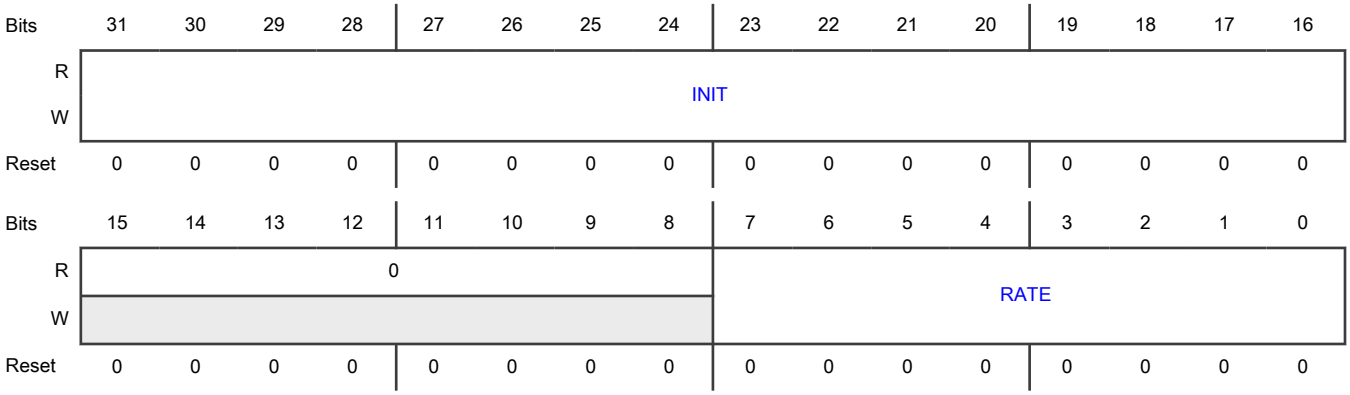
Register	Offset
PCFS_DIVC8	58h

Function

This register defines the rate of frequency change and initial change value on frequency ramp-up for the Progressive Clock Frequency switching of PLL_PHI0_CLK.

This register is reset only on destructive reset.

Diagram



Fields

Field	Function
31-16 INIT	Divider change initial value This field provides the initial change value of the clock divider for the clock ramp-up phase of PLL_PHI0_CLK.
15-8 —	This field is reserved and reads return zeros.
7-0 RATE	Divider change rate This value controls the change value of the clock divider for the clock ramp-up and ramp-down phase of PLL_PHI0_CLK.

25.5.4 PCFS Divider End 8 Register (PCFS_DIVE8)

Offset

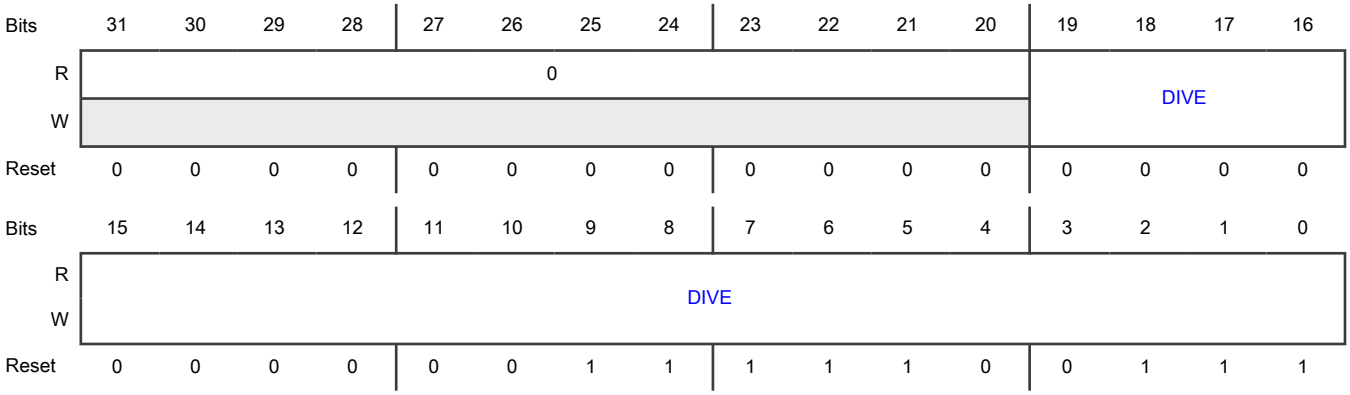
Register	Offset
PCFS_DIVE8	5Ch

Function

This register defines the final division value on frequency ramp-down for the progressive system clock switching of PLL_PHI0_CLK.

This registers is reset only on destructive reset.

Diagram



Fields

Field	Function
31-20 —	This field is reserved and reads return zeros.
19-0 DIVE	Divider end value This field provides the end value of the clock divider for the PLL_PHI0_CLK ramp-down phase.

25.5.5 PCFS Divider Start 8 Register (PCFS_DIVS8)

Offset

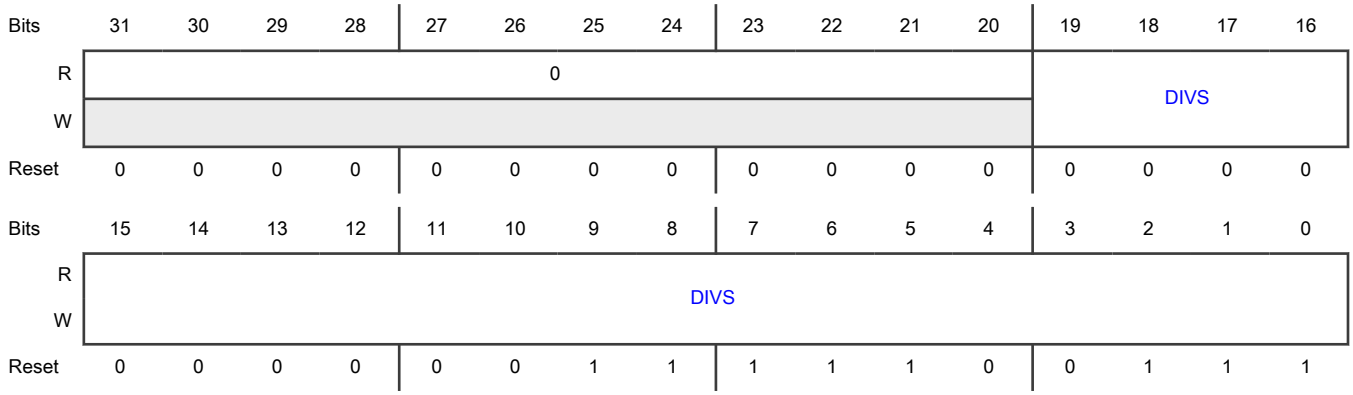
Register	Offset
PCFS_DIVS8	60h

Function

This register defines the initial division value on frequency ramp-up for the progressive system clock switching of PLL_PHI0_CLK.

This register is reset only on destructive reset.

Diagram



Fields

Field	Function
31-20 —	This field is reserved and reads return zeros.
19-0 DIVS	Divider start value This field provides the start value of the clock divider for the PLL_PHI0_CLK ramp-up phase

25.5.6 Clock Mux 0 Select Control Register (MUX_0_CSC)

Offset

Register	Offset
MUX_0_CSC	300h

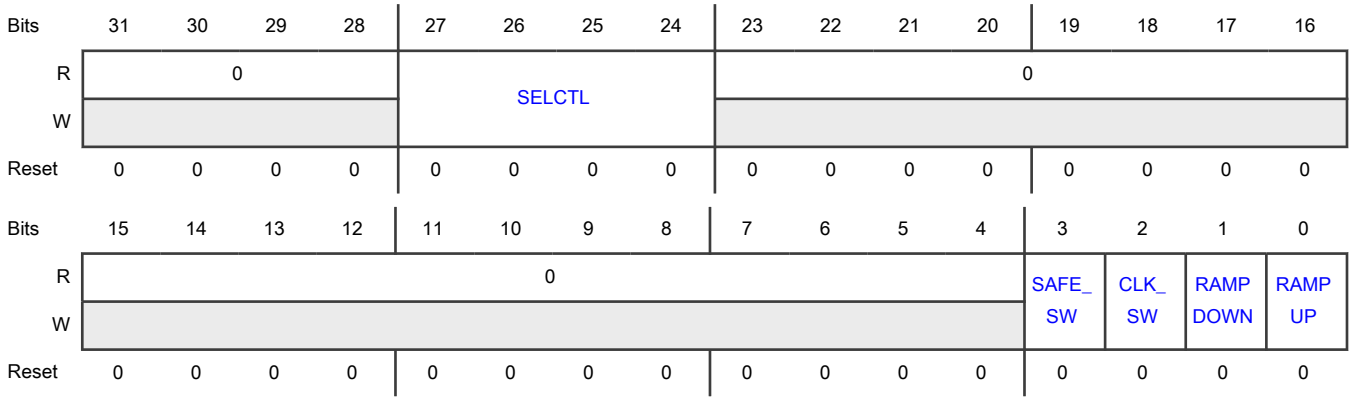
Function

This register provides the clock source selection control for clock mux 0. Clock mux 0 implements hardware control clock switching ensuring that the clock switch happens in a graceful manner (without glitches). See the "Hardware-controlled clock multiplexer" section for details.

This register is reset on destructive reset only.

An update to all the PCFS-related fields of this register must be an atomic write, which means a single write must update the CLK_SW, RAMPDOWN, and RAMPUP fields. It is necessary to set both RAMPUP and RAMPDOWN bits together even if you want to trigger either RAMPUP or RAMPDOWN process otherwise the desired PCFS sequence will not be executed.

Diagram



Fields

Field	Function
31-28 —	This field is reserved and reads return zeros.
27-24 SELCTL	Clock source selection control Selects the source clock for clock mux 0. The reserved values are not displayed. 0000b - FIRC 1000b - PLL_PHI0_CLK
23-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request Writing 1 to this bit makes a safe clock switch request to FIRC. After a safe clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
2 CLK_SW	Clock switch Writing 1 to this bit makes a clock switch request to clock mux 0. After a clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
1 RAMPDOWN	PCFS ramp-down Writing 1 to this bit makes a PCFS ramp-down request to clock mux 0. After a PCFS ramp-down operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
0 RAMPUP	PCFS ramp-up Writing 1 to this bit makes a PCFS ramp-up request to clock mux 0. After a PCFS ramp-up operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.

25.5.7 Clock Mux 0 Select Status Register (MUX_0_CSS)

Offset

Register	Offset
MUX_0_CSS	304h

Function

This register provides the current clock source selection status for clock mux 0.

This register is reset on destructive reset only.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				SELSTAT				0				SWTRG			SWIP
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SAFE_SW	CLK_SW	RAMP DOWN	RAMP UP
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	This field is reserved and reads return zeros.
27-24 SELSTAT	Clock source selection status This value indicates the current source selected for clock mux 0. The reserved values are not displayed. 0000b - FIRC 1000b - PLL_PHI0_CLK
23-20 —	This field is reserved and reads return zeros.
19-17 SWTRG	Switch trigger cause This value indicates the cause for the latest clock source switch.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE If the clock fails, followed by multiple safe clock switch requests for MC_CGM hardware clock mux, the value of the SWTRG field can be either 4 or 5. 000b - Reserved 001b - Switch after request succeeded. 010b - Switch after the request failed because of an inactive target clock and the current clock is FIRC. 011b - Switch after the request failed because of an inactive current clock and the current clock is FIRC. 100b - Switch to FIRC because of a safe clock request or reset succeeded. 101b - Switch to FIRC because of a safe clock request or reset succeeded, but the previous current clock source was inactive. 110b - Reserved 111b - Reserved
16 SWIP	Switch in progress **NOTE** New clock switch request can only be given three clock cycles after the completion of the previous request. 0b - Clock source switching is complete. 1b - Clock source switching is in progress.
15-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request This field provides an indication of whether a switch to safe clock operation was requested during the previous/ongoing request on clock mux 0. 0b - No safe clock switch operation was requested. 1b - Safe clock switch operation was requested.
2 CLK_SW	Clock switch This field provides an indication of whether a clock switch operation was requested during the previous/ongoing request on clock mux 0. 0b - No clock switch operation was requested. 1b - Clock switch operation was requested.
1 RAMPDOWN	PCFS ramp-down

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field provides an indication of whether a PCFS ramp-down operation was requested during the previous/ongoing request on clock mux 0. **NOTE** In case of safe clock switching, the ramp-down operation runs internally, but the value of the corresponding status field is not set to 1. 0b - No ramp-down operation was requested. 1b - Ramp-down operation was requested.
0 RAMPUP	PCFS ramp-up This field provides an indication of whether a PCFS ramp-up operation was requested during the previous/ ongoing request on clock mux 0. 0b - No ramp-up operation was requested. 1b - Ramp-up operation was requested.

25.5.8 Clock Mux 0 Divider 0 Control Register (MUX_0_DC_0)

Offset

Register	Offset
MUX_0_DC_0	308h

Function

This register controls the clock divider 0 for clock mux 0.

This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DE	0											DIV			
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 DE	Divider enable 0b - Unused 1b - Divider is enabled.
30-19 —	This field is reserved and reads return zeros.
18-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-0 —	This field is reserved and reads return zeros.

25.5.9 Clock Mux 0 Divider 1 Control Register (MUX_0_DC_1)

Offset

Register	Offset
MUX_0_DC_1	30Ch

Function

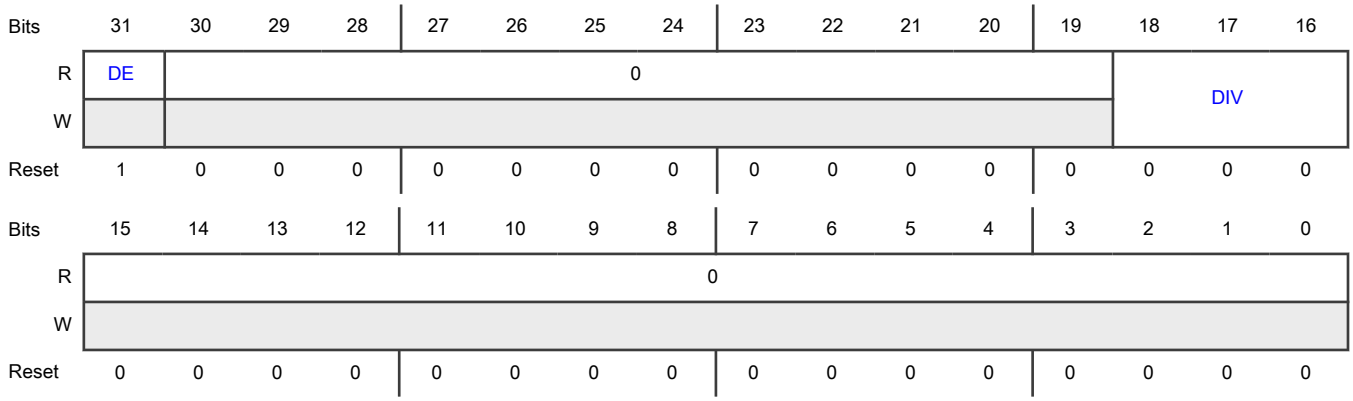
This register controls the clock divider 1 for clock mux 0.

This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram



Fields

Field	Function
31 DE	Divider enable 0b - Unused 1b - Divider is enabled.
30-19 —	This field is reserved and reads return zeros.
18-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-0 —	This field is reserved and reads return zeros.

25.5.10 Clock Mux 0 Divider 2 Control Register (MUX_0_DC_2)

Offset

Register	Offset
MUX_0_DC_2	310h

Function

This register controls the clock divider 2 for clock mux 0.

This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DE	0											DIV			
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 DE	Divider enable 0b - Unused 1b - Divider is enabled.
30-20 —	This field is reserved and reads return zeros.
19-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-0 —	This field is reserved and reads return zeros.

25.5.11 Clock Mux 0 Divider 3 Control Register (MUX_0_DC_3)

Offset

Register	Offset
MUX_0_DC_3	314h

Function

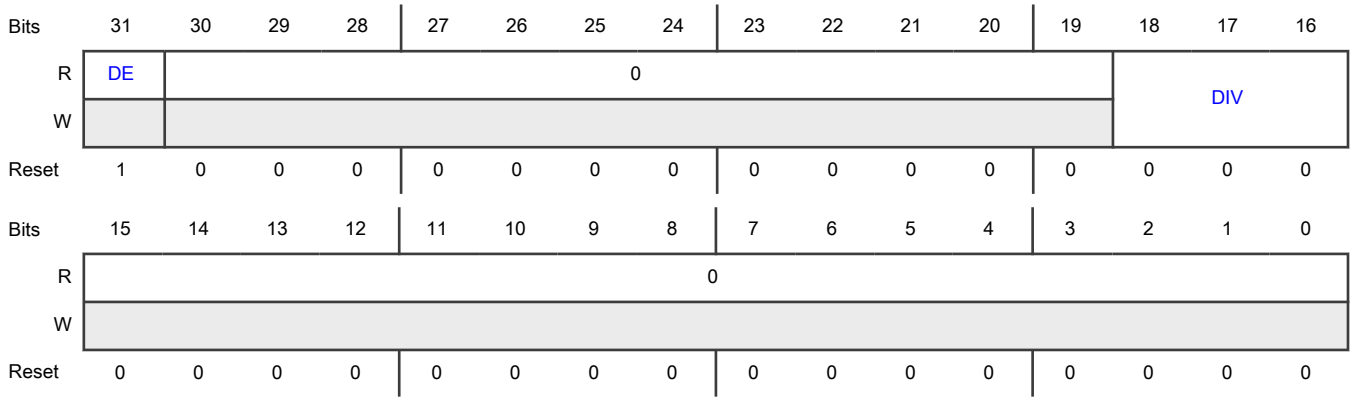
This register controls the clock divider 3 for clock mux 0.

This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram



Fields

Field	Function
31 DE	Divider enable 0b - Unused 1b - Divider is enabled.
30-19 —	This field is reserved and reads return zeros.
18-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-0 —	This field is reserved and reads return zeros.

25.5.12 Clock Mux 0 Divider 4 Control Register (MUX_0_DC_4)

Offset

Register	Offset
MUX_0_DC_4	318h

Function

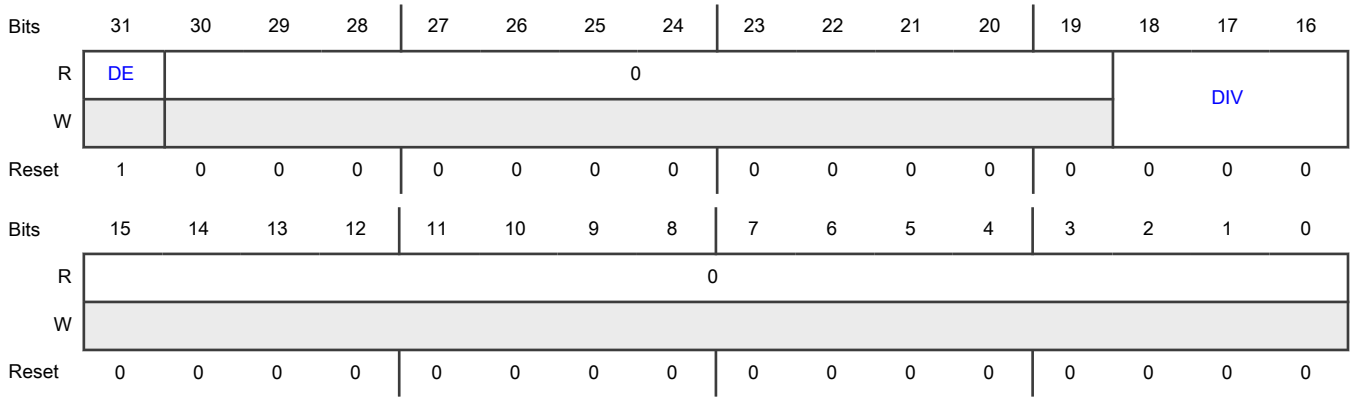
This register controls the clock divider 4 for clock mux 0.

This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram



Fields

Field	Function
31 DE	Divider enable 0b - Unused 1b - Divider is enabled.
30-19 —	This field is reserved and reads return zeros.
18-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-0 —	This field is reserved and reads return zeros.

25.5.13 Clock Mux 0 Divider 5 Control Register (MUX_0_DC_5)

Offset

Register	Offset
MUX_0_DC_5	31Ch

Function

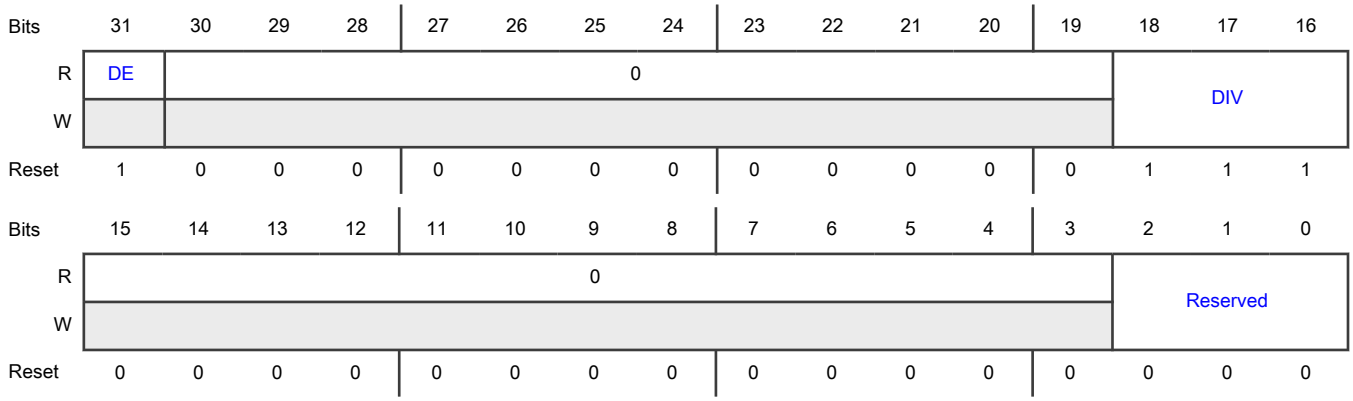
This register controls the clock divider 5 for clock mux 0.

This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram



Fields

Field	Function
31 DE	Divider enable 0b - Unused 1b - Divider is enabled.
30-19 —	This field is reserved and reads return zeros.
18-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-3 —	Reserved
2-0 —	Reserved

25.5.14 Clock Mux 0 Divider 6 Control Register (MUX_0_DC_6)

Offset

Register	Offset
MUX_0_DC_6	320h

Function

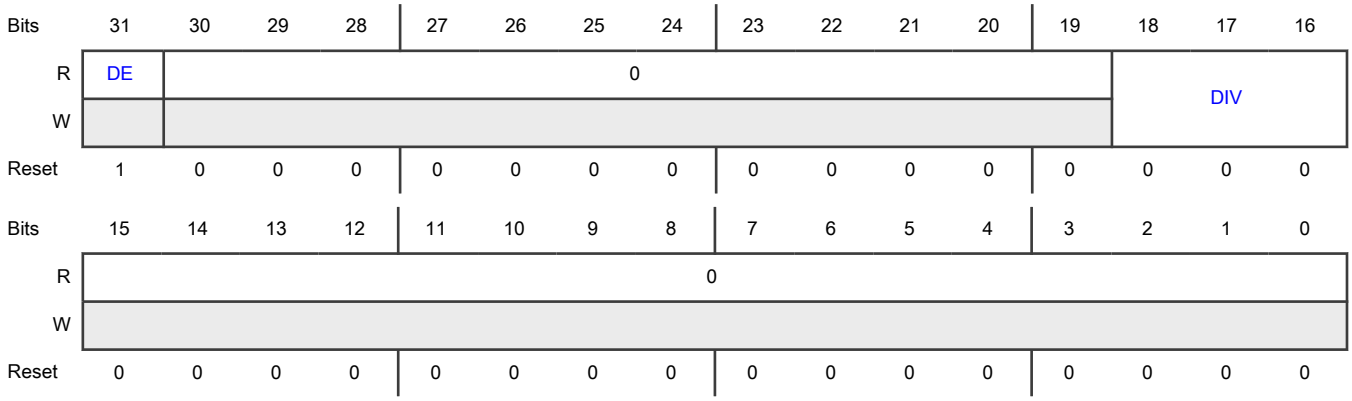
This register controls the clock divider 6 for clock mux 0.

This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram



Fields

Field	Function
31 DE	Divider enable 0b - Unused 1b - Divider is enabled.
30-19 —	This field is reserved and reads return zeros.
18-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-0 —	This field is reserved and reads return zeros.

25.5.15 Clock Mux 0 Divider 7 Control Register (MUX_0_DC_7)

Offset

Register	Offset
MUX_0_DC_7	324h

Function

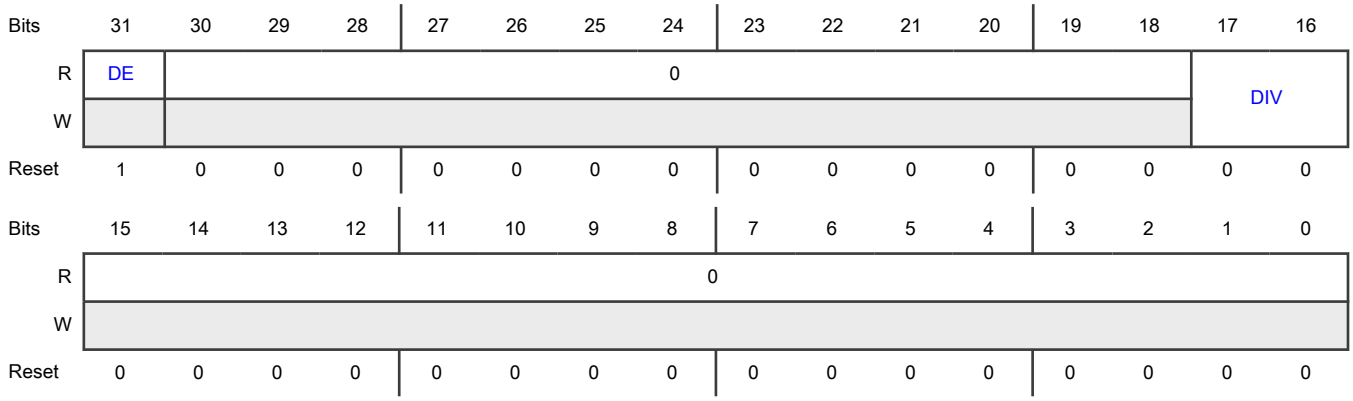
This register controls the clock divider 7 for clock mux 0.

This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram



Fields

Field	Function
31 DE	Divider enable 0b - Unused 1b - Divider is enabled.
30-18 —	This field is reserved and reads return zeros.
17-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-0 —	This field is reserved and reads return zeros.

25.5.16 Clock Mux 0 Divider Trigger Control Register (MUX_0_DIV_TRIG_CTRL)

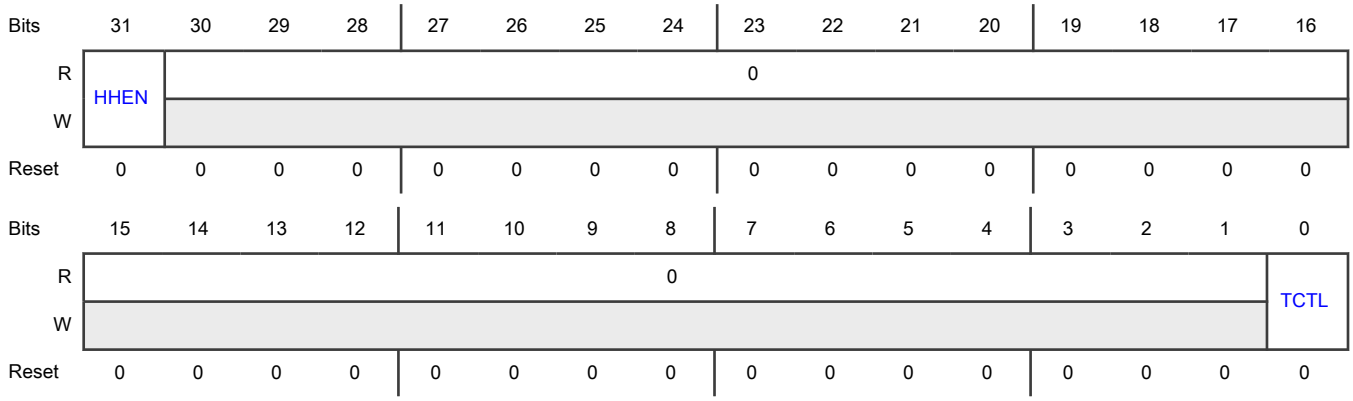
Offset

Register	Offset
MUX_0_DIV_TRIG_CTRL	334h

Function

This register selects whether the dividers associated with clock mux 0 are updated immediately on writing to the corresponding divider configuration register (referred to as immediate divider update) or only on writing to the MC_CGM_MUX_0_DIV_TRIG register (referred to as common trigger update). When common trigger update is configured, this register also controls initiation of the halt handshake protocol with the on-chip AXBS. Software is required to configure HHEN field for handshaking with on-chip AXBS when the ratio of division value among the clock dividers need to be changed.

Diagram



Fields

Field	Function
31 HHEN	Halt handshake enable This field controls the initiation of the halt handshake protocol with AXBS when a common trigger divider update is initiated. 0b - No halt handshake protocol is initiated. 1b - Halt handshake protocol is initiated.
30-1 —	This field is reserved and reads return zeros.
0 TCTL	Trigger control This field controls the divider update configuration between immediate and common update. 0b - Immediate divider update 1b - Common trigger divider update

25.5.17 Clock Mux 0 Divider Trigger Register (MUX_0_DIV_TRIG)

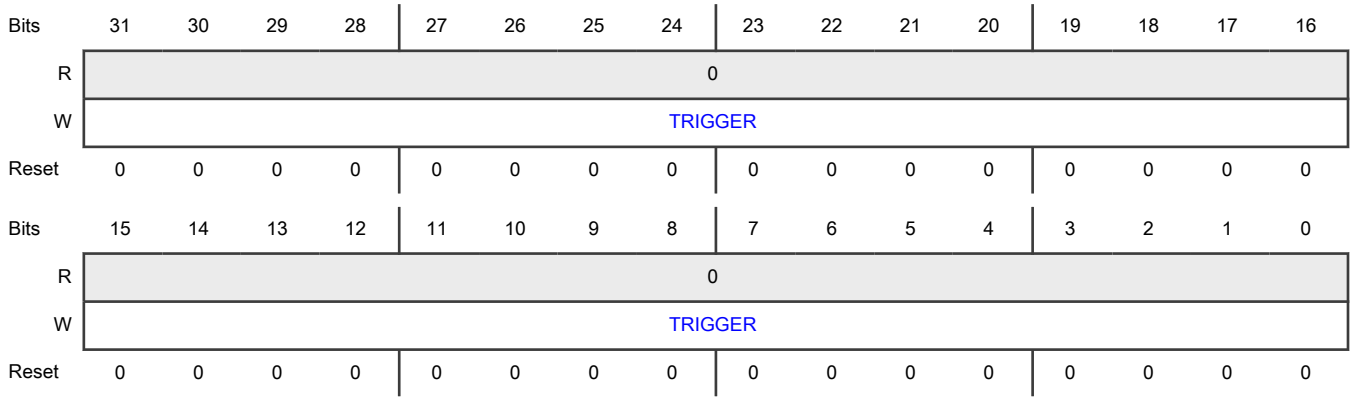
Offset

Register	Offset
MUX_0_DIV_TRIG	338h

Function

This register provides a common trigger for the clock dividers (only 50% duty cycle dividers) of clock mux 0. Writing any value to this register provides a trigger to the dividers. This register should only be written after appropriately configuring the MC_CGM_MUX_0_DIV_TRIG_CTRL register.

Diagram



Fields

Field	Function
31-0 TRIGGER	Trigger for divider update

25.5.18 Clock Mux 0 Divider Update Status Register (MUX_0_DIV_UPD_STAT)

Offset

Register	Offset
MUX_0_DIV_UPD_STAT	33Ch

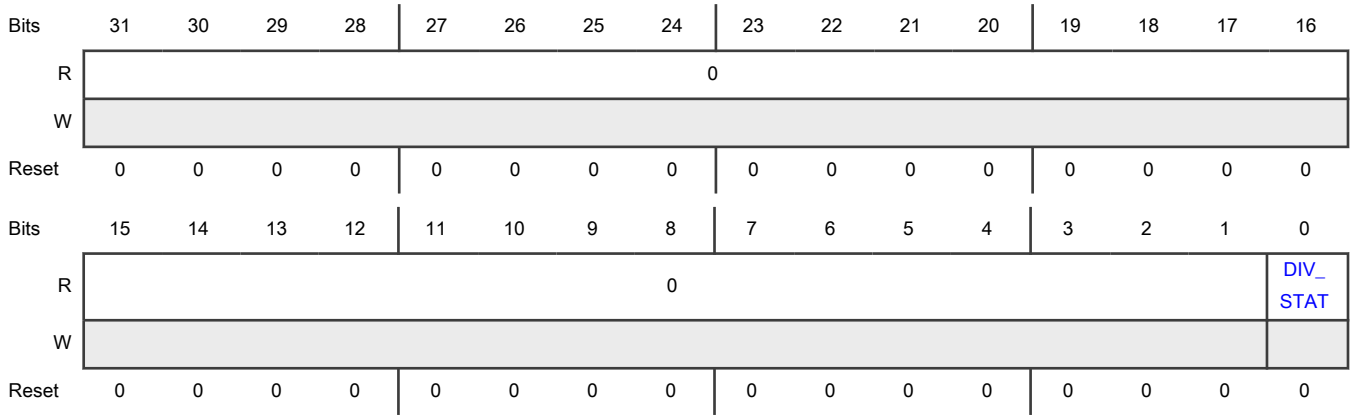
Function

This register provides the update status of the clock dividers corresponding to clock mux 0. When a write operation on any divider control register is performed, the divider status bit in this register is set to logic-1. The bit is set to logic-0 when the divider has sampled the new divider configuration. Performing multiple writes without tracking the status bit on same or other clock dividers inside the same clock mux leads to inconsistent reporting, that is, the divider status maybe be set to logic-0 when the corresponding divider update is pending.

NOTE

Read accesses to MUX_n_DIV_UPD_STAT always complete without returning bus transfer error independent of whether any divider(s) are implemented inside MC_CGM clock mux.

Diagram



Fields

Field	Function
31-1 —	This field is reserved and reads return zeros.
0 DIV_STAT	Divider status for clock mux 0 On reading MUX_n_DIV_UPD_STAT after updating a divider control register, if the value of this field is fixed to 1 because of an error in the selected clock source, perform the following steps to switch the mux to a new clock source: 1. Switch the mux to a working clock source without polling this field. 2. Update MUX_n_DC_m and poll this field. **NOTE** This field clears once divider configuration is updated or on destructive reset. If functional reset comes when this field is 1 then it can remain fixed to 1 until divider input clock is restored. 0b - No divider configuration update is pending. 1b - Divider configuration update on at least one divider associated with this multiplexer is pending.

25.5.19 Clock Mux 1 Select Control Register (MUX_1_CSC)

Offset

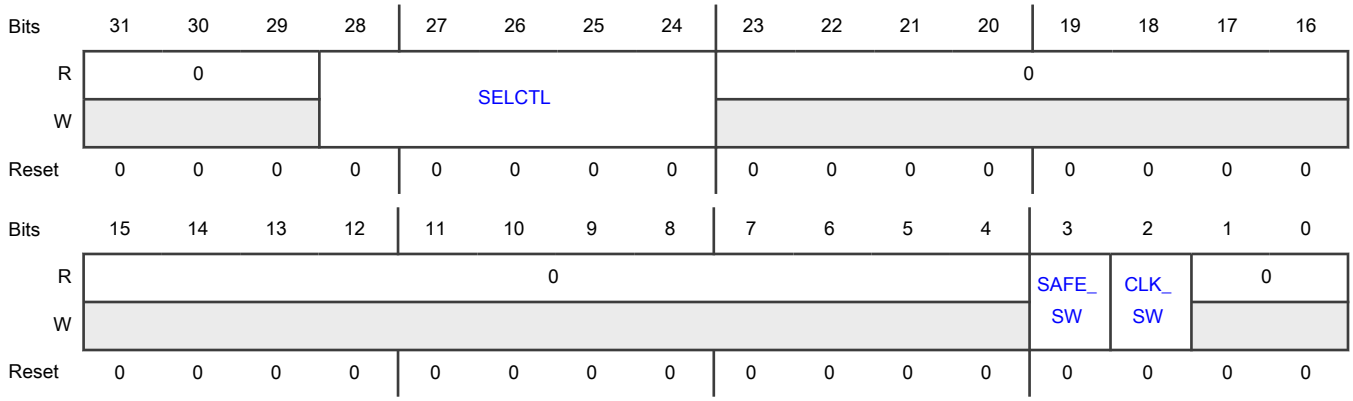
Register	Offset
MUX_1_CSC	340h

Function

This register provides the clock source selection control for clock mux 1. Clock mux 1 implements hardware control clock switching ensuring that the clock switch happens in a graceful manner (without glitches). See the "Hardware-controlled clock multiplexer" section for details.

This register is reset on destructive reset only.

Diagram



Fields

Field	Function
31-29 —	This field is reserved and reads return zeros.
28-24 SELCTL	Clock source selection control Selects the source clock for clock mux 1. The reserved values are not displayed. 0_0000b - FIRC 0_0010b - FXOSC 1_0110b - AIPS_PLAT_CLK
23-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request Writing 1 to this bit makes a safe clock switch request to FIRC. After a safe clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
2 CLK_SW	Clock switch Writing 1 to this bit makes a clock switch request to clock mux 1. After a clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
1-0 —	This field is reserved and reads return zeros.

25.5.20 Clock Mux 1 Select Status Register (MUX_1_CSS)

Offset

Register	Offset
MUX_1_CSS	344h

Function

This register provides the current clock source selection status for clock mux 1.

This register is reset on destructive reset only.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				SELSTAT				0				SWTRG			SWIP
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SAFE_SW	CLK_SW	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	This field is reserved and reads return zeros.
28-24 SELSTAT	Clock source selection status This value indicates the current source selected for clock mux 1. The reserved values are not displayed. 0_0000b - FIRC 0_0010b - FXOSC 1_0110b - AIPS_PLAT_CLK
23-20 —	This field is reserved and reads return zeros.
19-17 SWTRG	Switch trigger cause This value indicates the cause for the latest clock source switch.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE If the clock fails, followed by multiple safe clock switch requests for MC_CGM hardware clock mux, the value of the SWTRG field can be either 4 or 5. 000b - Reserved 001b - Switch after request succeeded. 010b - Switch after the request failed because of an inactive target clock and the current clock is FIRC. 011b - Switch after the request failed because of an inactive current clock and the current clock is FIRC. 100b - Switch to FIRC because of a safe clock request or reset succeeded. 101b - Switch to FIRC because of a safe clock request or reset succeeded, but the previous current clock source was inactive. 110b - Reserved 111b - Reserved
16 SWIP	Switch in progress **NOTE** New clock switch request can only be given three clock cycles after the completion of the previous request. 0b - Clock source switching is complete. 1b - Clock source switching is in progress.
15-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request This field provides an indication of whether a switch to safe clock operation was requested during the previous/ongoing request on clock mux 1. 0b - No safe clock switch operation was requested. 1b - Safe clock switch operation was requested.
2 CLK_SW	Clock switch This field provides an indication of whether a clock switch operation was requested during the previous/ongoing request on clock mux 1. 0b - No clock switch operation was requested. 1b - Clock switch operation was requested.
1-0 —	This field is reserved and reads return zeros.

25.5.21 Clock Mux 1 Divider 0 Control Register (MUX_1_DC_0)

Offset

Register	Offset
MUX_1_DC_0	348h

Function

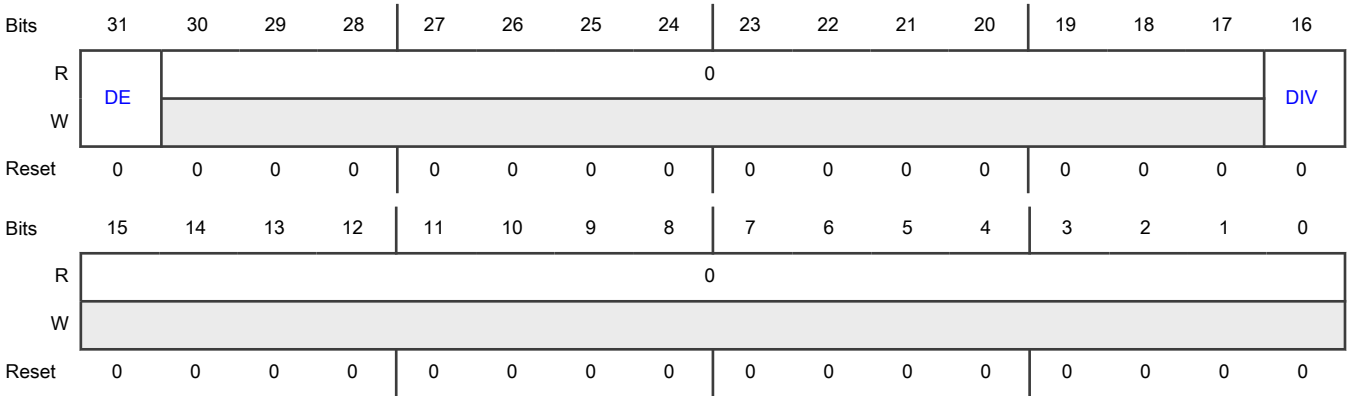
This register controls the clock divider 0 for clock mux 1.

This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram



Fields

Field	Function
31 DE	Divider enable 0b - Divider is disabled. 1b - Divider is enabled.
30-17 —	This field is reserved and reads return zeros.
16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-0 —	This field is reserved and reads return zeros.

25.5.22 Clock Mux 1 Divider Update Status Register (MUX_1_DIV_UPD_STAT)

Offset

Register	Offset
MUX_1_DIV_UPD_STAT	37Ch

Function

This register provides the update status of the clock dividers corresponding to clock mux 1. When a write operation on any divider control register is performed, the divider status bit in this register is set to logic-1. The bit is set to logic-0 when the divider has sampled the new divider configuration. Performing multiple writes without tracking the status bit on same or other clock dividers inside the same clock mux leads to inconsistent reporting, that is, the divider status maybe be set to logic-0 when the corresponding divider update is pending.

NOTE

Read accesses to MUX_n_DIV_UPD_STAT always complete without returning bus transfer error independent of whether any divider(s) are implemented inside MC_CGM clock mux.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DIV_STAT
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-1 —	This field is reserved and reads return zeros.
0 DIV_STAT	Divider status for clock mux 1 On reading MUX_n_DIV_UPD_STAT after updating a divider control register, if the value of this field is fixed to 1 because of an error in the selected clock source, perform the following steps to switch the mux to a new clock source: 1. Switch the mux to a working clock source without polling this field. 2. Update MUX_n_DC_m and poll this field.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field clears once divider configuration is updated or on destructive reset. If functional reset comes when this field is 1 then it can remain fixed to 1 until divider input clock is restored. 0b - No divider configuration update is pending. 1b - Divider configuration update on at least one divider associated with this multiplexer is pending.

25.5.23 Clock Mux 2 Select Control Register (MUX_2_CSC)

Offset

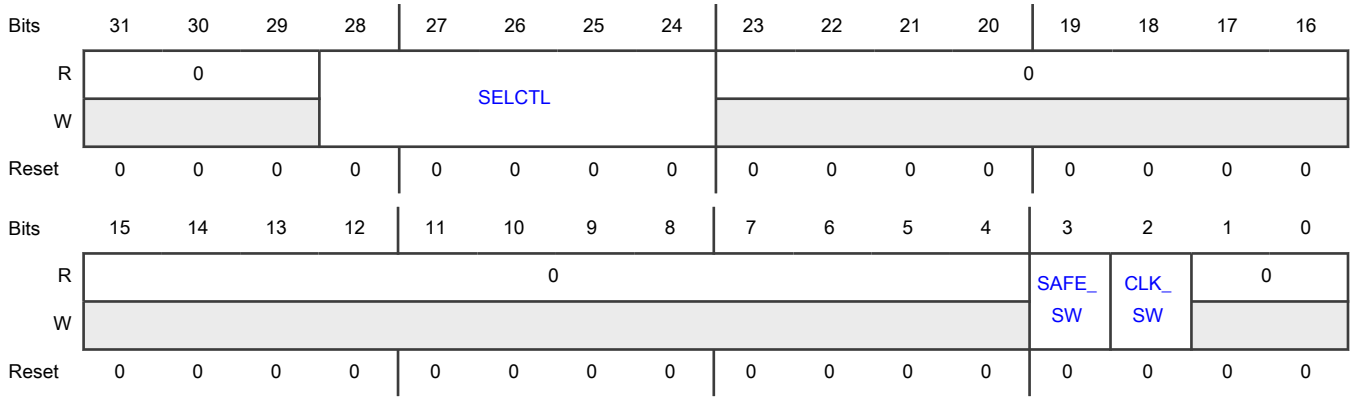
Register	Offset
MUX_2_CSC	380h

Function

This register provides the clock source selection control for clock mux 2. Clock mux 2 implements hardware control clock switching ensuring that the clock switch happens in a graceful manner (without glitches). See the "Hardware-controlled clock multiplexer" section for details.

This register is reset on destructive reset only.

Diagram



Fields

Field	Function
31-29	This field is reserved and reads return zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
28-24 SELCTL	Clock source selection control Selects the source clock for clock mux 2. The reserved values are not displayed. 0_0000b - FIRC 0_0010b - FXOSC 1_0110b - AIPS_PLAT_CLK
23-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request Writing 1 to this bit makes a safe clock switch request to FIRC. After a safe clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
2 CLK_SW	Clock switch Writing 1 to this bit makes a clock switch request to clock mux 2. After a clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
1-0 —	This field is reserved and reads return zeros.

25.5.24 Clock Mux 2 Select Status Register (MUX_2_CSS)

Offset

Register	Offset
MUX_2_CSS	384h

Function

This register provides the current clock source selection status for clock mux 2.

This register is reset on destructive reset only.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				SELSTAT				0				SWTRG			SWIP
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SAFE_SW	CLK_SW	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	This field is reserved and reads return zeros.
28-24 SELSTAT	Clock source selection status This value indicates the current source selected for clock mux 2. The reserved values are not displayed. 0_0000b - FIRC 0_0010b - FXOSC 1_0110b - AIPS_PLAT_CLK
23-20 —	This field is reserved and reads return zeros.
19-17 SWTRG	Switch trigger cause This value indicates the cause for the latest clock source switch. **NOTE** If the clock fails, followed by multiple safe clock switch requests for MC_CGM hardware clock mux, the value of the SWTRG field can be either 4 or 5. 000b - Reserved 001b - Switch after request succeeded. 010b - Switch after the request failed because of an inactive target clock and the current clock is FIRC. 011b - Switch after the request failed because of an inactive current clock and the current clock is FIRC. 100b - Switch to FIRC because of a safe clock request or reset succeeded. 101b - Switch to FIRC because of a safe clock request or reset succeeded, but the previous current clock source was inactive.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	110b - Reserved 111b - Reserved
16 SWIP	Switch in progress **NOTE** New clock switch request can only be given three clock cycles after the completion of the previous request. 0b - Clock source switching is complete. 1b - Clock source switching is in progress.
15-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request This field provides an indication of whether a switch to safe clock operation was requested during the previous/ongoing request on clock mux 2. 0b - No safe clock switch operation was requested. 1b - Safe clock switch operation was requested.
2 CLK_SW	Clock switch This field provides an indication of whether a clock switch operation was requested during the previous/ongoing request on clock mux 2. 0b - No clock switch operation was requested. 1b - Clock switch operation was requested.
1-0 —	This field is reserved and reads return zeros.

25.5.25 Clock Mux 2 Divider 0 Control Register (MUX_2_DC_0)

Offset

Register	Offset
MUX_2_DC_0	388h

Function

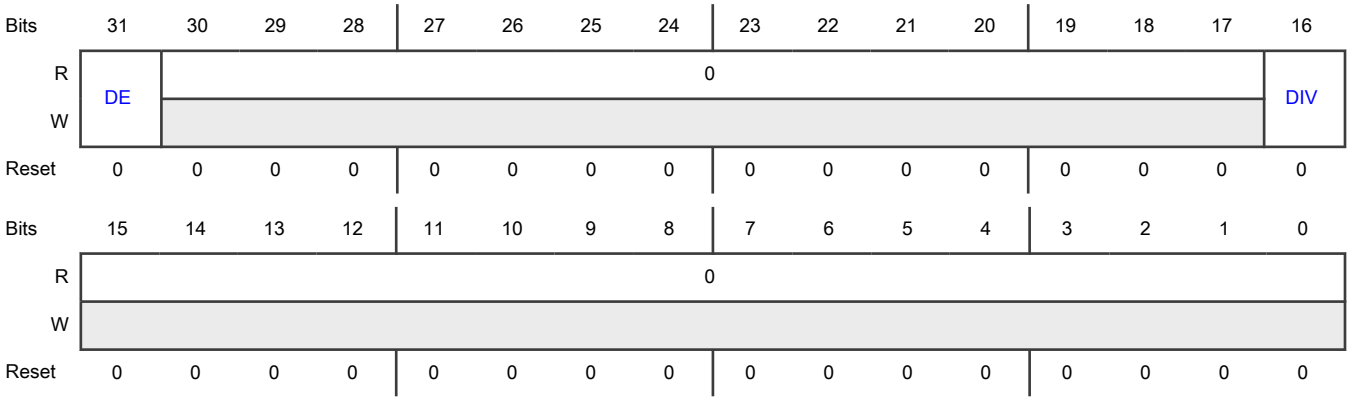
This register controls the clock divider 0 for clock mux 2.

This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram



Fields

Field	Function
31 DE	Divider enable 0b - Divider is disabled. 1b - Divider is enabled.
30-17 —	This field is reserved and reads return zeros.
16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-0 —	This field is reserved and reads return zeros.

25.5.26 Clock Mux 2 Divider Update Status Register (MUX_2_DIV_UPD_STAT)

Offset

Register	Offset
MUX_2_DIV_UPD_STAT	3BCh

Function

This register provides the update status of the clock dividers corresponding to clock mux 2. When a write operation on any divider control register is performed, the divider status bit in this register is set to logic-1. The bit is set to logic-0 when the divider has sampled the new divider configuration. Performing multiple writes without tracking the status bit on same or other

clock dividers inside the same clock mux leads to inconsistent reporting, that is, the divider status maybe be set to logic-0 when the corresponding divider update is pending.

NOTE

Read accesses to MUX_n_DIV_UPD_STAT always complete without returning bus transfer error independent of whether any divider(s) are implemented inside MC_CGM clock mux.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DIV_STAT
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-1 —	This field is reserved and reads return zeros.
0 DIV_STAT	Divider status for clock mux 2 On reading MUX_n_DIV_UPD_STAT after updating a divider control register, if the value of this field is fixed to 1 because of an error in the selected clock source, perform the following steps to switch the mux to a new clock source: 1. Switch the mux to a working clock source without polling this field. 2. Update MUX_n_DC_m and poll this field. **NOTE** This field clears once divider configuration is updated or on destructive reset. If functional reset comes when this field is 1 then it can remain fixed to 1 until divider input clock is restored. 0b - No divider configuration update is pending. 1b - Divider configuration update on at least one divider associated with this multiplexer is pending.

25.5.27 Clock Mux 3 Select Control Register (MUX_3_CSC)

Offset

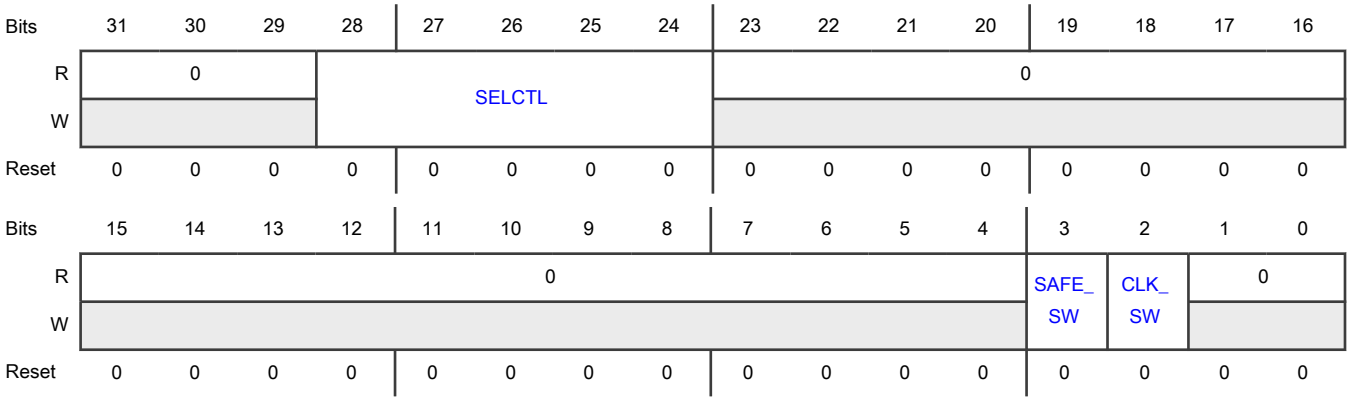
Register	Offset
MUX_3_CSC	3C0h

Function

This register provides the clock source selection control for clock mux 3. Clock mux 3 implements hardware control clock switching ensuring that the clock switch happens in a graceful manner (without glitches). See the "Hardware-controlled clock multiplexer" section for details.

This register is reset on destructive reset only.

Diagram



Fields

Field	Function
31-29 —	This field is reserved and reads return zeros.
28-24 SELCTL	Clock source selection control Selects the source clock for clock mux 3. The reserved values are not displayed. 0_0000b - FIRC 0_0010b - FXOSC 1_0110b - AIPS_PLAT_CLK
23-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Writing 1 to this bit makes a safe clock switch request to FIRC. After a safe clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
2 CLK_SW	Clock switch Writing 1 to this bit makes a clock switch request to clock mux 3. After a clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
1-0 —	This field is reserved and reads return zeros.

25.5.28 Clock Mux 3 Select Status Register (MUX_3_CSS)

Offset

Register	Offset
MUX_3_CSS	3C4h

Function

This register provides the current clock source selection status for clock mux 3.

This register is reset on destructive reset only.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				SELSTAT				0				SWTRG			SWIP
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SAFE_SW	CLK_SW	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	This field is reserved and reads return zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
28-24 SELSTAT	Clock source selection status This value indicates the current source selected for clock mux 3. The reserved values are not displayed. 0_0000b - FIRC 0_0010b - FXOSC 1_0110b - AIPS_PLAT_CLK
23-20 —	This field is reserved and reads return zeros.
19-17 SWTRG	Switch trigger cause This value indicates the cause for the latest clock source switch. **NOTE** If the clock fails, followed by multiple safe clock switch requests for MC_CGM hardware clock mux, the value of the SWTRG field can be either 4 or 5. 000b - Reserved 001b - Switch after request succeeded. 010b - Switch after the request failed because of an inactive target clock and the current clock is FIRC. 011b - Switch after the request failed because of an inactive current clock and the current clock is FIRC. 100b - Switch to FIRC because of a safe clock request or reset succeeded. 101b - Switch to FIRC because of a safe clock request or reset succeeded, but the previous current clock source was inactive. 110b - Reserved 111b - Reserved
16 SWIP	Switch in progress **NOTE** New clock switch request can only be given three clock cycles after the completion of the previous request. 0b - Clock source switching is complete. 1b - Clock source switching is in progress.
15-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field provides an indication of whether a switch to safe clock operation was requested during the previous/ongoing request on clock mux 3. 0b - No safe clock switch operation was requested. 1b - Safe clock switch operation was requested.
2 CLK_SW	Clock switch This field provides an indication of whether a clock switch operation was requested during the previous/ ongoing request on clock mux 3. 0b - No clock switch operation was requested. 1b - Clock switch operation was requested.
1-0 —	This field is reserved and reads return zeros.

25.5.29 Clock Mux 3 Divider 0 Control Register (MUX_3_DC_0)

Offset

Register	Offset
MUX_3_DC_0	3C8h

Function

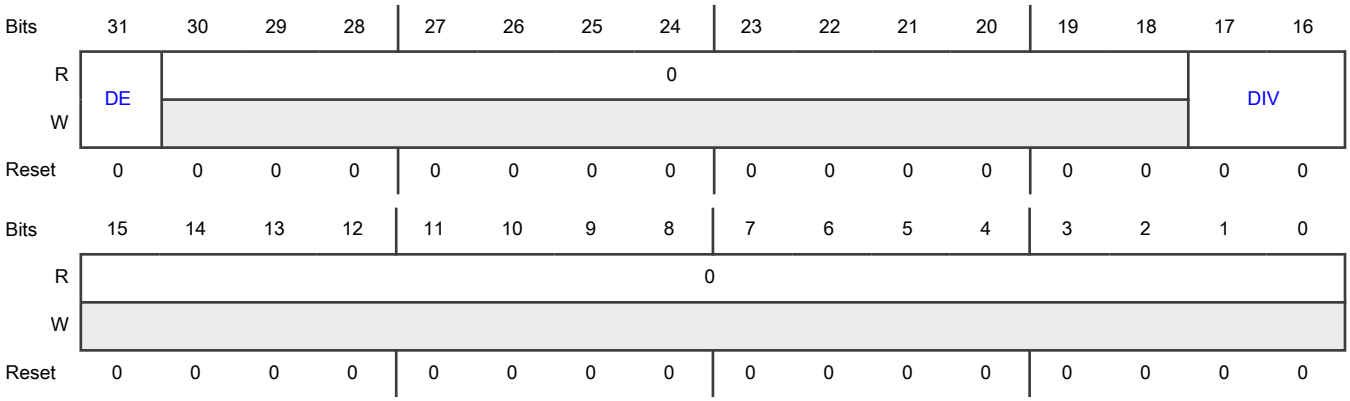
This register controls the clock divider 0 for clock mux 3.

This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram



Fields

Field	Function
31 DE	Divider enable 0b - Divider is disabled. 1b - Divider is enabled.
30-18 —	This field is reserved and reads return zeros.
17-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-0 —	This field is reserved and reads return zeros.

25.5.30 Clock Mux 3 Divider Update Status Register (MUX_3_DIV_UPD_STAT)

Offset

Register	Offset
MUX_3_DIV_UPD_STAT	3FCh

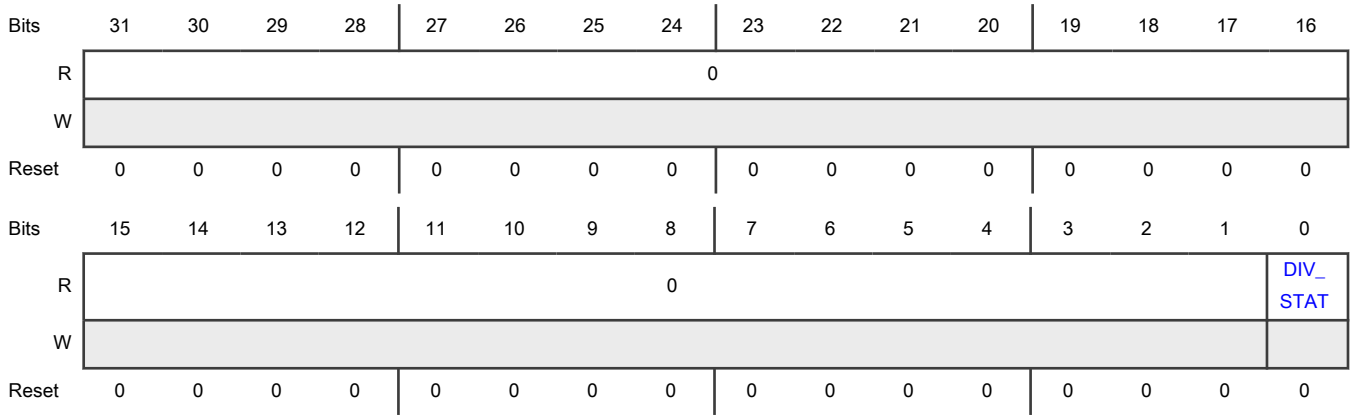
Function

This register provides the update status of the clock dividers corresponding to clock mux 3. When a write operation on any divider control register is performed, the divider status bit in this register is set to logic-1. The bit is set to logic-0 when the divider has sampled the new divider configuration. Performing multiple writes without tracking the status bit on same or other clock dividers inside the same clock mux leads to inconsistent reporting, that is, the divider status maybe be set to logic-0 when the corresponding divider update is pending.

NOTE

Read accesses to MUX_n_DIV_UPD_STAT always complete without returning bus transfer error independent of whether any divider(s) are implemented inside MC_CGM clock mux.

Diagram



Fields

Field	Function
31-1 —	This field is reserved and reads return zeros.
0 DIV_STAT	Divider status for clock mux 3 On reading MUX_n_DIV_UPD_STAT after updating a divider control register, if the value of this field is fixed to 1 because of an error in the selected clock source, perform the following steps to switch the mux to a new clock source: 1. Switch the mux to a working clock source without polling this field. 2. Update MUX_n_DC_m and poll this field. **NOTE** This field clears once divider configuration is updated or on destructive reset. If functional reset comes when this field is 1 then it can remain fixed to 1 until divider input clock is restored. 0b - No divider configuration update is pending. 1b - Divider configuration update on at least one divider associated with this multiplexer is pending.

25.5.31 Clock Mux 4 Select Control Register (MUX_4_CSC)

Offset

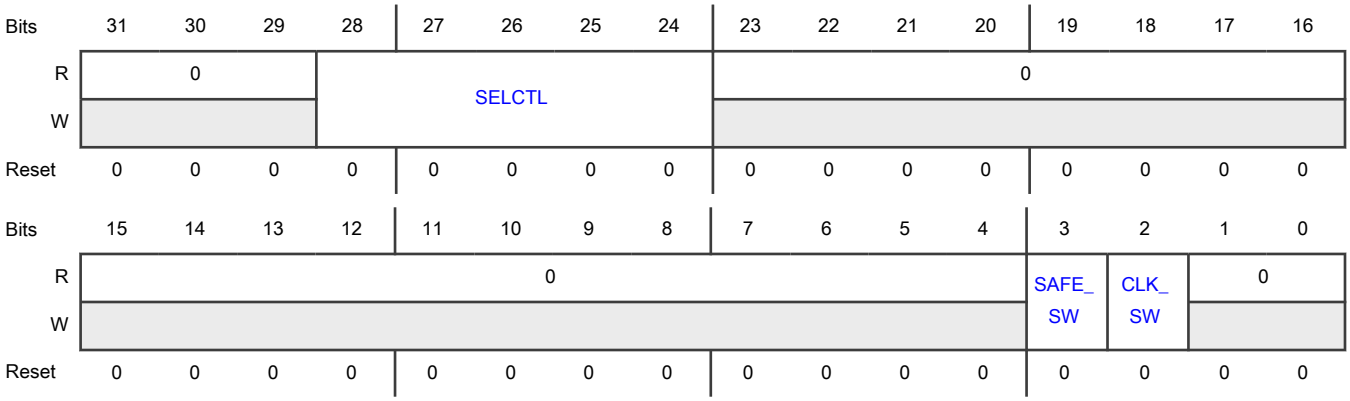
Register	Offset
MUX_4_CSC	400h

Function

This register provides the clock source selection control for clock mux 4. Clock mux 4 implements hardware control clock switching ensuring that the clock switch happens in a graceful manner (without glitches). See the "Hardware-controlled clock multiplexer" section for details.

This register is reset on destructive reset only.

Diagram



Fields

Field	Function
31-29 —	This field is reserved and reads return zeros.
28-24 SELCTL	Clock source selection control Selects the source clock for clock mux 4. The reserved values are not displayed. 0_0000b - FIRC 0_0010b - FXOSC 1_0110b - AIPS_PLAT_CLK
23-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request Writing 1 to this bit makes a safe clock switch request to FIRC. After a safe clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
2 CLK_SW	Clock switch Writing 1 to this bit makes a clock switch request to clock mux 4. After a clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
1-0 —	This field is reserved and reads return zeros.

25.5.32 Clock Mux 4 Select Status Register (MUX_4_CSS)

Offset

Register	Offset
MUX_4_CSS	404h

Function

This register provides the current clock source selection status for clock mux 4.

This register is reset on destructive reset only.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				SELSTAT				0				SWTRG			SWIP
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SAFE_SW	CLK_SW	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	This field is reserved and reads return zeros.
28-24 SELSTAT	Clock source selection status This value indicates the current source selected for clock mux 4. The reserved values are not displayed. 0_0000b - FIRC 0_0010b - FXOSC 1_0110b - AIPS_PLAT_CLK
23-20 —	This field is reserved and reads return zeros.
19-17 SWTRG	Switch trigger cause This value indicates the cause for the latest clock source switch.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE If the clock fails, followed by multiple safe clock switch requests for MC_CGM hardware clock mux, the value of the SWTRG field can be either 4 or 5. 000b - Reserved 001b - Switch after request succeeded. 010b - Switch after the request failed because of an inactive target clock and the current clock is FIRC. 011b - Switch after the request failed because of an inactive current clock and the current clock is FIRC. 100b - Switch to FIRC because of a safe clock request or reset succeeded. 101b - Switch to FIRC because of a safe clock request or reset succeeded, but the previous current clock source was inactive. 110b - Reserved 111b - Reserved
16 SWIP	Switch in progress **NOTE** New clock switch request can only be given three clock cycles after the completion of the previous request. 0b - Clock source switching is complete. 1b - Clock source switching is in progress.
15-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request This field provides an indication of whether a switch to safe clock operation was requested during the previous/ongoing request on clock mux 4. 0b - No safe clock switch operation was requested. 1b - Safe clock switch operation was requested.
2 CLK_SW	Clock switch This field provides an indication of whether a clock switch operation was requested during the previous/ongoing request on clock mux 4. 0b - No clock switch operation was requested. 1b - Clock switch operation was requested.
1-0 —	This field is reserved and reads return zeros.

25.5.33 Clock Mux 4 Divider 0 Control Register (MUX_4_DC_0)

Offset

Register	Offset
MUX_4_DC_0	408h

Function

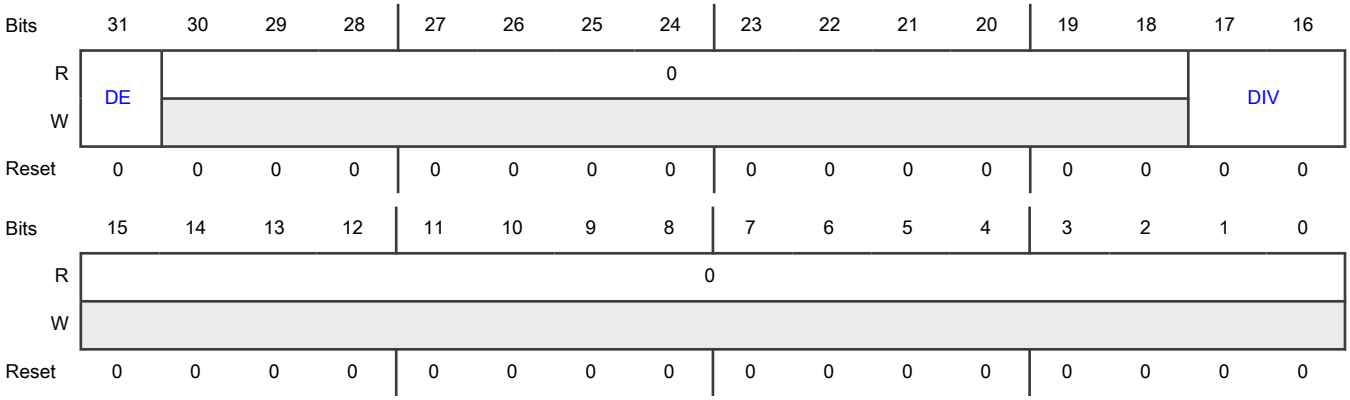
This register controls the clock divider 0 for clock mux 4.

This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram



Fields

Field	Function
31 DE	Divider enable 0b - Divider is disabled. 1b - Divider is enabled.
30-18 —	This field is reserved and reads return zeros.
17-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-0 —	This field is reserved and reads return zeros.

25.5.34 Clock Mux 4 Divider Update Status Register (MUX_4_DIV_UPD_STAT)

Offset

Register	Offset
MUX_4_DIV_UPD_STAT	43Ch

Function

This register provides the update status of the clock dividers corresponding to clock mux 4. When a write operation on any divider control register is performed, the divider status bit in this register is set to logic-1. The bit is set to logic-0 when the divider has sampled the new divider configuration. Performing multiple writes without tracking the status bit on same or other clock dividers inside the same clock mux leads to inconsistent reporting, that is, the divider status maybe be set to logic-0 when the corresponding divider update is pending.

NOTE

Read accesses to MUX_n_DIV_UPD_STAT always complete without returning bus transfer error independent of whether any divider(s) are implemented inside MC_CGM clock mux.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DIV_STAT
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-1 —	This field is reserved and reads return zeros.
0 DIV_STAT	Divider status for clock mux 4 On reading MUX_n_DIV_UPD_STAT after updating a divider control register, if the value of this field is fixed to 1 because of an error in the selected clock source, perform the following steps to switch the mux to a new clock source: 1. Switch the mux to a working clock source without polling this field. 2. Update MUX_n_DC_m and poll this field.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field clears once divider configuration is updated or on destructive reset. If functional reset comes when this field is 1 then it can remain fixed to 1 until divider input clock is restored. 0b - No divider configuration update is pending. 1b - Divider configuration update on at least one divider associated with this multiplexer is pending.

25.5.35 Clock Mux 5 Select Control Register (MUX_5_CSC)

Offset

Register	Offset
MUX_5_CSC	440h

Function

This register provides the clock source selection control of clock mux 5. Clock mux 5 implements software control clock switching, and a graceful clock switch can be performed by executing a sequence of steps in software. See "Software-controlled clock multiplexer" section for details.

This register is reset on destructive reset only.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				SELCTL				0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												FCG	CG	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29	This field is reserved and reads return zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
28-24 SELCTL	Clock source selection control Selects the source clock for clock mux 5. The reserved values are not displayed. 0_0000b - FIRC 0_0001b - SIRC 0_0010b - FXOSC 0_0100b - SXOSC 1_0111b - AIPS_SLOW_CLK
23-4 —	This field is reserved and reads return zeros.
3 FCG	Force clock gate Writing 1 to this bit gates the clock at the output of clock mux 5 to logic-0 irrespective of the logic level of the currently selected clock. Clock gating using this bit should only be performed when it is insured that current clock source is inactive.
2 CG	Clock gate Writing 1 to this bit gates the clock at the output of clock mux 5 to logic-0. Using this bit it is insured that no glitches are resulted when gating the clock.
1-0 —	This field is reserved and reads return zeros.

25.5.36 Clock Mux 5 Select Status Register (MUX_5_CSS)

Offset

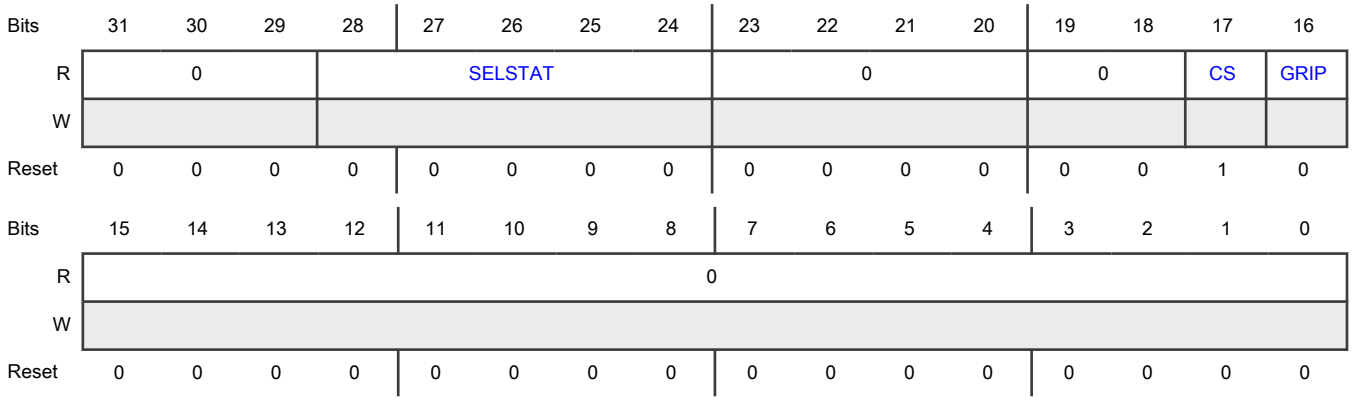
Register	Offset
MUX_5_CSS	444h

Function

This register provides the current clock source selection status for clock mux 5.

This register is reset on destructive reset only.

Diagram



Fields

Field	Function
31-29 —	This field is reserved and reads return zeros.
28-24 SELSTAT	Clock source selection status This value indicates the current source selected for clock mux 5. The reserved values are not displayed. 0_0000b - FIRC 0_0001b - SIRC 0_0010b - FXOSC 0_0100b - SXOSC 1_0111b - AIPS_SLOW_CLK
23-20 —	This field is reserved and reads return zeros.
19-18 —	This field is reserved and reads return zeros.
17 CS	Clock status This field indicates state of the clock at the output of the clock mux. 0b - Clock is gated to logic-0 at output of clock mux 1b - Clock mux is transparent. Active clock pulses at input of clock mux results in same number of pulses at its output
16 GRIP	Gating request is in progress. When a clock gate request is given this bit indicates if the clock gating at the output of mux has completed or not. 0b - Clock source gating or ungating has completed.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Clock source gating or ungating is in progress.
15-0 —	This field is reserved and reads return zeros.

25.5.37 Clock Mux 5 Divider 0 Control Register (MUX_5_DC_0)

Offset

Register	Offset
MUX_5_DC_0	448h

Function

This register controls the clock divider 0 for clock mux 5.

This divider is a 50% duty cycle divider.

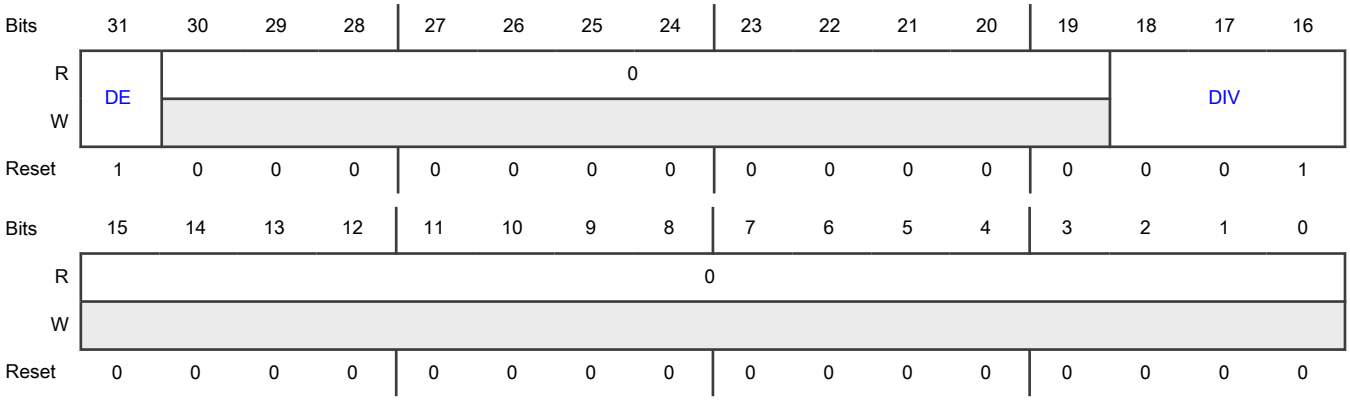
NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

NOTE

Software-controlled clock multiplexer dividers are not expected to return to the default state on the hardware transitions and handshakes occurring as part of the functional reset entry sequence (only hardware-controlled clock multiplexer dividers return to the default state).

Diagram



Fields

Field	Function
31 DE	Divider enable 0b - Divider is disabled. 1b - Divider is enabled.
30-19 —	This field is reserved and reads return zeros.
18-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-0 —	This field is reserved and reads return zeros.

25.5.38 Clock Mux 5 Divider Update Status Register (MUX_5_DIV_UPD_STAT)

Offset

Register	Offset
MUX_5_DIV_UPD_STAT	47Ch

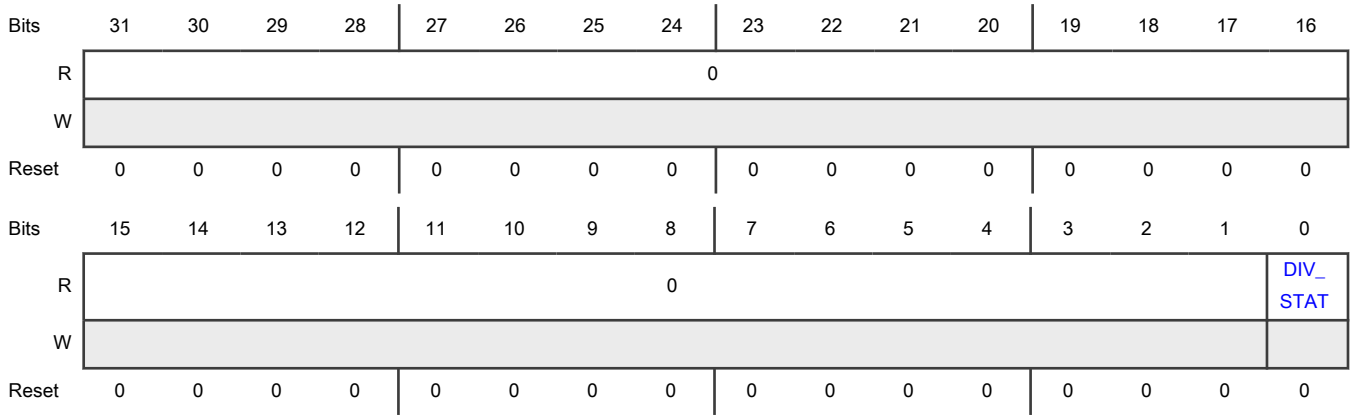
Function

This register provides the update status of the clock dividers corresponding to clock mux 5. When a write operation on any divider control register is performed, the divider status bit in this register is set to logic-1. The bit is set to logic-0 when the divider has sampled the new divider configuration. Performing multiple writes without tracking the status bit on same or other clock dividers inside the same clock mux leads to inconsistent reporting, that is, the divider status maybe be set to logic-0 when the corresponding divider update is pending.

NOTE

Read accesses to MUX_n_DIV_UPD_STAT always complete without returning bus transfer error independent of whether any divider(s) are implemented inside MC_CGM clock mux.

Diagram



Fields

Field	Function
31-1 —	This field is reserved and reads return zeros.
0 DIV_STAT	Divider status for clock mux 5 On reading MUX_n_DIV_UPD_STAT after updating a divider control register, if the value of this field is fixed to 1 because of an error in the selected clock source, perform the following steps to switch the mux to a new clock source: 1. Switch the mux to a working clock source without polling this field. 2. Update MUX_n_DC_m and poll this field. **NOTE** This field clears once divider configuration is updated or on destructive reset. If functional reset comes when this field is 1 then it can remain fixed to 1 until divider input clock is restored. 0b - No divider configuration update is pending. 1b - Divider configuration update on at least one divider associated with this multiplexer is pending.

25.5.39 Clock Mux 6 Select Control Register (MUX_6_CSC)

Offset

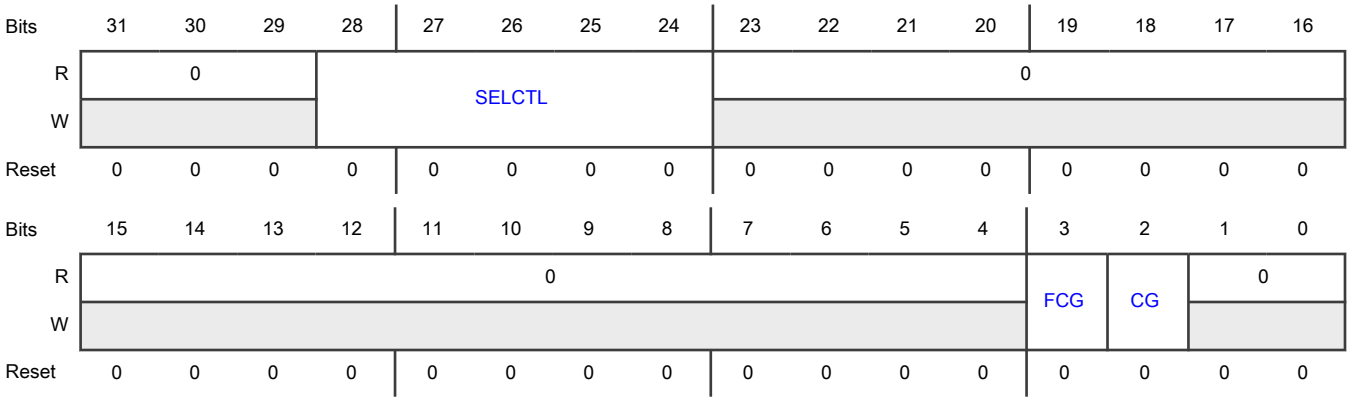
Register	Offset
MUX_6_CSC	480h

Function

This register provides the clock source selection control of clock mux 6. Clock mux 6 implements software control clock switching, and a graceful clock switch can be performed by executing a sequence of steps in software. See "Software-controlled clock multiplexer" section for details.

This register is reset on destructive reset only.

Diagram



Fields

Field	Function
31-29 —	This field is reserved and reads return zeros.
28-24 SELCTL	Clock source selection control Selects the source clock for clock mux 6. The reserved values are not displayed. 0_0000b - FIRC 0_0001b - SIRC 0_0010b - FXOSC 0_0100b - SXOSC 0_1000b - PLL_PHI0_CLK 0_1001b - PLL_PHI1_CLK 0_1100b - PLL_AUX_PHI0_CLK 0_1101b - PLL_AUX_PHI1_CLK 1_0000b - CORE_CLK 1_0011b - HSE_CLK 1_0110b - AIPS_PLAT_CLK 1_0111b - AIPS_SLOW_CLK 1_1000b - GMAC0_MII_RMII_RGMII_TX_CLK 1_1001b - GMAC0_MII_RGMII_RX_CLK
23-4 —	This field is reserved and reads return zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
3 FCG	Force clock gate Writing 1 to this bit gates the clock at the output of clock mux 6 to logic-0 irrespective of the logic level of the currently selected clock. Clock gating using this bit should only be performed when it is insured that current clock source is inactive.
2 CG	Clock gate Writing 1 to this bit gates the clock at the output of clock mux 6 to logic-0. Using this bit it is insured that no glitches are resulted when gating the clock.
1-0 —	This field is reserved and reads return zeros.

25.5.40 Clock Mux 6 Select Status Register (MUX_6_CSS)

Offset

Register	Offset
MUX_6_CSS	484h

Function

This register provides the current clock source selection status for clock mux 6.

This register is reset on destructive reset only.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				SELSTAT				0				0		CS	GRIP
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29	This field is reserved and reads return zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
28-24 SELSTAT	Clock source selection status This value indicates the current source selected for clock mux 6. The reserved values are not displayed. 0_0000b - FIRC 0_0001b - SIRC 0_0010b - FXOSC 0_0100b - SXOSC 0_1000b - PLL_PHI0_CLK 0_1001b - PLL_PHI1_CLK 0_1100b - PLL_AUX_PHI0_CLK 0_1101b - PLL_AUX_PHI1_CLK 1_0000b - CORE_CLK 1_0011b - HSE_CLK 1_0110b - AIPS_PLAT_CLK 1_0111b - AIPS_SLOW_CLK 1_1000b - GMAC0_MII_RMII_RGMII_TX_CLK 1_1001b - GMAC0_MII_RGMII_RX_CLK
23-20 —	This field is reserved and reads return zeros.
19-18 —	This field is reserved and reads return zeros.
17 CS	Clock status This field indicates state of the clock at the output of the clock mux. 0b - Clock is gated to logic-0 at output of clock mux 1b - Clock mux is transparent. Active clock pulses at input of clock mux results in same number of pulses at its output
16 GRIP	Gating request is in progress. When a clock gate request is given this bit indicates if the clock gating at the output of mux has completed or not. 0b - Clock source gating or ungating has completed. 1b - Clock source gating or ungating is in progress.
15-0	This field is reserved and reads return zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	

25.5.41 Clock Mux 6 Divider 0 Control Register (MUX_6_DC_0)

Offset

Register	Offset
MUX_6_DC_0	488h

Function

This register controls the clock divider 0 for clock mux 6.

This divider is a 50% duty cycle divider.

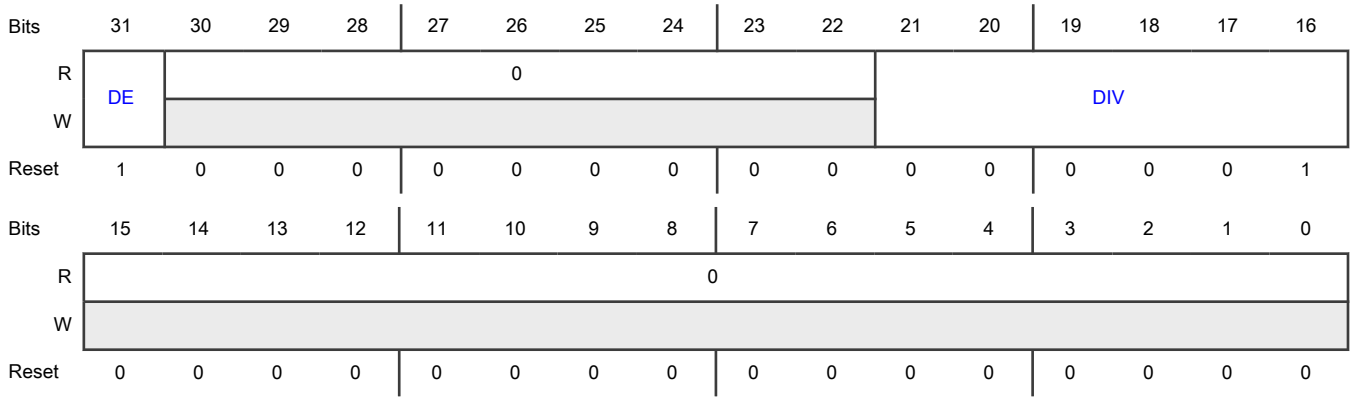
NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

NOTE

Software-controlled clock multiplexer dividers are not expected to return to the default state on the hardware transitions and handshakes occurring as part of the functional reset entry sequence (only hardware-controlled clock multiplexer dividers return to the default state).

Diagram



Fields

Field	Function
31 DE	Divider enable 0b - Divider is disabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Divider is enabled.
30-22 —	This field is reserved and reads return zeros.
21-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-0 —	This field is reserved and reads return zeros.

25.5.42 Clock Mux 6 Divider Update Status Register (MUX_6_DIV_UPD_STAT)

Offset

Register	Offset
MUX_6_DIV_UPD_STAT	4BCh

Function

This register provides the update status of the clock dividers corresponding to clock mux 6. When a write operation on any divider control register is performed, the divider status bit in this register is set to logic-1. The bit is set to logic-0 when the divider has sampled the new divider configuration. Performing multiple writes without tracking the status bit on same or other clock dividers inside the same clock mux leads to inconsistent reporting, that is, the divider status maybe be set to logic-0 when the corresponding divider update is pending.

NOTE

Read accesses to MUX_n_DIV_UPD_STAT always complete without returning bus transfer error independent of whether any divider(s) are implemented inside MC_CGM clock mux.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DIV_STAT
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-1 —	This field is reserved and reads return zeros.
0 DIV_STAT	Divider status for clock mux 6 On reading MUX_n_DIV_UPD_STAT after updating a divider control register, if the value of this field is fixed to 1 because of an error in the selected clock source, perform the following steps to switch the mux to a new clock source: 1. Switch the mux to a working clock source without polling this field. 2. Update MUX_n_DC_m and poll this field. **NOTE** This field clears once divider configuration is updated or on destructive reset. If functional reset comes when this field is 1 then it can remain fixed to 1 until divider input clock is restored. 0b - No divider configuration update is pending. 1b - Divider configuration update on at least one divider associated with this multiplexer is pending.

25.5.43 Clock Mux 7 Select Control Register (MUX_7_CSC)

Offset

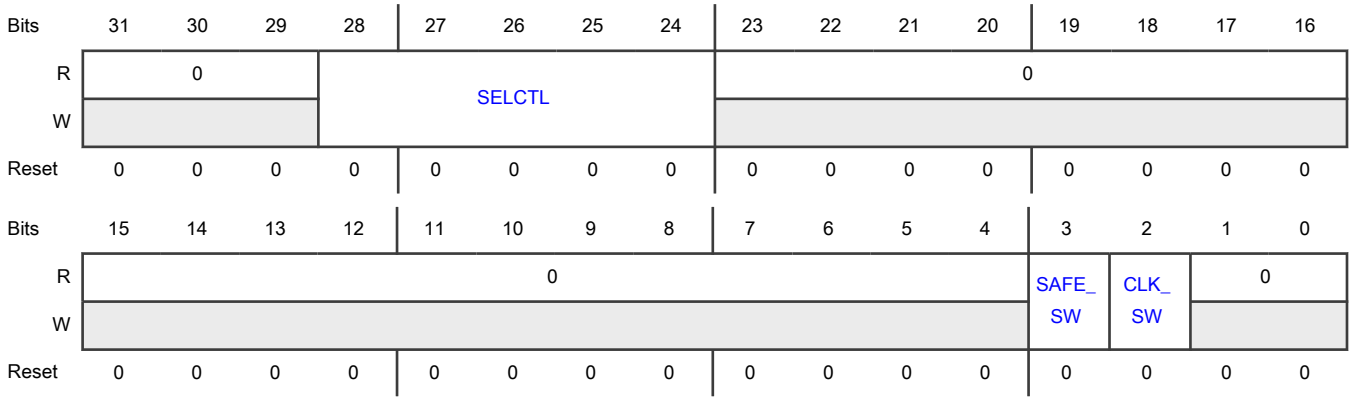
Register	Offset
MUX_7_CSC	4C0h

Function

This register provides the clock source selection control for clock mux 7. Clock mux 7 implements hardware control clock switching ensuring that the clock switch happens in a graceful manner (without glitches). See the "Hardware-controlled clock multiplexer" section for details.

This register is reset on destructive reset only.

Diagram



Fields

Field	Function
31-29 —	This field is reserved and reads return zeros.
28-24 SELCTL	Clock source selection control Selects the source clock for clock mux 7. The reserved values are not displayed. 0_0000b - FIRC 1_1000b - GMAC0_MII_RMII_RGMII_TX_CLK
23-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request Writing 1 to this bit makes a safe clock switch request to FIRC. After a safe clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
2 CLK_SW	Clock switch Writing 1 to this bit makes a clock switch request to clock mux 7. After a clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
1-0 —	This field is reserved and reads return zeros.

25.5.44 Clock Mux 7 Select Status Register (MUX_7_CSS)

Offset

Register	Offset
MUX_7_CSS	4C4h

Function

This register provides the current clock source selection status for clock mux 7.

This register is reset on destructive reset only.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				SELSTAT				0				SWTRG			SWIP
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SAFE_SW	CLK_SW	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	This field is reserved and reads return zeros.
28-24 SELSTAT	Clock source selection status This value indicates the current source selected for clock mux 7. The reserved values are not displayed. 0_0000b - FIRC 1_1000b - GMAC0_MII_RMII_RGMII_TX_CLK
23-20 —	This field is reserved and reads return zeros.
19-17 SWTRG	Switch trigger cause This value indicates the cause for the latest clock source switch. **NOTE** If the clock fails, followed by multiple safe clock switch requests for MC_CGM hardware clock mux, the value of the SWTRG field can be either 4 or 5. 000b - Reserved 001b - Switch after request succeeded. 010b - Switch after the request failed because of an inactive target clock and the current clock is FIRC. 011b - Switch after the request failed because of an inactive current clock and the current clock is FIRC.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	100b - Switch to FIRC because of a safe clock request or reset succeeded. 101b - Switch to FIRC because of a safe clock request or reset succeeded, but the previous current clock source was inactive. 110b - Reserved 111b - Reserved
16 SWIP	Switch in progress **NOTE** New clock switch request can only be given three clock cycles after the completion of the previous request. 0b - Clock source switching is complete. 1b - Clock source switching is in progress.
15-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request This field provides an indication of whether a switch to safe clock operation was requested during the previous/ongoing request on clock mux 7. 0b - No safe clock switch operation was requested. 1b - Safe clock switch operation was requested.
2 CLK_SW	Clock switch This field provides an indication of whether a clock switch operation was requested during the previous/ongoing request on clock mux 7. 0b - No clock switch operation was requested. 1b - Clock switch operation was requested.
1-0 —	This field is reserved and reads return zeros.

25.5.45 Clock Mux 7 Divider 0 Control Register (MUX_7_DC_0)

Offset

Register	Offset
MUX_7_DC_0	4C8h

Function

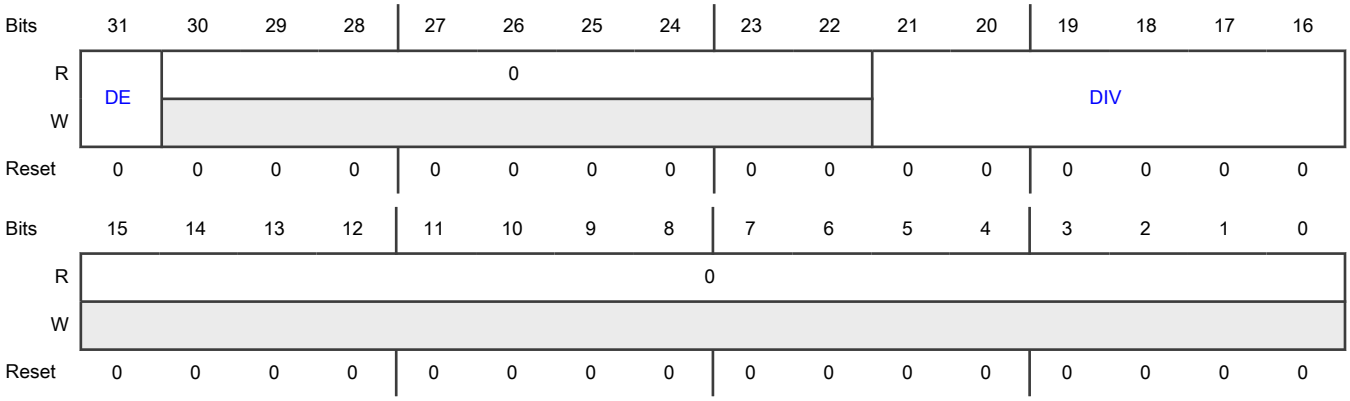
This register controls the clock divider 0 for clock mux 7.

This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram



Fields

Field	Function
31 DE	Divider enable 0b - Divider is disabled. 1b - Divider is enabled.
30-22 —	This field is reserved and reads return zeros.
21-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-0 —	This field is reserved and reads return zeros.

25.5.46 Clock Mux 7 Divider Update Status Register (MUX_7_DIV_UPD_STAT)

Offset

Register	Offset
MUX_7_DIV_UPD_STAT	4FCh

Function

This register provides the update status of the clock dividers corresponding to clock mux 7. When a write operation on any divider control register is performed, the divider status bit in this register is set to logic-1. The bit is set to logic-0 when the

divider has sampled the new divider configuration. Performing multiple writes without tracking the status bit on same or other clock dividers inside the same clock mux leads to inconsistent reporting, that is, the divider status maybe be set to logic-0 when the corresponding divider update is pending.

NOTE

Read accesses to MUX_n_DIV_UPD_STAT always complete without returning bus transfer error independent of whether any divider(s) are implemented inside MC_CGM clock mux.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DIV_STAT
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-1 —	This field is reserved and reads return zeros.
0 DIV_STAT	Divider status for clock mux 7 On reading MUX_n_DIV_UPD_STAT after updating a divider control register, if the value of this field is fixed to 1 because of an error in the selected clock source, perform the following steps to switch the mux to a new clock source: 1. Switch the mux to a working clock source without polling this field. 2. Update MUX_n_DC_m and poll this field. **NOTE** This field clears once divider configuration is updated or on destructive reset. If functional reset comes when this field is 1 then it can remain fixed to 1 until divider input clock is restored. 0b - No divider configuration update is pending. 1b - Divider configuration update on at least one divider associated with this multiplexer is pending.

25.5.47 Clock Mux 8 Select Control Register (MUX_8_CSC)

Offset

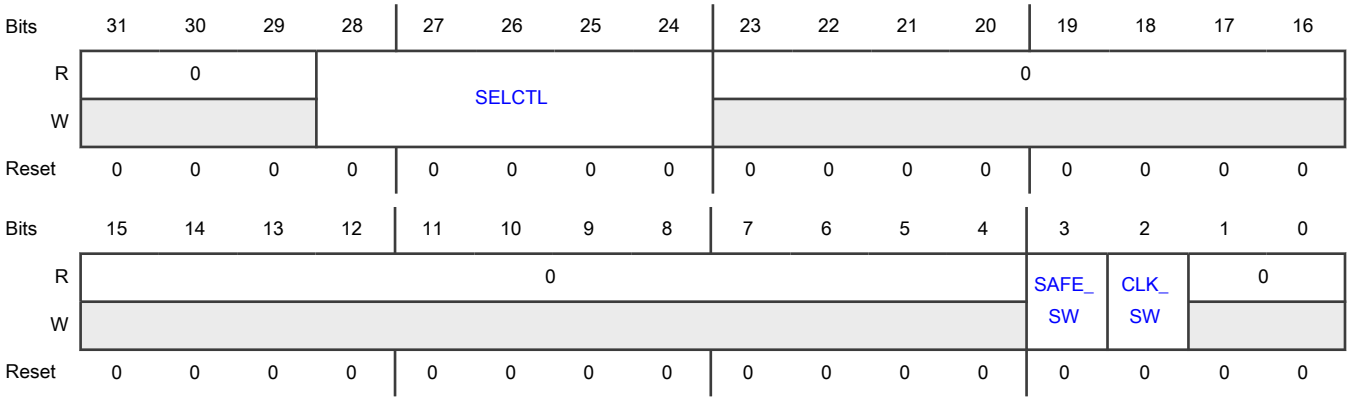
Register	Offset
MUX_8_CSC	500h

Function

This register provides the clock source selection control for clock mux 8. Clock mux 8 implements hardware control clock switching ensuring that the clock switch happens in a graceful manner (without glitches). See the "Hardware-controlled clock multiplexer" section for details.

This register is reset on destructive reset only.

Diagram



Fields

Field	Function
31-29 —	This field is reserved and reads return zeros.
28-24 SELCTL	Clock source selection control Selects the source clock for clock mux 8. The reserved values are not displayed. 0_0000b - FIRC 0_1100b - PLL_AUX_PHI0_CLK 1_1000b - GMAC0_MII_RMII_RGMII_TX_CLK
23-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Writing 1 to this bit makes a safe clock switch request to FIRC. After a safe clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
2 CLK_SW	Clock switch Writing 1 to this bit makes a clock switch request to clock mux 8. After a clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
1-0 —	This field is reserved and reads return zeros.

25.5.48 Clock Mux 8 Select Status Register (MUX_8_CSS)

Offset

Register	Offset
MUX_8_CSS	504h

Function

This register provides the current clock source selection status for clock mux 8.

This register is reset on destructive reset only.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				SELSTAT				0				SWTRG			SWIP
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SAFE_SW	CLK_SW	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	This field is reserved and reads return zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
28-24 SELSTAT	Clock source selection status This value indicates the current source selected for clock mux 8. The reserved values are not displayed. 0_0000b - FIRC 0_1100b - PLL_AUX_PHI0_CLK 1_1000b - GMAC0_MII_RMII_RGMII_TX_CLK
23-20 —	This field is reserved and reads return zeros.
19-17 SWTRG	Switch trigger cause This value indicates the cause for the latest clock source switch. **NOTE** If the clock fails, followed by multiple safe clock switch requests for MC_CGM hardware clock mux, the value of the SWTRG field can be either 4 or 5. 000b - Reserved 001b - Switch after request succeeded. 010b - Switch after the request failed because of an inactive target clock and the current clock is FIRC. 011b - Switch after the request failed because of an inactive current clock and the current clock is FIRC. 100b - Switch to FIRC because of a safe clock request or reset succeeded. 101b - Switch to FIRC because of a safe clock request or reset succeeded, but the previous current clock source was inactive. 110b - Reserved 111b - Reserved
16 SWIP	Switch in progress **NOTE** New clock switch request can only be given three clock cycles after the completion of the previous request. 0b - Clock source switching is complete. 1b - Clock source switching is in progress.
15-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field provides an indication of whether a switch to safe clock operation was requested during the previous/ongoing request on clock mux 8. 0b - No safe clock switch operation was requested. 1b - Safe clock switch operation was requested.
2 CLK_SW	Clock switch This field provides an indication of whether a clock switch operation was requested during the previous/ ongoing request on clock mux 8. 0b - No clock switch operation was requested. 1b - Clock switch operation was requested.
1-0 —	This field is reserved and reads return zeros.

25.5.49 Clock Mux 8 Divider 0 Control Register (MUX_8_DC_0)

Offset

Register	Offset
MUX_8_DC_0	508h

Function

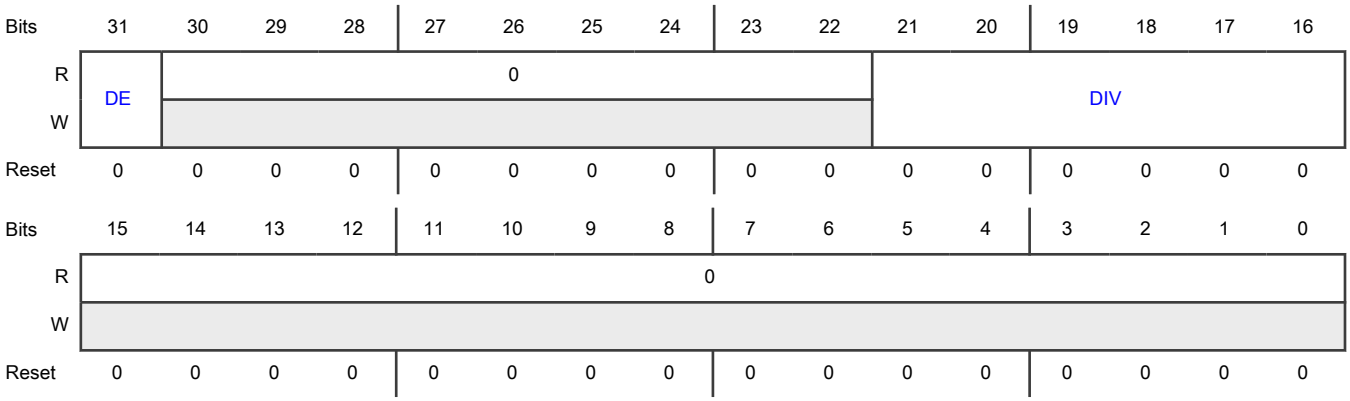
This register controls the clock divider 0 for clock mux 8.

This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram



Fields

Field	Function
31 DE	Divider enable 0b - Divider is disabled. 1b - Divider is enabled.
30-22 —	This field is reserved and reads return zeros.
21-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-0 —	This field is reserved and reads return zeros.

25.5.50 Clock Mux 8 Divider Update Status Register (MUX_8_DIV_UPD_STAT)

Offset

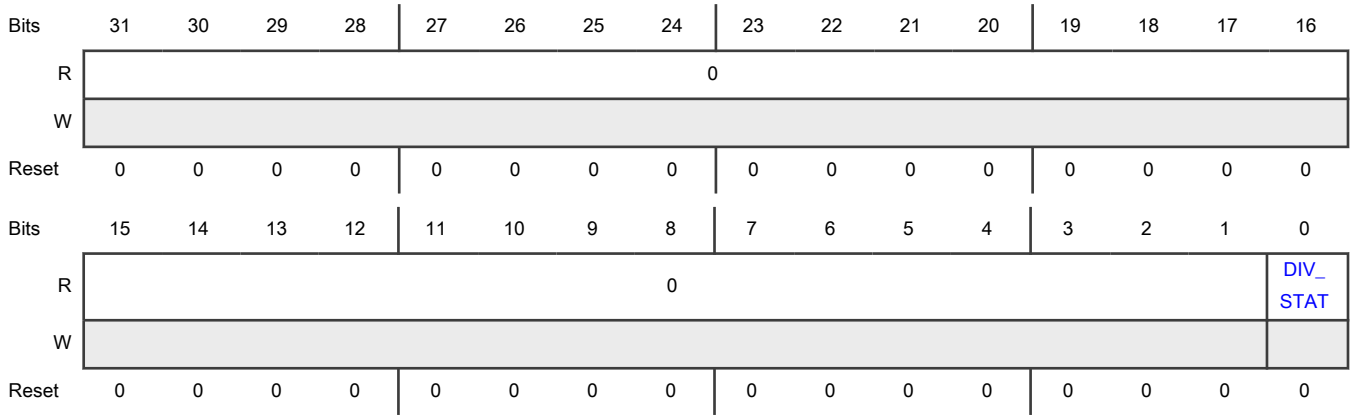
Register	Offset
MUX_8_DIV_UPD_STAT	53Ch

Function
This register provides the update status of the clock dividers corresponding to clock mux 8. When a write operation on any divider control register is performed, the divider status bit in this register is set to logic-1. The bit is set to logic-0 when the divider has sampled the new divider configuration. Performing multiple writes without tracking the status bit on same or other clock dividers inside the same clock mux leads to inconsistent reporting, that is, the divider status maybe be set to logic-0 when the corresponding divider update is pending.

NOTE

Read accesses to MUX_n_DIV_UPD_STAT always complete without returning bus transfer error independent of whether any divider(s) are implemented inside MC_CGM clock mux.

Diagram



Fields

Field	Function
31-1 —	This field is reserved and reads return zeros.
0 DIV_STAT	Divider status for clock mux 8 On reading MUX_n_DIV_UPD_STAT after updating a divider control register, if the value of this field is fixed to 1 because of an error in the selected clock source, perform the following steps to switch the mux to a new clock source: 1. Switch the mux to a working clock source without polling this field. 2. Update MUX_n_DC_m and poll this field. **NOTE** This field clears once divider configuration is updated or on destructive reset. If functional reset comes when this field is 1 then it can remain fixed to 1 until divider input clock is restored. 0b - No divider configuration update is pending. 1b - Divider configuration update on at least one divider associated with this multiplexer is pending.

25.5.51 Clock Mux 9 Select Control Register (MUX_9_CSC)

Offset

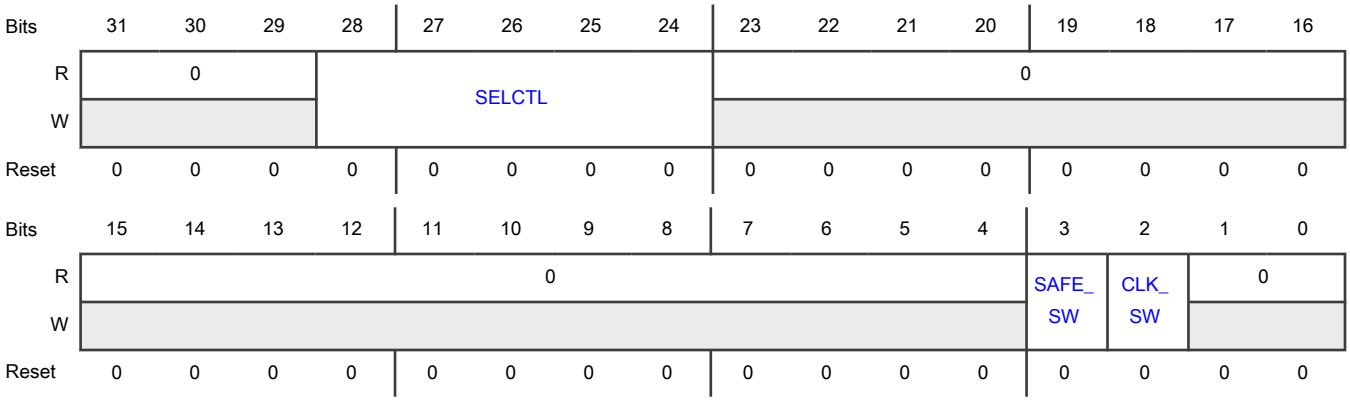
Register	Offset
MUX_9_CSC	540h

Function

This register provides the clock source selection control for clock mux 9. Clock mux 9 implements hardware control clock switching ensuring that the clock switch happens in a graceful manner (without glitches). See the "Hardware-controlled clock multiplexer" section for details.

This register is reset on destructive reset only.

Diagram



Fields

Field	Function
31-29 —	This field is reserved and reads return zeros.
28-24 SELCTL	Clock source selection control Selects the source clock for clock mux 9. The reserved values are not displayed. 0_0000b - FIRC 0_0010b - FXOSC 0_1100b - PLL_AUX_PHI0_CLK 1_1000b - GMAC0_MII_RMII_RGMII_TX_CLK 1_1001b - GMAC0_MII_RGMII_RX_CLK 1_1101b - GMAC1_MII_RGMII_RX_CLK 1_1110b - GMAC1_MII_RMII_RGMII_TX_CLK
23-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request Writing 1 to this bit makes a safe clock switch request to FIRC. After a safe clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
2 CLK_SW	Clock switch Writing 1 to this bit makes a clock switch request to clock mux 9. After a clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
1-0 —	This field is reserved and reads return zeros.

25.5.52 Clock Mux 9 Select Status Register (MUX_9_CSS)

Offset

Register	Offset
MUX_9_CSS	544h

Function

This register provides the current clock source selection status for clock mux 9.

This register is reset on destructive reset only.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				SELSTAT				0				SWTRG			SWIP
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SAFE_SW	CLK_SW	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	This field is reserved and reads return zeros.
28-24 SELSTAT	Clock source selection status This value indicates the current source selected for clock mux 9. The reserved values are not displayed. 0_0000b - FIRC 0_0010b - FXOSC 0_1100b - PLL_AUX_PHI0_CLK 1_1000b - GMAC0_MII_RMII_RGMII_TX_CLK 1_1001b - GMAC0_MII_RGMII_RX_CLK 1_1101b - GMAC1_MII_RGMII_RX_CLK 1_1110b - GMAC1_MII_RMII_RGMII_TX_CLK
23-20	This field is reserved and reads return zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
19-17 SWTRG	Switch trigger cause This value indicates the cause for the latest clock source switch. **NOTE** If the clock fails, followed by multiple safe clock switch requests for MC_CGM hardware clock mux, the value of the SWTRG field can be either 4 or 5. 000b - Reserved 001b - Switch after request succeeded. 010b - Switch after the request failed because of an inactive target clock and the current clock is FIRC. 011b - Switch after the request failed because of an inactive current clock and the current clock is FIRC. 100b - Switch to FIRC because of a safe clock request or reset succeeded. 101b - Switch to FIRC because of a safe clock request or reset succeeded, but the previous current clock source was inactive. 110b - Reserved 111b - Reserved
16 SWIP	Switch in progress **NOTE** New clock switch request can only be given three clock cycles after the completion of the previous request. 0b - Clock source switching is complete. 1b - Clock source switching is in progress.
15-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request This field provides an indication of whether a switch to safe clock operation was requested during the previous/ongoing request on clock mux 9. 0b - No safe clock switch operation was requested. 1b - Safe clock switch operation was requested.
2 CLK_SW	Clock switch This field provides an indication of whether a clock switch operation was requested during the previous/ongoing request on clock mux 9.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No clock switch operation was requested. 1b - Clock switch operation was requested.
1-0 —	This field is reserved and reads return zeros.

25.5.53 Clock Mux 9 Divider 0 Control Register (MUX_9_DC_0)

Offset

Register	Offset
MUX_9_DC_0	548h

Function

This register controls the clock divider 0 for clock mux 9.

This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DE	0									DIV					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 DE	Divider enable 0b - Divider is disabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Divider is enabled.
30-22 —	This field is reserved and reads return zeros.
21-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-0 —	This field is reserved and reads return zeros.

25.5.54 Clock Mux 9 Divider Update Status Register (MUX_9_DIV_UPD_STAT)

Offset

Register	Offset
MUX_9_DIV_UPD_STAT	57Ch

Function

This register provides the update status of the clock dividers corresponding to clock mux 9. When a write operation on any divider control register is performed, the divider status bit in this register is set to logic-1. The bit is set to logic-0 when the divider has sampled the new divider configuration. Performing multiple writes without tracking the status bit on same or other clock dividers inside the same clock mux leads to inconsistent reporting, that is, the divider status maybe be set to logic-0 when the corresponding divider update is pending.

NOTE

Read accesses to MUX_n_DIV_UPD_STAT always complete without returning bus transfer error independent of whether any divider(s) are implemented inside MC_CGM clock mux.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DIV_STAT
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-1 —	This field is reserved and reads return zeros.
0 DIV_STAT	Divider status for clock mux 9 On reading MUX_n_DIV_UPD_STAT after updating a divider control register, if the value of this field is fixed to 1 because of an error in the selected clock source, perform the following steps to switch the mux to a new clock source: 1. Switch the mux to a working clock source without polling this field. 2. Update MUX_n_DC_m and poll this field. **NOTE** This field clears once divider configuration is updated or on destructive reset. If functional reset comes when this field is 1 then it can remain fixed to 1 until divider input clock is restored. 0b - No divider configuration update is pending. 1b - Divider configuration update on at least one divider associated with this multiplexer is pending.

25.5.55 Clock Mux 10 Select Control Register (MUX_10_CSC)

Offset

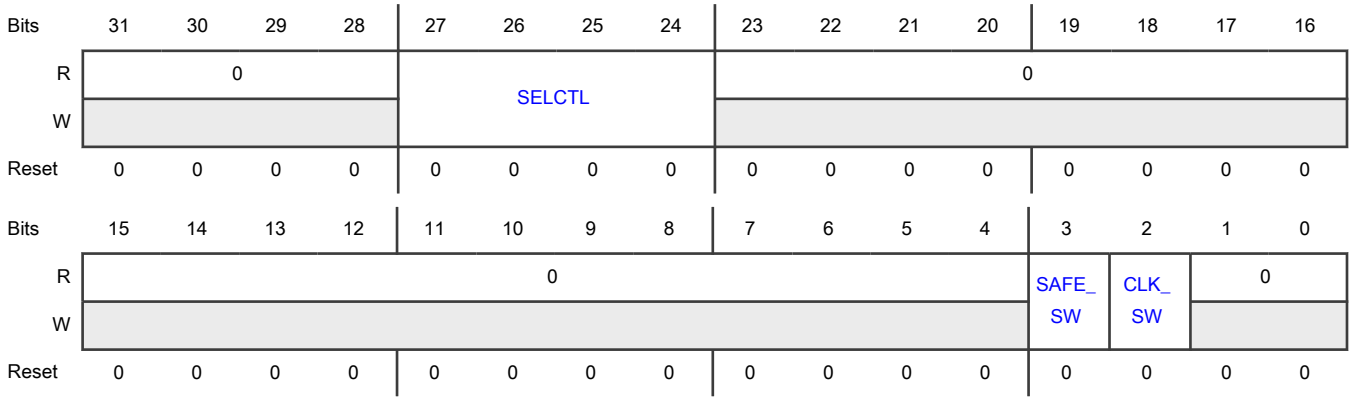
Register	Offset
MUX_10_CSC	580h

Function

This register provides the clock source selection control for clock mux 10. Clock mux 10 implements hardware control clock switching ensuring that the clock switch happens in a graceful manner (without glitches). See the "Hardware-controlled clock multiplexer" section for details.

This register is reset on destructive reset only.

Diagram



Fields

Field	Function
31-28 —	This field is reserved and reads return zeros.
27-24 SELCTL	Clock source selection control Selects the source clock for clock mux 10. The reserved values are not displayed. 0000b - FIRC 0010b - FXOSC 1001b - PLL_PHI1_CLK 1100b - PLL_AUX_PHI0_CLK
23-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request Writing 1 to this bit makes a safe clock switch request to FIRC. After a safe clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
2 CLK_SW	Clock switch Writing 1 to this bit makes a clock switch request to clock mux 10. After a clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
1-0 —	This field is reserved and reads return zeros.

25.5.56 Clock Mux 10 Select Status Register (MUX_10_CSS)

Offset

Register	Offset
MUX_10_CSS	584h

Function

This register provides the current clock source selection status for clock mux 10.

This register is reset on destructive reset only.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				SELSTAT				0				SWTRG			SWIP
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SAFE_SW	CLK_SW	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	This field is reserved and reads return zeros.
27-24 SELSTAT	Clock source selection status This value indicates the current source selected for clock mux 10. The reserved values are not displayed. 0000b - FIRC 0010b - FXOSC 1001b - PLL_PHI1_CLK 1100b - PLL_AUX_PHI0_CLK
23-20 —	This field is reserved and reads return zeros.
19-17	Switch trigger cause

Table continues on the next page...

Table continued from the previous page...

Field	Function
SWTRG	This value indicates the cause for the latest clock source switch. **NOTE** If the clock fails, followed by multiple safe clock switch requests for MC_CGM hardware clock mux, the value of the SWTRG field can be either 4 or 5. 000b - Reserved 001b - Switch after request succeeded. 010b - Switch after the request failed because of an inactive target clock and the current clock is FIRC. 011b - Switch after the request failed because of an inactive current clock and the current clock is FIRC. 100b - Switch to FIRC because of a safe clock request or reset succeeded. 101b - Switch to FIRC because of a safe clock request or reset succeeded, but the previous current clock source was inactive. 110b - Reserved 111b - Reserved
16 SWIP	Switch in progress **NOTE** New clock switch request can only be given three clock cycles after the completion of the previous request. 0b - Clock source switching is complete. 1b - Clock source switching is in progress.
15-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request This field provides an indication of whether a switch to safe clock operation was requested during the previous/ongoing request on clock mux 10. 0b - No safe clock switch operation was requested. 1b - Safe clock switch operation was requested.
2 CLK_SW	Clock switch This field provides an indication of whether a clock switch operation was requested during the previous/ ongoing request on clock mux 10. 0b - No clock switch operation was requested. 1b - Clock switch operation was requested.
1-0	This field is reserved and reads return zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	

25.5.57 Clock Mux 10 Divider 0 Control Register (MUX_10_DC_0)

Offset

Register	Offset
MUX_10_DC_0	588h

Function

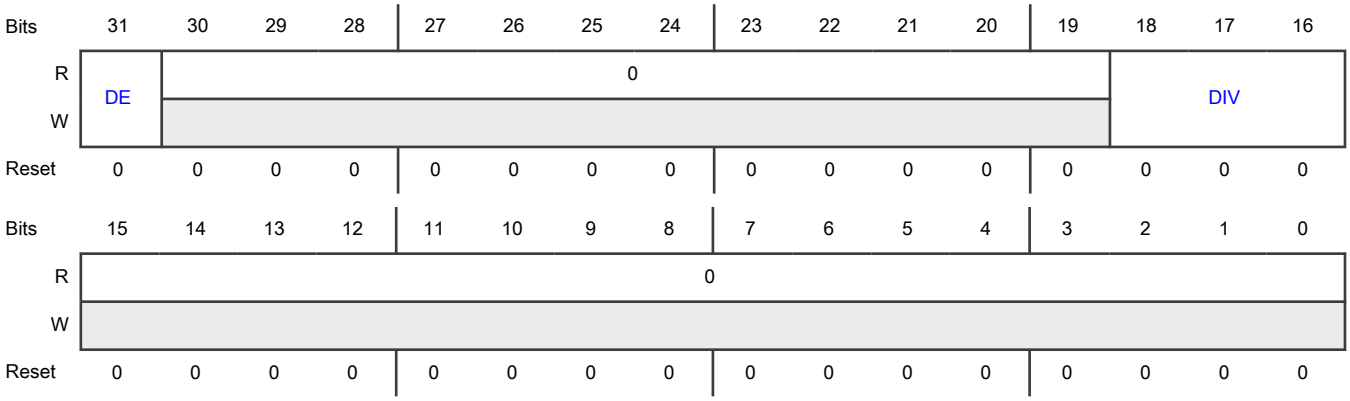
This register controls the clock divider 0 for clock mux 10.

This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram



Fields

Field	Function
31 DE	Divider enable 0b - Divider is disabled. 1b - Divider is enabled.
30-19 —	This field is reserved and reads return zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
18-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-0 —	This field is reserved and reads return zeros.

25.5.58 Clock Mux 10 Divider Update Status Register (MUX_10_DIV_UPD_STAT)

Offset

Register	Offset
MUX_10_DIV_UPD_STAT	5BCh

Function

This register provides the update status of the clock dividers corresponding to clock mux 10. When a write operation on any divider control register is performed, the divider status bit in this register is set to logic-1. The bit is set to logic-0 when the divider has sampled the new divider configuration. Performing multiple writes without tracking the status bit on same or other clock dividers inside the same clock mux leads to inconsistent reporting, that is, the divider status maybe be set to logic-0 when the corresponding divider update is pending.

NOTE

Read accesses to MUX_n_DIV_UPD_STAT always complete without returning bus transfer error independent of whether any divider(s) are implemented inside MC_CGM clock mux.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DIV_STAT
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-1 —	This field is reserved and reads return zeros.
0 DIV_STAT	Divider status for clock mux 10 On reading MUX_n_DIV_UPD_STAT after updating a divider control register, if the value of this field is fixed to 1 because of an error in the selected clock source, perform the following steps to switch the mux to a new clock source: 1. Switch the mux to a working clock source without polling this field. 2. Update MUX_n_DC_m and poll this field. **NOTE** This field clears once divider configuration is updated or on destructive reset. If functional reset comes when this field is 1 then it can remain fixed to 1 until divider input clock is restored. 0b - No divider configuration update is pending. 1b - Divider configuration update on at least one divider associated with this multiplexer is pending.

25.5.59 Clock Mux 11 Select Control Register (MUX_11_CSC)

Offset

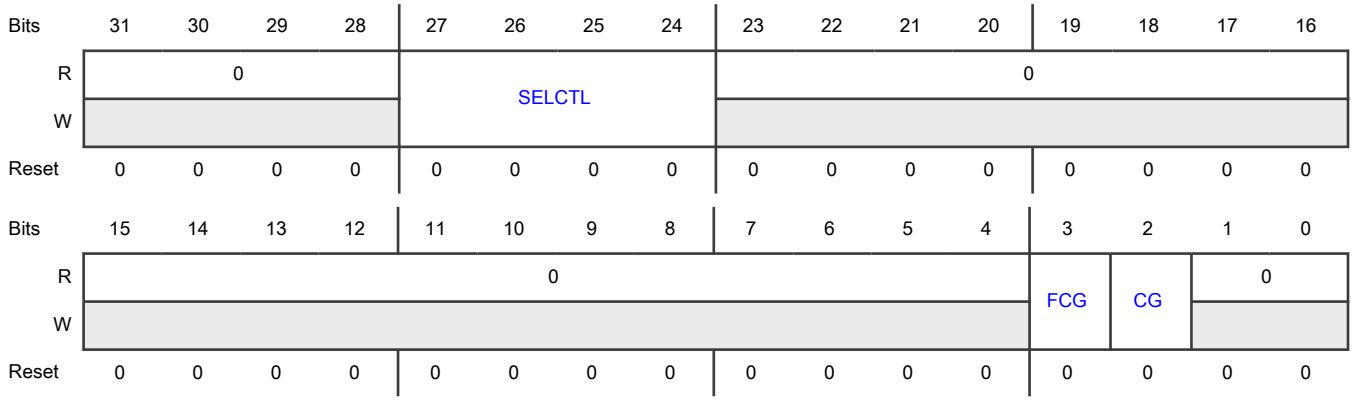
Register	Offset
MUX_11_CSC	5C0h

Function

This register provides the clock source selection control of clock mux 11. Clock mux 11 implements software control clock switching, and a graceful clock switch can be performed by executing a sequence of steps in software. See "Software-controlled clock multiplexer" section for details.

This register is reset on destructive reset only.

Diagram



Fields

Field	Function
31-28 —	This field is reserved and reads return zeros.
27-24 SELCTL	Clock source selection control Selects the source clock for clock mux 11. The reserved values are not displayed. 0000b - FIRC 0010b - FXOSC 1001b - PLL_PHI1_CLK 1100b - PLL_AUX_PHI0_CLK
23-4 —	This field is reserved and reads return zeros.
3 FCG	Force clock gate Writing 1 to this bit gates the clock at the output of clock mux 11 to logic-0 irrespective of the logic level of the currently selected clock. Clock gating using this bit should only be performed when it is insured that current clock source is inactive.
2 CG	Clock gate Writing 1 to this bit gates the clock at the output of clock mux 11 to logic-0. Using this bit it is insured that no glitches are resulted when gating the clock.
1-0 —	This field is reserved and reads return zeros.

25.5.60 Clock Mux 11 Select Status Register (MUX_11_CSS)

Offset

Register	Offset
MUX_11_CSS	5C4h

Function

This register provides the current clock source selection status for clock mux 11.

This register is reset on destructive reset only.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				SELSTAT				0				0		CS	GRIP
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	This field is reserved and reads return zeros.
27-24 SELSTAT	Clock source selection status This value indicates the current source selected for clock mux 11. The reserved values are not displayed. 0000b - FIRC 0010b - FXOSC 1001b - PLL_PHI1_CLK 1100b - PLL_AUX_PHI0_CLK
23-20 —	This field is reserved and reads return zeros.
19-18 —	This field is reserved and reads return zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
17 CS	Clock status This field indicates state of the clock at the output of the clock mux. 0b - Clock is gated to logic-0 at output of clock mux 1b - Clock mux is transparent. Active clock pulses at input of clock mux results in same number of pulses at its output
16 GRIP	Gating request is in progress. When a clock gate request is given this bit indicates if the clock gating at the output of mux has completed or not. 0b - Clock source gating or ungating has completed. 1b - Clock source gating or ungating is in progress.
15-0 —	This field is reserved and reads return zeros.

25.5.61 Clock Mux 11 Divider 0 Control Register (MUX_11_DC_0)

Offset

Register	Offset
MUX_11_DC_0	5C8h

Function

This register controls the clock divider 0 for clock mux 11.

This divider is a 50% duty cycle divider.

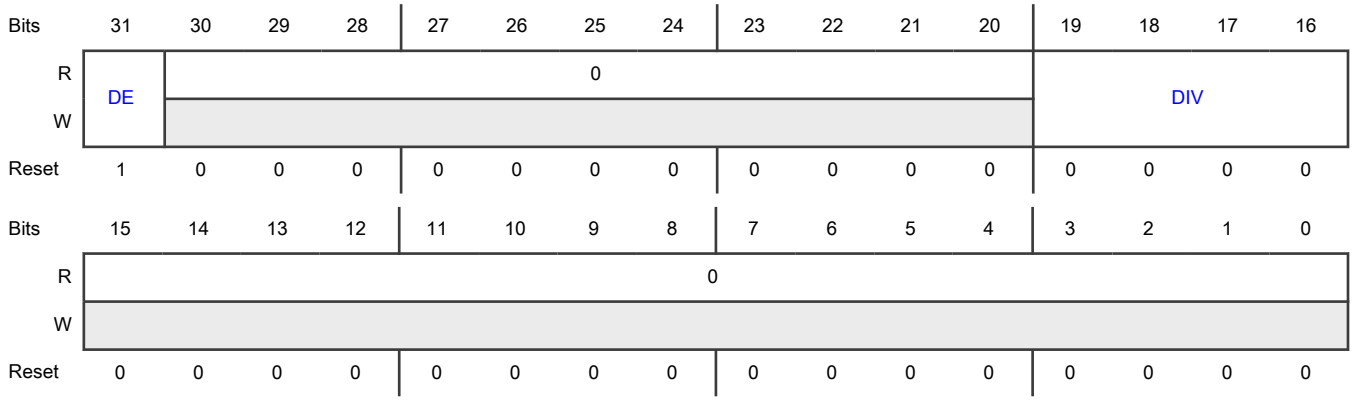
NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

NOTE

Software-controlled clock multiplexer dividers are not expected to return to the default state on the hardware transitions and handshakes occurring as part of the functional reset entry sequence (only hardware-controlled clock multiplexer dividers return to the default state).

Diagram



Fields

Field	Function
31 DE	Divider enable 0b - Divider is disabled. 1b - Divider is enabled.
30-20 —	This field is reserved and reads return zeros.
19-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-0 —	This field is reserved and reads return zeros.

25.5.62 Clock Mux 11 Divider Update Status Register (MUX_11_DIV_UPD_STAT)

Offset

Register	Offset
MUX_11_DIV_UPD_STAT	5FCh

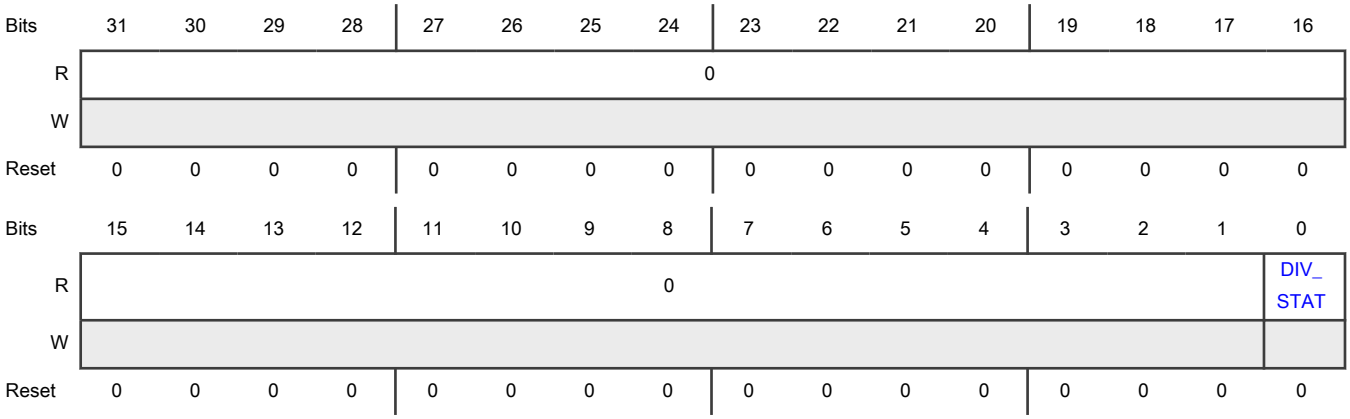
Function

This register provides the update status of the clock dividers corresponding to clock mux 11. When a write operation on any divider control register is performed, the divider status bit in this register is set to logic-1. The bit is set to logic-0 when the divider has sampled the new divider configuration. Performing multiple writes without tracking the status bit on same or other clock dividers inside the same clock mux leads to inconsistent reporting, that is, the divider status maybe be set to logic-0 when the corresponding divider update is pending.

NOTE

Read accesses to MUX_n_DIV_UPD_STAT always complete without returning bus transfer error independent of whether any divider(s) are implemented inside MC_CGM clock mux.

Diagram



Fields

Field	Function
31-1 —	This field is reserved and reads return zeros.
0 DIV_STAT	Divider status for clock mux 11 On reading MUX_n_DIV_UPD_STAT after updating a divider control register, if the value of this field is fixed to 1 because of an error in the selected clock source, perform the following steps to switch the mux to a new clock source: 1. Switch the mux to a working clock source without polling this field. 2. Update MUX_n_DC_m and poll this field. **NOTE** This field clears once divider configuration is updated or on destructive reset. If functional reset comes when this field is 1 then it can remain fixed to 1 until divider input clock is restored. 0b - No divider configuration update is pending. 1b - Divider configuration update on at least one divider associated with this multiplexer is pending.

25.5.63 Clock Mux 13 Select Control Register (MUX_13_CSC)

Offset

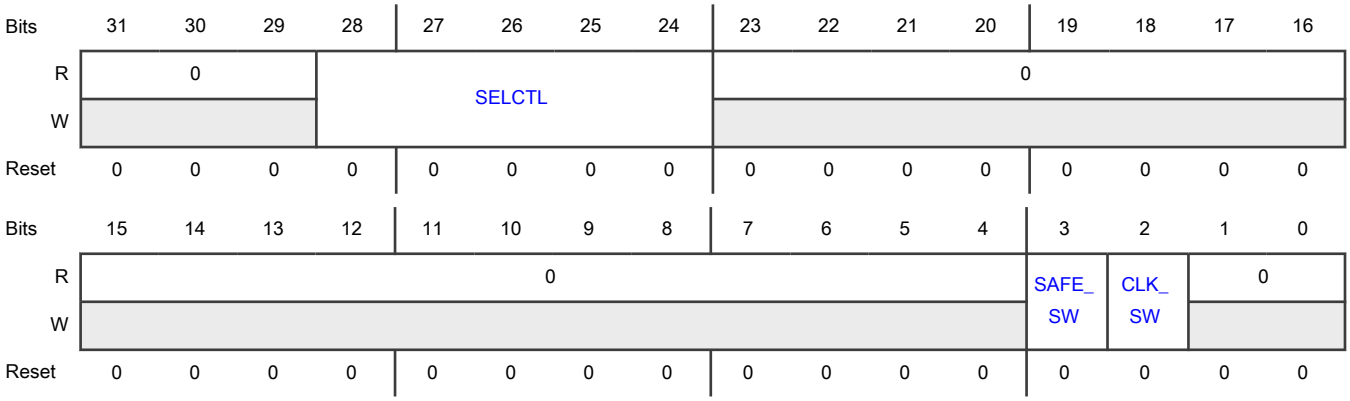
Register	Offset
MUX_13_CSC	640h

Function

This register provides the clock source selection control for clock mux 13. Clock mux 13 implements hardware control clock switching ensuring that the clock switch happens in a graceful manner (without glitches). See the "Hardware-controlled clock multiplexer" section for details.

This register is reset on destructive reset only.

Diagram



Fields

Field	Function
31-29 —	This field is reserved and reads return zeros.
28-24 SELCTL	Clock source selection control Selects the source clock for clock mux 13. The reserved values are not displayed. 0_0000b - FIRC 0_0010b - FXOSC 1_0110b - AIPS_PLAT_CLK
23-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request Writing 1 to this bit makes a safe clock switch request to FIRC. After a safe clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
2 CLK_SW	Clock switch Writing 1 to this bit makes a clock switch request to clock mux 13. After a clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
1-0 —	This field is reserved and reads return zeros.

25.5.64 Clock Mux 13 Select Status Register (MUX_13_CSS)

Offset

Register	Offset
MUX_13_CSS	644h

Function

This register provides the current clock source selection status for clock mux 13.

This register is reset on destructive reset only.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				SELSTAT				0				SWTRG			SWIP
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SAFE_SW	CLK_SW	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	This field is reserved and reads return zeros.
28-24 SELSTAT	Clock source selection status This value indicates the current source selected for clock mux 13. The reserved values are not displayed. 0_0000b - FIRC 0_0010b - FXOSC 1_0110b - AIPS_PLAT_CLK
23-20 —	This field is reserved and reads return zeros.
19-17 SWTRG	Switch trigger cause This value indicates the cause for the latest clock source switch.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE If the clock fails, followed by multiple safe clock switch requests for MC_CGM hardware clock mux, the value of the SWTRG field can be either 4 or 5. 000b - Reserved 001b - Switch after request succeeded. 010b - Switch after the request failed because of an inactive target clock and the current clock is FIRC. 011b - Switch after the request failed because of an inactive current clock and the current clock is FIRC. 100b - Switch to FIRC because of a safe clock request or reset succeeded. 101b - Switch to FIRC because of a safe clock request or reset succeeded, but the previous current clock source was inactive. 110b - Reserved 111b - Reserved
16 SWIP	Switch in progress **NOTE** New clock switch request can only be given three clock cycles after the completion of the previous request. 0b - Clock source switching is complete. 1b - Clock source switching is in progress.
15-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request This field provides an indication of whether a switch to safe clock operation was requested during the previous/ongoing request on clock mux 13. 0b - No safe clock switch operation was requested. 1b - Safe clock switch operation was requested.
2 CLK_SW	Clock switch This field provides an indication of whether a clock switch operation was requested during the previous/ongoing request on clock mux 13. 0b - No clock switch operation was requested. 1b - Clock switch operation was requested.
1-0 —	This field is reserved and reads return zeros.

25.5.65 Clock Mux 13 Divider 0 Control Register (MUX_13_DC_0)

Offset

Register	Offset
MUX_13_DC_0	648h

Function

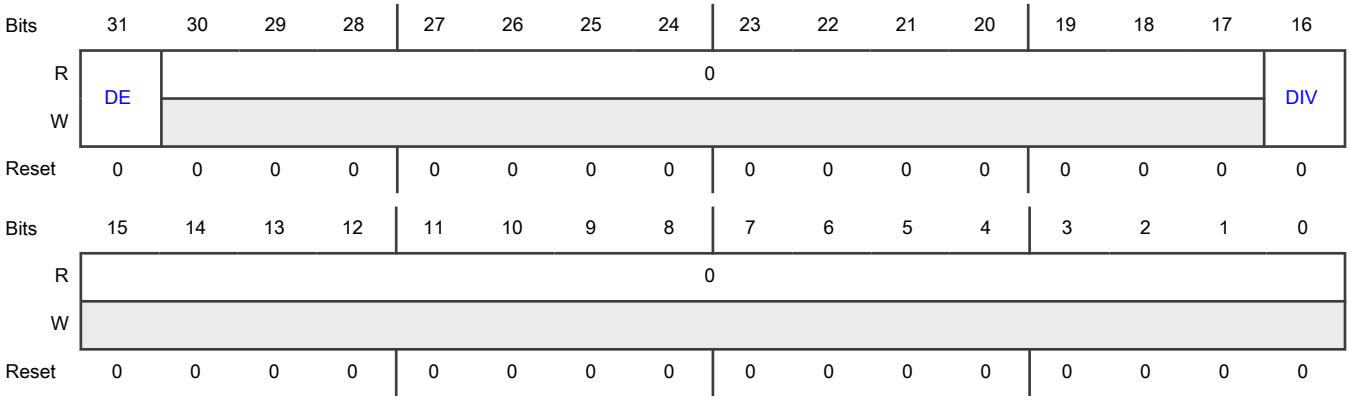
This register controls the clock divider 0 for clock mux 13.

This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram



Fields

Field	Function
31 DE	Divider enable 0b - Divider is disabled. 1b - Divider is enabled.
30-17 —	This field is reserved and reads return zeros.
16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-0 —	This field is reserved and reads return zeros.

25.5.66 Clock Mux 13 Divider Update Status Register (MUX_13_DIV_UPD_STAT)

Offset

Register	Offset
MUX_13_DIV_UPD_STAT	67Ch

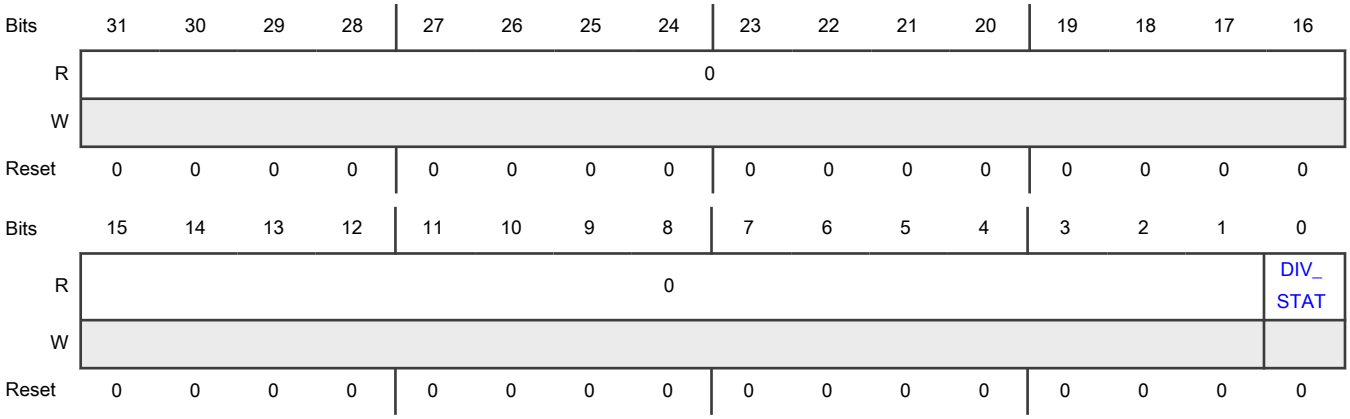
Function

This register provides the update status of the clock dividers corresponding to clock mux 13. When a write operation on any divider control register is performed, the divider status bit in this register is set to logic-1. The bit is set to logic-0 when the divider has sampled the new divider configuration. Performing multiple writes without tracking the status bit on same or other clock dividers inside the same clock mux leads to inconsistent reporting, that is, the divider status maybe be set to logic-0 when the corresponding divider update is pending.

NOTE

Read accesses to MUX_n_DIV_UPD_STAT always complete without returning bus transfer error independent of whether any divider(s) are implemented inside MC_CGM clock mux.

Diagram



Fields

Field	Function
31-1 —	This field is reserved and reads return zeros.
0 DIV_STAT	Divider status for clock mux 13 On reading MUX_n_DIV_UPD_STAT after updating a divider control register, if the value of this field is fixed to 1 because of an error in the selected clock source, perform the following steps to switch the mux to a new clock source: 1. Switch the mux to a working clock source without polling this field. 2. Update MUX_n_DC_m and poll this field.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field clears once divider configuration is updated or on destructive reset. If functional reset comes when this field is 1 then it can remain fixed to 1 until divider input clock is restored. 0b - No divider configuration update is pending. 1b - Divider configuration update on at least one divider associated with this multiplexer is pending.

25.5.67 Clock Mux 15 Select Control Register (MUX_15_CSC)

Offset

Register	Offset
MUX_15_CSC	6C0h

Function

This register provides the clock source selection control for clock mux 15. Clock mux 15 implements hardware control clock switching ensuring that the clock switch happens in a graceful manner (without glitches). See the "Hardware-controlled clock multiplexer" section for details.

This register is reset on destructive reset only.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				SELCTL				0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SAFE_SW	CLK_SW	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29	This field is reserved and reads return zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
28-24 SELCTL	Clock source selection control Selects the source clock for clock mux 15. The reserved values are not displayed. 0_0000b - FIRC 1_1110b - GMAC1_MII_RMII_RGMII_TX_CLK
23-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request Writing 1 to this bit makes a safe clock switch request to FIRC. After a safe clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
2 CLK_SW	Clock switch Writing 1 to this bit makes a clock switch request to clock mux 15. After a clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
1-0 —	This field is reserved and reads return zeros.

25.5.68 Clock Mux 15 Select Status Register (MUX_15_CSS)

Offset

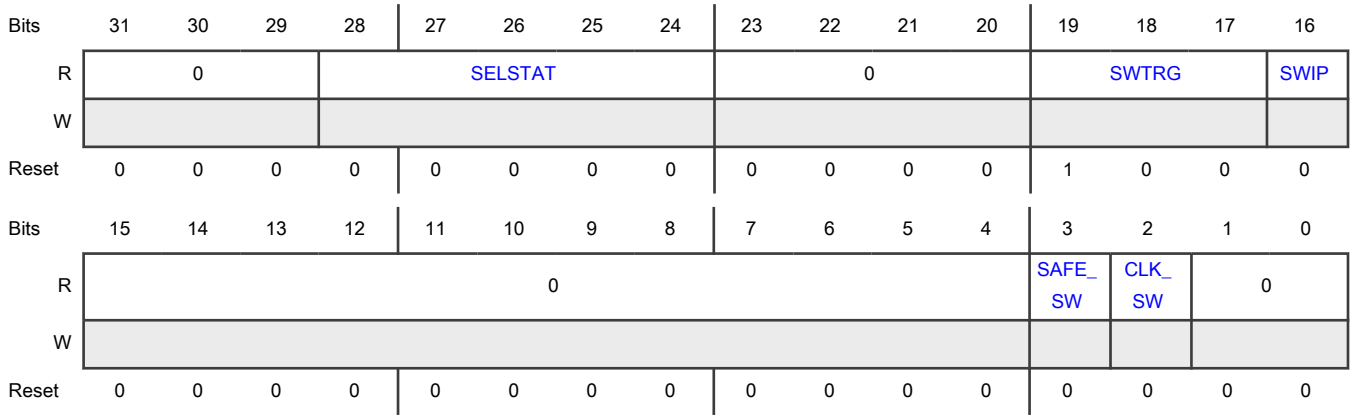
Register	Offset
MUX_15_CSS	6C4h

Function

This register provides the current clock source selection status for clock mux 15.

This register is reset on destructive reset only.

Diagram



Fields

Field	Function
31-29 —	This field is reserved and reads return zeros.
28-24 SELSTAT	Clock source selection status This value indicates the current source selected for clock mux 15. The reserved values are not displayed. 0_0000b - FIRC 1_1110b - GMAC1_MII_RMII_RGMII_TX_CLK
23-20 —	This field is reserved and reads return zeros.
19-17 SWTRG	Switch trigger cause This value indicates the cause for the latest clock source switch. **NOTE** If the clock fails, followed by multiple safe clock switch requests for MC_CGM hardware clock mux, the value of the SWTRG field can be either 4 or 5. 000b - Reserved 001b - Switch after request succeeded. 010b - Switch after the request failed because of an inactive target clock and the current clock is FIRC. 011b - Switch after the request failed because of an inactive current clock and the current clock is FIRC. 100b - Switch to FIRC because of a safe clock request or reset succeeded. 101b - Switch to FIRC because of a safe clock request or reset succeeded, but the previous current clock source was inactive.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	110b - Reserved 111b - Reserved
16 SWIP	Switch in progress **NOTE** New clock switch request can only be given three clock cycles after the completion of the previous request. 0b - Clock source switching is complete. 1b - Clock source switching is in progress.
15-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request This field provides an indication of whether a switch to safe clock operation was requested during the previous/ongoing request on clock mux 15. 0b - No safe clock switch operation was requested. 1b - Safe clock switch operation was requested.
2 CLK_SW	Clock switch This field provides an indication of whether a clock switch operation was requested during the previous/ongoing request on clock mux 15. 0b - No clock switch operation was requested. 1b - Clock switch operation was requested.
1-0 —	This field is reserved and reads return zeros.

25.5.69 Clock Mux 15 Divider 0 Control Register (MUX_15_DC_0)

Offset

Register	Offset
MUX_15_DC_0	6C8h

Function

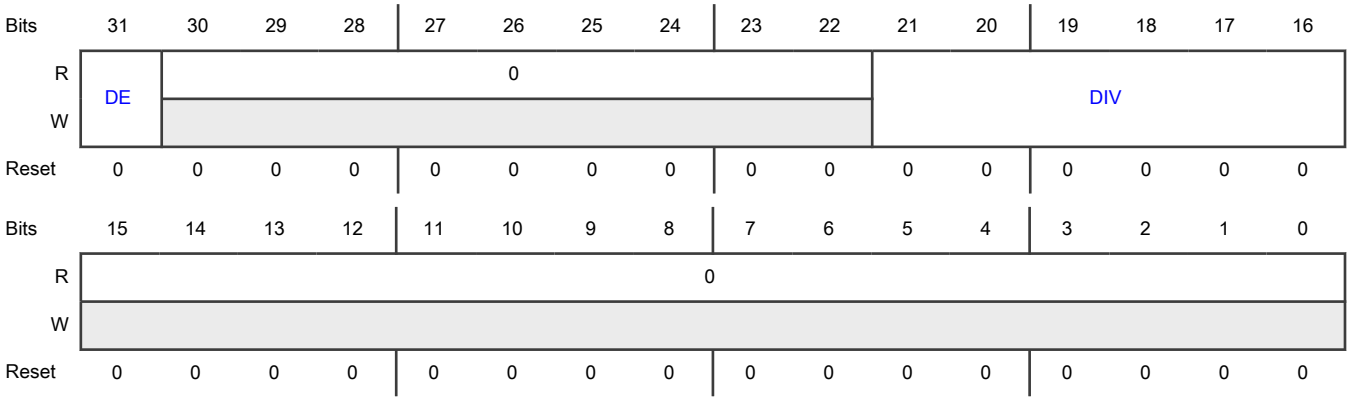
This register controls the clock divider 0 for clock mux 15.

This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram



Fields

Field	Function
31 DE	Divider enable 0b - Divider is disabled. 1b - Divider is enabled.
30-22 —	This field is reserved and reads return zeros.
21-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-0 —	This field is reserved and reads return zeros.

25.5.70 Clock Mux 15 Divider Update Status Register (MUX_15_DIV_UPD_STAT)

Offset

Register	Offset
MUX_15_DIV_UPD_STAT	6FCh

Function

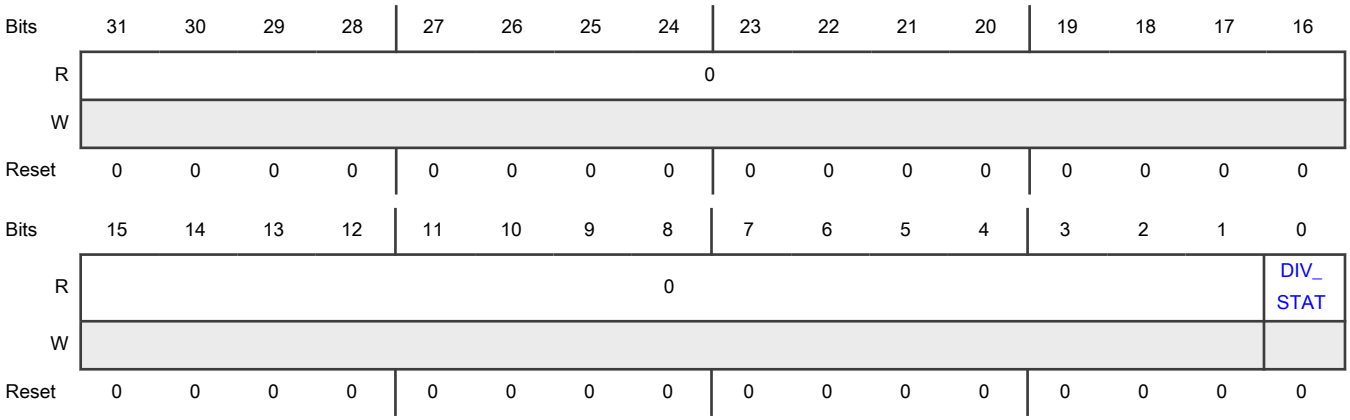
This register provides the update status of the clock dividers corresponding to clock mux 15. When a write operation on any divider control register is performed, the divider status bit in this register is set to logic-1. The bit is set to logic-0 when the divider has sampled the new divider configuration. Performing multiple writes without tracking the status bit on same or other

clock dividers inside the same clock mux leads to inconsistent reporting, that is, the divider status maybe be set to logic-0 when the corresponding divider update is pending.

NOTE

Read accesses to MUX_n_DIV_UPD_STAT always complete without returning bus transfer error independent of whether any divider(s) are implemented inside MC_CGM clock mux.

Diagram



Fields

Field	Function
31-1 —	This field is reserved and reads return zeros.
0 DIV_STAT	Divider status for clock mux 15 On reading MUX_n_DIV_UPD_STAT after updating a divider control register, if the value of this field is fixed to 1 because of an error in the selected clock source, perform the following steps to switch the mux to a new clock source: 1. Switch the mux to a working clock source without polling this field. 2. Update MUX_n_DC_m and poll this field. **NOTE** This field clears once divider configuration is updated or on destructive reset. If functional reset comes when this field is 1 then it can remain fixed to 1 until divider input clock is restored. 0b - No divider configuration update is pending. 1b - Divider configuration update on at least one divider associated with this multiplexer is pending.

25.5.71 Clock Mux 16 Select Control Register (MUX_16_CSC)

Offset

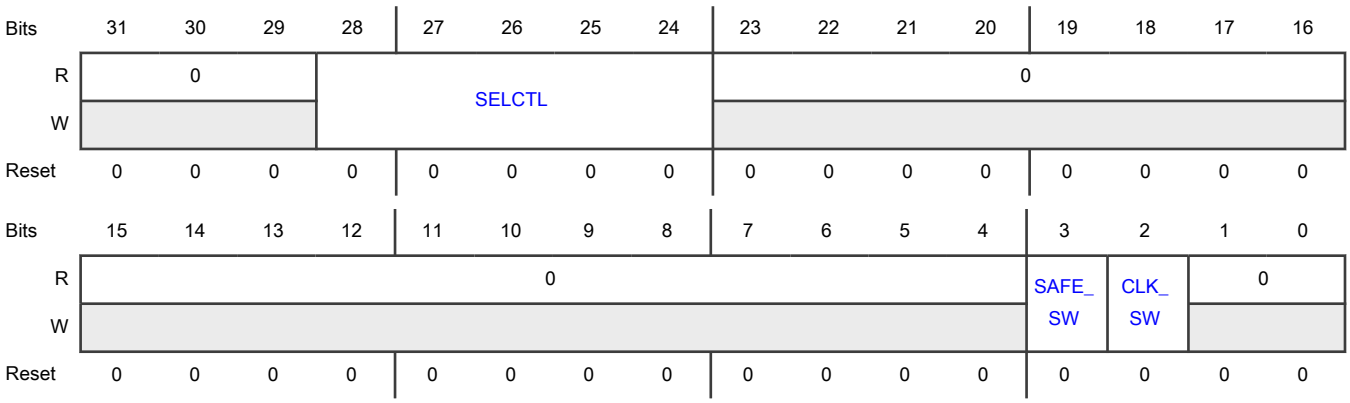
Register	Offset
MUX_16_CSC	700h

Function

This register provides the clock source selection control for clock mux 16. Clock mux 16 implements hardware control clock switching ensuring that the clock switch happens in a graceful manner (without glitches). See the "Hardware-controlled clock multiplexer" section for details.

This register is reset on destructive reset only.

Diagram



Fields

Field	Function
31-29 —	This field is reserved and reads return zeros.
28-24 SELCTL	Clock source selection control Selects the source clock for clock mux 16. The reserved values are not displayed. 0_0000b - FIRC 0_1100b - PLL_AUX_PHI0_CLK 1_1110b - GMAC1_MII_RMII_RGMII_TX_CLK
23-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Writing 1 to this bit makes a safe clock switch request to FIRC. After a safe clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
2 CLK_SW	Clock switch Writing 1 to this bit makes a clock switch request to clock mux 16. After a clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
1-0 —	This field is reserved and reads return zeros.

25.5.72 Clock Mux 16 Select Status Register (MUX_16_CSS)

Offset

Register	Offset
MUX_16_CSS	704h

Function

This register provides the current clock source selection status for clock mux 16.

This register is reset on destructive reset only.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				SELSTAT				0				SWTRG			SWIP
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SAFE_SW	CLK_SW	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	This field is reserved and reads return zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
28-24 SELSTAT	Clock source selection status This value indicates the current source selected for clock mux 16. The reserved values are not displayed. 0_0000b - FIRC 0_1100b - PLL_AUX_PHI0_CLK 1_1110b - GMAC1_MII_RMII_RGMII_TX_CLK
23-20 —	This field is reserved and reads return zeros.
19-17 SWTRG	Switch trigger cause This value indicates the cause for the latest clock source switch. **NOTE** If the clock fails, followed by multiple safe clock switch requests for MC_CGM hardware clock mux, the value of the SWTRG field can be either 4 or 5. 000b - Reserved 001b - Switch after request succeeded. 010b - Switch after the request failed because of an inactive target clock and the current clock is FIRC. 011b - Switch after the request failed because of an inactive current clock and the current clock is FIRC. 100b - Switch to FIRC because of a safe clock request or reset succeeded. 101b - Switch to FIRC because of a safe clock request or reset succeeded, but the previous current clock source was inactive. 110b - Reserved 111b - Reserved
16 SWIP	Switch in progress **NOTE** New clock switch request can only be given three clock cycles after the completion of the previous request. 0b - Clock source switching is complete. 1b - Clock source switching is in progress.
15-4 —	This field is reserved and reads return zeros.
3	Safe clock request

Table continues on the next page...

Table continued from the previous page...

Field	Function
SAFE_SW	This field provides an indication of whether a switch to safe clock operation was requested during the previous/ongoing request on clock mux 16. 0b - No safe clock switch operation was requested. 1b - Safe clock switch operation was requested.
2 CLK_SW	Clock switch This field provides an indication of whether a clock switch operation was requested during the previous/ ongoing request on clock mux 16. 0b - No clock switch operation was requested. 1b - Clock switch operation was requested.
1-0 —	This field is reserved and reads return zeros.

25.5.73 Clock Mux 16 Divider 0 Control Register (MUX_16_DC_0)

Offset

Register	Offset
MUX_16_DC_0	708h

Function

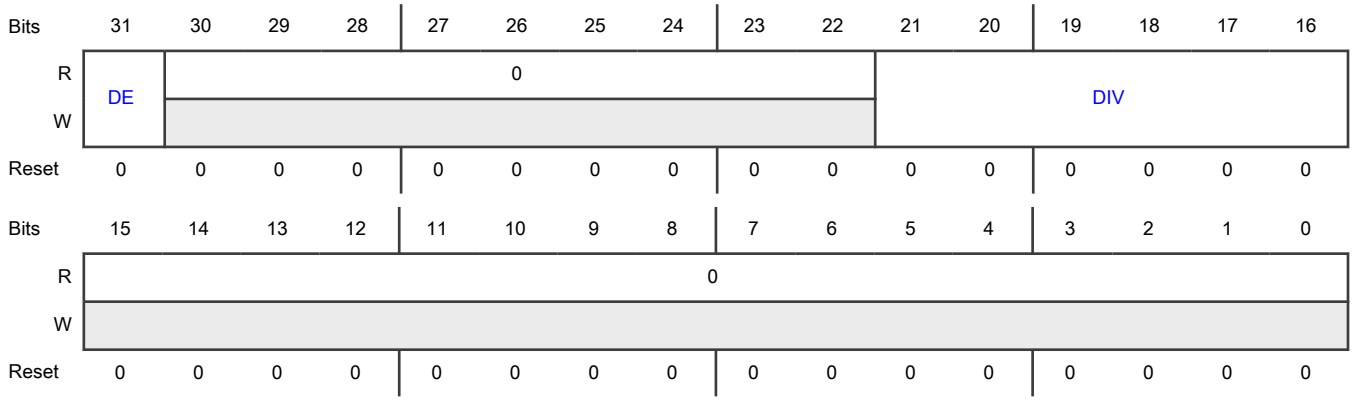
This register controls the clock divider 0 for clock mux 16.

This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram



Fields

Field	Function
31 DE	Divider enable 0b - Divider is disabled. 1b - Divider is enabled.
30-22 —	This field is reserved and reads return zeros.
21-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-0 —	This field is reserved and reads return zeros.

25.5.74 Clock Mux 16 Divider Update Status Register (MUX_16_DIV_UPD_STAT)

Offset

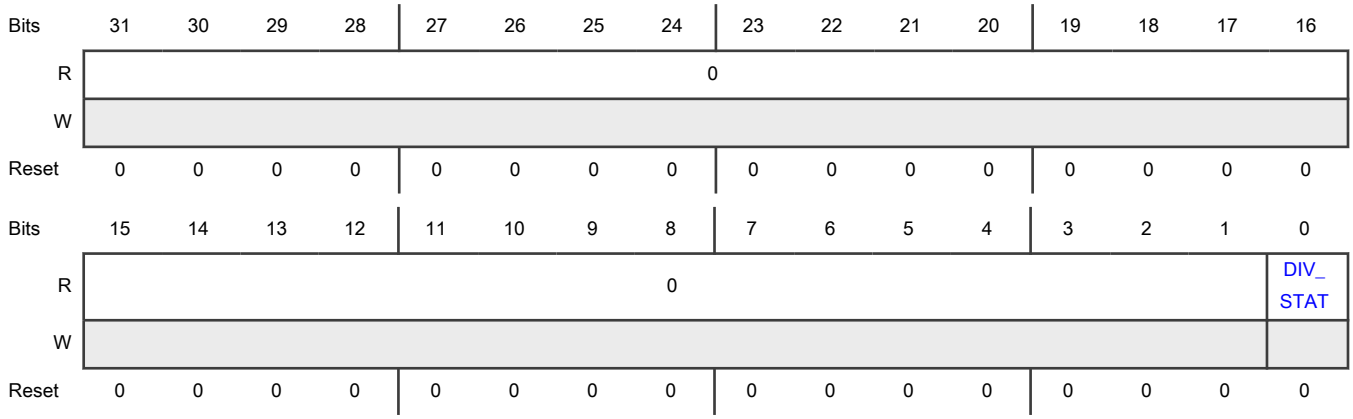
Register	Offset
MUX_16_DIV_UPD_STAT	73Ch

Function
This register provides the update status of the clock dividers corresponding to clock mux 16. When a write operation on any divider control register is performed, the divider status bit in this register is set to logic-1. The bit is set to logic-0 when the divider has sampled the new divider configuration. Performing multiple writes without tracking the status bit on same or other clock dividers inside the same clock mux leads to inconsistent reporting, that is, the divider status maybe be set to logic-0 when the corresponding divider update is pending.

NOTE

Read accesses to MUX_n_DIV_UPD_STAT always complete without returning bus transfer error independent of whether any divider(s) are implemented inside MC_CGM clock mux.

Diagram



Fields

Field	Function
31-1 —	This field is reserved and reads return zeros.
0 DIV_STAT	Divider status for clock mux 16 On reading MUX_n_DIV_UPD_STAT after updating a divider control register, if the value of this field is fixed to 1 because of an error in the selected clock source, perform the following steps to switch the mux to a new clock source: 1. Switch the mux to a working clock source without polling this field. 2. Update MUX_n_DC_m and poll this field. **NOTE** This field clears once divider configuration is updated or on destructive reset. If functional reset comes when this field is 1 then it can remain fixed to 1 until divider input clock is restored. 0b - No divider configuration update is pending. 1b - Divider configuration update on at least one divider associated with this multiplexer is pending.

25.5.75 Clock Mux 18 Select Control Register (MUX_18_CSC)

Offset

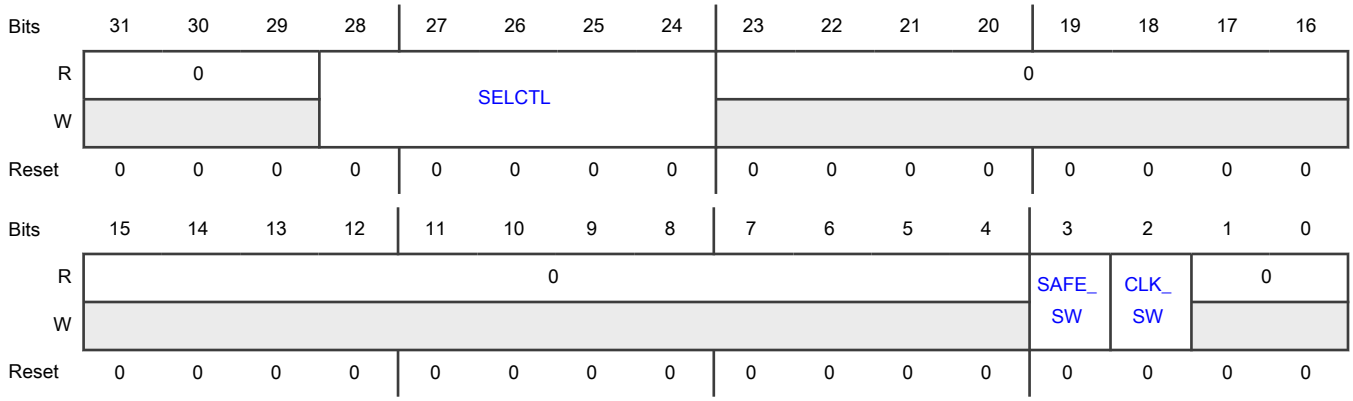
Register	Offset
MUX_18_CSC	780h

Function

This register provides the clock source selection control for clock mux 18. Clock mux 18 implements hardware control clock switching ensuring that the clock switch happens in a graceful manner (without glitches). See the "Hardware-controlled clock multiplexer" section for details.

This register is reset on destructive reset only.

Diagram



Fields

Field	Function
31-29 —	This field is reserved and reads return zeros.
28-24 SELCTL	Clock source selection control Selects the source clock for clock mux 18. The reserved values are not displayed. 0_0000b - FIRC 0_0010b - FXOSC 1_0110b - AIPS_PLAT_CLK
23-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request Writing 1 to this bit makes a safe clock switch request to FIRC. After a safe clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
2 CLK_SW	Clock switch Writing 1 to this bit makes a clock switch request to clock mux 18. After a clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
1-0 —	This field is reserved and reads return zeros.

25.5.76 Clock Mux 18 Select Status Register (MUX_18_CSS)

Offset

Register	Offset
MUX_18_CSS	784h

Function

This register provides the current clock source selection status for clock mux 18.

This register is reset on destructive reset only.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				SELSTAT				0				SWTRG			SWIP
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SAFE_SW	CLK_SW	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	This field is reserved and reads return zeros.
28-24 SELSTAT	Clock source selection status This value indicates the current source selected for clock mux 18. The reserved values are not displayed. 0_0000b - FIRC 0_0010b - FXOSC 1_0110b - AIPS_PLAT_CLK
23-20 —	This field is reserved and reads return zeros.
19-17 SWTRG	Switch trigger cause This value indicates the cause for the latest clock source switch.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE If the clock fails, followed by multiple safe clock switch requests for MC_CGM hardware clock mux, the value of the SWTRG field can be either 4 or 5. 000b - Reserved 001b - Switch after request succeeded. 010b - Switch after the request failed because of an inactive target clock and the current clock is FIRC. 011b - Switch after the request failed because of an inactive current clock and the current clock is FIRC. 100b - Switch to FIRC because of a safe clock request or reset succeeded. 101b - Switch to FIRC because of a safe clock request or reset succeeded, but the previous current clock source was inactive. 110b - Reserved 111b - Reserved
16 SWIP	Switch in progress **NOTE** New clock switch request can only be given three clock cycles after the completion of the previous request. 0b - Clock source switching is complete. 1b - Clock source switching is in progress.
15-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request This field provides an indication of whether a switch to safe clock operation was requested during the previous/ongoing request on clock mux 18. 0b - No safe clock switch operation was requested. 1b - Safe clock switch operation was requested.
2 CLK_SW	Clock switch This field provides an indication of whether a clock switch operation was requested during the previous/ongoing request on clock mux 18. 0b - No clock switch operation was requested. 1b - Clock switch operation was requested.
1-0 —	This field is reserved and reads return zeros.

25.5.77 Clock Mux 18 Divider 0 Control Register (MUX_18_DC_0)

Offset

Register	Offset
MUX_18_DC_0	788h

Function

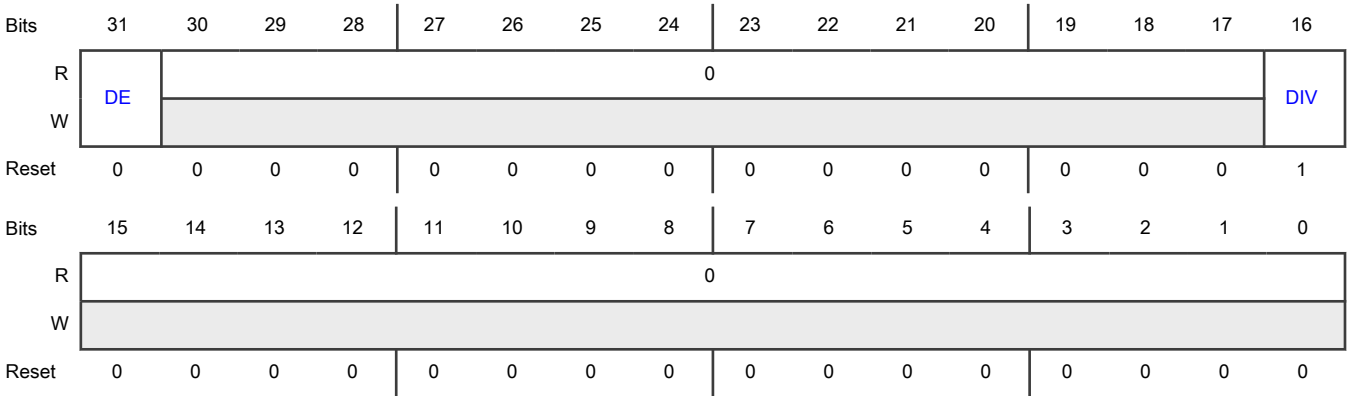
This register controls the clock divider 0 for clock mux 18.

This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram



Fields

Field	Function
31 DE	Divider enable 0b - Divider is disabled. 1b - Divider is enabled.
30-17 —	This field is reserved and reads return zeros.
16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-0 —	This field is reserved and reads return zeros.

25.5.78 Clock Mux 18 Divider Update Status Register (MUX_18_DIV_UPD_STAT)

Offset

Register	Offset
MUX_18_DIV_UPD_STAT	7BCh

Function

This register provides the update status of the clock dividers corresponding to clock mux 18. When a write operation on any divider control register is performed, the divider status bit in this register is set to logic-1. The bit is set to logic-0 when the divider has sampled the new divider configuration. Performing multiple writes without tracking the status bit on same or other clock dividers inside the same clock mux leads to inconsistent reporting, that is, the divider status maybe be set to logic-0 when the corresponding divider update is pending.

NOTE

Read accesses to MUX_n_DIV_UPD_STAT always complete without returning bus transfer error independent of whether any divider(s) are implemented inside MC_CGM clock mux.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DIV_STAT
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-1 —	This field is reserved and reads return zeros.
0 DIV_STAT	Divider status for clock mux 18 On reading MUX_n_DIV_UPD_STAT after updating a divider control register, if the value of this field is fixed to 1 because of an error in the selected clock source, perform the following steps to switch the mux to a new clock source: 1. Switch the mux to a working clock source without polling this field. 2. Update MUX_n_DC_m and poll this field.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field clears once divider configuration is updated or on destructive reset. If functional reset comes when this field is 1 then it can remain fixed to 1 until divider input clock is restored. 0b - No divider configuration update is pending. 1b - Divider configuration update on at least one divider associated with this multiplexer is pending.

25.5.79 Clock Mux 19 Select Control Register (MUX_19_CSC)

Offset

Register	Offset
MUX_19_CSC	7C0h

Function

This register provides the clock source selection control for clock mux 19. Clock mux 19 implements hardware control clock switching ensuring that the clock switch happens in a graceful manner (without glitches). See the "Hardware-controlled clock multiplexer" section for details.

This register is reset on destructive reset only.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				SELCTL				0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SAFE_SW	CLK_SW	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28	This field is reserved and reads return zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
27-24 SELCTL	Clock source selection control Selects the source clock for clock mux 19. The reserved values are not displayed. 0000b - FIRC 1101b - PLL_AUX_PHI1_CLK
23-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request Writing 1 to this bit makes a safe clock switch request to FIRC. After a safe clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
2 CLK_SW	Clock switch Writing 1 to this bit makes a clock switch request to clock mux 19. After a clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
1-0 —	This field is reserved and reads return zeros.

25.5.80 Clock Mux 19 Select Status Register (MUX_19_CSS)

Offset

Register	Offset
MUX_19_CSS	7C4h

Function

This register provides the current clock source selection status for clock mux 19.

This register is reset on destructive reset only.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				SELSTAT				0				SWTRG			SWIP
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SAFE_SW	CLK_SW	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	This field is reserved and reads return zeros.
27-24 SELSTAT	Clock source selection status This value indicates the current source selected for clock mux 19. The reserved values are not displayed. 0000b - FIRC 1101b - PLL_AUX_PHI1_CLK
23-20 —	This field is reserved and reads return zeros.
19-17 SWTRG	Switch trigger cause This value indicates the cause for the latest clock source switch. **NOTE** If the clock fails, followed by multiple safe clock switch requests for MC_CGM hardware clock mux, the value of the SWTRG field can be either 4 or 5. 000b - Reserved 001b - Switch after request succeeded. 010b - Switch after the request failed because of an inactive target clock and the current clock is FIRC. 011b - Switch after the request failed because of an inactive current clock and the current clock is FIRC. 100b - Switch to FIRC because of a safe clock request or reset succeeded. 101b - Switch to FIRC because of a safe clock request or reset succeeded, but the previous current clock source was inactive.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	110b - Reserved 111b - Reserved
16 SWIP	Switch in progress **NOTE** New clock switch request can only be given three clock cycles after the completion of the previous request. 0b - Clock source switching is complete. 1b - Clock source switching is in progress.
15-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request This field provides an indication of whether a switch to safe clock operation was requested during the previous/ongoing request on clock mux 19. 0b - No safe clock switch operation was requested. 1b - Safe clock switch operation was requested.
2 CLK_SW	Clock switch This field provides an indication of whether a clock switch operation was requested during the previous/ongoing request on clock mux 19. 0b - No clock switch operation was requested. 1b - Clock switch operation was requested.
1-0 —	This field is reserved and reads return zeros.

25.5.81 Clock Mux 19 Divider 0 Control Register (MUX_19_DC_0)

Offset

Register	Offset
MUX_19_DC_0	7C8h

Function

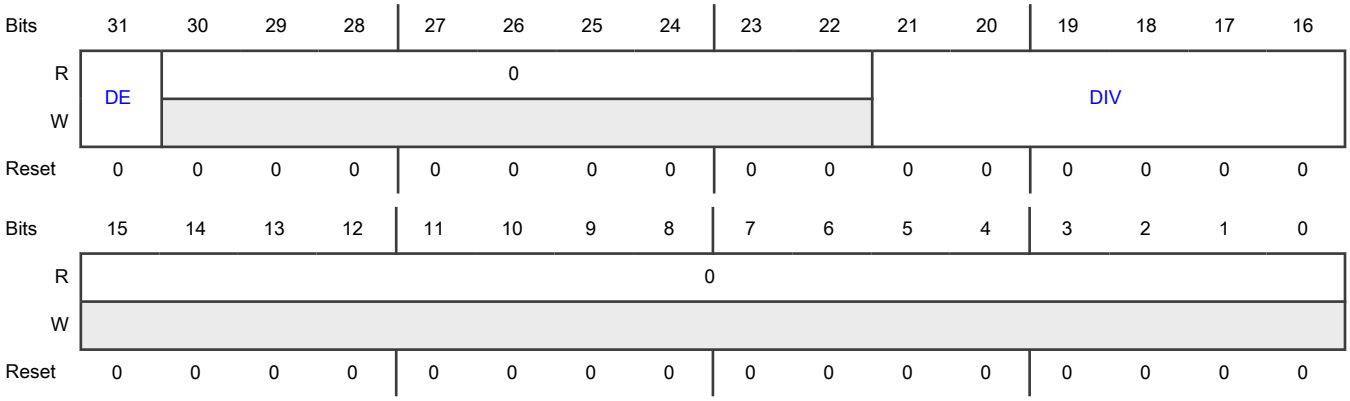
This register controls the clock divider 0 for clock mux 19.

This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram



Fields

Field	Function
31 DE	Divider enable 0b - Divider is disabled. 1b - Divider is enabled.
30-22 —	This field is reserved and reads return zeros.
21-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-0 —	This field is reserved and reads return zeros.

25.5.82 Clock Mux 19 Divider Update Status Register (MUX_19_DIV_UPD_STAT)

Offset

Register	Offset
MUX_19_DIV_UPD_STAT	7FCh

Function

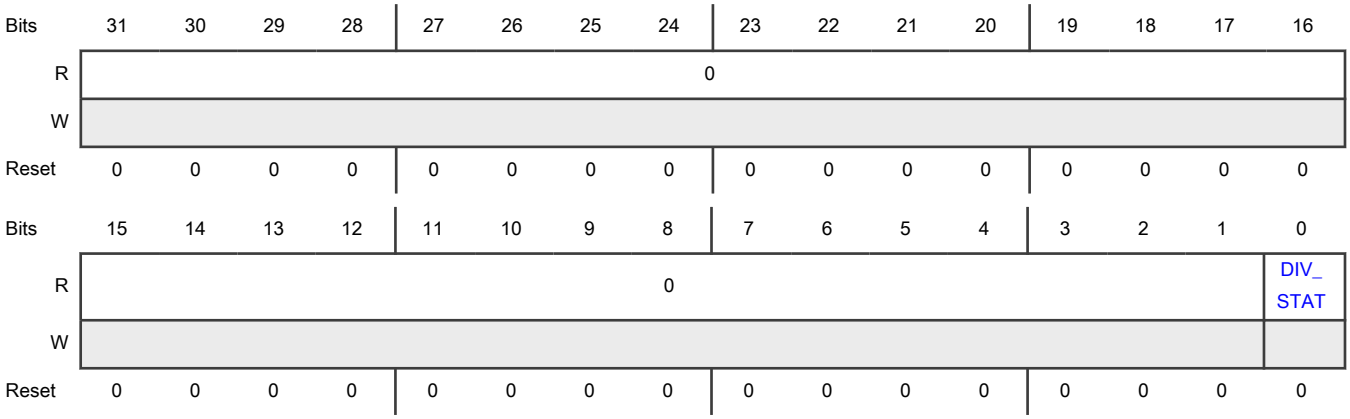
This register provides the update status of the clock dividers corresponding to clock mux 19. When a write operation on any divider control register is performed, the divider status bit in this register is set to logic-1. The bit is set to logic-0 when the divider has sampled the new divider configuration. Performing multiple writes without tracking the status bit on same or other

clock dividers inside the same clock mux leads to inconsistent reporting, that is, the divider status maybe be set to logic-0 when the corresponding divider update is pending.

NOTE

Read accesses to MUX_n_DIV_UPD_STAT always complete without returning bus transfer error independent of whether any divider(s) are implemented inside MC_CGM clock mux.

Diagram



Fields

Field	Function
31-1 —	This field is reserved and reads return zeros.
0 DIV_STAT	Divider status for clock mux 19 On reading MUX_n_DIV_UPD_STAT after updating a divider control register, if the value of this field is fixed to 1 because of an error in the selected clock source, perform the following steps to switch the mux to a new clock source: 1. Switch the mux to a working clock source without polling this field. 2. Update MUX_n_DC_m and poll this field. **NOTE** This field clears once divider configuration is updated or on destructive reset. If functional reset comes when this field is 1 then it can remain fixed to 1 until divider input clock is restored. 0b - No divider configuration update is pending. 1b - Divider configuration update on at least one divider associated with this multiplexer is pending.

25.5.83 Clock Mux 20 Select Control Register (MUX_20_CSC)

Offset

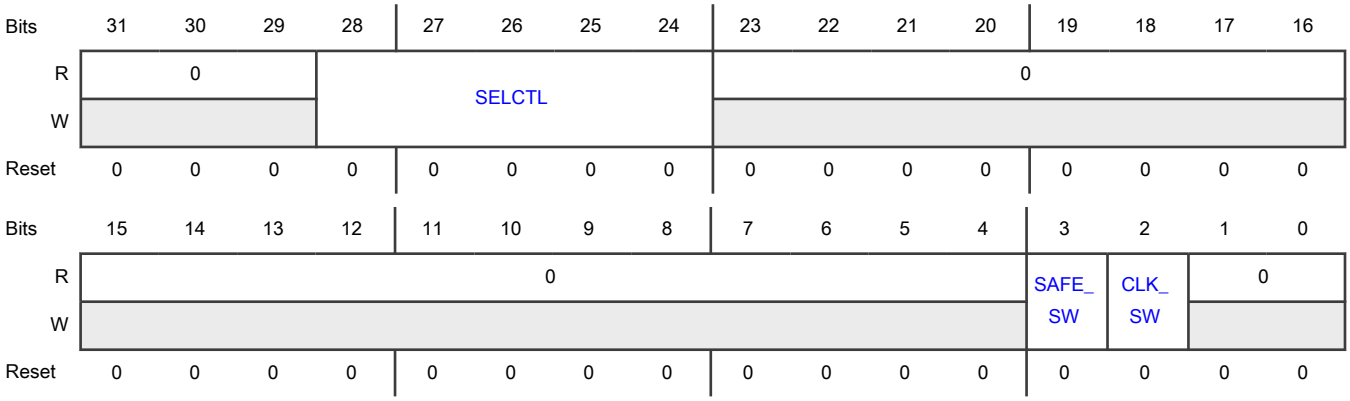
Register	Offset
MUX_20_CSC	800h

Function

This register provides the clock source selection control for clock mux 20. Clock mux 20 implements hardware control clock switching ensuring that the clock switch happens in a graceful manner (without glitches). See the "Hardware-controlled clock multiplexer" section for details.

This register is reset on destructive reset only.

Diagram



Fields

Field	Function
31-29 —	This field is reserved and reads return zeros.
28-24 SELCTL	Clock source selection control Selects the source clock for clock mux 20. The reserved values are not displayed. 0_0000b - FIRC 0_0010b - FXOSC 1_0110b - AIPS_PLAT_CLK
23-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Writing 1 to this bit makes a safe clock switch request to FIRC. After a safe clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
2 CLK_SW	Clock switch Writing 1 to this bit makes a clock switch request to clock mux 20. After a clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
1-0 —	This field is reserved and reads return zeros.

25.5.84 Clock Mux 20 Select Status Register (MUX_20_CSS)

Offset

Register	Offset
MUX_20_CSS	804h

Function

This register provides the current clock source selection status for clock mux 20.

This register is reset on destructive reset only.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				SELSTAT				0				SWTRG			SWIP
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SAFE_SW	CLK_SW	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	This field is reserved and reads return zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
28-24 SELSTAT	Clock source selection status This value indicates the current source selected for clock mux 20. The reserved values are not displayed. 0_0000b - FIRC 0_0010b - FXOSC 1_0110b - AIPS_PLAT_CLK
23-20 —	This field is reserved and reads return zeros.
19-17 SWTRG	Switch trigger cause This value indicates the cause for the latest clock source switch. **NOTE** If the clock fails, followed by multiple safe clock switch requests for MC_CGM hardware clock mux, the value of the SWTRG field can be either 4 or 5. 000b - Reserved 001b - Switch after request succeeded. 010b - Switch after the request failed because of an inactive target clock and the current clock is FIRC. 011b - Switch after the request failed because of an inactive current clock and the current clock is FIRC. 100b - Switch to FIRC because of a safe clock request or reset succeeded. 101b - Switch to FIRC because of a safe clock request or reset succeeded, but the previous current clock source was inactive. 110b - Reserved 111b - Reserved
16 SWIP	Switch in progress **NOTE** New clock switch request can only be given three clock cycles after the completion of the previous request. 0b - Clock source switching is complete. 1b - Clock source switching is in progress.
15-4 —	This field is reserved and reads return zeros.
3	Safe clock request

Table continues on the next page...

Table continued from the previous page...

Field	Function
SAFE_SW	This field provides an indication of whether a switch to safe clock operation was requested during the previous/ongoing request on clock mux 20. 0b - No safe clock switch operation was requested. 1b - Safe clock switch operation was requested.
2 CLK_SW	Clock switch This field provides an indication of whether a clock switch operation was requested during the previous/ ongoing request on clock mux 20. 0b - No clock switch operation was requested. 1b - Clock switch operation was requested.
1-0 —	This field is reserved and reads return zeros.

25.5.85 Clock Mux 20 Divider 0 Control Register (MUX_20_DC_0)

Offset

Register	Offset
MUX_20_DC_0	808h

Function

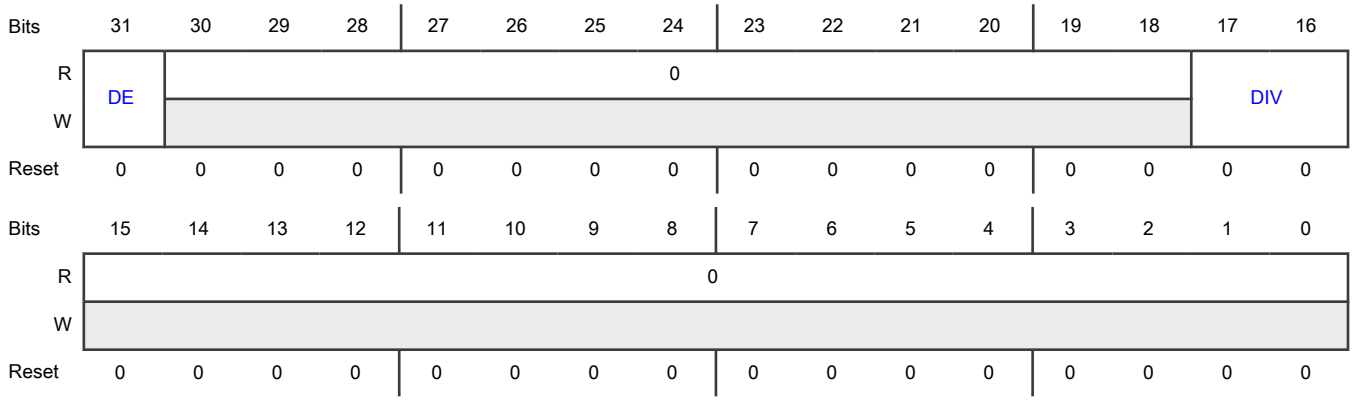
This register controls the clock divider 0 for clock mux 20.

This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram



Fields

Field	Function
31 DE	Divider enable 0b - Divider is disabled. 1b - Divider is enabled.
30-18 —	This field is reserved and reads return zeros.
17-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-0 —	This field is reserved and reads return zeros.

25.5.86 Clock Mux 20 Divider Update Status Register (MUX_20_DIV_UPD_STAT)

Offset

Register	Offset
MUX_20_DIV_UPD_STAT	83Ch

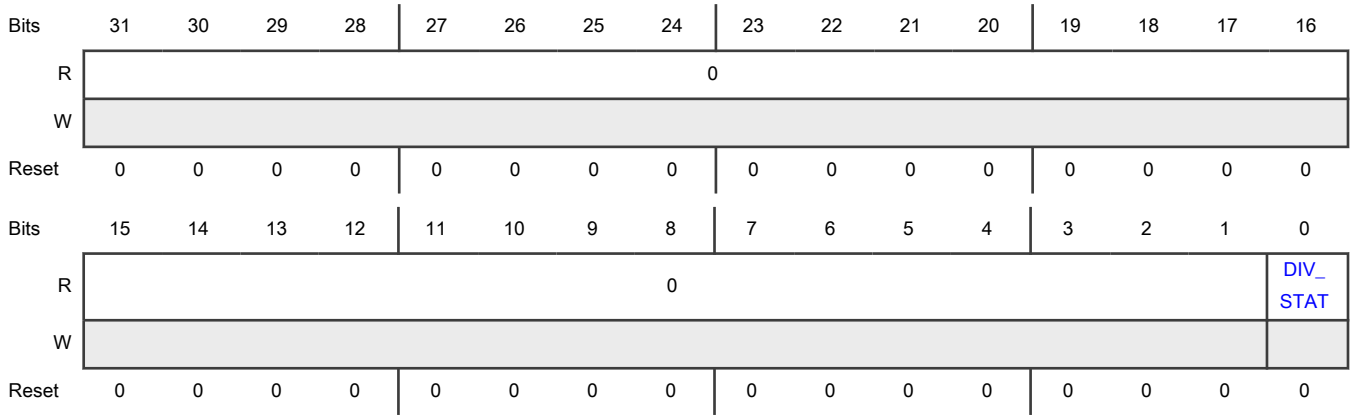
Function

This register provides the update status of the clock dividers corresponding to clock mux 20. When a write operation on any divider control register is performed, the divider status bit in this register is set to logic-1. The bit is set to logic-0 when the divider has sampled the new divider configuration. Performing multiple writes without tracking the status bit on same or other clock dividers inside the same clock mux leads to inconsistent reporting, that is, the divider status maybe be set to logic-0 when the corresponding divider update is pending.

NOTE

Read accesses to MUX_n_DIV_UPD_STAT always complete without returning bus transfer error independent of whether any divider(s) are implemented inside MC_CGM clock mux.

Diagram



Fields

Field	Function
31-1 —	This field is reserved and reads return zeros.
0 DIV_STAT	Divider status for clock mux 20 On reading MUX_n_DIV_UPD_STAT after updating a divider control register, if the value of this field is fixed to 1 because of an error in the selected clock source, perform the following steps to switch the mux to a new clock source: 1. Switch the mux to a working clock source without polling this field. 2. Update MUX_n_DC_m and poll this field. **NOTE** This field clears once divider configuration is updated or on destructive reset. If functional reset comes when this field is 1 then it can remain fixed to 1 until divider input clock is restored. 0b - No divider configuration update is pending. 1b - Divider configuration update on at least one divider associated with this multiplexer is pending.

25.6 Glossary

PCFS	Progressive clock frequency switching
LCM	Least common multiple

Chapter 26
Fast Internal RC Oscillator (FIRC)

26.1 Overview

The FIRC digital interface controls the internal 48 MHz RC oscillator system.

26.1.1 Features

FIRC can be disabled in Standby mode via software.

- Status register provides the current operating state:
  - On and stable
  - Off or on but not stable

26.2 External signals

This module has no external signals.

26.3 Initialization

This module does not require initialization.

26.4 FIRC register descriptions

26.4.1 FIRC memory map

FIRC base address: 402D_0000h

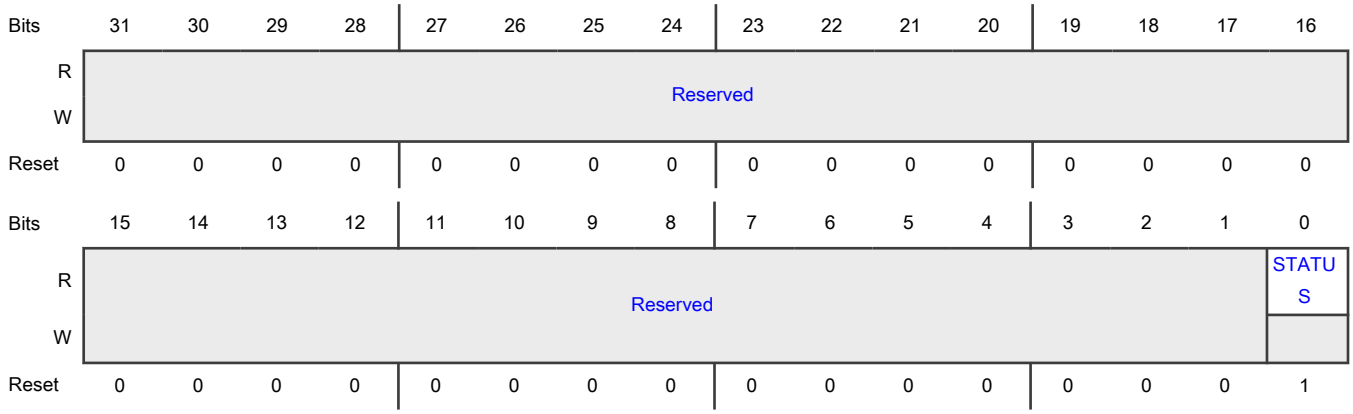
Offset	Register	Width (In bits)	Access	Reset value
4h	Status Register (Status_Register)	32	R	0000_0001h
8h	Standby Enable Register (STDBY_ENABLE)	32	RW	0000_0000h

26.4.2 Status Register (Status_Register)

Offset

Register	Offset
Status_Register	4h

Diagram



Fields

Field	Function
31-1 —	Reserved
0 STATUS	Status bit for FIRC 0b - FIRC is off or unstable. 1b - FIRC is on and stable.

26.4.3 Standby Enable Register (STDBY_ENABLE)

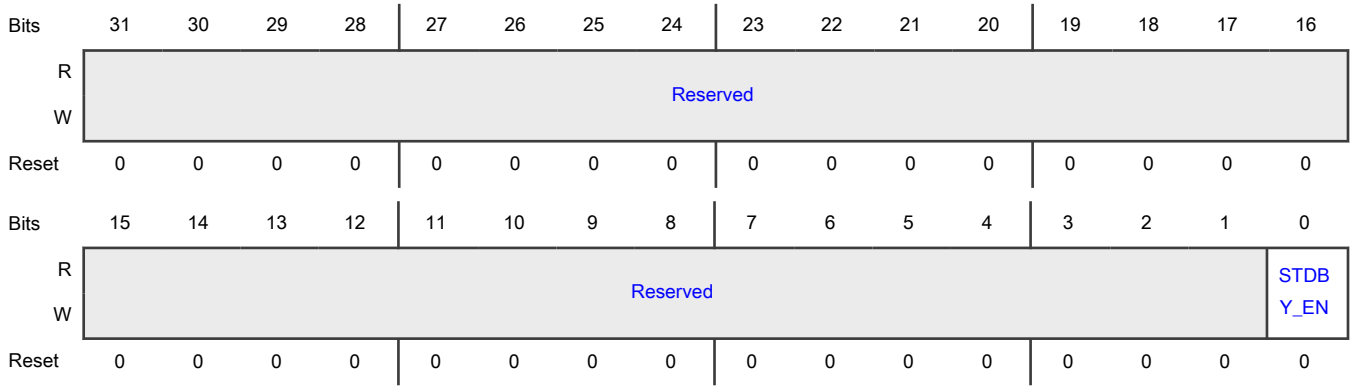
Offset

Register	Offset
STDBY_ENABLE	8h

Function

This register enables or disables FIRC in chip's Standby mode.

Diagram



Fields

Field	Function
31-1 —	RESERVED
0 STDBY_EN	Enables or disables FIRC in chip's Standby mode. 0b - Disabled 1b - Enabled

26.5 Glossary

Standby mode Power saving mode of the chip

Chapter 27
Slow Internal RC Oscillator (SIRC)

27.1 Overview

The SIRC digital interface controls the slow internal on-chip 32 KHz RC oscillator system.

27.1.1 Features

The SIRC module:

- Status register provides the current operating state:
  - On and stable
  - Off or on but not stable
- Operates at a frequency of 32 kHz in Functional mode

27.2 Operating mode

Only a POR reset will initialize the SIRC. Destructive or Functional resets do not impact the SIRC functionality.

SIRC stabilization occurs after 96 SIRC_CLK cycles.

The SIRC output clock remains invalid until the analog SIRC stabilizes. The output clock does not glitch or overshoot its frequency during enabling or disabling. Also, the clock does not get stuck or produce glitches on a very short hardware disable pulse.

27.3 External signals

This module has no external signals.

27.4 Initialization

This module does not require initialization.

27.5 SIRC register descriptions

27.5.1 SIRC memory map

Access to registers use 8-bit, 16-bit, or 32-bit addressing.

SIRC base address: 402C_8000h

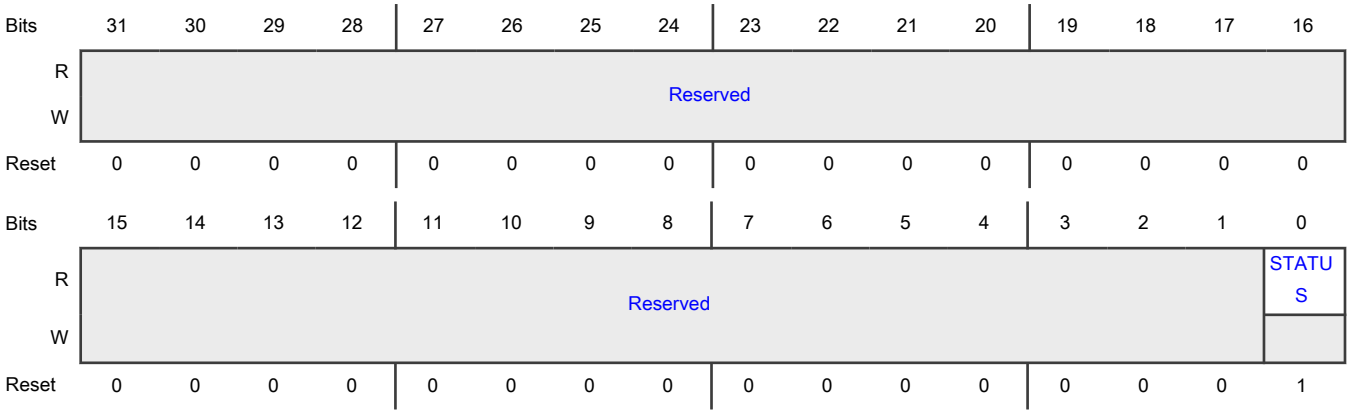
Offset	Register	Width (In bits)	Access	Reset value
4h	Status Register (SR)	32	R	0000_0001h
Ch	Miscellaneous input (MISCELLANEOUS_IN)	32	RW	0000_0000h

27.5.2 Status Register (SR)

Offset

Register	Offset
SR	4h

Diagram



Fields

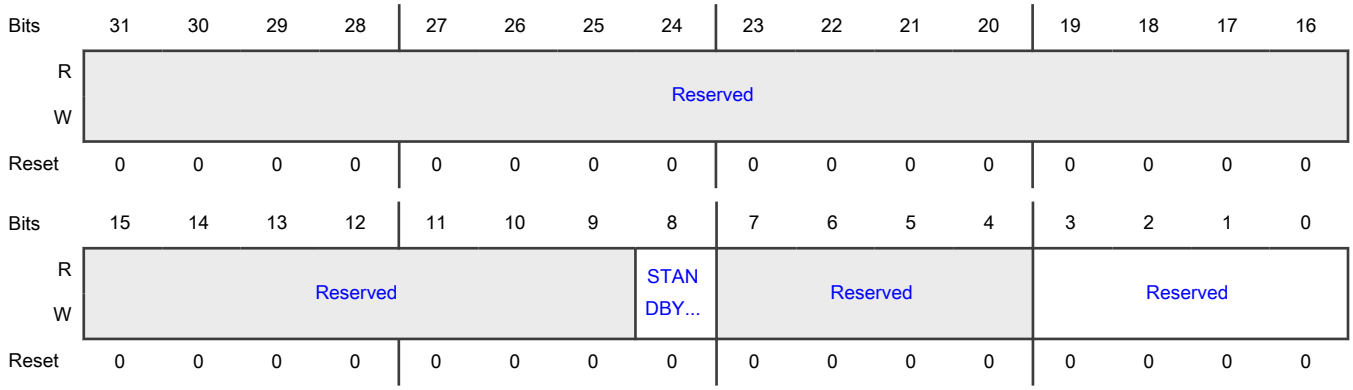
Field	Function
31-1 —	Reserved
0 STATUS	Status bit for SIRC 0b - SIRC is off or unstable 1b - SIRC is on and stable

27.5.3 Miscellaneous input (MISCELLANEOUS_IN)

Offset

Register	Offset
MISCELLANEOUS_IN	Ch

Diagram



Fields

Field	Function
31-9 —	Reserved
8 STANDBY_ENABLE	Standby Enable for SIRC 0b - SIRC disables in Standby mode 1b - SIRC enables in Standby mode
7-4 —	Reserved
3-0 —	Reserved

Chapter 28
Fast Crystal Oscillator Digital Controller (FXOSC)

28.1 Chip-specific FXOSC information

28.1.1 Chip-specific FXOSC information

For bypass mode applications, the EXTAL pin should be driven low when FXOSC is in off/disabled state.

- While initializing FXOSC: When the FXOSC is used in Bypass mode, the external clock source can only be enabled after the FXOSC is enabled.
- While disabling FXOSC: When the FXOSC is used in Bypass mode, the external clock source must already be inactive before disabling the FXOSC.

28.2 Overview

The Fast Crystal Oscillator (FXOSC) generates a clock which can be used at the SoC level. The FXOSC has a digital interface to control and configure the oscillator. When FXOSC is powered down at any time, it is designed not to generate any glitch at the output clock. A counter inside FXOSC handles different stabilization times. CG cell is clock gating cell, it gates the clock till stabilization time.

28.2.1 Block diagram

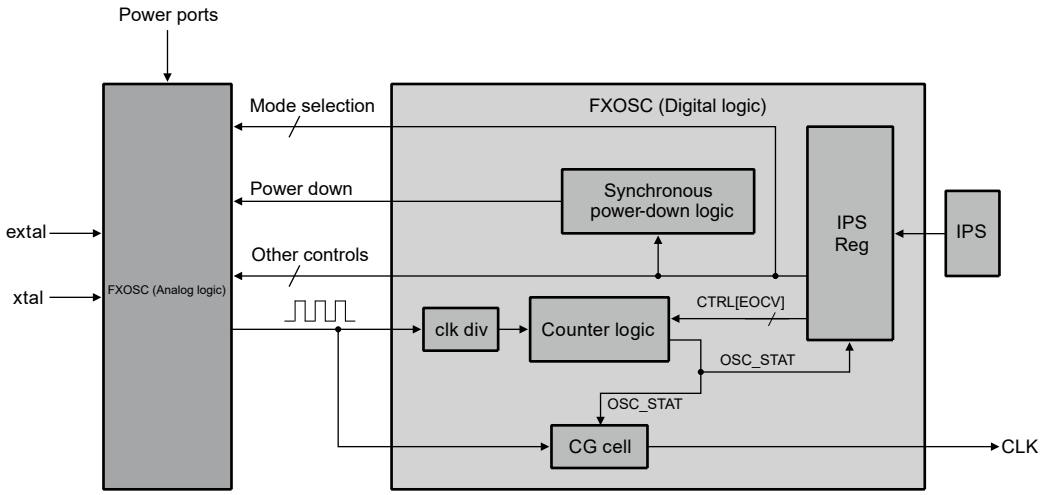


Figure 131. FXOSC block diagram

28.2.2 Features

FXOSC features are as follows:

- Status register shows current module state.
- Control register can:
  - Select a mode of operation:
    - Crystal mode
    - Single-Input Bypass mode using EXTAL clock input

— Disable the module (Power-Down mode)

28.3 Functional description

The table below shows configuration settings for the different FXOSC modes set by the CTRL register:

Table 177. FXOSC operation mode settings

Mode	CTRL[OSCON]	CTRL[OSC_BYP]	CTRL[COMP_EN]	FXOSC_CLK
Power-Down mode	0	X	X	0
Crystal mode	1	0	1	Crystal clock
Single-Input Bypass mode	1	1	0	EXTAL

Power-Down mode is the FXOSC default condition after any reset: POR, Destructive, or Functional.

28.3.1 Clock generation in Crystal mode

FXOSC_CLK start when counter value reaches FXOSC_CTRL[EOCV] × 128.

28.3.2 Clock generation in Single-Input Bypass mode

This mode bypasses the oscillator and uses a single-input external clock (EXTAL input) for FXOSC_CLK. When FXOSC enters Bypass mode:

- Enable the external clock after enabling FXOSC during FXOSC initialization.
- Disable the external clock beforehand when disabling FXOSC.

28.3.3 Clocking

This module has no clocking considerations.

28.3.4 Interrupts

This module has no interrupts.

28.4 External signals

This module has no external signals.

28.5 Initialization

Initializing FXOSC

Initialize FXOSC as follows:

1. Write the desired value to CTRL[OSC_BYP] and CTRL[COMP_EN] to select an operation mode as shown in Table 177.

   NOTE

   FXOSC must be disabled when the operation mode is modified.

2. Configure CTRL[GM_SEL].
   - In Crystal mode configure the transconductance based on the module specification in the chip data sheet.
   - In Single-Input Bypass mode write 0000b to this field.

NOTE

In Crystal mode FXOSC will not function with zero transconductance (GM_SEL = 0000b).

3. Set CTRL[EOCV] calculating the value as follows:
   - EOCV (in decimal) = (stabilization time in ns) ÷ (4×128×(period of clock in ns))
4. Write 1 to CRTL[OSCON] to enable FXOSC.

NOTE

When the FXOSC is used in Bypass mode, the clock from the external source can only be used after the FXOSC is enabled

5. Confirm the clock is stable (STAT[OSC_STAT] = 1) before using it.

NOTE

See Hardware design guide for further details and the recommended circuit for each mode.

Disabling FXOSC

Write 0 to CRTL[OSCON] to disable FXOSC when FXOSC_CLK is running and stable.

FXOSC enters Power-down mode after at least four crystal clocks. No glitches occur during the transition to Power-Down mode because synchronizers are used.

NOTE

After disabling FXOSC:

- Wait for at least 2µs before enabling FXOSC again.
- You must not change other values in FXOSC registers for at least 16 FXOSC_CLK cycles.

28.6 FXOSC register descriptions

This section provides the descriptions of all registers used for configuring the FXOSC.

28.6.1 FXOSC memory map

Use 8-bit, 16-bit, or 32-bit addressing to access registers.

FXOSC base address: 402D_4000h

Offset	Register	Width (In bits)	Access	Reset value
0h	FXOSC Control Register (CTRL)	32	RW	019D_00C0h
4h	Oscillator Status Register (STAT)	32	R	0000_0000h

28.6.2 FXOSC Control Register (CTRL)

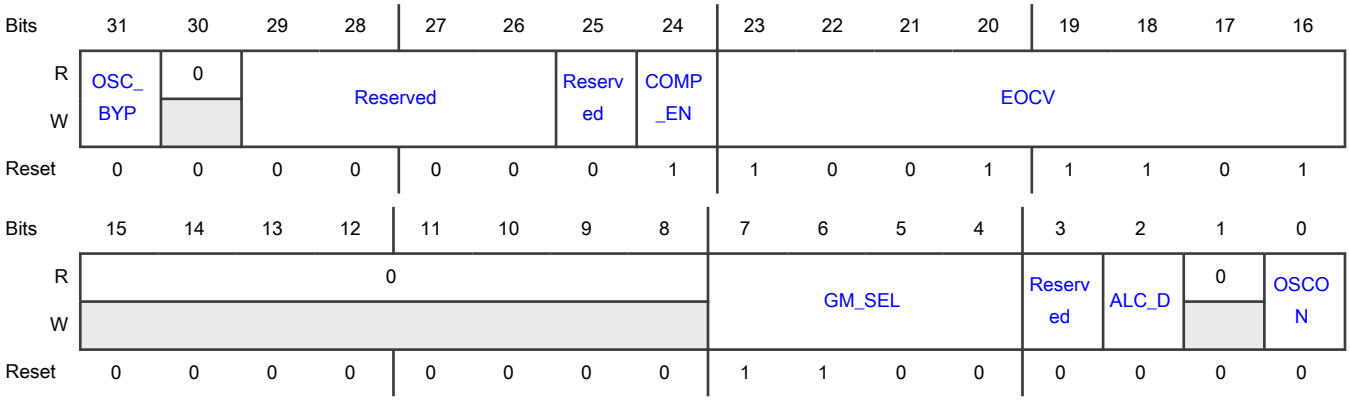
Offset

Register	Offset
CTRL	0h

Function

Configures FXOSC operation.

Diagram



Fields

Field	Function
31 OSC_BYP	Oscillator bypass Bypasses the internal oscillator. 0b - Internal oscillator not bypassed 1b - Internal oscillator bypassed
30 —	Reserved
29-26 —	Reserved
25 —	Reserved
24 COMP_EN	Comparator enable Enables or disables the comparator. • For Crystal mode set this field to 1. • For Single-Input Bypass mode set this field to 0. 0b - Comparator disabled 1b - Comparator enabled
23-16 EOCV	End of count value Specifies the end-of-count.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	The oscillator counter runs on the crystal clock divided by 4 and counts up to EOCV × 128. This counting period ensures that the external oscillator clock signal is stable before the system selects FXOSC as a source. **NOTE** • You must set EOCV to the appropriate value to allow clock and duty cycle to stabilize and guarantee that OSC_STAT becomes set within the crystal startup time. — In Crystal mode, EOCV value must to be calculated to appropriate value based on the crystal specification using the equation in Initializing FXOSC. — In Single-Input Bypass mode, EOCV value is irrelevant. FXOSC holds the counter in reset. • Before modifying EOCV, FXOSC must be disabled.
15-8 —	Reserved
7-4 GM_SEL	Crystal overdrive protection Selects the transconductance applied by the FXOSC amplifier. This setting depends on crystal specification. **NOTE** • In Crystal mode FXOSC will not function with zero transconductance (GM_SEL = 0000b). • For details on how to set this field, see Initializing FXOSC. 0000b - 0x 0001b - 0.1004x 0010b - 0.2009x 0011b - 0.3013x 0100b - 0.2343x 0101b - 0.3348x 0110b - 0.4345x 0111b - 0.5349x 1000b - 0.4679x 1001b - 0.5684x 1010b - 0.6681x 1011b - 0.7678x 1100b - 0.7016x 1101b - 0.8013x

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1110b - 0.9003x 1111b - 1x
3 —	Reserved
2 ALC_D	Automatic level controller enable The ALC feature of the FXOSC internal circuit automatically adjusts the bias current of the crystal oscillator's amplifier. It must be used in crystal oscillator mode. During startup the EXTAL/XTAL amplitude would be lower, so the ALC circuit dynamically increases the bias current of the internal amplifier to provide higher current and make crystal oscillator startup faster in normal oscillator mode. After EXTAL/XTAL amplitude has increased to a steady state, the ALC circuit automatically reduces the amplifier's bias current to save power. 0b - Enables automatic level controller 1b - Disables automatic level controller
1 —	Reserved
0 OSCON	Crystal oscillator power-down control Enables or disables FXOSC 0b - Disables FXOSC 1b - Enables FXOSC

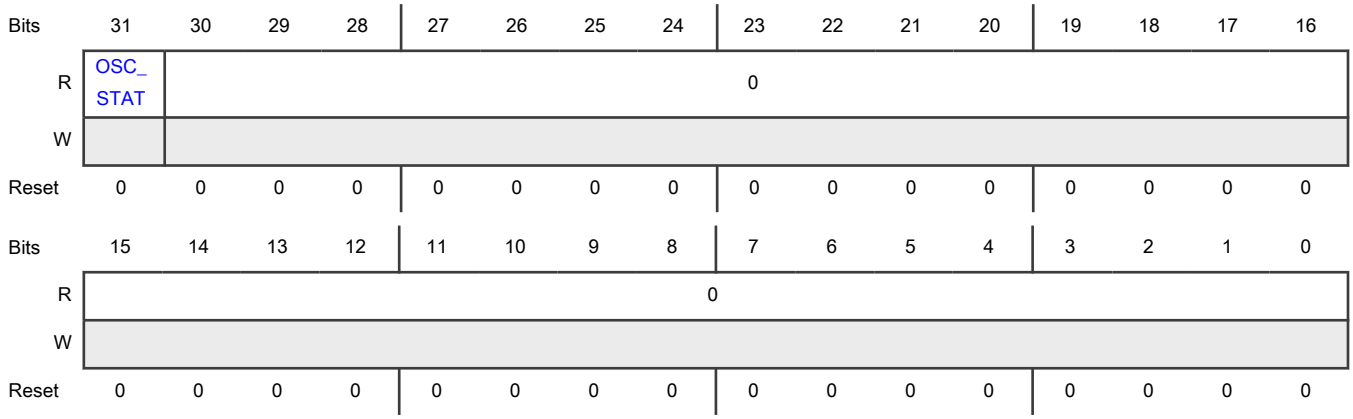
28.6.3 Oscillator Status Register (STAT)

Offset

Register	Offset
STAT	4h

Function
Shows current state of FXOSC.

Diagram



Fields

Field	Function
31 OSC_STAT	Crystal oscillator status Indicates the crystal oscillator status. **NOTE** OSC_STAT value is not valid if transconductance is set to 0. 0b - Crystal oscillator is off or on but not stable. 1b - Crystal oscillator is on and providing a stable clock.
30-0 —	Reserved

Chapter 29
Slow Crystal Oscillator Digital Controller (SXOSC)

29.1 Overview

The Slow crystal oscillator (SXOSC) generates a clock which can be used at the SoC level. The SXOSC has a digital interface to control and configure the oscillator. When SXOSC is powered down at any time, it is designed not to generate any glitch at the output clock. A counter inside SXOSC handles different stabilization times. CG cell is clock gating cell, it gates the clock till stabilization time.

29.1.1 Block diagram

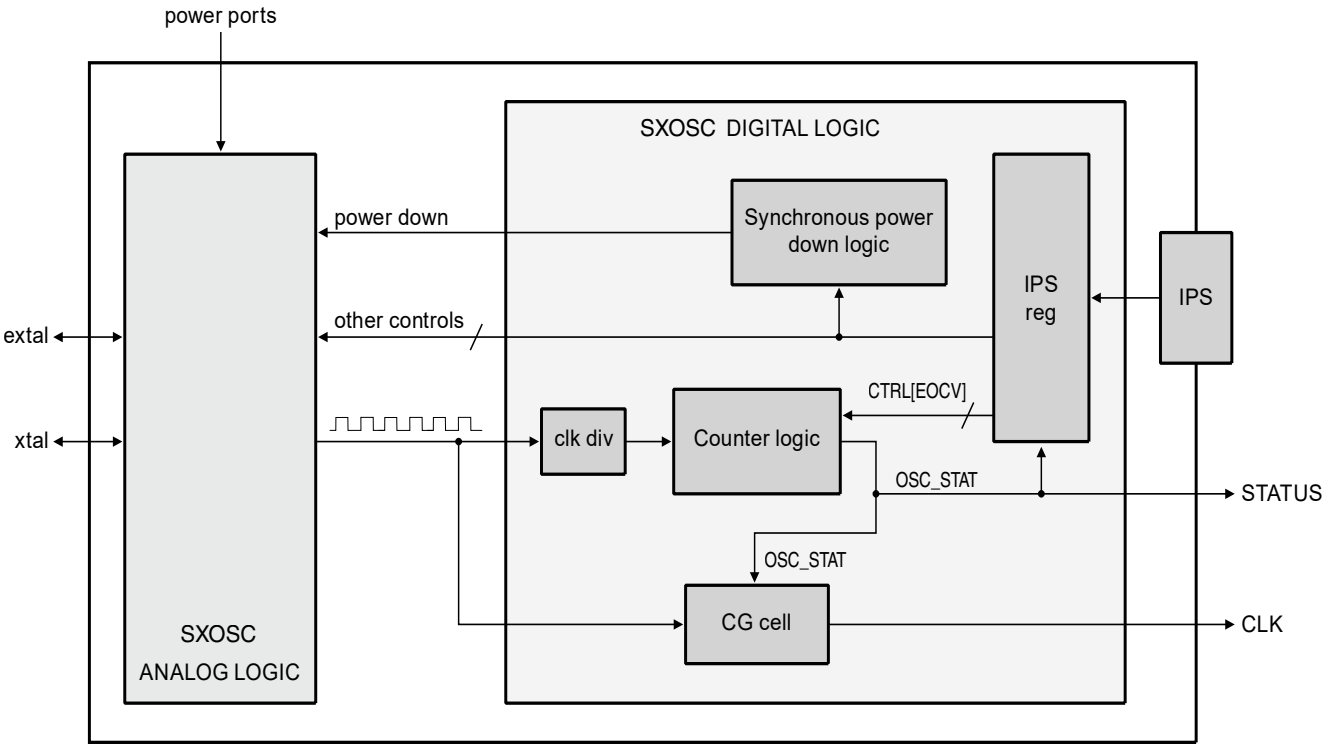


Figure 132. Block diagram

29.2 Features

- SXOSC generates a 32 KHz clock output in crystal mode
- SXOSC contains a status register, the value of which becomes 1 when the crystal stabilization time is complete
- SXOSC can be powered down through software bit.

29.3 Functional description

SXOSC generates control signals to configure the analog module to operate in specific modes.

The following table shows the mode of operation available for selection and its settings.

Table 178. Operation mode settings

Mode	Value of SXOSC_CTRL[OSCON]	Output clock
Functional Oscillator	0 (oscillator switched off)	0 (indicates no output)
	1 (oscillator switched on)	Crystal clock

29.3.1 Clock generation in crystal mode

After hard reset, the crystal oscillator is switched off by default. For clock generation in crystal mode, see Table 178. The counter logic starts counting and the stable clock starts running one clock cycle after reaching the value of SXOSC_CTRL[EOCV] x 128 counter value. The module writes 1 to SXOSC_STAT[OSC_STAT] after two module clock cycles.

29.3.2 Clock stopping in crystal mode

To stop a stable, running clock, configure the power down mode as specified in Table 178. A glitch does not occur because synchronizers are used.

29.3.1 Modes of operation

The SXOSC has following modes of operation.

29.3.1.1 Crystal mode

In this mode crystal is connected between extal and xtal ports, to select crystal mode see Table 178

29.3.1.2 Bypass mode

The bypass mode is handled outside the DA Wrapper.

NOTE

In this mode crystal is removed from extal and xtal ports and extal is driven by external clock and xtal is unconnected.

29.3.2 Clocking

This module has no clocking considerations.

29.3.3 Interrupts

This module has no interrupts.

29.4 External signals

This module has no external signals.

29.5 Initialization information

To enter into any mode the following sequences must be followed. By default IP is disabled.

- Power-down Mode:
  - When SXOSC is running in any mode, de-assert SXOSC_CTRL[OSCON]
- Crystal Mode:
  1. Disable the IP by de-asserting SXOSC_CTRL[OSCON] bit
  2. Connect the crystal between extal and xtal ports

3. Write an appropriate value to SXOSC_CTRL[EOCV]
4. Program recommended value in SXOSC_CTRL[GM_SEL]
5. Enable the IP by asserting SXOSC_CTRL[OSCON] bit
6. SXOSC_STAT[OSC_STAT] bit will be set after counter runs as per programming of SXOSC_CTRL[EOCV] and clock will be released to SoC

29.6 SXOSC register descriptions

This section provides the description of all registers for configuring the SXOSC.

29.6.1 SXOSC memory map

Addresses are given as offsets from the module base address. All registers can be accessed using 8-bit, 16-bit or 32-bit addressing.

NOTE

Some of the register reset values are specifically configured for each unique device by external configuration signals or parameters.

SXOSC base address: 402C_C000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Oscillator Control Register (SXOSC_CTRL)	32	RW	007D_0000h
4h	Oscillator Status Register (SXOSC_STAT)	32	R	0000_0000h

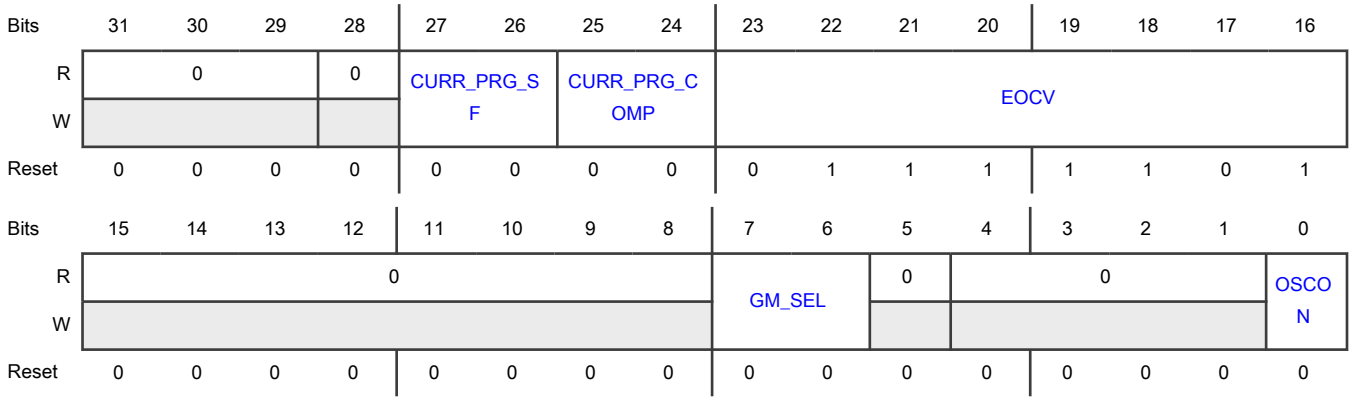
29.6.2 Oscillator Control Register (SXOSC_CTRL)

Offset

Register	Offset
SXOSC_CTRL	0h

Function
Oscillator Control Register

Diagram



Fields

Field	Function
31-29 —	Reserved
28 —	Reserved
27-26 CURR_PRG_SF	These bits specify programmability of level shifter current. 00b - 3x 01b - 2x 10b - 3.5x 11b - 4x
25-24 CURR_PRG_COMP	These bits specify programmability of comparator current. 00b - 1x 01b - 2x 10b - 3x 11b - 4x
23-16 EOCV	End of count value These bits specify the end of count value. This value is used by the oscillator Stabilization counter for comparison whenever it is switched On. This counting period ensures that the external oscillator clock signal is stable before it can be selected by the system. Oscillator counter runs on crystal clock divide by 4, and counts value upto EOCV * 128. **NOTE** In order to find the appropriate EOCV value, ensure that the internal counter is running for at least the stabilization time of the crystal as given in the Data Sheet.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE It is recommended to change the value of EOCV only when the IP is in disabled state.
15-8 —	Reserved
7-6 GM_SEL	Crystal overdrive protection This field setting decides the trans-conductance applied by SXOSC amplifier, and it will depend on crystal specification. 00b - 1x 01b - 1.25x 10b - 1.3x 11b - 1.6x
5 —	Reserved
4-1 —	Reserved
0 OSCON	Crystal oscillator power-down control **NOTE** When disabling the IP through software, program 0 to this bit-field, and ensure to not change any other values in the registers for at least 16 SXOSC clock cycles. 0b - Crystal oscillator is switched OFF 1b - Crystal oscillator is switched ON

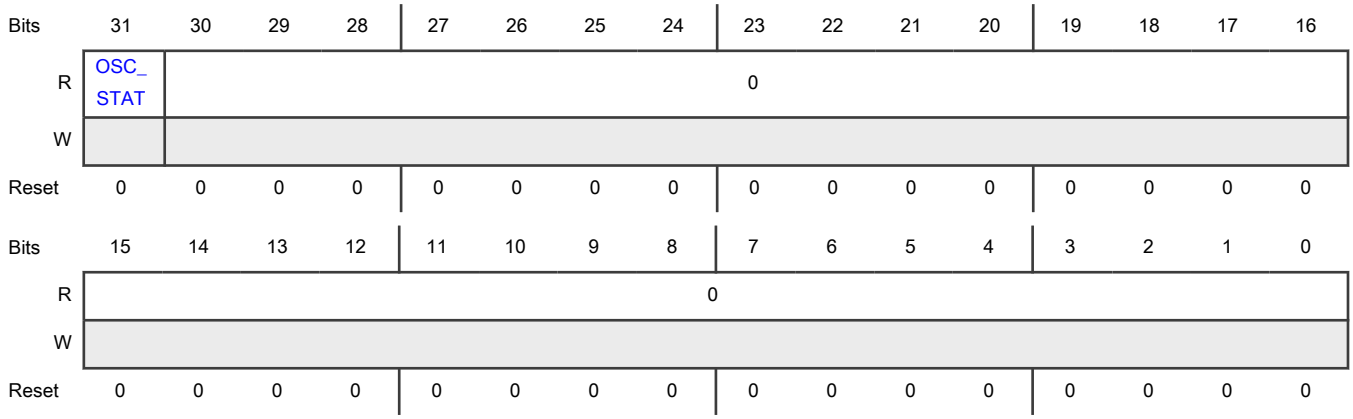
29.6.3 Oscillator Status Register (SXOSC_STAT)

Offset

Register	Offset
SXOSC_STAT	4h

Function
Oscillator Status Register

Diagram



Fields

Field	Function
31 OSC_STAT	Crystal oscillator status 0b - Crystal oscillator output clock is not stable. 1b - Crystal oscillator is providing a stable clock.
30-0 —	Reserved

Chapter 30
PLL Digital Interface (PLLDIG)

30.1 Chip-specific PLLDIG information

30.1.1 PLLDIG instances

This chip supports up to two instances of PLLDIG.

Table 179. PLLDIG instances

Instance	S32K388/S32K389/S32K358/S32K356/ S32K348/S32K338/S32K328	S32K310/S32K311/S32K312/ S32K314/S32K322/S32K324/S32K342/ S32K341/S32K344
PLL	Yes	Yes
PLL_AUX	Yes	No

Table 180. PLLDIG configuration

Instance	Frequency modulation supported	Number of reference clocks supported	Number of clock outputs supported
PLL	Yes	1[1]	2
PLL_AUX	No	1	2[2]

1. For S32K310 and S32K311: 2
2. For S32K328, S32K338, S32K348, S32K356, and S32K358: 3

30.1.2 PLL-supported accesses and frequencies

The PLLODIV_0 and PLLODIV_1 registers support only word accesses. When you write to these registers, you must retain the default values of the reserved fields.

PLLDIG supports a down-spread modulation of up to 500 MHz PLL PHI clock output only.

30.1.3 Register implementation

In S32K358, S32K356, S32K311 and S32K310 there are additional registers as compared to what is mentioned in section 'PLLDIG memory map'. See following table for details.

Table 181. Register details

Register/Bitfield	Offset	Availability
PLLCLKMUX[REFCLKSEL][1]	20h	Only available in S32K311 and S32K310
PLLODIV_2[2]	88h	Only available in S32K358, S32K356, S32K348, S32K338, and S32K328

1. See section 'PLLCLKMUX definition' for register definition
2. Implementation of this register is same as PLLODIV_0/1. See PLLODIV_0/1 in section 'PLLDIG memory map' for register definition

30.1.3.1 PLLCLKMUX definition

Register PLLCLKMUX (PLL Clock Multiplexer) is available at offset 20h. This register selects the PLL clock source. Bitfield definition of this register is as shown below:

Table 182. PLLCLKMUX Definition

Bitfield offset	Bitfield name	Bitfield description
0	REFCLKSEL	Reference Clock Select: Selects the PLL clock source. • 0b-FXOSC_CLK • 1b-FIRC_DIV2_CLK
1-31	Reserved	

30.1.4 Initialization information for S32K311 and S32K310

Perform the following steps to initialize PLL:

1. Confirm that PLLODIV_*n*[DE] is 0 for all dividers.
2. Confirm that PLLCR[PLLPD] is 1.
3. Program PLLCLKMUX to select the appropriate reference clock.
4. Program the following as needed:
   - PLLDV
   - PLLFD
   - PLLFM to the desired value
5. Program PLLDV[ODIV2] and PLLODIV_*n*[DIV] to the desired values.
6. Wait for the PLL reference clock to be stable.
7. Write 0 to PLLCR[PLLPD].
8. Wait for PLLSR[LOCK] to be 1.
9. Write 1 to PLLODIV_*n*[DE].

Perform the following steps to shut down PLL:

1. Write 0 to PLLODIV_n[DE] for all dividers.
2. Write 1 to PLLCR[PLLPD].

30.2 Overview

PLL can multiply or divide the frequency of a given clock input.

30.2.1 Block diagram

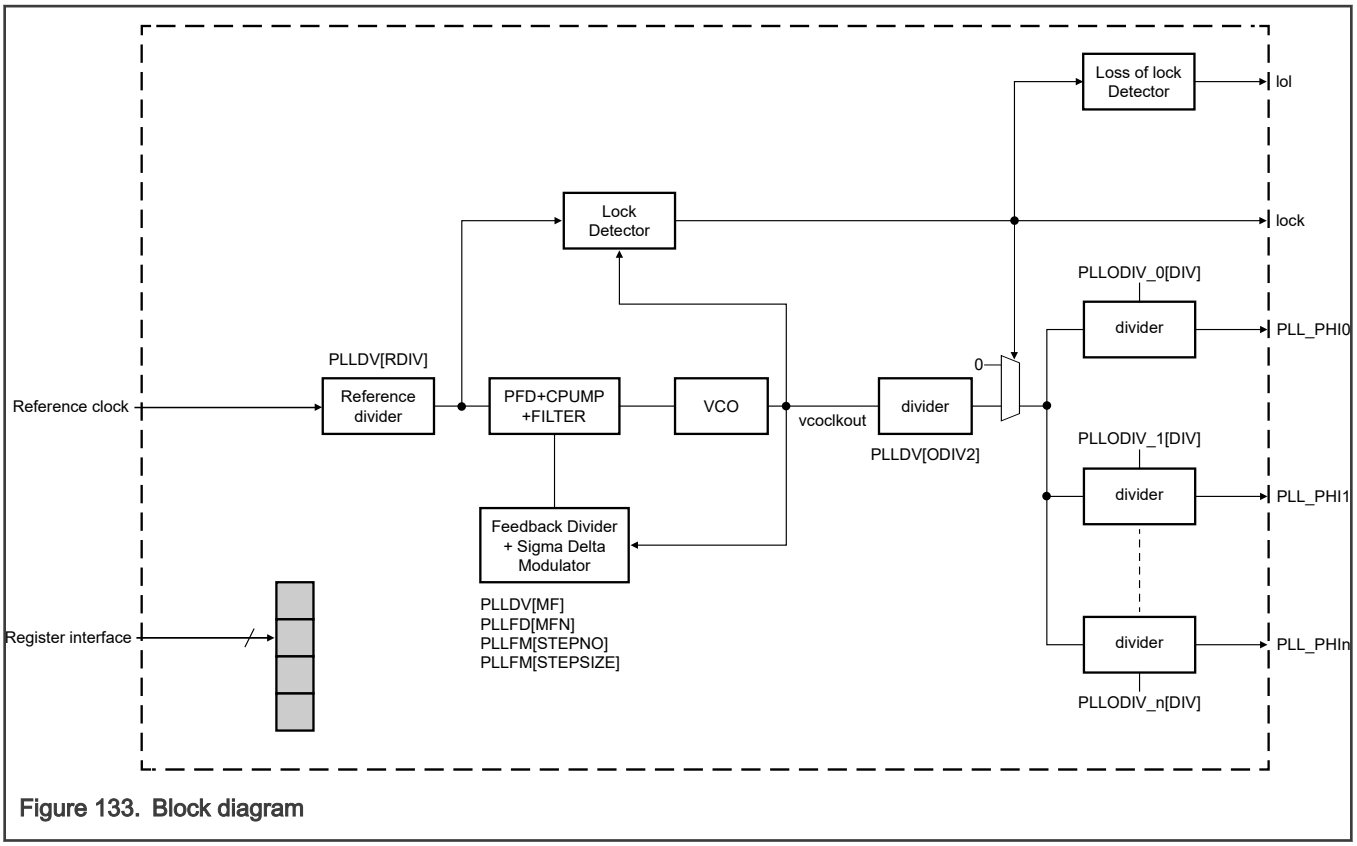


Figure 133. Block diagram

The number of output dividers can vary with the module instance. See the Clocking chapter to confirm the number of PLL output dividers.

30.2.2 Features

PLL includes the following features:

- Programmable frequency modulation
- Multiple integer dividers on PLL outputs
- Lock detection circuitry reports when PLL achieves frequency lock
- Continuous monitoring of lock status to report Loss of Lock (LOL) condition
- Powering down the module for low-power operation (Power-Down mode)

30.3 Functional description

This section explains PLL operation and configuration.

30.3.1 Modes of operation

Table 183. Modes of operation

PLLCR[PLLPD]	PLLFD[SDMEN]	PLLFM[SSCGBYP]	Description
1	x	x	PLL is disabled.

Table continues on the next page...

Table 183. Modes of operation (continued)

PLLCR[PLLPD]	PLLFD[SDMEN]	PLLFM[SSCGBYP]	Description
0	0	1	Functional mode – PLL operates in integer-only mode. See Clock configuration.
0	1	1	Functional mode – PLL operates in Fractional mode (non-Frequency modulation). See Clock configuration.
0	1	0	Functional mode – PLL operates in Frequency Modulation mode. See Frequency modulation.

30.3.2 Input clock frequency

PLL is designed to operate over a specified input clock frequency range. PLL source frequency limits are discussed in this chip's data sheet.

30.3.3 Clock configuration

See the equations below and the corresponding register configuration that determine the relationship between VCO frequency (f_{VCO}) and PLL reference frequency.

- Integer-only mode:
  - When PLLDV[RDIV] is 0:

$$f_{\text{pll_VCO}} = f\text{pll_ref} \times \text{PLLDV[MFI]}$$

Equation 1. PLL VCO frequency in integer-only mode when PLLDV[RDIV] is 0

  - When PLLDV[RDIV] is not 0:

$$f_{\text{pll_VCO}} = \frac{f_{\text{pll_ref}}}{\text{PLLDV[RDIV]}} \times \text{PLLDV[MFI]}$$

Equation 2. PLL VCO frequency in integer-only mode when PLLDV[RDIV] is not 0

- Fractional mode:
  - When PLLDV[RDIV] is 0:

$$f_{\text{pll_VCO}} = f_{\text{pll_ref}} \times \left(\text{PLLDV[MFI]} + \frac{\text{PLLFD[MFN]}}{18432}\right)$$

Equation 3. PLL VCO frequency in Fractional mode when PLLDV[RDIV] is 0

  - When PLLDV[RDIV] is not 0:

$$f_{\text{pll_VCO}} = \frac{f_{\text{pll_ref}}}{\text{PLLDV[RDIV]}} \times \left(\text{PLLDV[MFI]} + \frac{\text{PLLFD[MFN]}}{18432}\right)$$

Equation 4. PLL VCO frequency in Fractional mode when PLLDV[RDIV] is not 0

See the equation below and the corresponding register configuration that determine the relationship between reference and PLL_PHI*n* output frequencies.

$$f_{\text{pll_phi}} = \frac{f_{\text{pll_VCO}}}{\text{PLLDV[ODIV2]} \times (\text{PLLDIV_}n\text{[DIV]} + 1)}$$

Equation 5. PLL PHI output frequency

When configuring PLL, you must not violate the maximum system clock frequency or maximum and minimum VCO frequency specification of PLL (see this chip's data sheet for frequency limits).

You must disable PLL by writing 1 to PLLCR[PLLPD] before any PLL configuration or input clock are modified.

You must disable PLL by writing 1 to PLLCR[PLLPD] for at least 5 µs before writing 0 to PLLCR[PLLPD] to enable PLL.

The recommended procedure to program PLL and enter Normal mode is shown in Initialization information.

30.3.4 Loss of lock (LOL)

PLL provides LOL indication. The LOL indication can only be generated when PLL is in Functional mode (see Modes of operation). When PLL detects a LOL, it asserts its LOL event output.

PLL does not detect loss of reference clock. If the reference clock stops after PLL achieves lock, PLL continues to indicate lock. It is assumed that monitoring of PLL's reference clock that is done outside PLL is enabled while PLL is in operation.

PLL LOL is intended for detection of gross failures. Use CMUs for accurate frequency monitoring.

30.3.5 Frequency modulation

In Frequency Modulation mode, PLL generates a frequency-modulated clock. The modulation depth and modulation frequency are calculated using the equations shown in Frequency modulation programming.

Write 1 to PLLFM[SPREADCTL] to select down-spread modulation. See Figure 134 that shows an example of down-spread modulation.

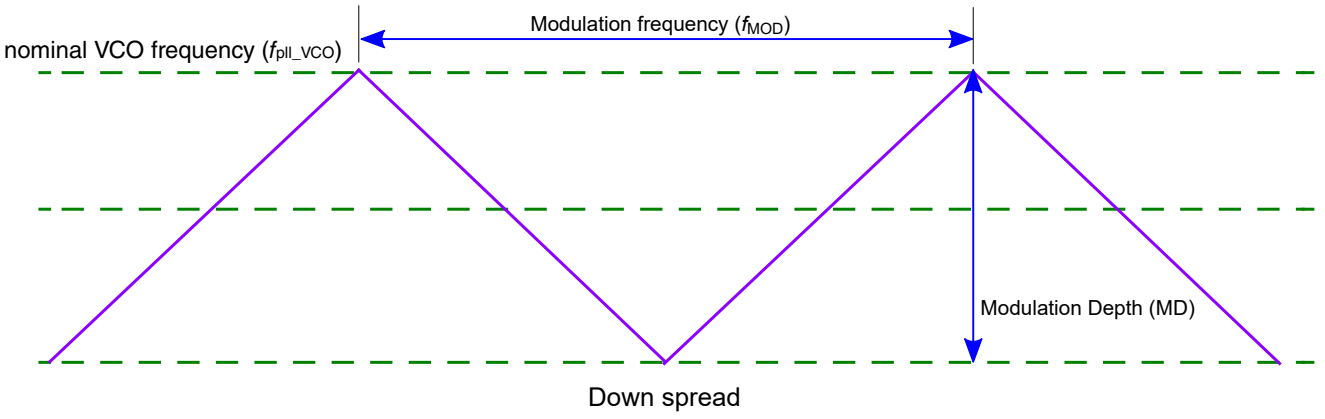


Figure 134. Frequency modulation

30.3.5.1 Frequency modulation programming

Modulation depth and modulation frequency programming uses step number (PLLFM[STEPNO]) and step size (PLLFM[STEPSIZE]). The table below shows variables used during calculations when programming PLL for frequency modulation.

Table 184. Variables for configuring modulation depth and frequency

Variable	Description
f_{REF}	Input clock frequency
f_{MOD}	Expected modulation frequency
MD	Expected modulation depth in percentage
LDF	Loop division factor
f_{pll_VCO}	Nominal VCO frequency

Use the following equations to configure PLL for frequency modulation.

$$LDF = PLLDV[MFI] + \frac{PLLFD[MFN]}{18432}$$

Equation 6. LDF

$$PLLFM[STEPNO] = \frac{f_{REF}}{\left(2 \times f_{MOD} \times PLLDV[RDIV]\right)}$$

Equation 7. Step number

$$PLLFM[STEPSIZE] = \frac{MD \times LDF}{100 \times PLLFM[STEPNO]} \times 18432$$

Equation 8. Step size

Frequency modulation is only possible if the condition shown in Equation 9 is met.

$$(PLLFM[STEPSIZE] \times PLLFM[STEPNO]) < 18432$$

Equation 9. Requirement to achieve FM

You must write 0 to PLLFM[SSCGBYP] and write 1 to PLLFD[SDMEN] to enable frequency modulation.

$$\text{Max (MD \%)} = \frac{(f_{REF} \times 100)}{f_{pll_VCO}}$$

Equation 10. Maximum possible modulation depth when PLL[RDIV] is 0

$$\text{Max (MD \%)} = \frac{(f_{REF} \times 100)}{PLLDV[RDIV] \times f_{pll_VCO}}$$

Equation 11. Maximum possible modulation depth when PLL[RDIV] is not 0

CAUTION

The effective modulation depth may differ from the intended modulation depth because of rounding operations applied to PLLFM[STEPSIZE] and PLLFM[STEPNO].

30.3.6 Interrupt signals

This module has no interrupt signals.

30.4 External signals

This module has no external signals.

30.5 Initialization information

Perform the following steps to initialize PLL:

1. Confirm that PLLODIV_*n*[DE] is 0 for all dividers.
2. Confirm that PLLCR[PLLPD] is 1.
3. Program the following as needed:
   - PLLDV
   - PLLFD
   - PLLFM to the desired value
4. Program PLLDV[ODIV2] and PLLODIV_*n*[DIV] to the desired values.
5. Wait for the PLL reference clock to be stable.
6. Write 0 to PLLCR[PLLPD].
7. Wait for PLLSR[LOCK] to be 1.
8. Write 1 to PLLODIV_*n*[DE].

Perform the following steps to shut down PLL:

1. Write 0 to PLLODIV_n[DE] for all dividers.
2. Write 1 to PLLCR[PLLPD].

30.6 PLLDIG register descriptions

This section provides the memory map and detailed descriptions of registers used for configuring PLL. The table below shows the memory map. Addresses are given as offsets from the module base address. All registers are accessed using 8-bit, 16-bit, or 32-bit addressing.

30.6.1 PLLDIG memory map

PLL base address: 402E_0000h

PLL_AUX base address: 402E_4000h

Offset	Register	Width (In bits)	Access	Reset value
0h	PLL Control (PLLCR)	32	RW	8000_0000h
4h	PLL Status (PLLSR)	32	RW	0000_0300h
8h	PLL Divider (PLLDV)	32	RW	0C3F_1032h
Ch	PLL Frequency Modulation (PLLFM)	32	RW	4000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
10h	PLL Fractional Divider (PLLFD)	32	RW	0000_0000h
18h	PLL Calibration Register 2 (PLLCAL2)	32	RW	0006_0000h
80h - 84h	PLL Output Divider (PLLODIV_0 - PLLODIV_1)	32	RW	0000_0000h

30.6.2 PLL Control (PLLCR)

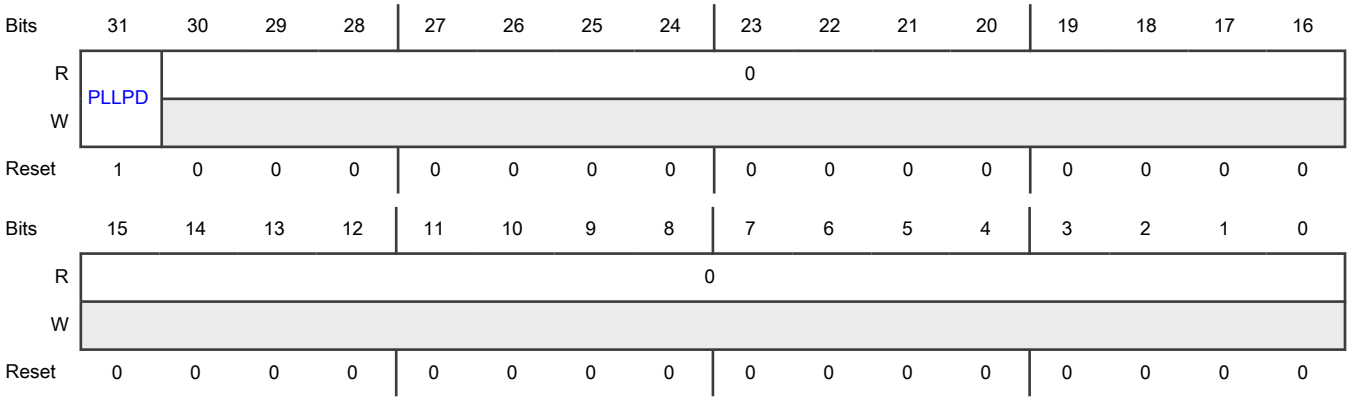
Offset

Register	Offset
PLLCR	0h

Function

Configures PLL functionality.

Diagram



Fields

Field	Function
31 PLLPD	PLL Power Down Powers down or powers up PLL. 0b - Powered up 1b - Powered down
30-0 —	Reserved

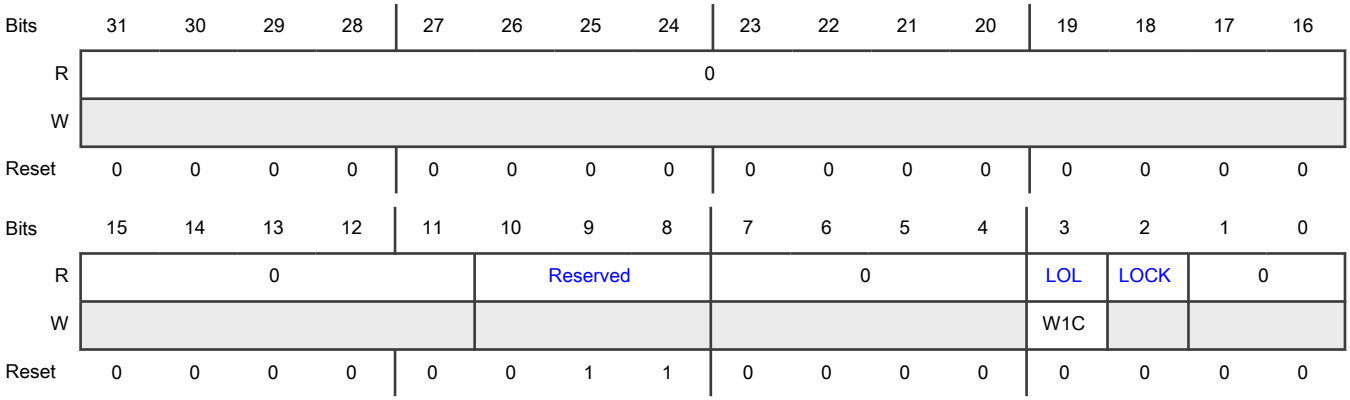
30.6.3 PLL Status (PLLSR)

Offset

Register	Offset
PLLSR	4h

Function
Shows the PLL status.

Diagram



Fields

Field	Function
31-11 —	Reserved
10-8 —	Reserved
7-4 —	Reserved
3 LOL	Loss-Of-Lock Flag Indicates the current PLL lock status. 0b - No loss of lock detected 1b - Loss of lock detected
2 LOCK	Lock Status Indicates that PLL has acquired lock. 0b - Unlocked

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Locked
1-0 —	Reserved

30.6.4 PLL Divider (PLLDV)

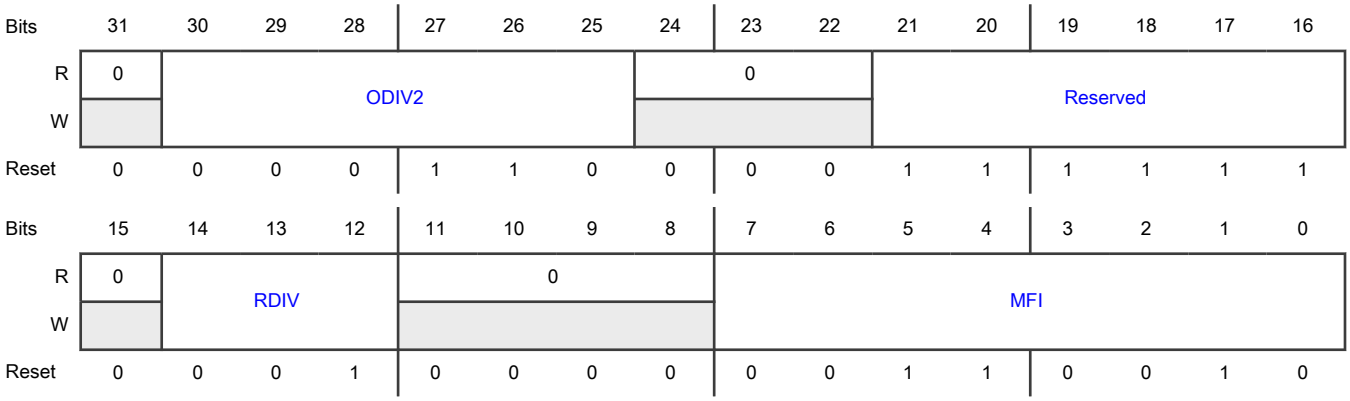
Offset

Register	Offset
PLLDV	8h

Function

Divides input clocks for PLL output generation.

Diagram



Fields

Field	Function
31 —	Reserved
30-25 ODIV2	Output frequency divider for raw PLL clock. 6-bit field determining the VCO clock post divider for driving the PHI output clock. 000000 – Divide by 1 000001 – Divide by 1 000010 – Divide by 2

Table continues on the next page...

Table continued from the previous page...

Field	Function
	000011 – Divide by 3 000100 – Divide by 4 ..... 111111 – Divide by 63
24-22 —	Reserved
21-16 —	Reserved
15 —	Reserved
14-12 RDIV	Input Clock Predivider Sets the input clock divider. The output of the predivider circuit generates the PLL loop reference clock. 000b - Divide by 1 001b - Divide by 1 010b - Divide by 2 011b - Divide by 3 100b - Divide by 4 101b - Divide by 5 110b - Divide by 6 111b - Divide by 7
11-8 —	Reserved
7-0 MFI	Integer Portion Of Loop Divider Sets the value of the divider in the PLL feedback loop. The value specified establishes the multiplication factor applied to the reference frequency. Write the divider value to this field, where the chosen value does not violate VCO frequency specifications.

30.6.5 PLL Frequency Modulation (PLLFM)

Offset

Register	Offset
PLLFM	Ch

Function

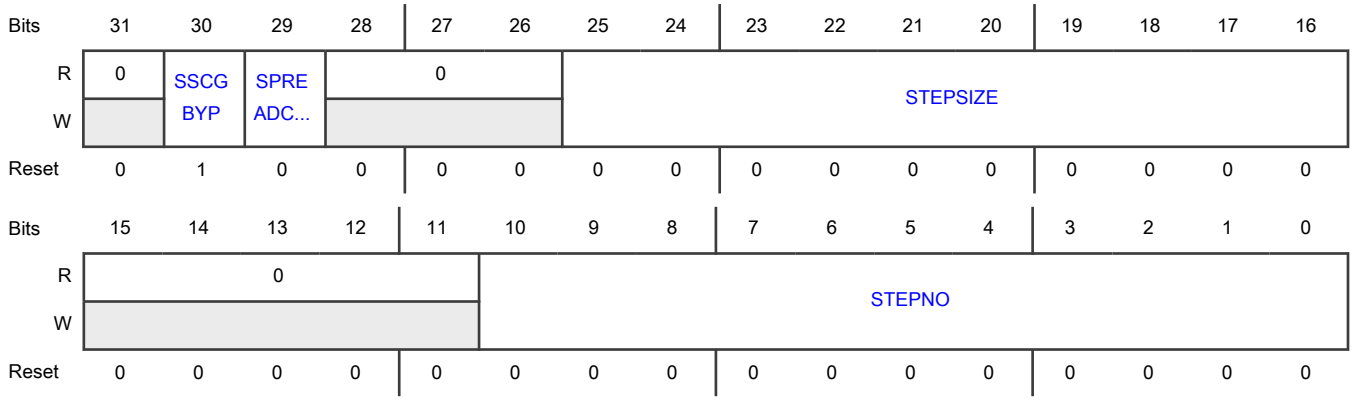
Configures PLL frequency modulation parameters.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
PLL	PLLFM	—
PLL_AUX	—	PLLFM

Diagram



Fields

Field	Function
31 —	Reserved
30 SSCGBYP	Frequency Modulation (Spread Spectrum Clock Generation) Bypass Bypasses frequency modulation. 0b - Not bypassed 1b - Bypassed
29	Modulation Type Selection

Table continues on the next page...

Table continued from the previous page...

Field	Function
SPREADCTL	Indicates that the modulation is spread below the nominal frequency. You must write 1 to this field. 0b - Reserved 1b - Spread below nominal frequency
28-26 —	Reserved
25-16 STEPSIZE	Frequency Modulation Step Size Provides the step size for modulation depth and frequency in Frequency Modulation mode (see Frequency modulation).
15-11 —	Reserved
10-0 STEPNO	Number Of Steps Of Modulation Period Or Frequency Modulation Provides the number of steps to achieve modulation depth in Frequency Modulation mode (see Frequency modulation).

30.6.6 PLL Fractional Divider (PLLFD)

Offset

Register	Offset
PLLFD	10h

Function

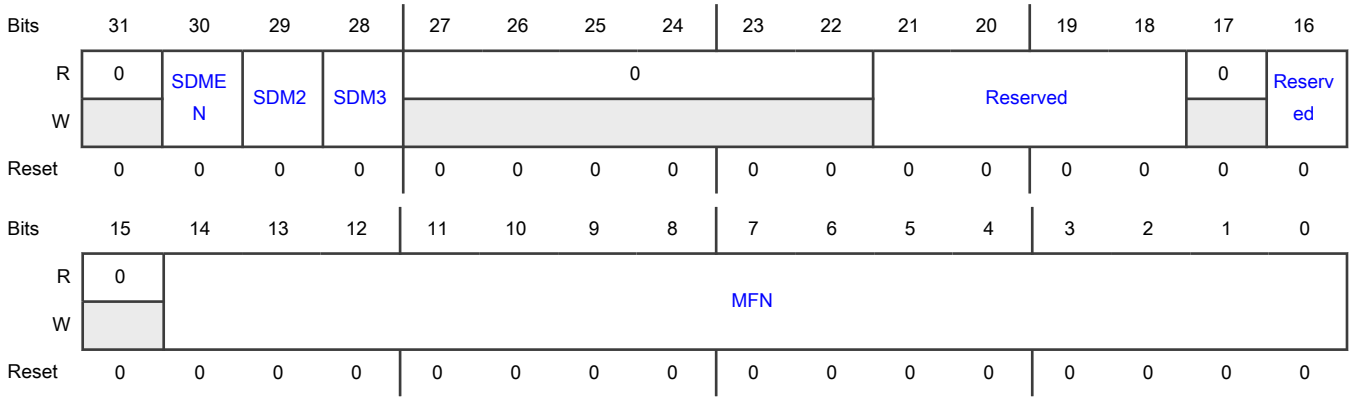
Enables and configures frequency modulation.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
PLL	PLLFD	—
PLL_AUX	—	PLLFD

Diagram



Fields

Field	Function
31 —	Reserved
30 SDMEN	Fractional Mode Enable Enables Fractional mode. 0b - Disabled 1b - Enabled
29 SDM2	Fractional Mode Configuration When you are in the fractional mode (SDMEN = 1), write 1 to this field. **NOTE** If SDMEN = 1, this field must be written 1.
28 SDM3	Fractional Mode Configuration When you are in the fractional mode (SDMEN = 1), write 1 to this field. **NOTE** If SDMEN = 1, this field must be written 1.
27-22 —	Reserved
21-18 —	Reserved
17 —	Reserved
16	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
15 —	Reserved
14-0 MFN	Numerator Of Fractional Loop Division Factor Sets the numerator of the fractional loop division factor. You must write a value of less than 18432 to this field. When Fractional mode is disabled, you must write 000_0000_0000_0000b to this field.

30.6.7 PLL Calibration Register 2 (PLLCAL2)

Offset

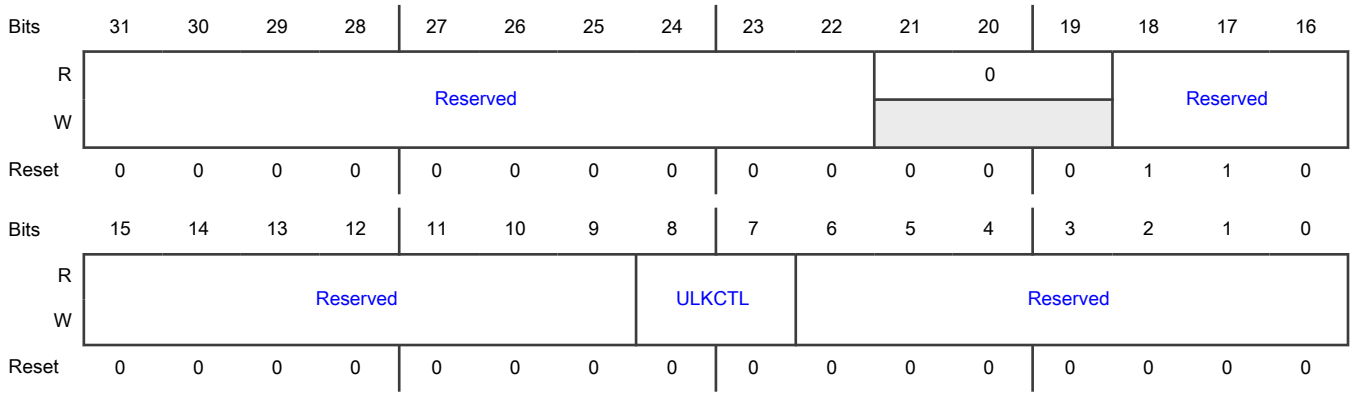
Register	Offset
PLLCAL2	18h

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
PLL	PLLCAL2	—
PLL_AUX	—	PLLCAL2

Diagram



Fields

Field	Function
31-22 —	Reserved
21-19 —	Reserved
18-16 —	Reserved
15-9 —	Reserved
8-7 ULKCTL	Unlock Control Accuracy Defines the accuracy necessary to achieve unlock. The lock counter determines unlock if the number of VCO clock cycles in the window of reference cycles is outside the number of cycles defined by this field. 00b - Unlock range = Expected value ± 9 (recommended when PLLFM[SSCGBYP] = 1). Unlock range = Expected value ± 9 (recommended when PLLFM[SSCGBYP] = 1) 01b - Unlock range = Expected value ± 17 (recommended when PLLFM[SSCGBYP] = 0). Unlock range = Expected value ± 17 (recommended when PLLFM[SSCGBYP] = 0) 10b - Unlock range = Expected value ± 33 11b - Unlock range = Expected value ± 5
6-0 —	Reserved

30.6.8 PLL Output Divider (PLLODIV_0 - PLLODIV_1)

Offset

Register	Offset
PLLODIV_0	80h
PLLODIV_1	84h

Function

Controls the PLL output clock divider settings.

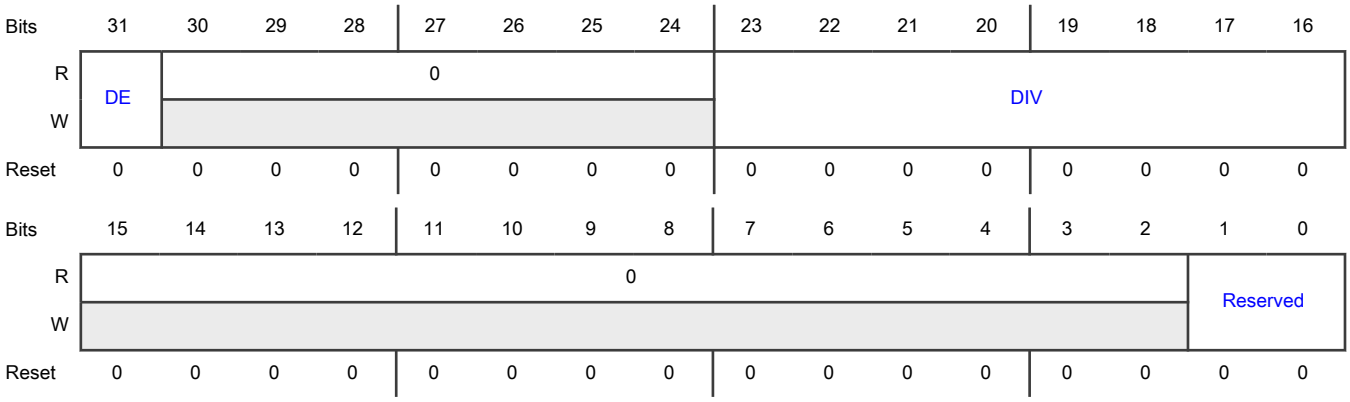
This divider has a 50% duty cycle.

NOTE

These registers support only word accesses. Other write accesses lead to the following:

- Unpredictable behavior
- No transfer error generated

Diagram



Fields

Field	Function
31 DE	Divider Enable Enables PLL output divider. Divider must be disabled before disabling PLL. 0b - Disabled 1b - Enabled
30-24 —	Reserved
23-16 DIV	Division Value Provides the division value for the output clock divider. The clock period of the clock after division is DIV + 1 times the time period of the divider input clock. **NOTE** This field is not supported in every instance. The following table includes only supported registers.
15-2	Reserved

Instance	Field supported in	Field not supported in
PLL	PLLODIV_0–PLLODIV_1	—
PLL_AUX	PLLODIV_0–PLLODIV_1[20–16]	PLLODIV_0–PLLODIV_1[23–21]

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
1-0 —	Reserved Do not write any value other than the reset value.

Chapter 31
Reset Overview

31.1 Introduction

This chip's reset logic consists of a reset sequence that leads the chip to a fixed deterministic state after predefined reset events occur. These events can pertain to chip failure events, the chip's special operating conditions, or certain software-governed events to initiate a chip reset sequence. This chapter discusses the chip reset scheme and related topics such as:

- Types of reset reactions
- Reset event sources
- Chip reset sequences
  - POR
  - Destructive reset
  - Functional reset
- RAM retention across functional reset
- Reset pin (RESET_b) behavior
- Debug system reset
- Signal-level reset flow

31.2 Chip reset types and reactions

31.2.1 Chip reset blocks

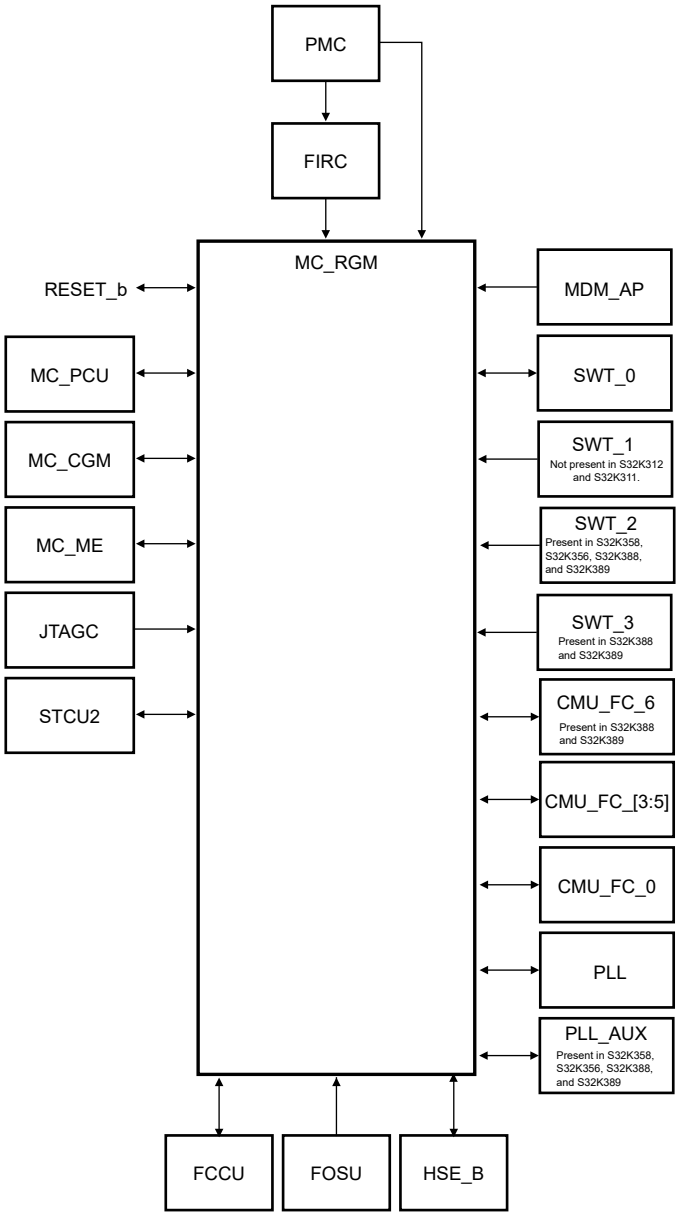


Figure 135. Chip reset blocks

31.2.2 Chip reset types

Table 185. Chip reset types

Reset event type	Functional description
POR	Leads to a complete chip reset.
Destructive	Leads most parts of the chip, except a few modules, to reset. SRAM content is lost after this reset event.

Table continues on the next page...

Table 185. Chip reset types (continued)

Reset event type	Functional description
Functional	Leads all the communication peripherals and cores to reset. The communication protocols' sanity is not guaranteed and they are assumed to be reinitialized after reset. The SRAM content, and the functionality of certain modules, is preserved across functional reset.

31.2.3 High-level reset sequence overview

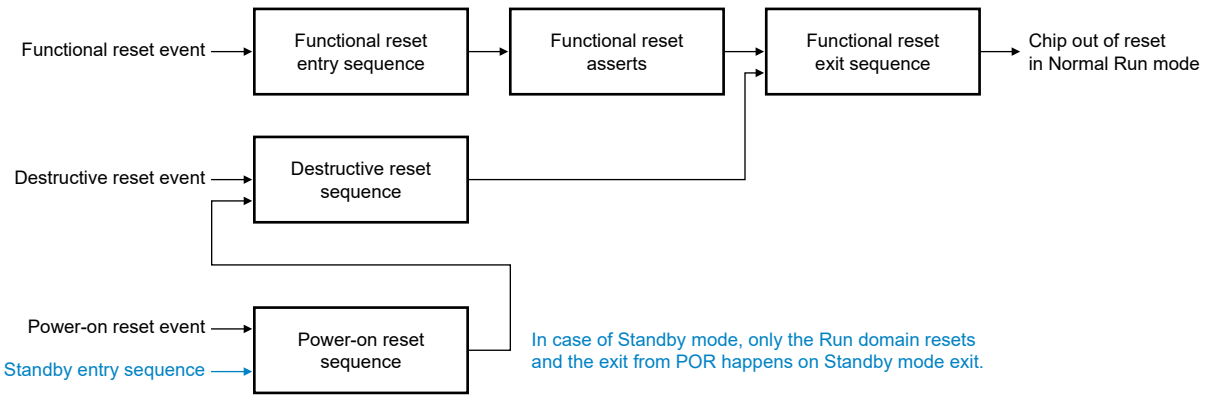


Figure 136. High-level reset sequence overview

31.2.3.1 Reset event reactions

Table 186. Reset event reactions

Reset event type	Triggered from	Reaction
POR	Anywhere	Moves to the beginning of the power-on sequence
Destructive reset	Power-on sequence	No reset sequence change
	Anywhere in the chip operation except in the power-on-sequence	Moves to the beginning of the destructive reset sequence
Functional reset	Out-of-reset	Moves to the beginning of the functional reset entry sequence
	Anywhere within the functional reset sequence	No reset sequence change

31.2.3.2 Chip action after reset event

For each reset event, immediately after MC_RGM captures it, the chip performs these actions:

1. Writes 1 to the corresponding reset event status fields in MC_RGM.DES and MC_RGM.FES (see the MC_RGM chapter for more information).
2. Places its pins in their default states (see the IOMUX file attached to this document for more information).
3. Asserts the RESET_b pin.

NOTE

After self-test completes, you can configure RESET_b assertion using MC_RGM.FES[ST_DONE].

4. Enters the reset sequence as described in Reset event reactions, depending on the current state and reset event type.

31.3 Reset sources—POR, destructive, and functional

MC_RGM records reset events in MC_RGM.FES and MC_RGM.DES, indicating the source of functional reset events and destructive reset events, respectively. You must read these fields to identify the reset source on reset recovery.

31.3.1 POR sources

Table 187. POR sources

Source module	Field in MC_RGM.DES	RESET_b assertion	Description
PMC	F_POR	Always	VDD_LV POR
			LVR on 1.1 V supply in Standby mode
			LVR on 1.1 V supply in Run mode
			LVR on 2.5 V supply in Standby mode
			LVR on 2.5 V supply in Run mode
			LVR on VDD_HV_A supply in Standby mode
			LVR on VDD_HV_A supply in Run mode
			LVR on VDD_HV_B supply in Standby mode [1]
			LVR on VDD_HV_B supply in Run mode [1]
POR_WDG			POR_WDG timeout (see the POR_WDG chapter for more information)

1. LVR on VDD_HV_B run mode and LVR on VDD_HV_B standby mode are not present in S32K312 and S32K311.

NOTE

You cannot escalate or demote POR to an interrupt.

31.3.2 Stages of the POR sequence

Table 188. Stages of the POR sequence

Stage	Process
PWRUP	1. Starts after a POR event (for example, a POR source assert). 2. Waits for the power-up sequence to complete. 3. Exits when all the POR sources clear. 4. Transitions to the FIRC_STRT stage after the procedure completes.
FIRC_STRT	1. Enters this stage after exiting the PWRUP stage. **NOTE** FIRC_CLK, if enabled, becomes available after it is stable. The duration depends on the clock startup time (see the chip datasheet for more information). The MC_RGM state machine proceeds further after FIRC_CLK is available. 2. Transitions to the destructive reset sequence after the procedure completes.

31.3.3 Destructive reset sources

Table 189. Destructive reset sources

Source module[1]	Field in MC_RGM.DES	Description
FOSU	FCCU_FTR	FCCU failure to react
STCU2	STCU_URF	STCU2 unrecoverable fault
MC_RGM	MC_RGM_FRE	Functional reset escalation
CMU_FC_0	FXOSC_FAIL	FXOSC failure
PLL	PLL_LOL	PLL loss of lock
CMU_FC_3	CORE_CLK_FAIL	Core clock failure
CMU_FC_4	AIPS_PLAT_CLK_FAIL	AIPS_PLAT_CLK failure
CMU_FC_5	HSE_CLK_FAIL	HSE_CLK failure
CMU_FC_6	CM7_CORE CLK FAIL	CM7_CORE_CLK Failure
MC_CGM	SYS_DIV_FAIL	System clock dividers alignment failure
HSE_B	HSE_TMPR_R ST	HSE_B tamper detect reset
HSE_B	HSE_SNVS_RST	HSE_B SNVS tamper detection
MC_ME	SW_DEST	Software destructive reset
MDM_AP	DEBUG_DEST	Debug destructive reset
RESET_b pin	EXT_RST	RESET_b pin assertion

1. All destructive resets can be escalated, but only the PLL LOL destructive reset can be demoted to an interrupt (see Destructive reset event bypass for PLL LOL destructive reset bypass details).

NOTE

All reset sources in the table above assert the RESET_b pin.

31.3.4 Functional reset sources

Table 190. Functional reset sources

Source module	Field in MC_RGM.FES	RESET_b assertion	Demotable to IRQ[1]	Escalation[2]	Description
FCCU soft reaction[3]	FCCU_RST	Always	Yes[4]	Yes	FCCU reset reaction
STCU2	ST_DONE	Configurable	No	No	Self-test done
SWT_0	SWT0_RST	Always	Yes[5]	Yes	SWT reset request
SWT_1[6]	SWT1_RST	Always	Yes[7]	Yes	SWT reset request
SWT_2[8]	SWT2_RST	Always	Yes[9]	Yes	SWT reset request
SWT_3[10]	SWT3_RST	Always	Yes[11]	Yes	SWT reset request
PLL_AUX[12]	PLL_AUX_RST[13]	Always	Yes[14]	Yes	PLL reset request
JTAGC	JTAG_RST	Always	Yes [15]	No	JTAG reset

Table continues on the next page...

Table 190. Functional reset sources (continued)

Source module	Field in MC_RGM.FES	RESET_b assertion	Demotable to IRQ[1]	Escalation[2]	Description
HSE_B	HSE_SWT_RST	Always	No	Yes	HSE_B SWT timeout
HSE_B	HSE_BOOT_RST	Always	No	Yes	HSE_B boot reset
MC_ME	SW_FUNC	Always	No	Yes	Software functional reset
MDM_AP	DEBUG_FUNC	Always	Yes [16]	Yes	Debug functional reset

1. See Functional reset demotion to an interrupt for more information.
2. See Reset escalation for more information.
3. An FCCU soft functional reset is a chip functional reset (see the FCCU chapter for more information).
4. Controlled by MC_RGM.FERD[D_FCCU_RST].
5. Controlled by MC_RGM.FERD[D_SWT0_RST].
6. SWT_1 is not present in S32K312, S32K311, S32K310, S32K348, S32K314, S32K341, S32K342, S32K344.
7. Controlled by MC_RGM.FERD[D_SWT1_RST].
8. SWT_2 is only present in S32K338/S32K388/S32K389.
9. Controlled by MC_RGM.FERD[D_SWT2_RST]
10. SWT_3 is only present in S32K388/S32K389.
11. Controlled by MC_RGM.FERD[D_SWT3_RST]
12. PLL_AUX is only present in S32K358, S32K356, S32K388, and S32K389.
13. In order to use PLL_Aux as functional reset /interrupt source, the 24th bit of Functional Reset Register (DCMRWF2) should be configured.
14. Controlled by MC_RGM.FERD[D_PLL_AUX_RST]
15. Controlled by MC_RGM.FERD[D_JTAG_RST].
16. Controlled by MC_RGM.FERD[D_DEBUG_FUNC].

31.3.5 Functional reset sequence descriptions

Table 191. Functional reset sequence descriptions

Stage	Series of events
Functional reset entry sequences	
FUNC0	This stage starts after any functional reset event. The FCCU fault monitoring and CMU_F*x_n* monitoring for FLL events is masked in this step to avoid any false fault or reset.
FUNC1	In this stage, a halt sequence that includes daisy chaining of all the gaskets halts, disabling the crossbar. The stage completes after the halt-handshake sequence completes.
FUNC2	In this stage, MC_RGM triggers the MC_CGM hardware clock multiplexers to switch to FIRC_CLK. • Software-based clock multiplexers do not support switching to FIRC_CLK on functional reset. • If PCFS is enabled, the system clock switching can be done via PCFS. This stage completes after MC_CGM switches the system clock to FIRC.
FUNC3	In this stage, MC_RGM triggers all the MC_CGM hardware-based clock multiplexers with PCFS enabled or disabled to move their dividers to default values.

Table continues on the next page...

Table 191. Functional reset sequence descriptions (continued)

Stage	Series of events
	• Software-based clock multiplexers do not support this feature. This stage completes after all the clock multiplexer dividers initialize to their corresponding default values.
FUNC4	In this stage, PLLDIG turns off synchronously. The stage completes after PLLDIG turns off.
FUNC5	FXOSC_CLK switches off synchronously. The FUNC4 and FUNC5 stages ensure that PLLDIG disables cleanly to ensure there are no glitches on the PLLDIG clock because of reset. The stage completes after FXOSC switches off.
FUNC6	In this stage, clocks of modules that are a part of LBIST and working on the destructive reset are enabled to meet their synchronous reset requirements, if any. For the self-test logic, in self-test, the destructive reset deasserts after this stage completes and after safe stating is removed. **NOTE** In the self-test sequence, the logic, which is a part of self-test (LBIST logic) resets when self-test completes. All the parts of logic in self-test (POR, destructive, and functional reset) reset after self-test completes whereas the rest of the chip undergoes a functional reset sequence because of self-test completion (MC_RGM.FES[ST_DONE] = 1). See the "Safety Overview" and STCU2 chapters for more information on the self-test operation.
FUNC7	MC_RGM asserts the functional reset and triggers a counter running on FIRC_CLK (for up to 64 cycles) to enable clocks for the modules having synchronous reset requirements. Flash memory comes out of reset after this stage completes. **NOTE** The flash memory resets after a functional reset event but comes out of reset, before the rest of the modules do, at the start of the functional reset exit sequence. The rest of the modules reset when the functional reset comes out of reset at the end of the functional reset exit sequence. Therefore, the reset to flash memory is an early functional reset that deasserts earlier than the functional reset, even if asserted at the same time.
Functional reset exit sequences	
FUNC8	This stage consists of flash memory and MC_RGM handshaking. Flash memory indicates the completion of its initialization to MC_RGM.
FUNC9	DCM initiates the scanning of flash memory DCF records. This state completes after the flash memory scanning completes. See the DCF clients file attached to this document for more information.
FUNC10	After DCM scans the DCF records from the flash memory, DCM initiates the trim loading sequence for analog blocks. The analog blocks are loaded with the configured trimmed values in this stage, which completes after a trim-loading sequence completes.

Table continues on the next page...

Table 191. Functional reset sequence descriptions (continued)

Stage	Series of events
FUNC11	In this stage, MG_RGM stops driving RESET_b and checks that the signal does not assert externally. If you enable low-power debug, MC_RGM waits for a debug acknowledge. The completion of this stage indicates that MC_RGM completed the reset sequence, deasserting the functional reset to the system.

31.4 Reset and boot sequence

The chip reset sequence consists of several reset stages based on the occurrence of a particular reset event. All reset events follow the same chip reset sequence; only the entry points vary depending on the type of reset event. MC_RGM triggers each stage after the previous stage completes. These stages execute in a specific order, which ensures a deterministic state of the chip when a reset event completes.

Figure 137 shows a high-level representation of the chip startup sequence.

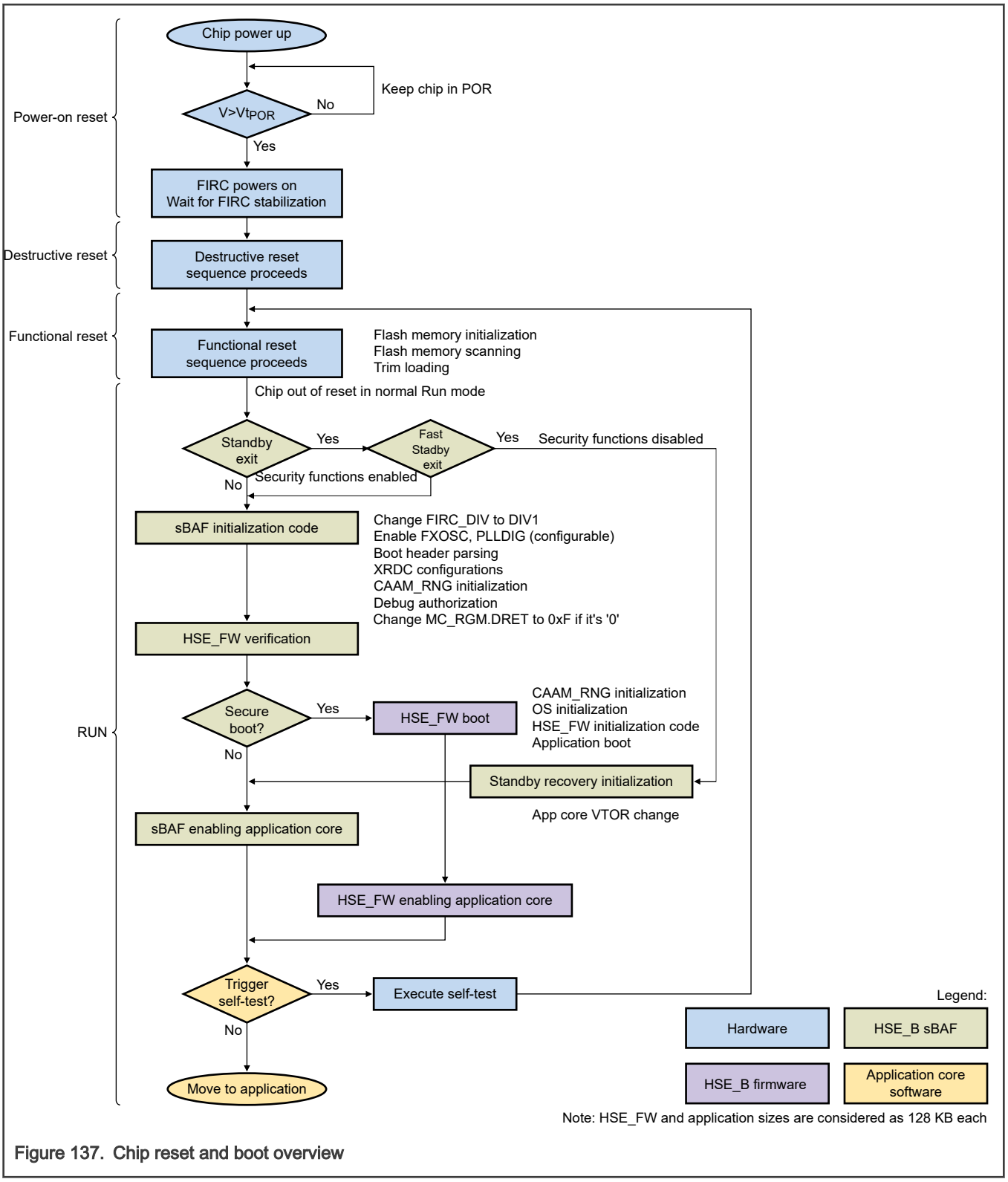


Figure 137. Chip reset and boot overview

31.4.1 POR

This stage starts when the POR event occurs, that is, when a POR source asserts. The logic within the Run power domain running on POR also resets in the chip Standby entry sequence.

NOTE

The logic in the Standby power domain does not reset in the chip Standby mode entry sequence.

The POR sequence consist of two stages:

1. Power-up (PWRUP)
2. FIRC oscillator start (FIRC_STRT)

See POR sequence for more information and Stages of the POR sequence for POR stages and their descriptions.

31.4.1.1 POR sequence

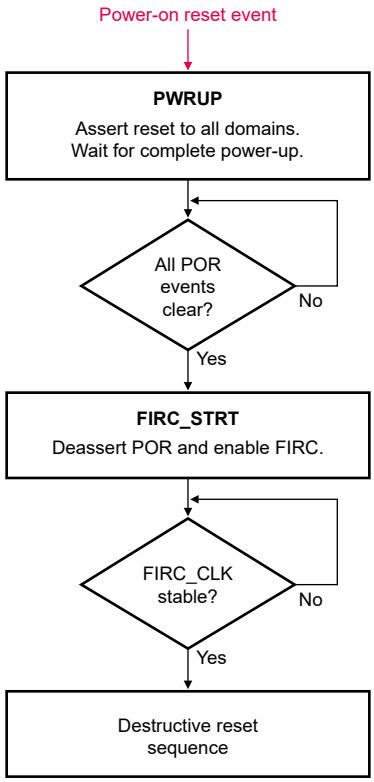


Figure 138. POR flow

31.4.2 Destructive reset

The chip enters a destructive reset sequence after the POR sequence or any destructive reset event completes. Destructive reset sequence illustrates the destructive reset sequence and Destructive sequence stage description discusses the stages of destructive sequence.

31.4.2.1 Destructive reset sequence

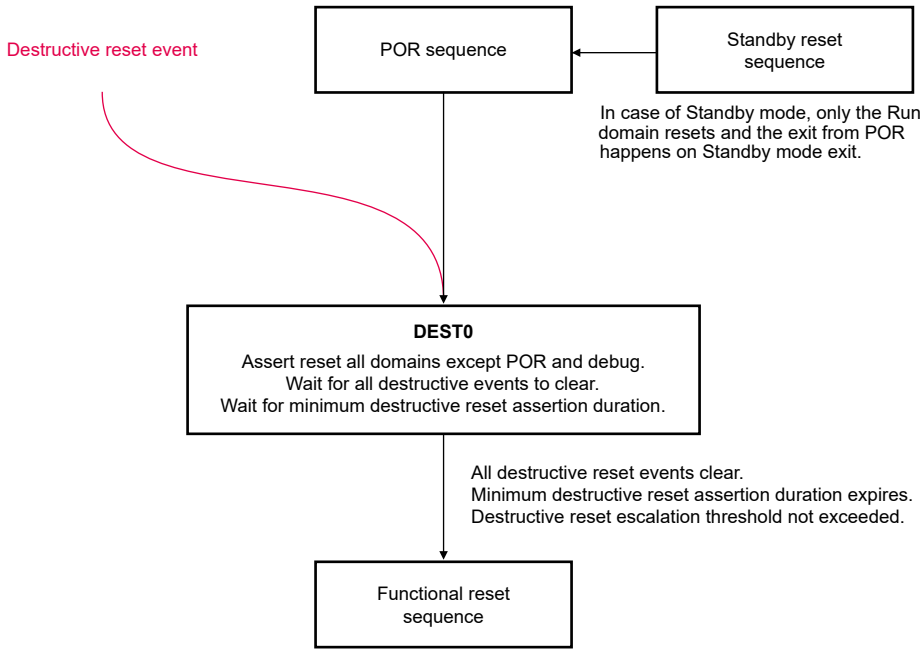


Figure 139. Destructive reset sequence

31.4.2.1.1 Destructive sequence stage description

Table 192. DEST0 description

Stage	Process
DEST0	1. Asserts reset to the entire chip, except logic running on POR. 2. Waits for all the destructive reset events to clear. 3. Waits for the minimum destructive reset assertion duration of eight FIRC_CLK cycles. 4. Deasserts after stage completion.

31.4.2.2 Destructive reset event bypass

This chip supports a destructive reset event demotion mechanism that the application software configures. The destructive reset bypasses and an interrupt event occurs (demotion). Table 193 discusses details related to GPR configuration and corresponding interrupt identification.

A successful chip operation is not guaranteed if a destructive reset event is bypassed.

Table 193. Destructive reset event bypass

Destructive reset event	Destructive reset event description	DCM field to bypass reset event	NVIC interrupt ID
MC_RGM.DES[PLL_LOL]	PLL loss of lock	DCM.DCMRWP3[9]	212

31.4.3 Functional reset

The chip enters the functional reset sequence when any of the following events occur:

- Functional reset
- POR or destructive reset (after the DEST0 stage completion)

On any functional reset event, the chip starts a functional reset entry sequence before the functional reset asserts and ensures the stability of logic running on a destructive reset and POR. On a destructive reset event or POR events the functional reset entry sequence does not execute.

The functional reset exit sequence consists of steps that ensure proper initialization of the chip after functional reset recovery.

31.4.3.1 Functional reset sequence

Functional reset flow illustrates the functional reset flow and Functional reset sequence descriptions discusses the functional reset stages and their descriptions.

Stages FUNC0 to FUNC6 present the functional reset entry sequence. It occurs on any functional reset event before the functional reset. In other words, when a functional reset event occurs, MC_RGM holds the asserted reset and executes the functional reset entry sequence. After the sequence completes, MC_RGM resets the chip, which remains in Run mode during the functional reset entry sequence.

Stages FUNC7 to FUNC11 present the functional reset exit sequence, which occurs after a functional reset event before deasserting the chip reset. This includes handshaking with the flash memory and analog blocks, ensuring correct operation after reset exit.

31.4.3.1.1 Functional reset flow

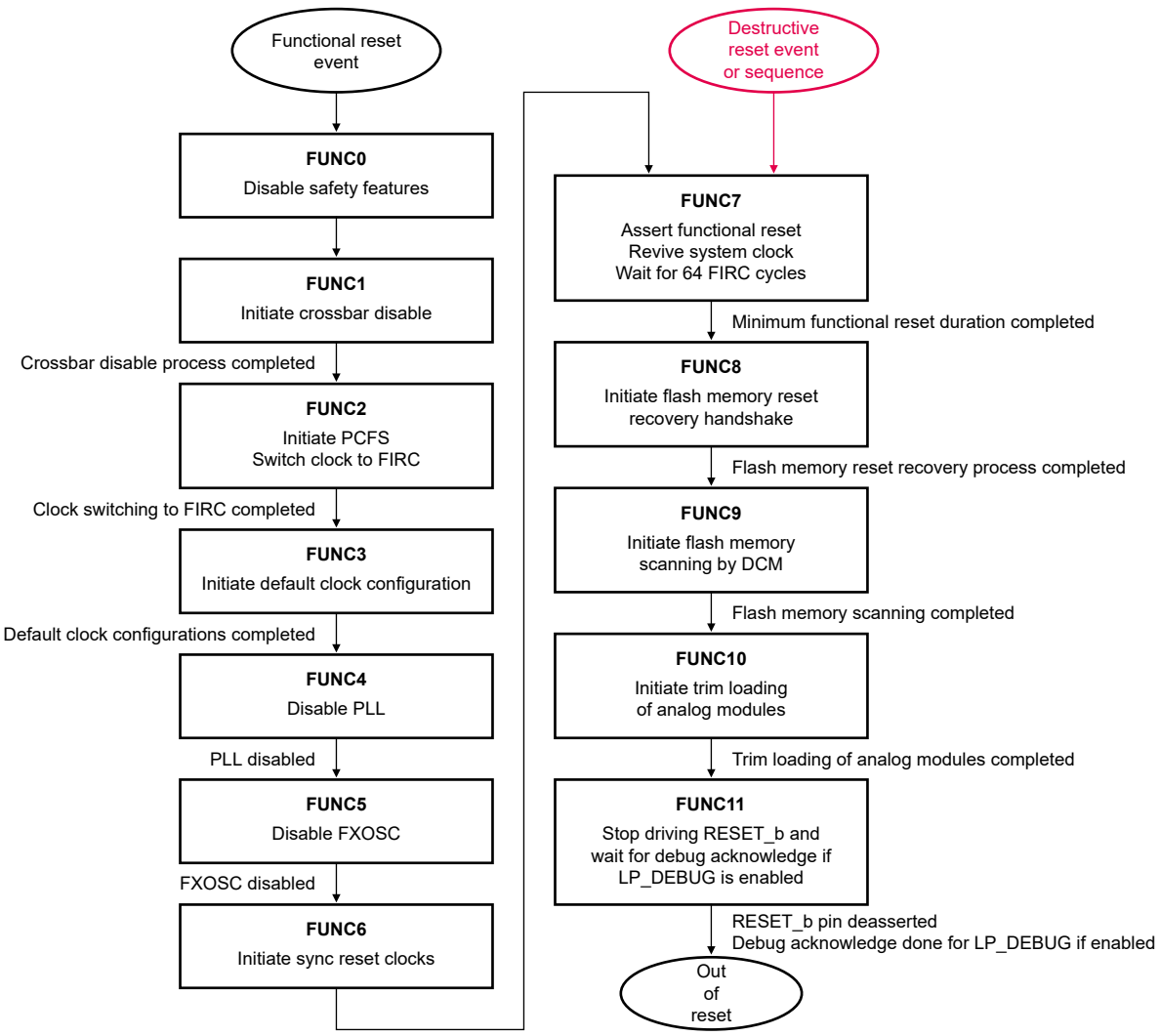


Figure 140. Functional reset flow

31.4.3.2 FUNC9 and FUNC10 stage bypass for faster Standby mode exit

The chip supports optional bypassing of the FUNC9 and FUNC10 (only FIRC and PMC phases) stages on Standby mode exit to considerably reduce the Standby mode exit duration. This feature is recommended only for Standby mode exit and must be configured on:

- Standby mode entry sequence
- SW3
- Disabled on Standby mode exit

See the "Faster Standby recovery" section in the "Power Management" chapter for more information.

31.4.3.3 Standby reset sequence

The logic within Run domain is reset additionally apart from the reset sequence in the chip standby entry sequence, wherein all the resets assert (POR, destructive, and functional) to the Run domain logic. The chip standby entry sequence does not have any impact on the logic in Standby power domain.

Wake-up from Standby mode removes the resets to the Run domain in the standby entry sequence.

See the "Peripheral reset status" section in the "Reset Overview" chapter for the logic present in Run domain.

31.4.3.4 Reset function redirection

Resets may escalate or demote to an IRQ, depending on the chip configuration.

31.4.3.4.1 Functional reset demotion to an interrupt

This chip supports the reset sequence demotion feature for functional resets. You can configure a functional reset to create an interrupt instead of a reset (see the MC_RGM chapter for details).

31.4.3.4.2 Reset escalation

The chip supports the reset escalation feature. If multiple functional or destructive resets occur, the related reset can escalate to a higher priority reset sequence (see the "Functional reset escalation" and "Destructive reset escalation" sections in the MC_RGM chapter).

31.4.3.4.2.1 Destructive reset escalation

You can enable destructive reset escalation by configuring a DCF client. You must also configure the destructive count threshold in MC_RGM.DRET (see "Destructive Reset Escalation Enable Register (DEST_RST_ESC):dcf_client_dest_rst_esc" in the DCF file attached to this document). The escalation event can individually be enabled or disabled for each reset source, and the fields in the dcf_client_dest_rst_esc register correspond with the fields in MC_RGM.DES register.

After being configured, MC_RGM immediately asserts a destructive reset escalation when the destructive event count reaches the threshold count in MC_RGM.DRET[DRET]. When the destructive and escalation reset assert, the reset sequence immediately enters the DEST0 state. The reset sequencing remains in DEST0 until a POR event occurs. If enabled, the destructive reset escalation counter increments with each destructive reset event. The application software clears the destructive reset escalation counter by writing any value to MC_RGM.DRET[DRET].

NOTE

You can configure GPR settings to allow demotion of destructive resets to interrupts instead of escalation (DCMRWP3[DEST_RST9_AS_IPI]). See Destructive reset event bypass for more information.

31.5 Reset timing diagram

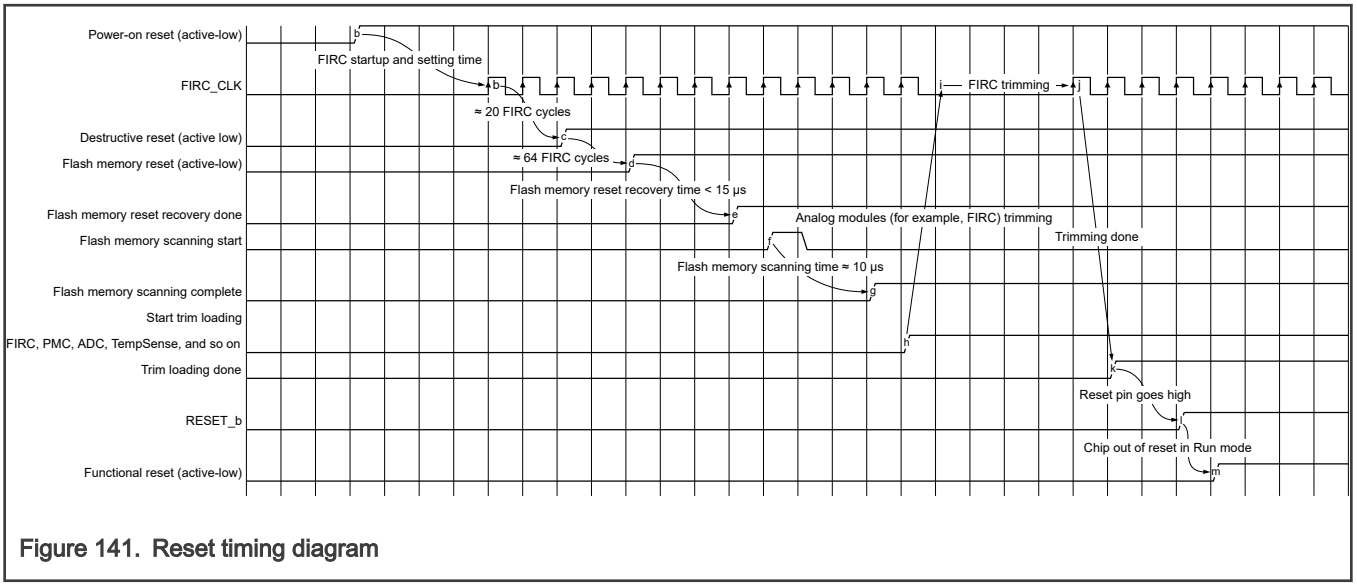


Figure 141. Reset timing diagram

31.6 Chip status after reset deassertion

Table 194. Chip status after reset deassertion

Function or feature	After POR deassertion	After destructive reset deassertion	After functional reset deassertion
Clock sources	• FIRC_CLK and SIRC_CLK on • Others off	• FIRC_CLK and SIRC_CLK on • Others off	• FIRC_CLK and SIRC_CLK on • SXOSC_CLK same as before functional reset • FXOSC_CLK and PLL_PHIn_CLK off
Clock selection	FIRC_CLK	FIRC_CLK	FIRC_CLK
Clock dividers	Default configuration	Default configuration	Default configuration
Reset status flags	MC_RGM.DES[F_POR] equals 1, others equal 0	MC_RGM.DES[F_DR_n] equals 1, others equal 0	MC_RGM.FES[F_FR_n] equals 1, others equal 0
MC_ME previous mode	Reset	Reset	Reset
FCCU fault information	Cleared	Cleared	Retained
SRAM content	Invalid	Invalid	Retained
DCF configurations in DCM	Reset value	Existing loaded value (reset value after POR)	Reloaded from flash memory
Cores	All off	All off	Cores initialized as per application configuration
Logic on POR[1]	Out of reset with default configuration	Out of reset with default configuration	Out of reset with default configuration
Logic on destructive reset [1]	Under reset with default configuration	Out of reset with default configuration	Out of reset with default configuration
Logic on functional reset [1]	Under reset with default configuration	Under reset with default configuration	Out of reset with default configuration

1. For the list of peripherals affected by POR, destructive reset, and functional reset events, see Module reset status.

31.7 Module reset status

Table 195. Module reset status

Module instances[1]	Destructive	Functional	Power domain[2]	Part of LBIST[3]
MC_RGM	Y	Y	Standby	No
PRAMC	Y	Y	Run	Yes
PFC	Y	Y	Run	Yes
SIUL_VIRTWRAPPER_PDAC0	Y	Y	Run	No

Table continues on the next page...

Table 195. Module reset status (continued)

Module instances[1]	Destructive	Functional	Power domain[2]	Part of LBIST[3]
SIUL_VIRTWRAPPER_PDAC1	Y	Y	Run	No
SIUL_VIRTWRAPPER_PDAC2	Y	Y	Run	No
SIUL_VIRTWRAPPER_PDAC3	Y	Y	Run	No
SIUL_VIRTWRAPPER_PDAC4[4]	Y	Y	Run	No
SIUL_VIRTWRAPPER_PDAC5[5]	Y	Y	Run	No
DCM	Y	Y	Run and Standby[6]	No
TRGMUX	Y	Y	Run	No
WKPU	Y	Y	Standby	No
CMU_F*x*_[0:5]	Y	Y	Run	CMU 0-3: No CMU 1-2: Yes CMU 4-5: Yes
FIRC	Y[7]	Y[7]	Standby	No
FXOSC	Y	Y	Standby	No
MC_CGM [8]	Y	N	Run	No
MC_ME	Y	Y	Run	No
PLL	Y	Y	Run	No
PLL_AUX[4]	Y	Y	Run	No
Configuration GPR	Y	Y	Run	No
eMIOS 0-2	Y	Y	Run	No
PIT_0	Y	Y	Standby[9]	No
PIT_[1:2]	Y	Y	Run	No
PIT_[3][11]	Y	Y	Run	No
FlexCAN_[0:5]	Y	Y	Run	No
FlexCAN_[6:7][4]	Y	Y	Run	No
FlexCAN_[8:11][10]	Y	Y	Run	No
FlexIO	Y	Y	Run	No
LPUART_[0:15]	Y	Y	Run	No
LPI2C_[0:1]	Y	Y	Run	No
LPSPI_[0:5]	Y	Y	Run	No
QuadSPI	Y	Y	Run	No
SAI_[0:1]	Y	Y	Run	No
uSDHC[4]	Y	Y	Run	No
ADC_[0:2]	Y	Y[14]	Run	No

Table continues on the next page...

Table 195. Module reset status (continued)

Module instances[1]	Destructive	Functional	Power domain[2]	Part of LBIST[3]
LPCMP_[0:2]	Y	Y	Standby	No
TempSense	Y	Y[14]	Run	No
CRC	Y	Y	Run	Yes
FCCU (+FOSU)	Y	N	Run	Yes
STCU2	Y	N	Run	No
HSE_B MUA-MUB	Y	Y	Run	No
MU_2 MUA-MUB[4]	Y	Y	Run	No
MU_3 MUA-MUB[11]	Y	Y	Run	No
MU_4 MUA-MUB[11]	Y	Y	Run	No
JDC	Y[12]	Y[12]	Run	No
DMAMUX_[0:1]	Y	Y	Run	No
PMC	Y[13]	N	Standby	No
Flash memory	Y	Y[14]	Run	No
SIRC	Y[7]	Y[7]	Standby	No
SXOSC	Y[15]	N	Standby	No
BCTU	Y	Y	Run	No
LCU[0:1]	Y	Y	Run	No
RTC	Y	N[16]	Standby	No
EMAC	Y	Y	Run	No
GMAC[4]/GMAC_0	Y	Y	Run	No
GMAC_1[11]	Y	Y	Run	No
HSE_B	Y	Y	Run	No
SWT_0	Y	Y	Standby	No
SWT_1	Y	Y	Run	Yes
SWT_2[4]	Y	Y	Run	Yes
SWT_3[11]	Y	Y	Run	Yes
STM_0	Y	Y	Run	No
STM_1	Y	Y	Run	No
STM_2[4]	Y	Y	Run	No
STM_3[11]	Y	Y	Run	No
MSCM	Y	Y	Run	No
ERM_0	Y	Y	Run	Yes

Table continues on the next page...

Table 195. Module reset status (continued)

Module instances[1]	Destructive	Functional	Power domain[2]	Part of LBIST[3]
ERM_1[4]	Y	Y	Run	Yes
EIM_0	Y	Y	Run	Yes
EIM_1[4]	Y	Y	Run	Yes
EIM_2[4]	Y	Y	Run	Yes
EIM_3[11]	Y	Y	Run	Yes
eDMA	Y	Y	Run	Yes
JTAGC	N	N	Run	No
MDM_AP	Y	N	Run	No
APB_AP	Y	N	Run	No
Cortex-M7_0	Y	Y[17]	Run	No
Cortex-M7_1	Y	Y[17]	Run	No
Cortex-M7_2[4]	Y	Y	Run	No
Cortex-M7_0 AHB-AP	Y	N	Run	No
Cortex-M7_0 AHB-AP	Y	N	Run	No
Cortex-M7_3[11]	Y	Y	Run	No
MC_PCU	Y	N	Standby	No

Legends:	
Y	The entire module resets on this particular reset.
Y	Only a portion of the module resets on this particular reset.
N	No portion of the module resets on this particular reset

1. All the modules listed in this table are reset on a POR event. See the memory map file attached to this document for the availability of modules across various parts in the S32K3xx family.
2. The modules in the RUN domain get reset on standby exit. The modules in STANDBY domain are not impacted by standby exit and retain their contents. However in case of standby exit via functional reset or destructive reset event, the corresponding flops within the STANDBY domain modules will also get reset.
3. The modules in the LBIST logic get reset on selftest completion.
4. Applicable for S32K358, S32K356, S32K388, and S32K389 only.
5. Applicable for S32K388 and S32K389 only.
6. Flash memory scanning logic is available in the Run domain. GPRs and LC decode logic are available in Standby domain.
7. All memory-mapped registers are on functional reset. The trimming logic is on destructive reset. Rest of counter and other stuff is on POR.
8. During functional reset stages FUNC2 and FUNC3 (see the Functional reset sequence descriptions for functional reset stage descriptions), MC_CGM.MUX_*n*_CSC and MC_CGM.MUX_*n*_DIV_*m* are automatically set to their default values. The default value of the MC_CGM.MUX_*n*_CSC[SELCTL] selects FIRC_CLK as the source clock for all multiplexers. The default value of the MC_CGM.MUX_*n*_DIV_*m* is register instance specific (see the "MC_CGM register descriptions" section in the "Clock Generation Module (MC_CGM)" chapter).
9. Only PIT_0 supports the RTI feature, and exists in the Standby domain.
10. Applicable for S32K389 only.
11. Applicable for S32K388 and S32K389 only
12. The system domain is reset on functional reset. A POR will reset it completely.
13. PMC registers are reset on a destructive reset except PMC.LVSC, which is reset only by own PMC.LVSC[PORF] flag. The LVR and POR logic is reset on an LVR or own PORF (see PMC.CONFIG and PMC.LVSC descriptions for details).

14. Reset on a functional reset; however, reset recovery occurs before the chip functional reset recovery, at the functional reset exit sequence start, for proper trim scanning and loading.
15. SXOSC is reset on a destructive reset so that the RTC operates properly across a functional reset.
16. RTC operates during a functional reset.
17. The functional reset maps to nSYSRESET to Arm Cortex-M7 and the destructive reset maps to nPORESET to Arm Cortex-M7. See the Cortex-M7 TRM for further description on part of logic on different domains within Cortex-M7.

31.8 System RAM retention across functional reset

System RAM retains content during functional reset through the crossbar halt handshake. The system crossbar halts during the functional reset entry sequence. Therefore, the accesses do not cause any content corruption (see Functional reset sequence for more information).

Follow this sequence for the crossbar halt handshake (see Halt handshake using the daisy-chaining method for gasket locations):

1. Send a halt request to the HSE_B AXBS, DMA AXBS, and EMAC IAHB bridges in parallel.
2. Wait for a halt acknowledgement from HSE_AXBS and DMA AXBS.
3. Send a halt request to the DMA IAHB and HSE_B IAHB gaskets.
4. Wait for a halt acknowledgement from all the gaskets listed in the aforementioned steps.
5. Send a halt request to the system AXBS.
6. Wait for a halt acknowledgement from the system AXBS.
7. Send a halt request to a peripheral AXBS.
8. Wait for a halt acknowledgement from a peripheral AXBS.
9. Send a halt request to the TCM IAHB and QSPI IAHB gaskets.
10. Wait for a halt acknowledgement from all the gaskets listed in the aforementioned steps.
11. Send a halt request to the AIPS0 IAHB and AIPS1 IAHB gaskets.
12. Wait for a acknowledgement from all of the gaskets listed in the aforementioned steps.
13. Send a halt request to TCM AXBS.
14. Wait for a halt acknowledgement from TCM AXBS.

NOTE

AXBS halt handshake sequence is automatically performed by the hardware.

After this halt sequence completes, the crossbar halt acknowledgement sequence also completes and the chip proceeds to the FUNC1 stage in the functional reset entry sequence.

NOTE

RAM retention is supported across the functional reset event for system RAMs only and not for HSE_B or peripheral memories.

31.8.1 Halt handshake using the daisy-chaining method

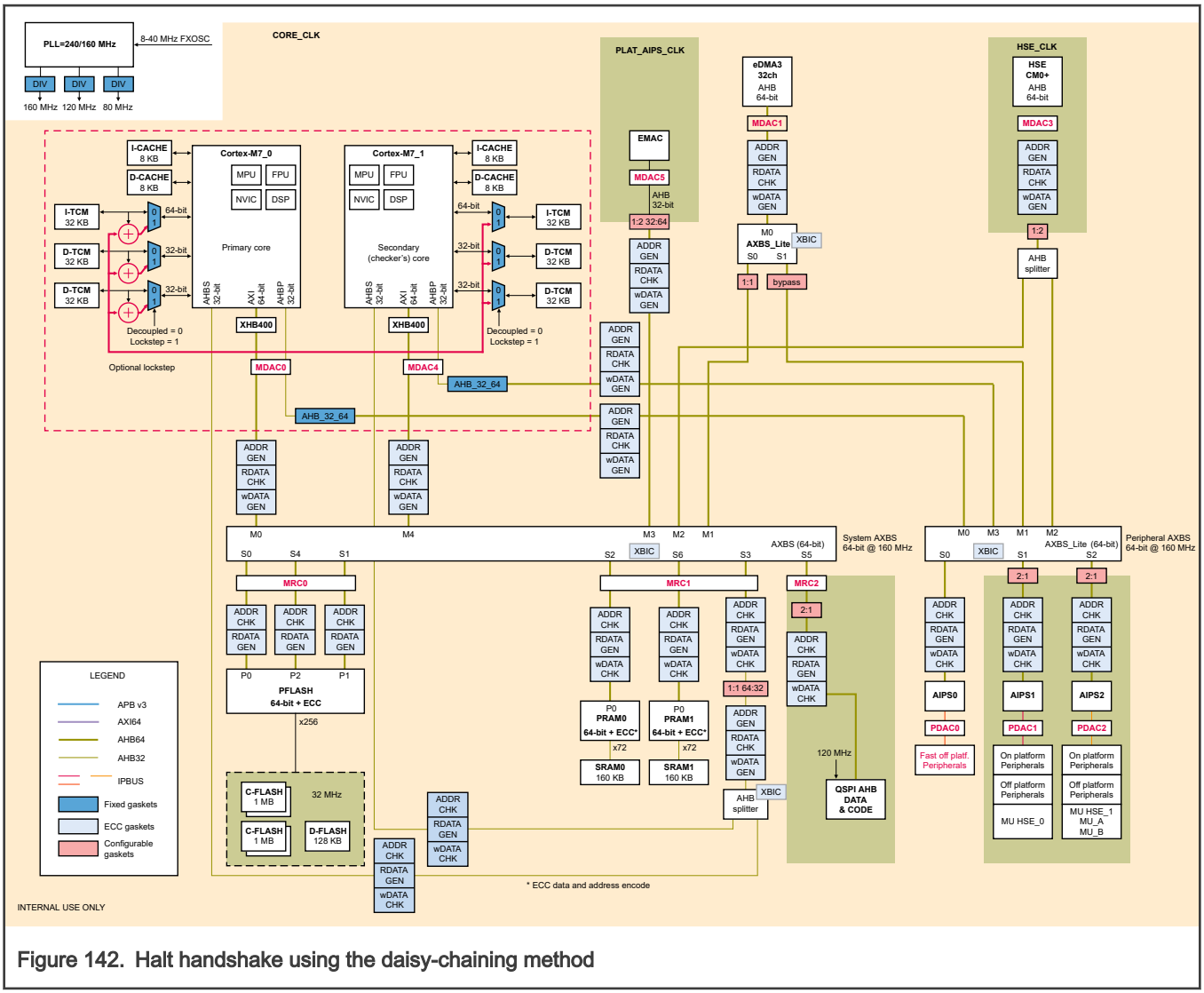


Figure 142. Halt handshake using the daisy-chaining method

31.9 Pad state during reset and after reset

SIUL2 controls the GPIO functionality. It sets to its default state on a functional reset, and ensures that every pad initializes to its default state (see the default configurations and reset states of the chip's GPIO in the IOMUX file attached to this document).

31.10 Reset pin

This chip contains a bidirectional reset pin (RESET_b) indicating the reset state. RESET_b multiplexes with the other functions on port PTA5 (see the IOMUX file attached to this document).

The DCF configuration controls the multiplexing capabilities of RESET_b. The default configuration of the RESET_b port is that of a dedicated reset signal (for example, not multiplexed. See the DCF clients file attached to this document for more information).

The RESET_b pin offers the following uses if you configure it for the reset functionality:

- Acts as an external destructive reset source
- Acts as an indicator for the chip reset sequence for both functional and destructive reset sequences

NOTE

MC_RGM.FES[F_EXR] captures an externally sourced RESET_b assertion for a destructive reset.

The RESET_b pin also indicates an internally asserted reset to external modules. It has a weak internal pullup. In normal Run mode, it keeps the chip out of reset.

31.10.1 Reset pin control during self-test

You can write 1 to MC_RGM.ERCTRL[ERASSERT] to assert RESET_b (writes only in Supervisor mode), before LBIST or MBIST executes. MC_RGM then asserts RESET_b and tristates the GPIO pins placing them in a safe state. Tristating GPIO ensures a safe state for the chip pins when LBIST or MBIST executes. Following BIST, the chip executes a reset sequence. The chip configures again for the application software, before executing the safety function. MC_RGM.ERCTRL[ERASSERT] clears on a functional reset. See Figure 8 for an illustration.

NOTE

Writing 1 to SIUL2.MSCR*n*[SMC] enables the GPIO pins and the chip continues its normal I/O functionality.

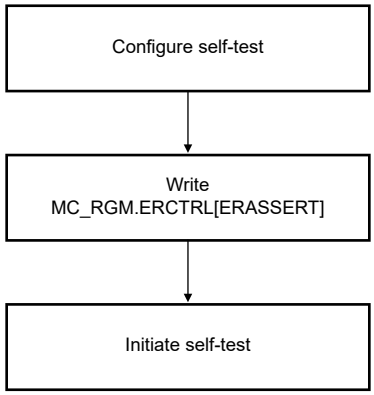


Figure 143. Reset pin control before self-test

Multiple chip configuration scenarios cause RESET_b to react differently after self-test completion:

- You can write 1 to MC_RGM.ERCTRL[ERASSERT] causing RESET_b to assert. This assertion does not impact the reset sequence, and the reset indicates that the chip is not available in Functional mode (although the chip is not in reset sequence).

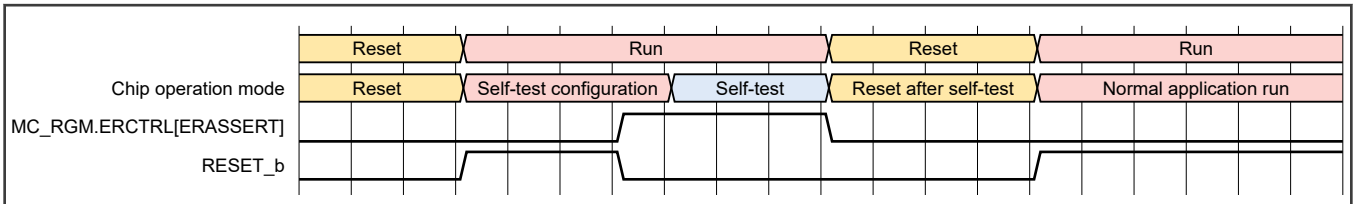


Figure 144. Reaction when MC_RGM.ERCTRL[ERASSERT] = 1

- If MC_RGM.ERCTRL[ERASSERT] = 0 (the default value), the RESET_b pin goes low after self-test completes. After self-test, the chip undergoes a functional reset in which the chip hardware writes 0 to MC_RGM.ERCTRL[ERASSERT]. The RESET_b pin goes high after the reset deasserts.

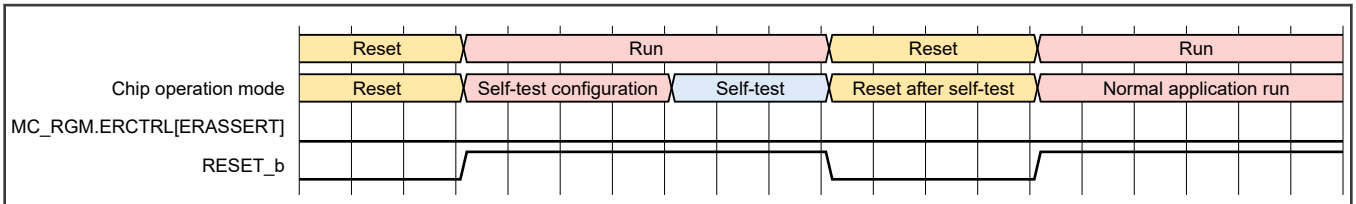


Figure 145. Reaction when MC_RGM.ERCTRL[ERASSERT] = 0

- If MC_RGM.ERCTRL[ERASSERT] = 0 (the default value), but you write 1 to MC_RGM.FBRE[ST_DONE], the RESET_b pin does not assert after self-test completes.

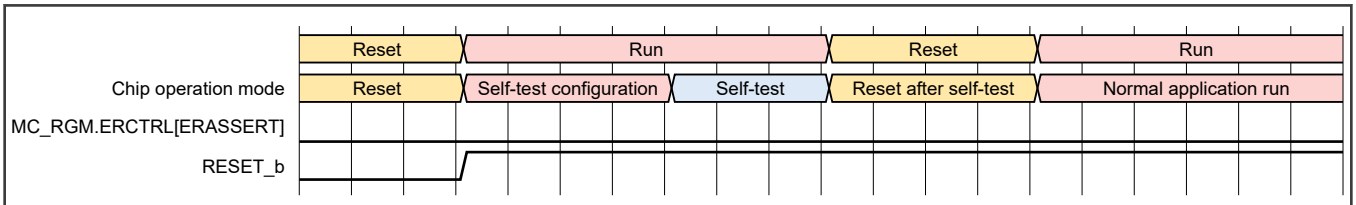


Figure 146. Reaction when MC_RGM.ERCTRL[ERASSERT] = 0 and MC_RGM.FBRE[ST_DONE] = 1

31.11 Reset control in Lockstep mode

In Lockstep mode, a two-cycle delayed lockstep implementation controls the reset to both the application cores, Cortex-M7_0 and Cortex-M7_1. This lockstep implementation means Cortex-M7_0 starts two clock cycles before Cortex-M7_1. You can configure the dcf_client_utest_misc[LOCSTEP_EN] field to control the lockstep (see the DCF clients file attached to this document for more information).

The Cortex-M7 cores consist of two reset domains:

- PORESETn (see PORESETn control in Lockstep mode)
- SYSRESETn (see SYSRESETn control in lockstep)

.

NOTE

Lockstep feature is available in S32K341, S32K342, S32K344, S32K348, S32K358, S32K356, S32K388, and S32K389. For all these variants, Cortex-M7_0 and Cortex-M7_1 are split lock capable. For S32K388 and S32K389, Cortex-M7_2, Cortex-M7_2 checker are in permanent lockstep.

31.11.1 PORESETn control in Lockstep mode

PORESETn includes debug modules that work across chip warm resets. The PORESETn domain resets on any destructive reset event. PORESETn deasserts after two cycles of reset deassertion synchronization delay, as soon as destructive reset sequence exits. In Lockstep mode, the reset deassertion to Cortex-M7_1 is further delayed by two CORE_CLK cycles as shown in PORESETn control in Lockstep mode.

31.11.1.1 PORESETn control in Lockstep mode

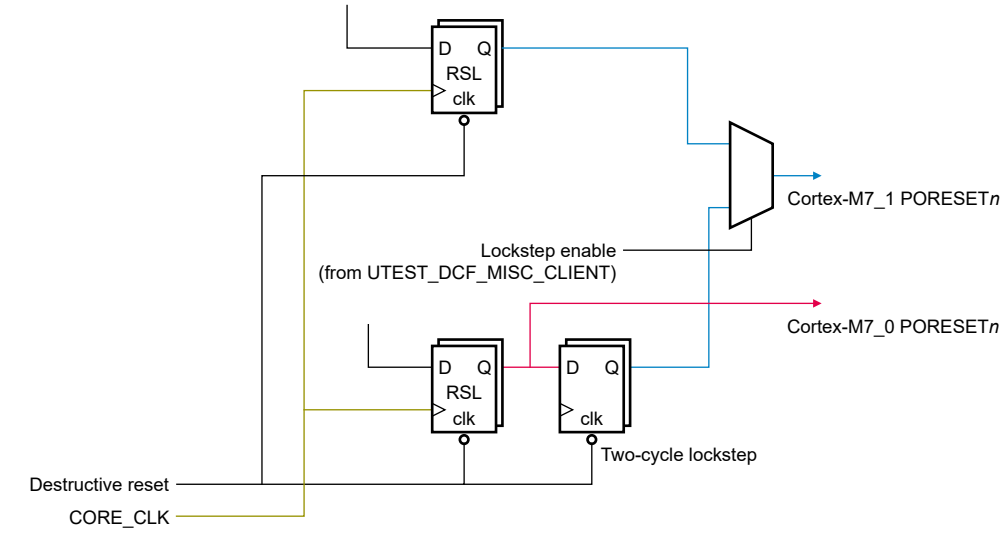


Figure 147. PORESETn control in Lockstep mode

31.11.2 SYSRESETn control in lockstep

Most of the components within the Cortex-M7 cores reside in the SYSRESETn domain, except the debug modules. These components reset on any functional reset event. The SYSRESETn reset remains gated even if it exits the functional reset until MC_ME.PRTN0_COREn_PCONF[CCE] enables the core clocks. The debugger can hold the core's SYSRESETn domain in reset, as described in section "Application core debug from first instruction" in the "Debug Subsystem" chapter.

After a chip's functional reset deasserts, the core clocks are functional, and there is no gating from the debugger, the application core's SYSRESETn is released after a two-cycle reset deassertion synchronizer delay. In Lockstep mode, the reset deassertion to Cortex-M7_1 is delayed by two CORE_CLK cycles, as shown in SYSRESETn control in Lockstep mode.

31.11.2.1 SYSRESETn control in Lockstep mode

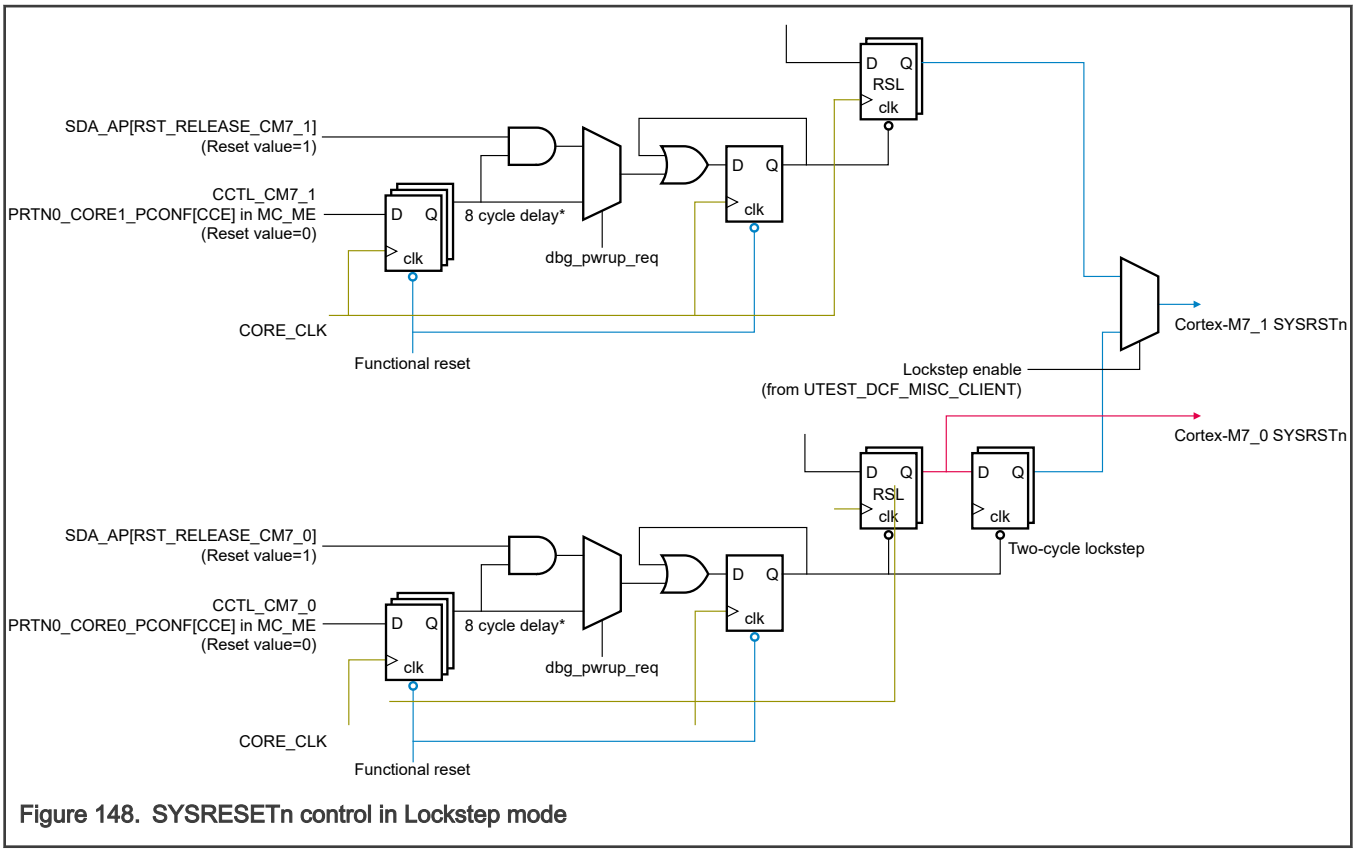


Figure 148. SYSRESETn control in Lockstep mode

31.12 Glossary

DCF	Device configuration format
POR	Power-on reset

Chapter 32
Boot Overview

32.1 Overview

32.1.1 Introduction

This chapter describes the system boot sequence and provides details about boot options.

After the hardware reset sequence completes, the only CPU available is in the HSE subsystem that is referred to as the HSE CPU.

The HSE CPU starts executing firmware code in the HSE code flash memory from a fixed location that contains the SBAF code. This code provides the boot sequence until the control is passed, based on the type of boot:

- Nonsecure boot: Passes control to the customer software that executes on one or all the application cores.
- Secure boot: Passes control to the HSE firmware running on the HSE CPU.

32.2 Appendix

SBAF takes into account the following scenarios to prevent stuck in reset of chip:

- After observing the eight functional resets in the chip, BAF enters Recovery mode sequence to recover the application core’s failing status.
- SBAF does not allow the chip's LC to advance to the OEM_PROD or IN_FIELD stage, if the application does not program CUST_DB_PSWD/A.
- SBAF boots the application from the system-RAM during recovery mode sequence to avoid unpredictable behavior.

32.2.1 Features

The features of SBAF are as follows:

- Supports secure and nonsecure boot modes
- Supports application boot core selection
- Allows chip LC advancement
- Supports debug authorization
- Supports XRDC configuration

32.3 Chip configuration

This section describes the chip configuration details for S32K3xx variants after you program SBAF and clear the HSE firmware feature flag.

If the security is enabled, see the HSE Firmware Reference manual for more information. Please contact NXP sales executive for details.

32.3.1 Memory map

This section explains the memory sections used by SBAF in various configurations.

32.3.1.1 Configuration details when the HSE firmware usage feature flag is disabled

Table 196. Configuration details when the HSE firmware usage feature flag is disabled

Memory section	S32K311	S32K341	S32K312, S32K322, S32K342	S32K314, S32K324, S32K344	S32K328, S32K338, S32K348, S32K358, S32K388	S32K356	S32K389
Flash memory	1 MB	1 MB	2 MB	4 MB	8 MB	6 MB	12 MB **NOTE** See Table 197.
IVT locations in priority order	0040_0000h, 0048_0000h, 1000_0000h ()	0040_0000h, 1000_0000h (256 Bytes)	0040_0000h, 0050_0000h, 1000_0000h (256 Bytes)	0040_0000h, 0050_0000h, 0060_0000h, 0070_0000h, 1000_0000h (256 Bytes)	0040_0000h, 0060_0000h, 0080_0000h, 00A0_0000h, 1000_0000h (256 Bytes)	40_0000h, 60_0000h, 80_0000h, 1000_0000h (256 Bytes)	0040_0000h, 0060_0000h, 0080_0000h, 00A0_0000h, 1000_0000h (256 Bytes)
Reserved	004F_4000h to 004F_FFFFh (48 KB)	005F_4000h to 005F_FFFFh (48 KB)		007F_4000h to 007F_FFFFh (48 KB)	00BF_4000h to 00BF_FFFFh (48 KB)	00A0_0000h to 00BF_FFFFh (2048 KB)	00FF_0000h to FF_FFFFh (64KB)
Application flash memory area	0040_0000h to 004F_3FFFh (976 KB)	0040_0000h to 004F_FFFFh (1024 KB)	0040_0000h to 005F_3FFFh (2000 KB)	0040_0000h to 007F_3FFFh (4048 KB)	0040_0000h to 00BF_3FFFh (8144 KB)	0040_0000h to 9F_FFFFh (6144 KB)	0040_0000h to FE_FFFFh (12,224 KB)
Application data flash memory	1000_0000h to 1000_FFFFh (64 KB)	1000_0000h to 1001_FFFFh (128 KB)					1000_0000h to 1003_FFFFh (256 KB)

Table 197. PFLASH configuration in S32K389

Block start address	Block end address	Size (KB)	Flash memory configuration
0x40_0000	0x4F_FFFF	1024	Program flash (PFC1 Block 0)
0x50_0000	0x5F_FFFF	1024	Program flash (PFC1 Block 1)
0x60_0000	0x7F_FFFF	2048	Program flash (PFC0 Block 0)
0x80_0000	0x9F_FFFF	2048	Program flash (PFC0 Block 1)
0xA0_0000	0xAF_FFFF	1024	Program flash (PFC1 Block 2)
0xB0_0000	0xBF_FFFF	1024	Program flash (PFC1 Block 3)
0xC0_0000	0xDF_FFFF	2048	Program flash (PFC0 Block 2)

Table continues on the next page...

Table 197. PFLASH configuration in S32K389 (continued)

Block start address	Block end address	Size (KB)	Flash memory configuration
0xE0_0000	0xFF_FFFF	2048	Program flash (PFC0 Block 3)
0x1000_0000	0x1003_FFFF	256	Program flash (PFC0 Block 4)
0x1004_0000	0x1007_FFFF	256	Program flash (PFC1 Block 4)

NOTE

For PFLASH configuration in other variants, see the memory map file attached to this document.

32.3.1.2 Configuration details when the HSE_B firmware usage feature flag is enabled

The following table explains the memory sections used by the SBAF in case HSE firmware feature flag is enabled in UTEST.

Table 198. Configuration details when the HSE_B firmware usage feature flag is enabled

Memory section	S32K311	S32K341	S32K312, S32K322, S32K342	S32K314, S32K324, S32K344	S32K328, S32K338, S32K348, S32K358, S32K388	S32K356	S32K389
Flash memory	1 MB	1 MB	2 MB	4 MB	8 MB	6 MB	12 MB **NOTE** See Table 197.
IVT locations in priority order	0040_0000h, 0048_0000h, 1000_0000h (256 Bytes)	0040_0000h, 1000_0000h (256 Bytes)	0040_0000h, 0050_0000h, 1000_0000h (256 Bytes)	0040_0000h, 0050_0000h, 0060_0000h, 0070_0000h 1000_0000h (256 Bytes)	0040_0000h, 0060_0000h, 0080_0000h, 00A0_0000h 1000_0000h (256 Bytes)	0040_0000h, 0060_0000h, 0080_0000h, (256 Bytes)	0040_0000h, 0060_0000h, 0080_0000h, 00A0_0000h, 1000_0000h (256 Bytes)
Reserved	004D_4000h to 004F_FFFFh (176 KB)	005D_4000h to 005F_FFFFh (176 KB)		007D_4000h to 007F_FFFFh (176 KB)	00BD_4000h to 00BF_FFFFh (176 KB)	00A0_0000h to 00BF_FFFFh (2048 KB)	00FD_0000h to 00FF_FFFFh (204 KB)
Application flash memory area when full memory "HSE firmware" is present	0040_0000h to 004D_3FFFh (848 KB)	0040_0000h to 004F_FFFFh (1024 KB)	0040_0000h to 005D_3FFFh (1872 KB)	0040_0000h to 007D_3FFFh (3920 KB)	0040_0000h to 00BD_3FFFh (8016 KB)	0040_0000h to 009F_FFFFh (6144 KB)	0040_0000h to 00FC_FFFFh (12,105 KB)
Application data flash memory	1000_0000h to 1000_FFFFh (64 KB)	1000_0000h to 1001_5FFFh (88 KB)					1000_0000h to 1003_FFFFh (256 KB)

32.3.1.3 AB swap configuration

Table 199. AB swap configuration

Memory section	S32K311	S32K341	S32K312, S32K322, S32K342	S32K314, S32K324, S32K344	S32K328, S32K338, S32K348, S32K358, S32K388	S32K356	S32K389
Flash memory	1 MB	1 MB	2 MB	4 MB	8 MB	6 MB	12 MB **NOTE** See Table 197.
IVT locations in priority order	0040_0000h, 1000_0000h	0040_0000h, 1000_0000h	0040_0000h, 1000_0000h	0040_0000h, 0050_0000h, 1000_0000h	0040_0000h, 0060_0000h, 1000_0000h	0040_0000h, 0060_0000h, 1000_0000h	0040_0000h, 0060_0000h, 1000_0000h
Reserved code area in active block	0045_4000h to 0047_FFFFh (176 KB)	004D_4000h to 004F_FFFFh (176 KB)	004D_4000h to 004F_FFFFh (176 KB)	005D_4000h to 005F_FFFFh (176 KB)	007D_4000h to 007F_FFFFh (176 KB)	0070_0000h to 007F_FFFFh (1024 KB)	009D_0000h to 009F_FFFFh (204 KB)
Reserved code area in passive block	004D_4000h to 004F_FFFFh (176 KB)	005D_4000h to 005F_FFFFh (176 KB)	005D_4000h to 005F_FFFFh (176 KB)	007D_4000h to 007F_FFFFh (176 KB)	00BD_4000h to 00BF_FFFFh (176 KB)	00B0_0000h to 00BF_FFFFh (1024 KB)	00FD_0000h to 00FF_FFFFh (204 KB)
Application flash memory area in active block	0040_0000h to 0045_3FFFh (336 KB)	0040_0000h to 0047_FFFFh (512 KB)	0040_0000h to 004D_3FFFh (848 KB)	0040_0000h to 005D_3FFFh (1872 KB)	0040_0000h to 007D_3FFFh (3920 KB)	0040_0000h to 006F_FFFFh (3072 KB)	0040_0000h to 009C_FFFFFh (5,938 KB)
Application flash memory area in passive block	0048_0000h to 004D_3FFFh (336 KB)	0050_0000h to 0057_FFFFh (512 KB)	0050_0000h to 005D_3FFFh (848 KB)	0060_0000h to 007D_3FFFh (1872 KB)	0080_0000h to 00BD_3FFFh (3920 KB)	0080_0000h to 00AF_FFFFh (2992 KB)	00A0_0000h to 00FC_FFFF (5,961 KB)
Application data flash memory	1000_0000h to 1000_FFFFh (64 KB)	1000_0000h to 1001_FFFFh (128 KB)					1000_0000h to 1003_ffffh (256 KB)

32.4 Common configuration pertaining to the chip

32.4.1 Feature configuration in CUST_DEL Device Life Cycle

The following table describes the status of various features when chip's LC is in the CUST_DEL stage. The application software configures features directly or indirectly.

Table 200. Feature configuration in CUST_DEL Device Life Cycle

Feature	Configuration information	Status	Configurability
OTA functionality		Disabled	Yes Application requests the HSE firmware or SBAF to enable this feature.
HSE firmware usage feature flag	Indicates whether the firmware installation is allowed in the chip. By default, this flag is unprogrammed, and SBAF assumes that the firmware installation is not allowed.	Unprogrammed	Yes To enable this feature, program in the UTEST location (see UTEST flag description for more information).
SBAF firmware		Programmed	No
HSE firmware	You must program the HSE firmware. An encrypted and signed firmware image is always delivered to you.	Not programmed	Yes SBAF can install this feature when the application software requests.
Image vector table		Not programmed	Yes The application software can program this feature.
SWT0		Disabled	By SWT bit in boot configuration word.
Boot sequence	Boot sequence is a nonsecure boot, which means, the SBAF boots the application without any authentication.	Nonsecure boot	Yes It can be changed to secure boot by programming the BOOT_SEQ bit in IVT.
Life Cycle		Customer delivery	Yes Advance by SBAF or HSE firmware when requested by application software.
Application core enablement status	Applications cores are booted in recovery mode sequence at address 2040_0100h.	Recovery mode sequence	Yes Program the required fields in the IVT to enable single or all application cores at the required address.
FIRC frequency value		48 MHz	Yes

Table continues on the next page...

Table 200. Feature configuration in CUST_DEL Device Life Cycle (continued)

Feature	Configuration information	Status	Configurability
			The application can configure after SBAF moves to WFI.
Application debug authorization mode	This mode is password-based. You can program the configuration in UTEST to change the mode to Challenge Response.	Password-based approach	When the HSE firmware feature flag clears, you cannot change Debug Authorization mode. When the HSE firmware feature flag is set, you can request the HSE firmware to change Debug Authorization mode to Challenge Response mode.
Application core debug status	Debug of application cores is enabled in the customer delivery life cycle.	Enabled	No

32.4.2 UTEST memory location usage by SBAF

Table 201. UTEST memory location usage by SBAF

Start address	End address	Size (bytes)	Description	Programmed by	Write protected
1B00_0000h	1B00_0007h	8	HSE firmware feature usage flag. See UTEST flag description for more information.	Application software	No
1B00_0040h	1B00_0047h	8	Unique Chip Identifier (UID)	NXP	No
1B00_0050h	1B00_0057h	8	FXOSC configuration flag See UTEST flag description for more information.	Application software	No
1B00_0080h	1B00_009Fh	32	Debug password (CUST_DB_PSWD_A) After the HSE firmware usage feature flag clears, SBAF uses this location to run the debug authorization feature. SBAF copies this value in the application expected response register, which derives the HSE expected response register. The size of this register is 16 bytes, and 1B00_0090h – 1B00_009Fh is reserved.	Application software	No

Table continues on the next page...

Table 201. UTEST memory location usage by SBAF (continued)

Start address	End address	Size (bytes)	Description	Programmed by	Write protected
			After the HSE firmware usage feature flag sets, the HSE firmware programs the password at a different location. DCM scans the password during reset only and retains the password in standby.		

See the DCF clients file attached to this document for more information.

32.4.3 UTEST flag description

32.4.3.1 HSE firmware usage feature flag

This flag indicates to SBAF that the application intends to use the HSE firmware on the chip. By default, this flag is unprogrammed, and SBAF assumes that the HSE firmware installation is not allowed in the secure samples. However, if application allows the installation of the HSE firmware, the HSE firmware usage feature flag can be programmed in the UTEST at the 1B00_0000h location.

Table 202. HSE firmware usage feature flag

Field type	Description	Remarks
Size	64 bits	
Default value	0xFFFFFFFFFFFFFFFF	SBAF does not allow the installation of HSE firmware.
UTEST location	0x1B000000	
Configurability	Application software in CUST_DEL lifecycle	
Enable flag	Program any value other than the default value	This enables the installation of the HSE firmware.

32.4.3.2 Crystal oscillator configuration flag

See the below table for description of the flag:

Table 203. Flag fields

Field type	Description
Size	64 bits
Default value	FFFF_FFFF_FFFF_FFFFh (Boot via FIRC)
UTEST location	1B00_0050h
Configurability	Application software in any LC

Table 204. Crystal oscillator configuration flag in UTEST

	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
R	FXOSC_ENABLE_MAGIC_NUMBER															
W																
	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
R																
W																

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	GMSEL				EOCV											
W																
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CRYSTAL_OSCILLATOR_FREQUENCY															
W																

Table 205. Crystal oscillator configuration bit definition

Field	Description
63 - 33	FXOSC_ENABLE_MAGIC_NUMBER AAAA_5555h – Enable (FXOSC.CTRL[OSCON] = 1h) FFFF_FFFFh – Disable (FXOSC.CTRL[OSCON] = 0h)
31 - 28	Crystal overdrive protection (GMSEL) For values = 0h or Fh Uses the default value, Ch.
27 - 20	End Of Count Value (EOCV) For value = 0h Uses the default value, 9Dh.
19 - 16	Reserved
15 - 0	CRYSTAL_OSCILLATOR_FREQUENCY Frequency value of used external crystal oscillator in kHz. The valid crystal frequency range is 8000–40000 kHz.

32.5 Image vector table

The following section describes the fields in the image vector table that the application programs. SBAF scans the IVT after the chip is out of reset. The structure is 256 bytes in size. This structure contains application cores start addresses. IVT must be programmed at least at one of the locations described in Chip configuration.

After reset, SBAF searches for the first valid IVT starting from the lowest address. If there are multiple valid IVT at IVT locations at the same time, the IVT with lowest address is used.

Table 206. Image vector table

Address offset	Size in bytes	Content	Comments
00h	4	Image vector table marker	marks the starting of the image vector table location. Its value must be 5AA5_5AA5h.
04h	4	Boot configuration word	Indicates the configuration word that allows you to select the various configurations in which you can boot the chip. See the upcoming section for more information.
08h	4	Reserved	
0Ch	4	Cortex-M7_0 core start address	Specifies the boot address of the Cortex-M7_0 core in the code flash memory area. It must honor core Vector Table Offset Register (VTOR) alignment restrictions.
10h	4	Reserved	
14h	4	Cortex-M7_1 core start address	Specifies the boot address of the Cortex-M7_1 core in the code flash memory area. It must honor core VTOR register alignment restrictions.
18h	4	Reserved	
1Ch	4	Cortex-M7_2 core start address	Specifies the boot address of the Cortex-M7_2 core in the code flash memory area. It must honor core VTOR register alignment restrictions.
20h	4	Reserved	
24h	4	Address of LC configuration	Specifies the address of the configuration word that allows you to advance the LC. See the upcoming section for more information.
28h	4	Cortex-M7_3 core start address	Specifies the boot address of the Cortex-M7_3 core in the code flash memory area. It must honor core VTOR register alignment restrictions.

32.5.1 Boot configuration word

This register informs SBAF to allow booting of selected applications.

Table 207. Boot configuration register

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W																

Table continues on the next page...

Table 207. Boot configuration register (continued)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R								CM7_3_ENABLE	RESERVED	RESERVED	APP_SWT_INIT	RESERVED	BOOT_SEQ	CM7_2_ENABLE	CM7_1_ENABLE	CM7_0_ENABLE
W																

Table 208. Boot configuration register field definition

Field	Description
31 – 9	Reserved
8	CM7_3_ENABLE Indicates whether the Cortex-M7_3 application core clock is gated after boot. 0b - Gated 1b - Ungated
7	Reserved
6	Reserved
5	APP_SWT_INIT Controls the SWT0 enablement before passing the control to the application core(s). 0b - Disables 1b - Enables SBAF initializes SWT0 before enabling the application cores. SBAF scans this field only when the HSE firmware usage feature flag is enabled and the BOOT_SEQ field is 0.
4	Reserved
3	BOOT_SEQ Controls the boot flow of the application when HSE Firmware usage feature flag is enabled. 0b - Nonsecure boot: SBAF starts the application image without any authentication in parallel to the HSE firmware. 1b - Secure boot: The HSE firmware executes the application image after authentication. SBAF only starts the HSE firmware after successful authentication.
2	CM7_2_ENABLE Indicates whether the Cortex-M7_2 application core clock is gated after boot. 0b - Gated 1b - Ungated

Table continues on the next page...

Table 208. Boot configuration register field definition (continued)

1	CM7_1_ENABLE Indicates whether the Cortex-M7_1 application core clock is gated after boot. 0b - Gated 1b - Ungated
0	CM7_0_ENABLE Indicates whether the Cortex-M7_0 application core clock is gated after boot. 0b - Gated 1b - Ungated

32.5.2 Address LC configuration word

This field allows you to advance the LC. To advance LC, you must program a valid 32-bit wide value at an address given in IVT at an address offset of 24h. This address must be 4 bytes aligned and should not lie in HSE reserved area.

The following table shows the valid values for advancement in the next LCs. For all other values at the given address, LC advancement is not allowed.

Table 209. Valid values for LC advancement

Life cycle stage	Valid values for LC advancement
OEM_PROD	DADA_DADAh
IN_FIELD	BABA_BABAh

Depending on the HSE firmware feature flag, the application password on the location must program before LC advancement; otherwise, SBAF does not attempt LC advancement.

The chip provides an LC mechanism for an irreversible progression of restrictions to access the chip's security-related content. You cannot reverse the chip's LC, so it is only possible to mature the chip. SBAF advances the chip through the LC:

- CUST_DEL --> OEM_PROD or IN_FIELD
- OEM_PROD --> IN_FIELD

To advance the LC through SBAF involves inserting the LC configuration word address in the IVT. SBAF issues a destructive reset on successful advancement. If the chip is found in the same LC as the IVT indicated, you can ignore LC advancement.

32.5.3 Structure definition of image vector table

Application can use the following structure to configure the IVT.

```
typedef const struct image_vector_table
{
uint32_t Header; /*Header of IVT Structure */
uint32_t BootConfig; /*Boot Configuration Word */
const uint32_t Reserved1; /* Reserved */
const uint32_t * CM7_0_StartAddress; /*Start Address of Application on CM7_0 Core */
const uint32_t Reserved2; /* Reserved */
const uint32_t * CM7_1_StartAddress; /*Start Address of Application on CM7_1 Core */
const uint32_t Reserved3; /* Reserved */
const uint32_t * CM7_2_StartAddress; /*Start Address of Application on CM7_2 Core */
const uint32_t * Reserved4; /* Reserved */
```

```
const uint32_t * LCConfig; /*Address of LC configuration Word */
const uint32_t * CM7_3_StartAddress; /*Start Address of Application on CM7_3 Core */
const uint32_t * HseFwHeaderStartAddress; /*Start address of HSE firmware pink header image */
const uint32_t * Reserved5[2U]; /* Reserved */
const uint32_t * XrdcMDACConfiguration; /*XRDC input MDAC configuration Data */
const uint32_t * Reserved6; /* Reserved */
const uint32_t * Recoveryapp_StartAddress; /*Start Address of Application Core for Recovery
Application */
const uint32_t * LengthRecoevryApp; /*Length of Application */
const uint8_t * Reserved7[156U]; /* Reserved */
const uint8_t * Random_IV[12U]; /*Random IV for GMAC calculation of IVT */
const uint8_t * GMAC[16U]; /*GMAC of the IVT. Reserved for Unsecure BAF */
}ivt_t;
```

32.6 Boot flow

Below diagram explains boot sequence flow of SBAF.

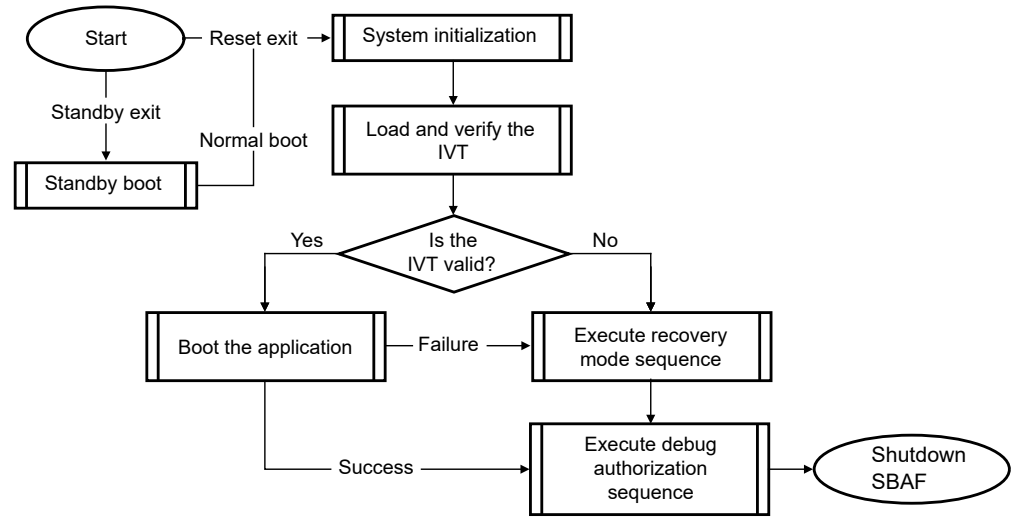


Figure 149. Boot flow

32.7 Standby boot

SBAF supports boot from standby exit. Chip register DCM.DCMRWF5 (address 402AC610h) supports boot on standby exit. This register clears on POR. See the "Device Configuration Module (DCM)" chapter in the S32K3xx reference manual for more information.

You must program this register before entering Standby mode.

There are two types of boot mode on exit from standby.

- Fast Standby
- Normal boot on exit from standby

In Fast Standby mode, the SBAF boots the Cortex-M7_0 core and halts the HSE CPU. The flow of Standby Boot is explained below:

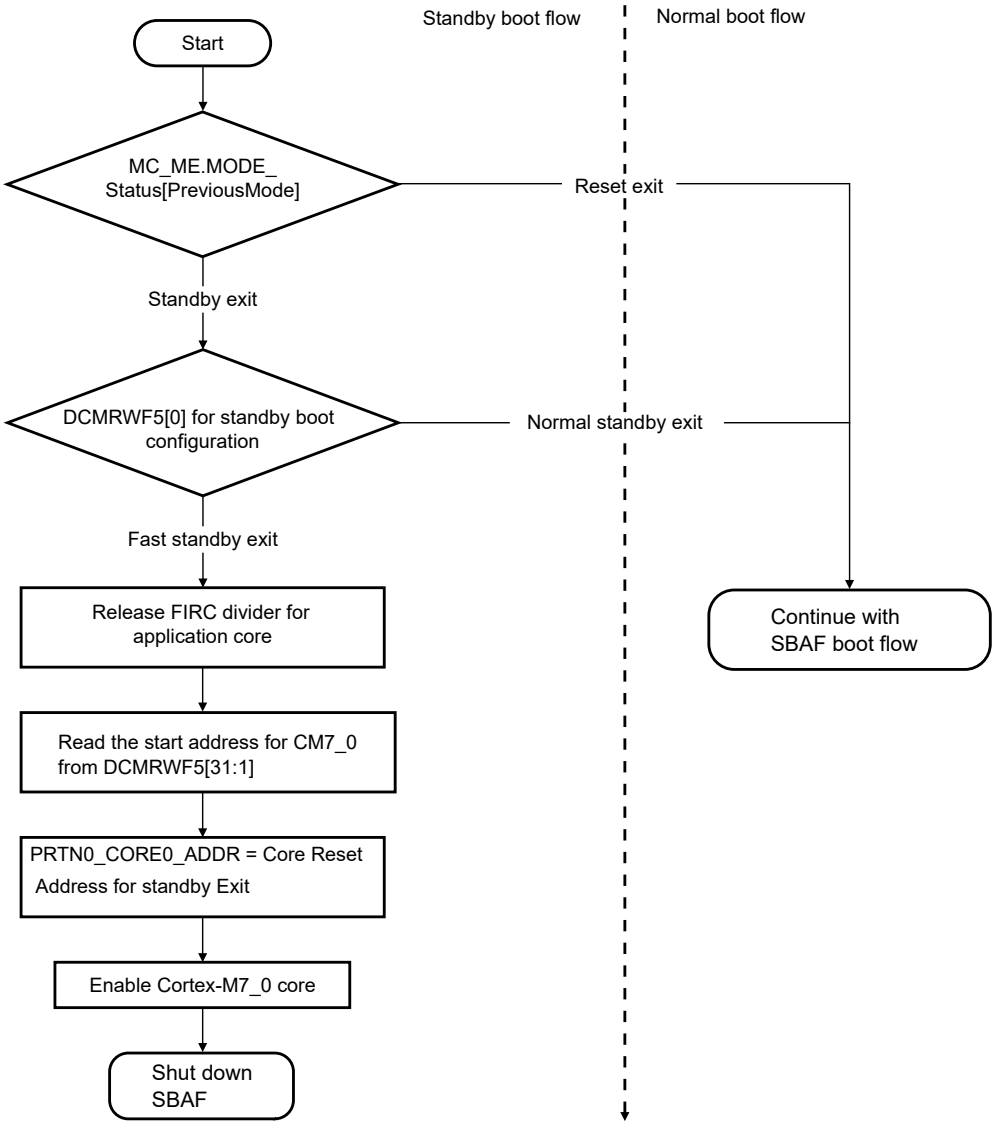


Figure 150. Standby boot flow

32.8 Reduced clock mode configuration

If you use clocking option B (Reduced clock mode configuration), the application sets the “dcf_client_utest_misc” DCF record to enable Reduced Clock mode. See the DCF clients file attached to the S32K3xx reference manual for more information on DCF records.

After reset, SBAF checks for DCMROF21[HSE_CLK_MODE_OPTION]. If you configure this field for clocking option B, the SBAF configures the following dividers in MC_CGM:

- MUX_0_DC_1
- MUX_0_DC_2

Below figure explains steps for reduced clock mode configuration steps by SBAF.

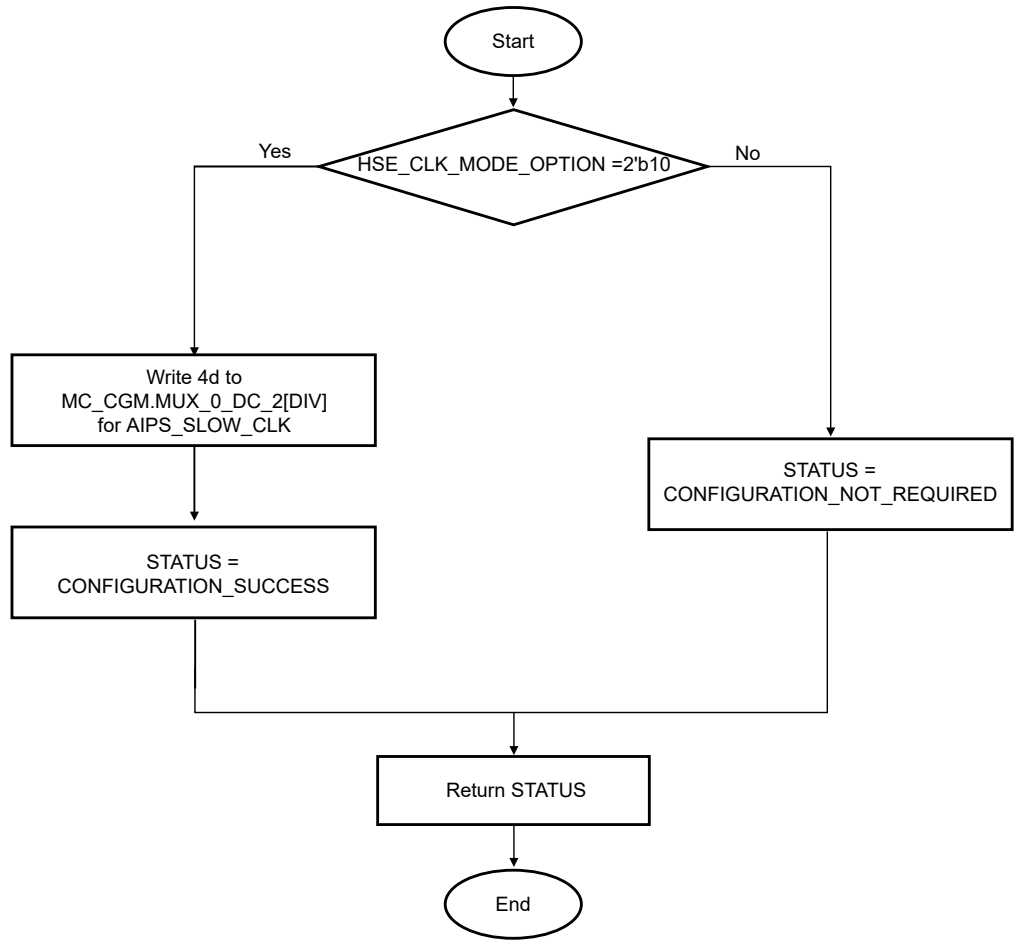


Figure 151. Reduced clock mode configuration

32.9 Debug authorization

You must program CUST_DB_PSWD_A at location 1B00_0080h. The application core debug is always password-based if the HSE firmware usage feature flag is cleared. See UTEST memory location usage by SBAF for more information.

32.10 FIRC divider register control

HSE.CONFIG_REG_GPR, at register address 4039C064h, controls the FIRC divider. This register is write-protected by default for the FIRC divider, and you cannot modify its settings.

After SBAF executes WFI, it provides write access to HSE.CONFIG_REG_GPR[FIRC_DIV_SEL], and you can configure this register. Before accessing this register, you must wait for the SBAF to enter WFI by reading core status register of HSE CPU (PRTN0_CORE2_STAT).

32.11 Application boot

The HSE firmware is responsible for securely booting the application image and not the SBAF. If secure boot is not requested (by writing 0 to the BOOT_SEQ field in Boot configuration word in IVT structure), SBAF always loads the application image, without authentication, in all chips, LC. By default, SBAF releases the reset of every core that you configure to enable.

Following flow chart explains the steps of application boot performed by SBAF.

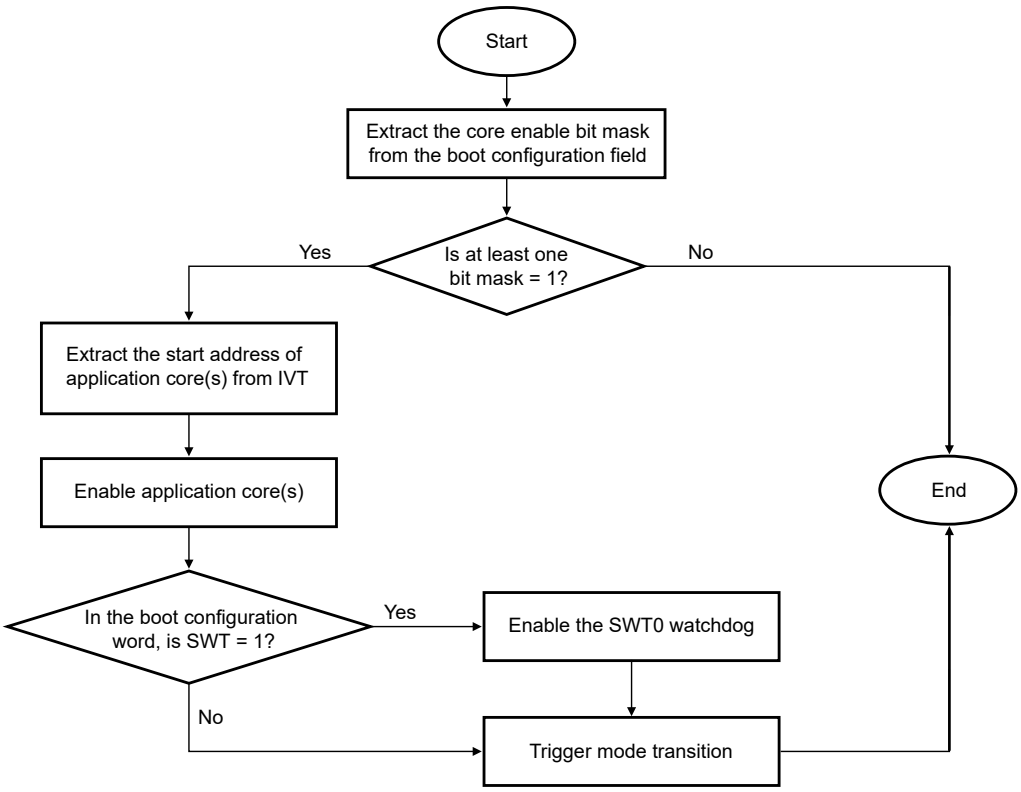


Figure 152. Application boot sequence

Before configuring HSE_CLK, you must wait for the SBAF to enter WFI by reading core status register of HSE CPU (PRTN0_CORE2_STAT).

32.12 Recovery mode sequence

This feature allows you to program the application image in LC = MCU_PROD and LC = CUST_DEL and debug the reason of corruption of IVT and re-program the IVT in other LC.

The following scenarios take place when SBAF executes the recovery mode sequence.

- Valid IVT is not found (corrupted or not programmed).
- SBAF does not boot the application.
- Boot configuration word in IVT does not program the application enablement field.
- The application issued more than eight functional resets and Disable Recovery mode on functional reset field is not set in DCM.DCMRWP1.
- The application issued more than eight destructive resets and Disable Recovery mode on destructive reset field is not set in DCM.DCMRWP1.

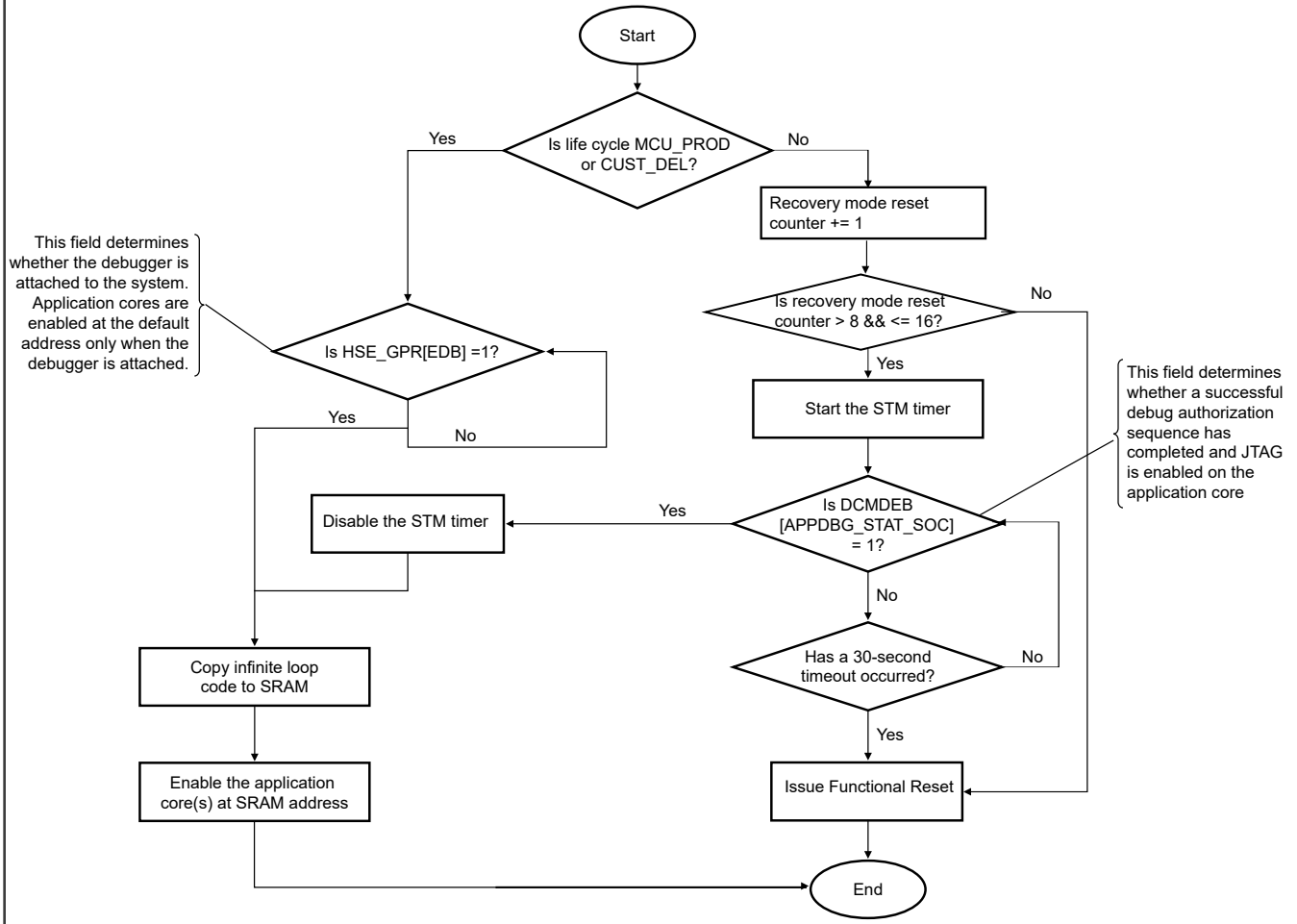


Figure 153. Recovery mode sequence

The infinite loop with the WFI code copies to SRAM1 that is 2040_0100h, and the code size is 16 bytes. To prevent prefetching errors in the application cores, place the infinite loop instruction at 2040_012Ch.

32.13 XRDC configuration

SBAF ensures that it has access to its resources and the application does not have access to area reserved for SBAF.
The following sections describe the XRDC default configuration values when you clear the HSE firmware usage feature flag.
Application should configure its XRDC and enable the XRDC by itself as SBAF does not enable the XRDC.

32.13.1 XRDC configuration of MDAC

Following table list down the default configuration of MDAC by SBAF.

Table 210. XRDC configuration of MDAC

Serial No.	Chip variant	MDAC register HSE CPU	Value	Domain number assigned to HSE CPU
1	S32K31x	MDA_W0_3_DFMT0	C000_0001h	1

Table continues on the next page...

Table 210. XRDC configuration of MDAC (continued)

Serial No.	Chip variant	MDAC register HSE CPU	Value	Domain number assigned to HSE CPU
2	S32K32x and S32K34x	MDA_W0_3_DFMT0	C000_0002h	2
3	S32K35x and S32K33x	MDA_W0_3_DFMT0	C000_0003h	3
4	S32K388 and S32K389	MDA_W0_3_DFMT0	C000_0004h	4

32.13.2 XRDC configuration of MRC

Following table list down the MRC used by SBAF.

Table 211. XRDC configuration of MRC

MRC number	Region descriptor number	Remarks
0	13	Reserved for HSE
0	14	Reserved for HSE
0	15	Reserved for HSE

32.13.3 XRDC configuration of PDAC

SBAF configures and lock the following peripherals for its use. SBAF provides all permissions to all domains.

Table 212. XRDC configuration of PDAC

Serial No.	Peripheral name	Peripheral PDAC number
1	Flash controller alternate	155
2	Flash memory alternate	188
3	HSE_GPR	231

32.14 BAF flash programming controls

The platform flash controller generates an exception of read-while-write if you simultaneously perform read and write on the same block. HSE CPU uses the Configuration PE Lock Register (CONFIG_PE_LOCK) in the HSE space to control the block program and erase for application.

BAF locks the high addressing code flash memory block during its execution, and it is cleared when the HSE CPU enters WFI. See the "Chip Configuration" chapter for the address of the high addressing code flash memory area.

Before programming, erasing, or executing from this address space, the application core polls for PRTN0_CORE2_STAT[WFI] to ensure that the HSE CPU is in the WFI state.

For boot sequence 1 or flash synchronization with the HSE firmware, see the HSE Firmware Reference Manual.

Below table explains PE lock bits setting in CONFIG_PE_LOCK register of HSE GPR during SBAF execution.

Table 213. PE lock fields setting in HSE.CONFIG_PE_LOCK

Chip	UTEST block	Data flash block	Code flash block 3	Code flash block 2	Code flash block 1	Code flash block 0
S32K3x1	1	0	NA	NA	NA	1
S32K3x2	1	0	NA	NA	1	1
S32K3x4	1	0	1	0	0	1
S32K3x8	1	0	1	0	0	1
S32K389	1	0	1	0	0	1

32.15 Status registers for application usage

This section explains various status registers to provide status information to the application.

32.15.1 SBAF version information

The SBAF version is a 64-bit field. The application can read the SBAF from address 4039_C020h. The following table describes the version information.

Table 214. SBAF version information

Bits	Field name	Description
0 – 7	Reserved	Reserved
8 - 15	SOC_TYPE_ID	This field represents the SBAF firmware, which is targeted for S32K3XX chip variant. Values of this field are: 0x5 – used for HSE-B S32K344/S32K314/S32K324 0xB - used for HSE-B S32K310 0xC - used for HSE-B S32K311/S32K341 0xD - used for HSE-B S32K312/S32K322/S32K342 0xE - used for HSE-B S32K358/S32K356/S32K348/S32K328/S32K328 0x10- used for HSE-B S32K388 0x11- used for HSE-B S32K389
16 - 31	FW_TYPE	This field represents the SBAF firmware type. Values of this field are: 0 – used for standard generic firmware targeting all customers 1-7 – Reserved >=8 used for Custom 1, Custom 2 (for example: Custom 1 = customer X's project A, Custom 2 = customer Y's project B)
32 - 39	Reserved	Reserved
40 - 47	BASELINE_NUMBER	Incremented when the compatibility with the previous version is broken.

Table continues on the next page...

Table 214. SBAF version information (continued)

48 - 55	INCREMENTAL_ NUMBER	Incremented when new features are added but compatibility kept.
56 - 63	RC_NUMBER	Release Candidate Number

32.15.2 DCM.DCMRWP1

The application can disable Recovery mode entry by SBAF after programming bits 23 and 22 of DCM.DCMRWP1.

Table 215. DCM.DCMRWP1 (address 0x402AC400)

Bits	Number of bits	R/W access by application	Description
24-31	8		Reserved
23	1	R/W	Disable Recovery Mode On Destructive Reset Indicates that this field resets by default, and SBAF allows Recovery mode sequence if the application issues > 8 destructive resets. The application can set this field to disable Recovery mode when the application issues > 8 destructive resets.
22	1	R/W	Disable Recovery Mode On Functional Reset Indicates that this field resets by default, and SBAF allows Recovery mode sequence if the application issues > 8 functional resets. The application can set this field to disable Recovery mode when the application issues > 8 functional resets.
21	1	R	Reserved
16-20	5	R	Recovery Mode Reset Counter Indicates that to enable Recovery mode functionality for the OEM_PROD and IN_FIELD LC stages, SBAF increments this counter when a functional or destructive reset is issued.
15	1	R	Reserved
11-14	4	R	Destructive Reset Counter Indicates that SBAF increments this counter when a destructive reset is issued.
0-10	11	R	Reserved

32.15.3 Status bits on HSE.GPR

SBAF writes HSE.GPR at address 4039_C028h to show status information as described in the following table.

Table 216. Status bits on HSE.GPR

Bit	Description
0	Indicates that SBAF presents and boots the HSE FW.

Table continues on the next page...

Table 216. Status bits on HSE.GPR (continued)

Bit	Description
1-4	Reserved.
5	Indicates that SBAF boots the application cores in Recovery mode sequence.
6	Indicates that SBAF performs the debug authentication.
7-31	Reserved.

32.16 Interrupt and exception handling

- **Interrupt handling:** No special interrupt handling routines are required during the boot process. Interrupts are disabled during SBAF execution.
- **Exception handling:** SBAF enters the recovery mode sequence after enabling debug authorization. After eight consecutive functional resets or destructive resets from Application Firmware, the device enters the recovery mode sequence.
- **Boot target watchdog:** SBAF enables/disables SWT0 watchdog with default timeout, that is 25 ms according to the boot configuration word, before enabling the application core(s). It is expected that the application services this watchdog before expiration.

32.17 Hardware modules used by SBAF

- **MC_ME:** SBAF uses MC_ME to enable the application cores, mode switch, and other operations during its execution.
- **FXOSC:** SBAF configures FXOSC according to the crystal oscillator configuration flag in UTEST.
- **Clock Generation Module (MC_CGM):** SBAF configures MC_CGM in Reduced clock mode configuration.
- **DCM:** SBAF uses the DCM to identify the Life Cycle, standby boot mode configuration, lockstep, and clock mode configuration.
- **HSE_GPR:** SBAF configures hardware protection, program erase lock, FIRC divider control, and SBAF version number.
- **XRDC:** SBAF configures the XRDC module.
- **Flash Module:** SBAF always perform the write and erase operation on alternate interface. See Chip configuration for details of flash memory usage by SBAF.

32.18 Hardware IP registers details modified by SBAF

Below table summarizes the registers which are modified by SBAF when HSE Firmware usage feature flag is disabled.

Table 217. Hardware IP registers modified by SBAF

IP	Register Name	Default value (hex)	Modified value (hex)
MC_ME	MC_ME.PRTN1_COFB1_STAT	1CFE_2FFCh	FXOSC and MC_CGM clocks are enabled by default. However, for 120 MHz clock requests, SBAF ensures if the MC_CGM clock is enabled.
	MC_ME.PRTN0_COREx_ADDR	0040_0000h	SBAF updates this address if you request the application

Table continues on the next page...

Table 217. Hardware IP registers modified by SBAF (continued)

IP	Register Name	Default value (hex)	Modified value (hex)
			core 0 boot during the normal boot sequence in IVT or Recovery mode (0x2040012C).
	MC_ME.PRTN0_COREx_PCONF	0000_0000h	The core clock is enabled when the recovery mode sequence is executed or when the clock is enabled in Boot configuration word.
	MC_ME.PRTN0_COREx_PUPD	0000_0000h	
	MC_ME.CTL_KEY	0000_5AF0h	SBAF updates this register if SBAF boots any one of the application core. 0000_5AF0h and then 0000_A50Fh.
FXOSC	FXOSC.CTRL	019D_00C0h	See Crystal oscillator configuration flag
MC_RGM	MC_RGM.DRET	0000_0000h	0000_000Fh
SWT_0	SWT_0.CR	FF00_010Ah	FF00_000Bh, when SWT_0 is enabled in the boot configuration word and FF00_000Ah when the SWT_0 is disabled in the boot configuration word.
	SWT_0.IR	0000_0000h	0000_0001h

32.19 Glossary

BAF Boot assist flash

CUST_DEL Chip's life cycle stage, customer delivery

IN_FIELD Chip's life cycle stage, in field

IVT Image vector table

LC Chip's life cycle—limits by design the configuration and debug/test possibilities of the chip for in-field usage

MCU_PROD Chip's life cycle stage, MCU production

OEM_PROD Chip's life cycle stage, OEM production

OTA Over the air

SBAF Secure boot assist flash

Chapter 33
Reset Generation Module (MC_RGM)

33.1 Chip-specific MC_RGM information

33.1.1 MC_RGM configuration

The "Reset sources—POR, Destructive, and Functional" section of the "Reset Overview" chapter provides information about MC_RGM's reset sources. The chapter also provides details about the chip's reset architecture.

NOTE

The ERCTRL register configuration takes several cycles to be effective. Any further access to MC_RGM must happen after at least nine AIPS_SLOW_CLK cycles of writing to ERCTRL.

Table 218. Register fields and applicability

Register	Bit field	Chips where applicable
Functional /External Reset Status Register (FES)	SWT1_RST	S32K324, S32K322, S32K328, S32K338, S32K358, S32K356, S32K388, S32K389
	SWT2_RST	S32K338, S32K388, S32K389
	SWT3_RST	S32K388, S32K389
Functional Event Reset Disable Register (FERD)	D_SWT1_RST	S32K324, S32K322, S32K328, S32K338, S32K358, S32K356, S32K388, S32K389
	D_SWT2_RST	S32K338, S32K388, S32K389
	D_SWT3_RST	S32K388, S32K389

33.1.2 Functional reset entry timer implementation

The default timeout value of the functional reset entry timer is 2048 clocks of MC_RGM clock (FIRC). If the RGM entry sequence hangs, then POR_WDG would trigger and the status of this functional reset entry sequence timeout is indicated at chip-level status register. The status of timeout in such case, is indicated at DCM.DCMROPP1[28]. There is no impact to the device behavior if the functional reset entry sequence gets completed within the POR_WDG timeout window.

33.1.3 S32K3xx reset state machine

The reset sequence of the S32K3xx products is depicted in the figure below:

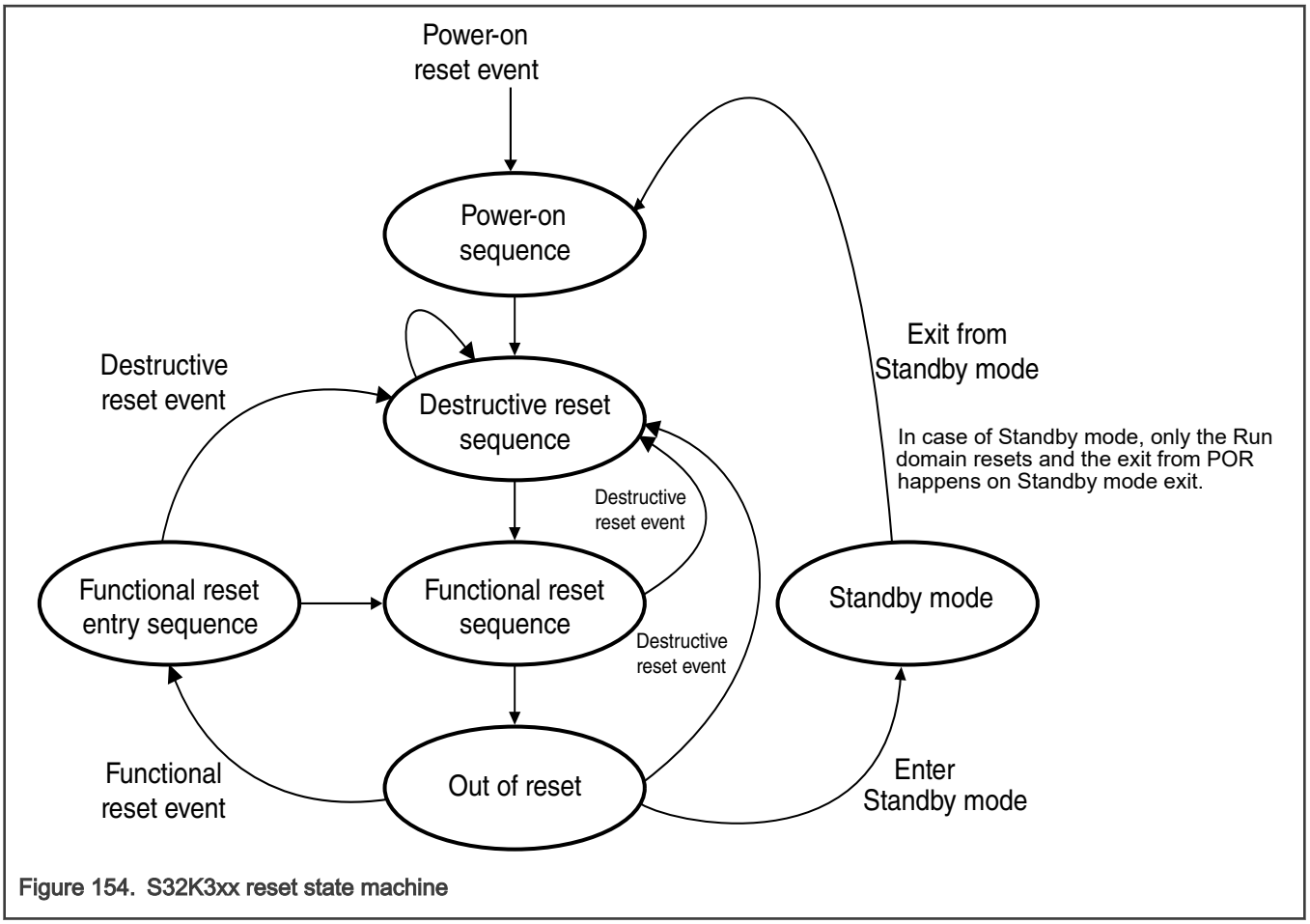


Figure 154. S32K3xx reset state machine

33.2 Introduction

The Reset Generation Module (MC_RGM) centralizes the different reset sources and manages the reset sequence of the chip. It provides a register interface and the reset sequencer. There are various registers available in this module to monitor and control the chip reset sequence.

The following figure shows the block diagram of MC_RGM.

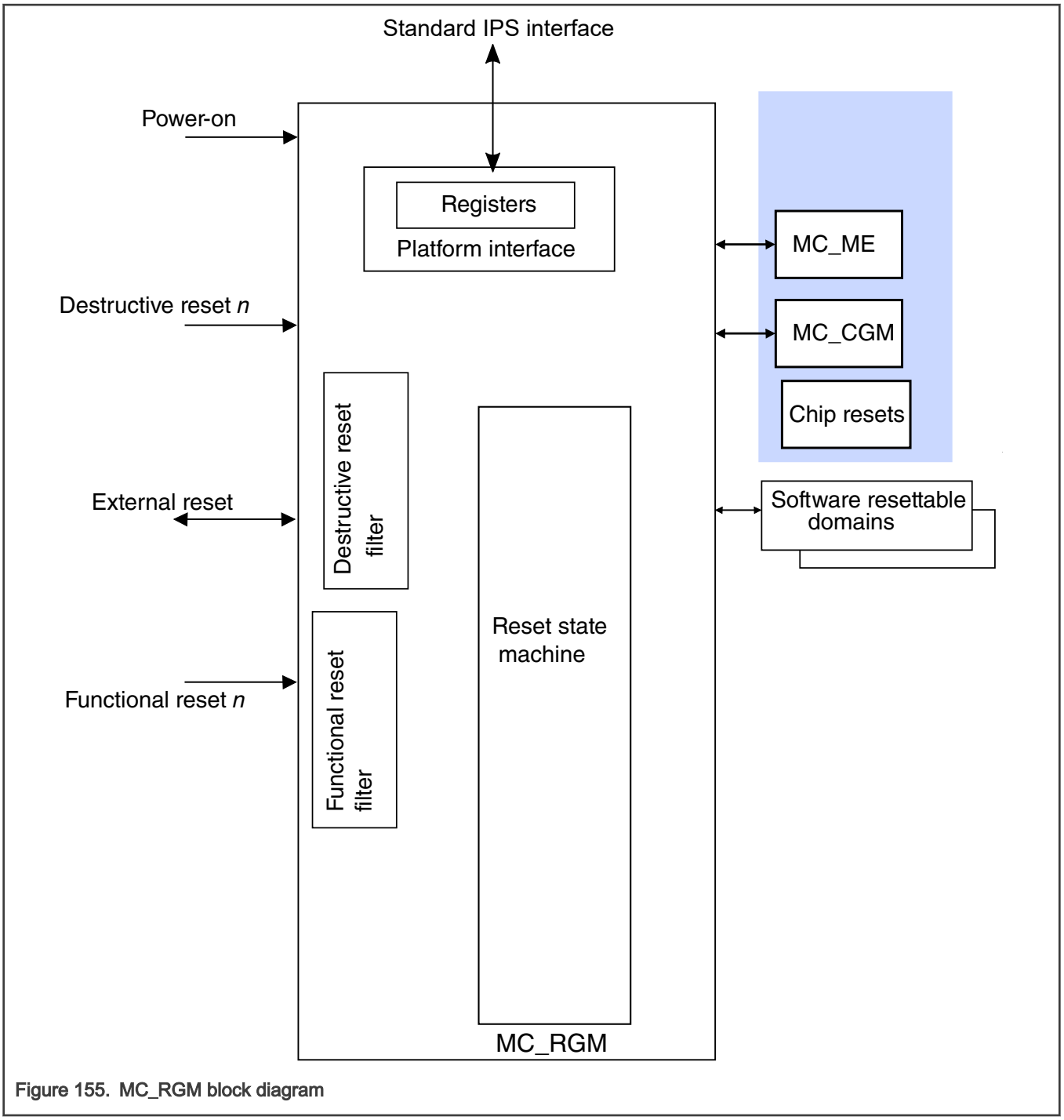


Figure 155. MC_RGM block diagram

33.3 Features

Here are the key features of MC_RGM:

- Destructive and functional reset management
- Capturing the reset sources for each reset sequence (reset status flags)
- Assertion the RESET_B pin to propagate the reset sequence out of chip
- Configurable escalation of recurring 'functional' resets to 'destructive' reset

- Configurable escalation of recurring 'destructive' resets to keep the chip in the reset state until the next power-on reset
- Software controllable reset assertion
- Pad safe state control generation

33.4 Reset sources

The reset sources are organized in three categories: power-on, destructive, and functional.

A power-on reset source is associated with an event typically related to power-up or low-voltage scenarios. When a power-on reset occurs, the full reset sequence is applied to the chip. This resets the full chip, including the MC_RGM, and the memory content must be considered to be invalid.

A destructive reset source is associated with an event related to a critical, usually hardware, error or dysfunction. When a destructive reset event occurs, the full reset sequence is applied to the chip. This resets the full chip ensuring a safe start-up state for both digital and analog modules, and the memory content must be considered to be invalid.

A functional reset source is associated with an event related to a less-critical, usually non-hardware, error or dysfunction. When a functional reset event occurs, a partial reset sequence is applied to the chip. In this case, most digital modules are reset normally, while the state of analog modules or specific digital modules as well as the system memory content is preserved.

33.5 External signal description

The MC_RGM interfaces with the RESET_B pin.

The following table describes the signals that are connected to the I/O pad ring.

Table 219. MC_RGM external signals

Signal name	Reset value	Description
RESET_B	0	Active low external reset. A bidirectional reset pin indicating the reset state.

33.6 RESET_B pin assertion and pin safe state control

The MC_RGM asserts the RESET_B pin when the device is in a reset sequence, and it remains asserted until the end of the reset sequence. During this reset sequence, most of the chip's pins are safe/pad stated according to the values shown in the IOMUX table/spreadsheet. Note that the safe state values may vary according to the reset sequence type.

In addition, the MC_RGM has a feature to assert the RESET_B pin through software, without initiating a reset sequence. This is achieved by writing 1b to the ERCTRL[ERASSERT]. When this occurs, most of the chip's pins are safe-stated according to the values shown in the IOMUX table/spread sheet. The RESET_B assertion and pin safe-stating remain active until the end of the next reset sequence.

This features has to be used only with selftest of the main reset domain.

33.7 Functional description

33.7.1 Reset state machine

The main role of MC_RGM is the generation of the reset sequence that ensures that the correct parts of the chip are reset based on the reset source event.

For each reset event, immediately after it is captured by the MC_RGM, the following takes place:

1. The corresponding reset event status bit is set in the MC_RGM_DES and MC_RGM_FES registers.
2. The pins are put into their default states

3. The RESET_B pin is asserted.

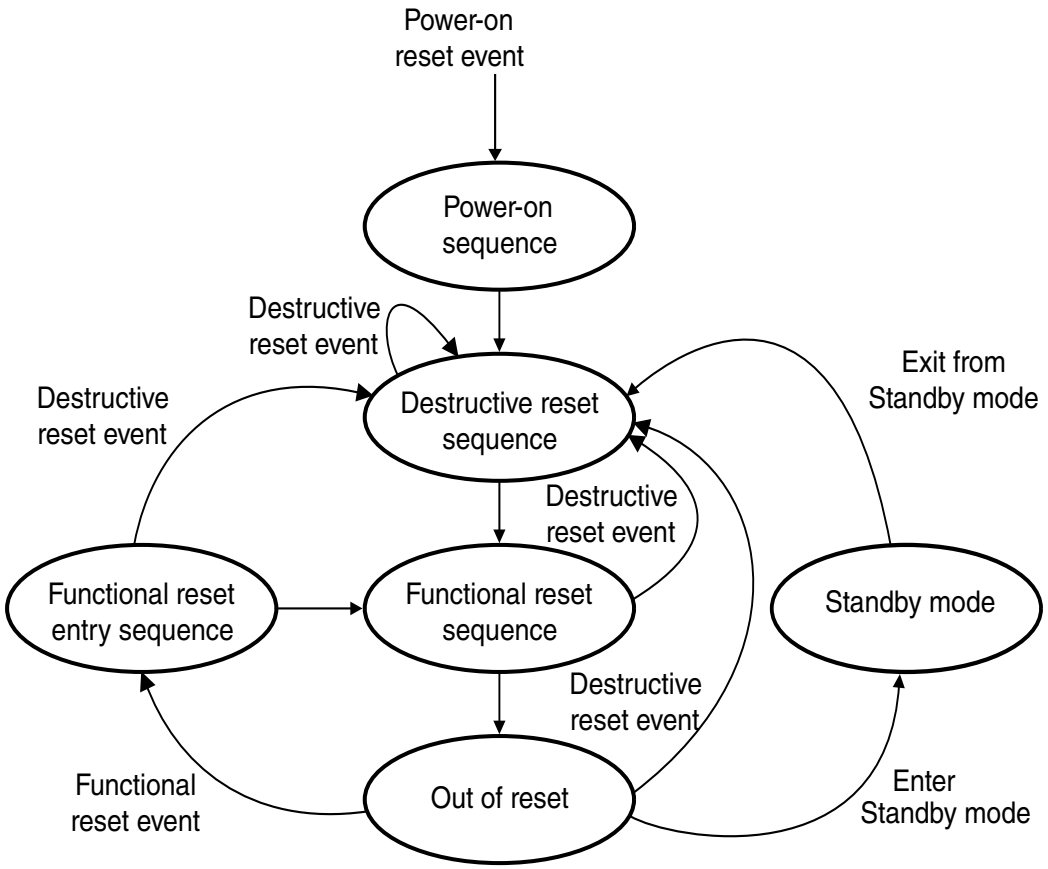


Figure 156. Reset sequence

NOTE

See chip-specific MC_RGM information for chip-specific reset sequence details.

33.7.1.1 Power-on reset sequence

A reset is always generated when the power-on reset source is asserted, and it has priority over all other reset sources. Such a power-on reset forces the reset state machine to enter the power-on sequence resulting in the assertion of all reset signals. The reset state machine starts progressing when the following two conditions are verified:

1. All the power-on reset events are cleared
2. The MC_RGM's clock source (the FIRC) has started up and stabilized

If a power-on reset event has occurred, the DES[F_POR] is set.

The power-on reset cannot be demoted by the software.

33.7.1.2 Destructive reset sequence

The 'Destructive reset sequence' is comprised of a number of phases, where DEST0 is the first phase and is followed by DEST1 and so forth.

This phase is entered immediately from any phase on a power-on, standby reset sequence, or enabled destructive reset event . A destructive reset counter starts immediately on entry in the DEST0 phase. The DEST0 state is exited to the DEST1 state on the rising edge of FIRC_CLK immediately after all of the following conditions have been established:

- The DEST0 duration time has expired
- All destructive reset inputs are cleared

The DEST0 state is immediately exited to the power-on state if a power-on reset event occurs.

The reset state machine exits the destructive reset sequence and enters the functional reset sequence when:

- All the destructive reset events are cleared.
- All the processes that take place during the destructive reset sequence have completed. For details, see Reset chapter.
- The 'destructive' reset escalator counter has not reached the value in DRET[DRET].

33.7.1.3 Functional reset sequence

There are two functional reset sequence, the functional reset entry sequence and the functional reset exit sequence.

33.7.1.3.1 Functional reset entry sequence

The functional reset entry sequence is only entered when a functional reset event occurs during the idle phase.

If a functional reset event occurs during an ongoing reset sequence, the corresponding event status flag is set, and the RESET_B pin is asserted per the reset event's configuration. However, the reset sequence is not influenced, and it continues to progress without interruption.

Functional reset is not asserted during functional reset entry sequence.

The functional reset entry sequence is exited to the DEST0 on the next rising edge of FIRC if a destructive reset event has occurred. The sequence immediately enters the power up sequence if a POR event occurs

In all other cases, the sequence exits to the first stage of functional reset exit sequence.

33.7.1.3.2 Functional reset exit sequence

The functional reset exit sequence is entered either on exit from the destructive reset sequence or on completion of the functional reset entry sequence. The reset state machine exits this sequence and enters the idle phase on verification of the following:

- All the functional reset events are cleared.
- All the processes that take place during the functional reset sequence have completed. For details, see Reset chapter.

If a functional reset event occurs during an ongoing reset sequence, the corresponding event status flag is set, and the RESET_B pin is asserted per the reset event's configuration. However, the reset sequence is not influenced, and it continues to progress without interruption.

33.7.1.4 Idle phase

This is the final phase and is entered on exit from the functional reset exit sequence . When this phase is reached, MC_RGM releases control of the system to the platform and waits for the new reset events that can trigger a reset sequence.

33.7.2 Destructive resets

A destructive reset indicates that an event has occurred after which critical register or memory content can no longer be guaranteed.

The status flag associated with a given destructive reset event (Destructive Event Status Register (DES)) is set when the destructive reset is asserted and the power-on reset is not asserted. It is possible for multiple status bits to be set simultaneously and the software determines which reset source is the most critical for the application.

The low-voltage detector threshold ensures that when the reset corresponding to the core supply low-voltage detect is enabled, the supply is sufficient to have the destructive event correctly propagated through the digital logic. Therefore, if a given destructive reset is enabled, MC_RGM ensures that the associated reset event is correctly triggered to the full system.

An enabled destructive reset triggers a reset sequence starting from the beginning of DEST0.

33.7.3 External reset

MC_RGM manages the external reset coming from RESET_B. The detection of a falling edge on RESET_B starts the reset sequence from the beginning of the destructive reset entry sequence.

The status flag associated with the external reset falling edge event (the FES[F_EXR]) is set when the external reset is asserted and the power-on reset is not asserted.

33.7.4 Functional resets

A functional reset indicates that an event has occurred after which it can be guaranteed that critical register and memory content is still intact.

The status flag associated with a given functional reset event (Functional /External Reset Status Register (FES)) is set when the functional reset is asserted and the power-on reset is not asserted. It is possible for multiple status bits to be set simultaneously and the software determines which reset source is the most critical for the application.

An enabled functional reset triggers a reset sequence starting from the beginning of the functional reset entry sequence.

33.7.5 Alternate event generation

MC_RGM provides alternative events to be generated on reset source assertion. When a reset source is asserted, MC_RGM normally enters the reset sequence. Alternatively, it is possible for some reset source events to be converted from a reset to an interrupt request issued to the core. Alternate event selection for a given reset source is made through the RGM_FERD register as shown in the following table.

Table 220. Functional Reset Disable Register (RGM_FERD) field descriptions

RGM bit FERD value	Generated event
0	Reset
1	Interrupt request

The alternate event is cleared by deasserting the source of the request (that is, at the reset source that caused the alternate request) and also clearing the appropriate RGM_FES status bit.

33.7.6 RESET_B assertion control

The software indicates to the MC_RGM that the RESET_B is to be asserted by writing to the **ERASSERT** bit in the RGM_ERCTRL register. When this bit is set by the software, RESET_B gets asserted. Setting of this field does not impact the reset sequence in any way.

An example where the ERCTRL[ERASSERT] field could be set by the software is when entering the self test sequence, during which RESET_B is to be asserted. This indicates the chip is not available in the functional mode although a reset sequence is not in progress. The deassertion of RESET_B is not controlled by the software. Instead, the RESET_B pin remains asserted until the next time the chip exits a reset sequence.

ERASSERT bit is also cleared during the reset sequence.

MC_RGM asserts the external reset if the reset sequence is triggered by one of the following:

- A power-on reset
- A destructive reset event
- A functional reset event

In this case, external reset is asserted until all conditions for the exiting reset sequence have been met, with the exception of the RESET_B assertion check

33.7.7 Functional reset escalation

Functional reset escalation can be used to generate a destructive reset if a number of functional resets is occurred between software writes to the RGM_FRET register. This function is enabled by writing a non-zero value to the FRET field of this register.

After the functional reset escalation is enabled, MC_RGM increases a counter on each functional reset that causes a reset sequence to be initiated (which means, entrance into FUNC0 from the IDLE phase). This counter is cleared on a write of any value to the RGM_FRET register and on any power-on or destructive reset. If the counter reaches the value in the FRET field of the RGM_FRET register, MC_RGM asserts a destructive reset.

The following figure shows the functional reset escalation counter.

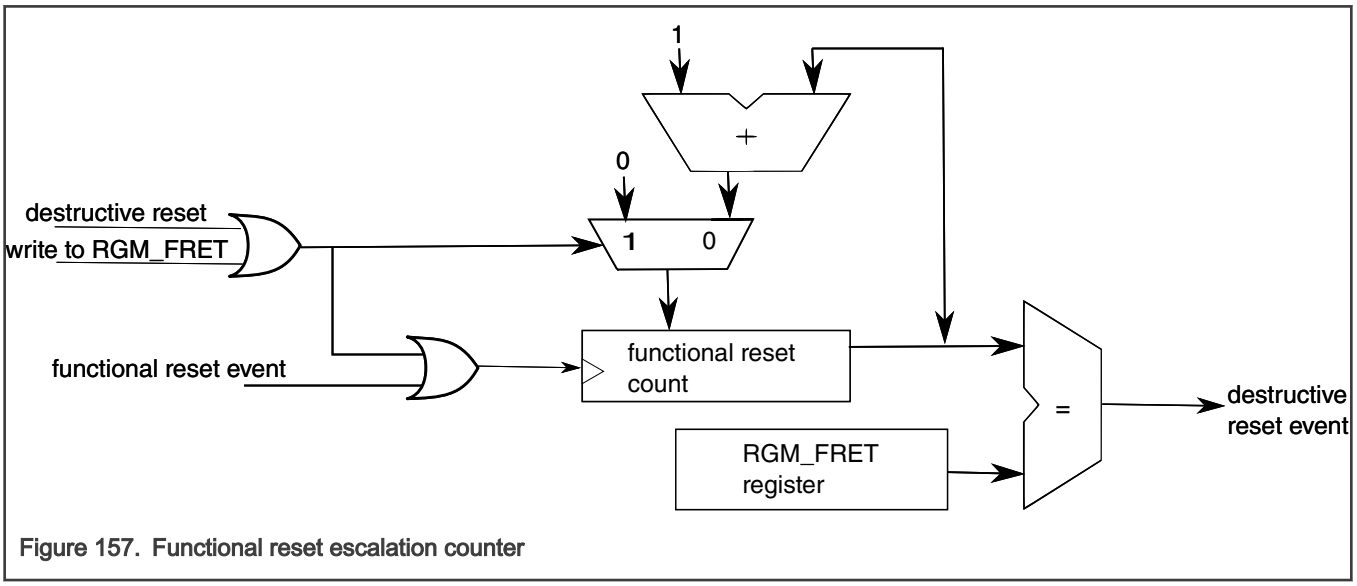


Figure 157. Functional reset escalation counter

NOTE

Functional counter increments for each reset source for which escalation is enabled. For details, see "Reset sources" table in the Reset chapter.

33.7.8 Destructive reset escalation

Destructive reset escalation can be used to keep the chip in the reset state until the power-on triggers a reset sequence if a number of destructive resets are occurred between software writes to the RGM_DRET register. This function is enabled by writing a non-zero value to the DRET field of this register.

After destructive reset escalation is enabled, MC_RGM increases a counter on each destructive reset that is enabled. . This causes a reset sequence to be initiated (that is, entrance into DEST0 from the idle phase or any other reset phase) or an ongoing reset sequence to restart (that is, entrance into DEST0 from any other reset phase). This counter is cleared on a write of any value to the RGM_DRET register and on any power-on reset. If the counter reaches the value in the DRET field of the RGM_DRET register, MC_RGM enters reset DEST0 and stays there until the next power-on reset occurs .

The following figure shows the destructive reset escalation counter.

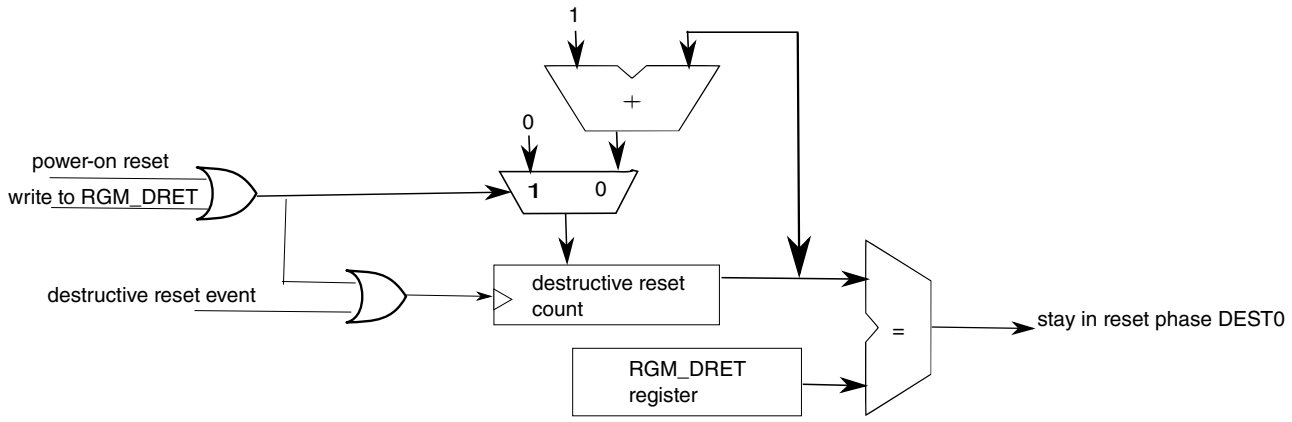


Figure 158. Destructive reset escalation counter

NOTE

Destructive counter increments for each reset source for which escalation is enabled. For details, see "Reset sources" table in the Reset chapter.

33.8 MC_RGM register descriptions

Access to the following locations do not generate transfer error:

- 2Ch

All registers can be accessed as read/write in supervisor mode.

33.8.1 MC RGM Register Map memory map

MC_RGM base address: 4028_C000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Destructive Event Status Register (DES)	32	RW	0000_0001h
8h	Functional /External Reset Status Register (FES)	32	RW	0000_0000h
Ch	Functional Event Reset Disable Register (FERD)	32	RW	0000_0000h
10h	Functional Bidirectional Reset Enable Register (FBRE)	32	RW	0000_0000h
14h	Functional Reset Escalation Counter Register (FREC)	32	RW	0000_0000h
18h	Functional Reset Escalation Threshold Register (FRET)	32	RW	0000_000Fh
1Ch	Destructive Reset Escalation Threshold Register (DRET)	32	RW	0000_0000h
20h	External Reset Control Register (ERCTRL)	32	RW	0000_0000h
24h	Reset During Standby Status Register (RDSS)	32	RW	0000_0000h

33.8.2 Destructive Event Status Register (DES)

Offset

Register	Offset
DES	0h

Function

This register contains the status of the 'destructive' reset sources. This register can be accessed as read/write in supervisor mode and read-only in user mode. Register bits are cleared on write '1'. This register is reset only on power-on reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	DEBUG_D...	SW_DEST	0	0	0	0	0	0	0	0	0	0	HSE_SNV...	HSE_TMP...	CM7_COR...
W		W1C	W1C											W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SYS_DIV...	HSE_CLK...	0	AIPS_PL...	0	CORE_CL...	PLL_LOL	FXOSC_F...	0	MC_RGM_...	0	STCU_URF	FCCU_FTR	0	0	F_POR
W	W1C	W1C		W1C		W1C	W1C	W1C		W1C		W1C	W1C			W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Fields

Field	Function
31 —	Reserved
30 DEBUG_DEST	Flag for 'Destructive' Reset DEBUG_DEST 0b - 'Destructive' reset event DEBUG_DEST has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Destructive' reset event DEBUG_DEST has occurred.
29 SW_DEST	Flag for 'Destructive' Reset SW_DEST 0b - 'Destructive' reset event SW_DEST has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Destructive' reset event SW_DEST has occurred.
28 —	Reserved
27	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
26 —	Reserved
25 —	Reserved
24 —	Reserved
23 —	Reserved
22 —	Reserved
21 —	Reserved
20 —	Reserved
19 —	Reserved
18 HSE_SNVS_RST	Flag for 'Destructive' Reset HSE_SNVS_RST 0b - 'Destructive' reset event HSE_SNVS_RST has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Destructive' reset event HSE_SNVS_RST has occurred.
17 HSE_TMPR_RST	Flag for 'Destructive' Reset HSE_TMPR_RST 0b - 'Destructive' reset event HSE_TMPR_RST has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Destructive' reset event HSE_TMPR_RST has occurred.
16 CM7_CORE_CLK_FAIL	Flag for 'Destructive' Reset CM7_CORE_CLK_FAIL 0b - 'Destructive' reset event CM7_CORE_CLK_FAIL has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Destructive' reset event CM7_CORE_CLK_FAIL has occurred.
15 SYS_DIV_FAIL	Flag for 'Destructive' Reset SYS_DIV_FAIL

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - 'Destructive' reset event SYS_DIV_FAIL has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Destructive' reset event SYS_DIV_FAIL has occurred.
14 HSE_CLK_FAIL	Flag for 'Destructive' Reset HSE_CLK_FAIL 0b - 'Destructive' reset event HSE_CLK_FAIL has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Destructive' reset event HSE_CLK_FAIL has occurred.
13 —	Reserved
12 AIPS_PLAT_CLK_FAIL	Flag for 'Destructive' Reset AIPS_PLAT_CLK_FAIL 0b - 'Destructive' reset event AIPS_PLAT_CLK_FAIL has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Destructive' reset event AIPS_PLAT_CLK_FAIL has occurred.
11 —	Reserved
10 CORE_CLK_FAIL	Flag for 'Destructive' Reset CORE_CLK_FAIL 0b - 'Destructive' reset event CORE_CLK_FAIL has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Destructive' reset event CORE_CLK_FAIL has occurred.
9 PLL_LOL	Flag for 'Destructive' Reset PLL_LOL 0b - 'Destructive' reset event PLL_LOL has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Destructive' reset event PLL_LOL has occurred.
8 FXOSC_FAIL	Flag for 'Destructive' Reset FXOSC_FAIL 0b - 'Destructive' reset event FXOSC_FAIL has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Destructive' reset event FXOSC_FAIL has occurred.
7 —	Reserved
6 MC_RGM_FRE	Flag for 'Destructive' Reset MC_RGM_FRE 0b - 'Destructive' reset event MC_RGM_FRE has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Destructive' reset event MC_RGM_FRE has occurred.

Table continues on the next page...

Table continued from the previous page...

Field	Function
5 —	Reserved
4 STCU_URF	Flag for 'Destructive' Reset STCU_URF 0b - 'Destructive' reset event STCU_URF has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Destructive' reset event STCU_URF has occurred.
3 FCCU_FTR	Flag for 'Destructive' Reset FCCU_FTR 0b - 'Destructive' reset event FCCU_FTR has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Destructive' reset event FCCU_FTR has occurred.
2 —	Reserved
1 —	Reserved
0 F_POR	Flag for power-on reset **NOTE** If this field is set, ignore all the fields of Destructive Event Status Register (DES) and Functional /External Reset Status Register (FES) registers at power up. 0b - No power-on event has occurred since the last clear. 1b - A power-on event has occurred.

33.8.3 Functional /External Reset Status Register (FES)

Offset

Register	Offset
FES	8h

Function

This register contains the status of the 'functional' and external reset sources. This register can be accessed as read/write in supervisor mode and read-only in user mode. Register fields are cleared on write '1' if the triggering event has already been cleared at the source. This register is reset only on power-on reset.

NOTE

- The startup self-test functional reset event to MC_RGM initiates a functional reset sequence after which the FES[ST_DONE] is set. This field is sticky in nature and gets cleared if software clears this field.
- If functional reset escalation to destructive reset is disabled, then the status of this register must be ignored if the fields of Destructive Event Status Register (DES) other than DES[F_POR] are set.
- If functional reset escalation to destructive reset is enabled and if the fields of Destructive Event Status Register (DES), other than DES[F_POR], are set, then based on these fields user should check if the cause of the destructive reset was due to functional reset escalation or if it was triggered directly by a destructive reset source, in which case FES needs to be ignored.
- See chip-specific MC_RGM information for applicability of the FES[ST_DONE] field.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	DEBUG_F...	SW_FUNC	0	0	0	0	0	0	0	0	HSE_BOO...	0	0	0	HSE_SWT...
W		W1C	W1C									W1C				W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	PLL_AUX	0	SWT3_RST	JTAG_RST	SWT2_RST	SWT1_RST	SWT0_RST	0	ST_DONE	FCCU_RST	0	0	F_EXR
W				W1C		W1C	W1C	W1C	W1C	W1C		W1C	W1C			W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30 DEBUG_FUNC	Flag for 'Functional' Reset DEBUG_FUNC 0b - 'Functional' reset event DEBUG_FUNC has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Functional' reset event DEBUG_FUNC has occurred.
29 SW_FUNC	Flag for 'Functional' Reset SW_FUNC 0b - 'Functional' reset event SW_FUNC has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Functional' reset event SW_FUNC has occurred.
28 —	Reserved
27	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
26 —	Reserved
25 —	Reserved
24 —	Reserved
23 —	Reserved
22 —	Reserved
21 —	Reserved
20 HSE_BOOT_RST	Flag for 'Functional' Reset HSE_BOOT_RST 0b - 'Functional' reset event HSE_BOOT_RST has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Functional' reset event HSE_BOOT_RST has occurred.
19 —	Reserved
18 —	Reserved
17 —	Reserved
16 HSE_SWT_RST	Flag for 'Functional' Reset HSE_SWT_RST 0b - 'Functional' reset event HSE_SWT_RST has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Functional' reset event HSE_SWT_RST has occurred.
15 —	Reserved
14	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
13 —	Reserved
12 PLL_AUX	Flag for 'Functional' Reset PLL_AUX 0b - 'Functional' reset event PLL_AUX has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Functional' reset event PLL_AUX has occurred.
11 —	Reserved
10 SWT3_RST	Flag for 'Functional' Reset SWT3_RST 0b - 'Functional' reset event SWT3_RST has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Functional' reset event SWT3_RST has occurred.
9 JTAG_RST	Flag for 'Functional' Reset JTAG_RST 0b - 'Functional' reset event JTAG_RST has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Functional' reset event JTAG_RST has occurred.
8 SWT2_RST	Flag for 'Functional' Reset SWT2_RST 0b - 'Functional' reset event SWT2_RST has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Functional' reset event SWT2_RST has occurred.
7 SWT1_RST	Flag for 'Functional' Reset SWT1_RST 0b - 'Functional' reset event SWT1_RST has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Functional' reset event SWT1_RST has occurred.
6 SWT0_RST	Flag for 'Functional' Reset SWT0_RST 0b - 'Functional' reset event SWT0_RST has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Functional' reset event SWT0_RST has occurred.
5 —	Reserved
4 ST_DONE	Flag for 'Functional' Reset ST_DONE

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - 'Functional' reset event ST_DONE has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Functional' reset event ST_DONE has occurred.
3 FCCU_RST	Flag for 'Functional' Reset FCCU_RST 0b - 'Functional' reset event FCCU_RST has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Functional' reset event FCCU_RST has occurred.
2 —	Reserved
1 —	Reserved
0 F_EXR	Flag for External Reset External reset is a source of destructive reset. 0b - No external reset event has occurred since either the last clear or the last power-on reset assertion. 1b - An external reset event has occurred.

33.8.4 Functional Event Reset Disable Register (FERD)

Offset

Register	Offset
FERD	Ch

Function

This register provides dedicated fields to disable functional reset sources. When any of these reset sources are disabled, the associated functional event is demoted to trigger an interrupt request. This register can be accessed as read/write in supervisor mode and read-only in user mode. Each byte can be written to only once after a destructive or power-on reset and this register is reset only on power-on and any destructive reset.

NOTE

It is important to clear the Functional /External Reset Status Register (FES) before writing 1 to any of the fields in this register. Otherwise a interrupt request may occur.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	D_DEBUG...	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	D_SWT3_...	D_JTAG_...	D_SWT2_...	D_SWT1_...	D_SWT0_...	0	0	D_FCCU_...	0	0	Reserved
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30 D_DEBUG_FUNC	DEBUG_FUNC Disable Control 0b - Functional reset event DEBUG_FUNC triggers a reset sequence. 1b - Functional reset event DEBUG_FUNC generates an interrupt request.
29 —	Reserved
28 —	Reserved
27 —	Reserved
26 —	Reserved
25 —	Reserved
24 —	Reserved
23 —	Reserved
22	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
21 —	Reserved
20 —	Reserved
19 —	Reserved
18 —	Reserved
17 —	Reserved
16 —	Reserved
15 —	Reserved
14 —	Reserved
13 —	Reserved
12 —	Reserved
11 —	Reserved
10 D_SWT3_RST	SWT3_RST Disable Control 0b - Functional reset event SWT3_RST triggers a reset sequence. 1b - Functional reset event SWT3_RST generates an interrupt request.
9 D_JTAG_RST	JTAG_RST Disable Control 0b - Functional reset event JTAG_RST triggers a reset sequence. 1b - Functional reset event JTAG_RST generates an interrupt request.

Table continues on the next page...

Table continued from the previous page...

Field	Function
8 D_SWT2_RST	SWT2_RST Disable Control 0b - Functional reset event SWT2_RST triggers a reset sequence. 1b - Functional reset event SWT2_RST generates an interrupt request.
7 D_SWT1_RST	SWT1_RST Disable Control 0b - Functional reset event SWT1_RST triggers a reset sequence. 1b - Functional reset event SWT1_RST generates an interrupt request.
6 D_SWT0_RST	SWT0_RST Disable Control 0b - Functional reset event SWT0_RST triggers a reset sequence. 1b - Functional reset event SWT0_RST generates an interrupt request.
5 —	Reserved
4 —	Reserved
3 D_FCCU_RST	FCCU_RST Disable Control 0b - Functional reset event FCCU_RST triggers a reset sequence. 1b - Functional reset event FCCU_RST generates an interrupt request.
2 —	Reserved
1 —	Reserved
0 —	Reserved

33.8.5 Functional Bidirectional Reset Enable Register (FBRE)

Offset

Register	Offset
FBRE	10h

Function

This register enables the generation of an external reset on 'functional' reset.This register can be accessed as read/write in supervisor mode and read-only in user mode. This register is reset on power-on and any 'destructive' reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	BE_DEBU...	BE_SW_F...	0	0	0	0	0	0	0	0	BE_HSE_...	0	0	0	BE_HSE_...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	BE_PL_...	0	BE_SWT3...	BE_JTAG...	BE_SWT2...	BE_SWT1...	BE_SWT0...	0	BE_ST_D...	BE_FCCU...	0	0	Reserved
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30 BE_DEBUG_FUNC	Bidirectional Reset Enables for 'Functional' Reset DEBUG_FUNC 0b - External reset pin is asserted on a 'Functional' reset DEBUG_FUNC event if the reset is enabled. 1b - External reset pin is not asserted on a 'functional' reset DEBUG_FUNC event.
29 BE_SW_FUNC	Bidirectional Reset Enables for 'Functional' Reset SW_FUNC 0b - External reset pin is asserted on a 'Functional' reset SW_FUNC event if the reset is enabled. 1b - External reset pin is not asserted on a 'functional' reset SW_FUNC event.
28 —	Reserved
27 —	Reserved
26 —	Reserved
25 —	Reserved
24 —	Reserved
23	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
22 —	Reserved
21 —	Reserved
20 BE_HSE_BOOT_RST	Bidirectional Reset Enables for 'Functional' Reset HSE_BOOT_RST 0b - External reset pin is asserted on a 'Functional' reset HSE_BOOT_RST event if the reset is enabled. 1b - External reset pin is not asserted on a 'functional' reset HSE_BOOT_RST event.
19 —	Reserved
18 —	Reserved
17 —	Reserved
16 BE_HSE_SWT_RST	Bidirectional Reset Enables for 'Functional' Reset HSE_SWT_RST 0b - External reset pin is asserted on a 'Functional' reset HSE_SWT_RST event if the reset is enabled. 1b - External reset pin is not asserted on a 'functional' reset HSE_SWT_RST event.
15 —	Reserved
14 —	Reserved
13 —	Reserved
12 BE_PLL_AUX	Bidirectional Reset Enables for 'Functional' Reset PLL_AUX 0b - External reset pin is asserted on a 'Functional' reset PLL_AUX event if the reset is enabled. 1b - External reset pin is not asserted on a 'functional' reset PLL_AUX event.
11 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
10 BE_SWT3_RST	Bidirectional Reset Enables for 'Functional' Reset SWT3_RST 0b - External reset pin is asserted on a 'Functional' reset SWT3_RST event if the reset is enabled. 1b - External reset pin is not asserted on a 'functional' reset SWT3_RST event.
9 BE_JTAG_RST	Bidirectional Reset Enables for 'Functional' Reset JTAG_RST 0b - External reset pin is asserted on a 'Functional' reset JTAG_RST event if the reset is enabled. 1b - External reset pin is not asserted on a 'functional' reset JTAG_RST event.
8 BE_SWT2_RST	Bidirectional Reset Enables for 'Functional' Reset SWT2_RST 0b - External reset pin is asserted on a 'Functional' reset SWT2_RST event if the reset is enabled. 1b - External reset pin is not asserted on a 'functional' reset SWT2_RST event.
7 BE_SWT1_RST	Bidirectional Reset Enables for 'Functional' Reset SWT1_RST 0b - External reset pin is asserted on a 'Functional' reset SWT1_RST event if the reset is enabled. 1b - External reset pin is not asserted on a 'functional' reset SWT1_RST event.
6 BE_SWT0_RST	Bidirectional Reset Enables for 'Functional' Reset SWT0_RST 0b - External reset pin is asserted on a 'Functional' reset SWT0_RST event if the reset is enabled. 1b - External reset pin is not asserted on a 'functional' reset SWT0_RST event.
5 —	Reserved
4 BE_ST_DONE	Bidirectional Reset Enables for 'Functional' Reset ST_DONE 0b - External reset pin is asserted on a 'Functional' reset ST_DONE event if the reset is enabled. 1b - External reset pin is not asserted on a 'functional' reset ST_DONE event.
3 BE_FCCU_RST	Bidirectional Reset Enables for 'Functional' Reset FCCU_RST 0b - External reset pin is asserted on a 'Functional' reset FCCU_RST event if the reset is enabled. 1b - External reset pin is not asserted on a 'functional' reset FCCU_RST event.
2 —	Reserved
1 —	Reserved
0 —	Reserved

33.8.6 Functional Reset Escalation Counter Register (FREC)

Offset

Register	Offset
FREC	14h

Function

This register provides the current value of functional reset escalation counter. It can be accessed in read/write, in supervisor mode. It can be accessed in read in the user mode. This register is reset by power-on reset, destructive reset, when you reconfigure the FREC field to Fh and when you write any value to the Functional Reset Escalation Threshold Register (FRET) register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												FREC			
W													W1C			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-4 —	Reserved
3-0 FREC	Functional' Reset Escalation Counter This field provides the value of functional reset escalation counter.

33.8.7 Functional Reset Escalation Threshold Register (FRET)

Offset

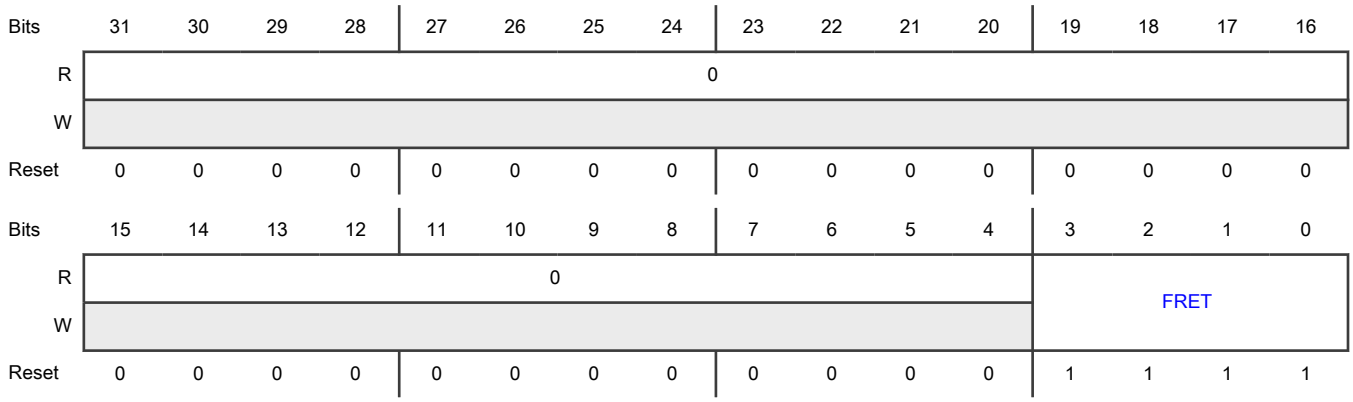
Register	Offset
FRET	18h

Function

This register sets the threshold for 'functional' reset escalation to a 'destructive' reset. It can be accessed in read/write, either in supervisor mode. It can be accessed in read-only in the user mode. Writing a non-zero value to the FRET field enables

the 'functional' reset escalation function. Writing any value to this register resets the 'functional' reset escalation counter. See Functional reset escalation for details on the 'functional' reset escalation function. This register is reset on power-on and any 'destructive' reset.

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 FRET	'Functional' Reset Escalation Threshold If the value of this field is 0, the 'functional' reset escalation function is disabled. Any other value is the number of 'functional' resets that causes a 'destructive' reset.

33.8.8 Destructive Reset Escalation Threshold Register (DRET)

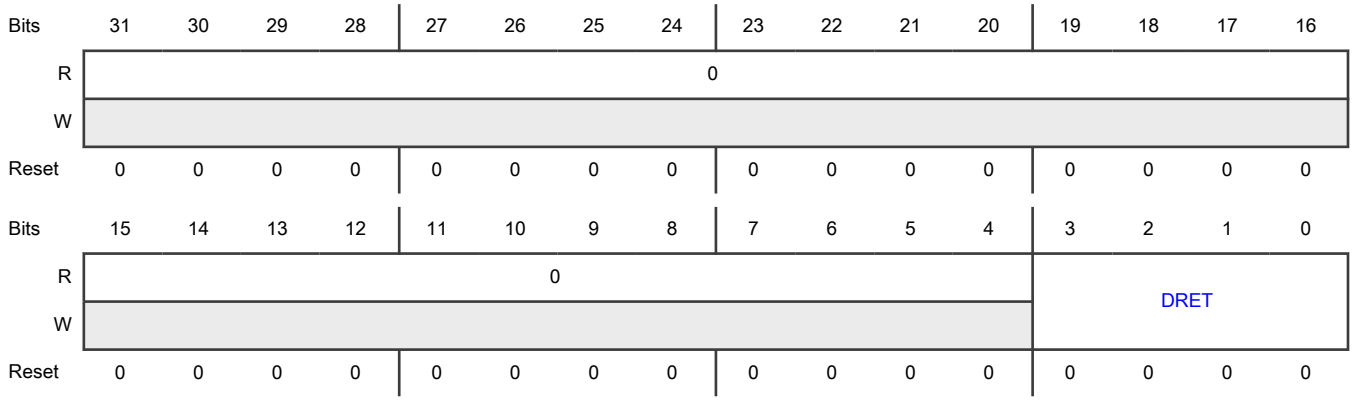
Offset

Register	Offset
DRET	1Ch

Function

This register sets the threshold for 'destructive' reset escalation to keeping the chip in the reset state until the next power-on reset triggers a new reset sequence. It can be accessed in read/write, either in supervisor mode. It can be accessed in read-only in the user mode. Writing a non-zero value to the DRET field enables the 'destructive' reset escalation function. Writing any value to this register resets the 'destructive' reset counter. See Destructive reset escalation for details on the 'destructive' reset escalation function. This register is reset only on power-on reset.

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 DRET	'Destructive' Reset Escalation Threshold If the value of this field is 0, the 'destructive' reset escalation function is disabled. Any other value is the number of 'destructive' resets which keeps the chip in the reset state until the next power-on reset triggers a new reset sequence.

33.8.9 External Reset Control Register (ERCTRL)

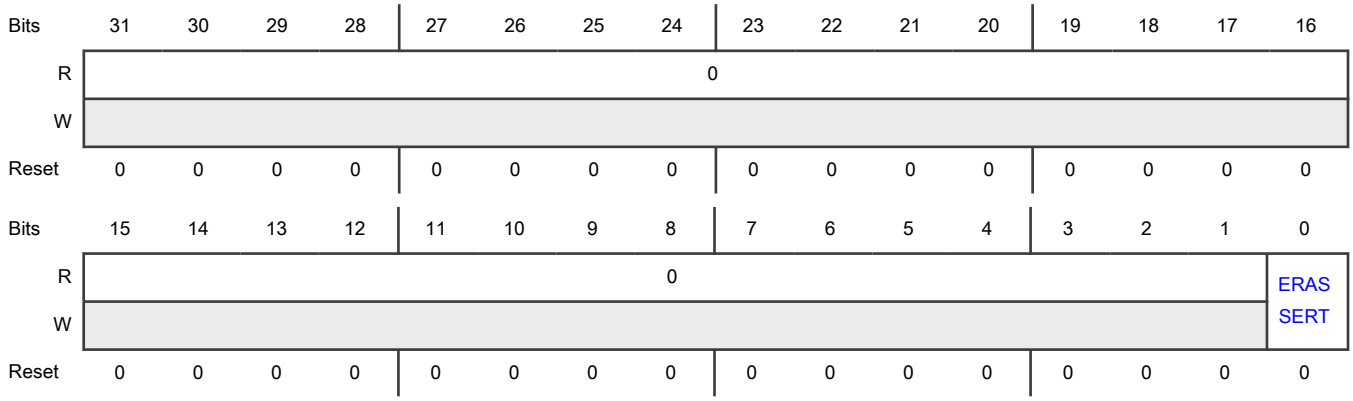
Offset

Register	Offset
ERCTRL	20h

Function

This register allows software to control the assertion of External reset pin. It can be accessed in read/write, in supervisor mode. It can be accessed in read-only in the user mode. This register is reset on all resets.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 ERASSERT	ERASSERT **NOTE** Setting ERASSERT to 1b also safe/pad states most of the chip's pins. See the IOMUX table/spreadsheet for each pin's safe/pad state value. Software must use the ERASSERT field for this purpose only as part of the main reset domain self-test entry procedure. Using it at any other time may result in unpredictable system behavior. 0b - No change 1b - External reset is asserted

33.8.10 Reset During Standby Status Register (RDSS)

Offset

Register	Offset
RDSS	24h

Function

This register provides status of whether a reset event occurred during standby mode. Register bits are cleared on write '1'. This register is reset only on power-on reset.

NOTE

On exiting a reset sequence after standby exit, the software must perform a read operation on MC_ME[PREV_MODE] and RDSS register. If any field of the RDSS register is set, the software must ignore the status reported by MC_ME[PREV_MODE] register otherwise the status of MC_ME[PREV_MODE] register reports the device status.

If MC_ME indicates last mode as RESET, then perform a reset exit in software else a standby exit.

NOTE

Set flags in the RDSS register must be cleared before a new standby entry.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0														FES_RES	DES_RES
W															W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-2 —	Reserved
1 FES_RES	FES_RES 0b - No functional reset event occurred during standby mode. 1b - Functional reset event occurred during standby mode.
0 DES_RES	DES_RES 0b - No destructive reset event occurred during standby mode. 1b - Destructive reset event occurred during standby mode.

Chapter 34
Power-on Reset Watchdog (POR_WDG)

34.1 Introduction

POR_WDG monitors the chip for stuck or hang scenarios when in reset and standby sequences. It generates a chip power-on-reset event to recover the chip in case it remains stuck in reset or standby entry/exit duration for a pre-configured period of time.

POR_WDG consists of a watchdog counter with a configurable threshold. After the threshold is reached POR_WDG generates a chip power-on-reset event.

SIRC clocks POR_WDG.

34.2 Features

- Counter with four possible configurable threshold limits
- Associate register interfaces to capture the status of input signals
- Monitors reset, standby entry, and standby exit

34.3 Configurations

POR_WDG is enabled by default out of reset to actively monitor the chip reset sequence with the default timeout threshold duration. The POR_WDG configuration controls support provisions for enabling or disabling POR_WDG and the counter threshold, as described in the following sections.

34.3.1 Enable/disable configurations

By default POR_WDG is enabled for reset sequence monitoring, as well as standby entry/exit sequence monitoring for stuck scenarios. The following table describes the controls to enable or disable POR_WDG operation.

Table 221. POR_WDG enable/disable control

POR_WDG monitoring	Default configuration	Enable/disable
Reset monitoring	Enabled	dcf_client_utest_misc[16] Enables or disables POR_WDG for reset monitoring. 0: Disable 1: Enable See the DCF clients file attached to this document for details.
Standby entry/exit monitoring	Disabled	DCM's DCMRWP1[8] Enables or disables POR_WDG for standby entry/exit monitoring. 0: Enable 1: Disable See the DCM chapter for details.

34.3.2 Timeout configurations

POR_WDG supports four configurable counter threshold levels for its counter, as listed in the table below. The default POR_WDG timeout is 25 ms.

Table 222. POR_WDG timeout configuration

DCM's DCMRWP1[10]	DCM's DCMRWP1[9]	POR_WDG timeout
0	0	6.25 ms
0	1	12.50 ms
1	0	25.00 ms
1	1	50.00 ms

34.3.3 Event and status registers

POR_WDG provides indication of a POR_WDG event along with the status of the chip at the point when the POR_WDG counter overflows.

POR_WDG event status register: A POR_WDG event is captured in DCM status register DCMROPP4[0].

POR_WDG status registers: The status of the chip at the POR_WDG event is captured in DCM status registers DCMROPP1–DCMROPP3. See the DCM chapter for details.

DCM's DCMROPP*n* are reset on a POR caused by PMC and are unaffected by a POR_WDG reset.

34.4 Reset sequence monitoring

34.4.1 Introduction

At power-up POR_WDG is enabled for reset monitoring (DCF record in UTEST Misc can bypass POR_WDG, as described in Table 221).

On every reset event, POR_WDG starts monitoring the reset sequence (including the functional reset entry sequence, in case of functional reset event).

If the chip does not exit the reset sequence within the POR_WDG timeout threshold, then POR_WDG initiates a POR sequence to recover the chip from a stuck or hung scenario. You can view the POR_WDG status in DCM's DCMROPPn registers to obtain:

- Diagnostic status
- Chip details in the event of a POR_WDG reset

34.4.2 Inactive windows or bypass operation

POR_WDG monitoring of the reset sequence is inactive in the following conditions:

- POR_WDG is disabled for reset monitoring by chip configuration in UTEST via dcf_client_utest_misc[16]. See the DCF clients file attached to this document for details.
- MC_RGM destructive reset escalation (ensures that the chip stays in reset).
- Extend pin reset by keeping it pressed externally.
- During selftest, when MC_RGM.RGM_ERCTRL[ERASSERT] is configured as 1.

34.4.3 Windows of operation

POR_WDG monitors the reset sequence in the following scenarios:

- Reset pin is active (when chip is in reset, including the functional reset entry sequence).
- Standby domain functional reset is asserted (when reset pad is pulled up but functional reset is asserted).
- RUN domain destructive reset is asserted but flash memory is not in Low-Power mode (when RUN domain destructive reset fires unexpectedly in Run mode).

- RUN domain functional reset is asserted but flash memory is not in Low-Power mode (when RUN domain functional reset fires unexpectedly in RUN mode).

34.5 Standby entry/exit sequence monitoring

34.5.1 Introduction

On every low-power entry or exit event, POR_WDG starts monitoring the low-power sequence (standby entry and standby exit sequence).

If the chip does not exit the standby entry or exit sequence within the POR_WDG timeout threshold, POR_WDG initiates a POR sequence to recover the chip from the stuck or hung scenario. You can then check the POR_WDG status registers in DCM to check the chip status when POR_WDG raised the POR to the chip.

34.5.2 Inactive windows/bypass operation

You can configure DCM's DCMRWP1[8] to disable POR_WDG for standby entry and exit sequence monitoring.

34.5.3 Windows of operation

POR_WDG monitors the standby sequence (standby entry and standby exit) in these scenarios:

- Standby entry sequence monitoring: Monitoring from main core WFI execution until PMC acknowledges Low-Power mode entry.
- Standby exit sequence monitoring: Monitoring from wakeup event until the RUN domain reset recovery.

34.6 Glossary

WFI	Wait for Interrupt

Chapter 35
Security Overview

35.1 Introduction

This chip has a comprehensive set of customer-configurable security features designed to protect code and data from unauthorized access. The content of this section is for non-secure operation only. Contact your NXP representative for details if you need secure operation.

35.2 Security features

S32K3xx:

- Exceeds the EVITA- full [7] functionality and offers high performance for edge nodes applications.
- Bases its chip censorship on the life cycle model. Access to chip code and data becomes progressively more restricted as the chip matures through a defined set of life cycle steps.
- Offers these memory security features:
  - NVM censorship support
  - Debug password protection
  - OTP flash memory areas
- Supports a unique ID—The chip has a unique ID stored in an OTP flash memory area. Any core on the chip can read this unique ID.
- Includes HSE_B, which is a dedicated security system providing:
  - A processor core
  - Dedicated SRAM
  - Symmetric Hardware Accelerator
  - Asymmetric Hardware Accelerator
  - True Random Number Generator (TRNG)
  - Pseudo Random Number Generator (PRNG)
  - Exclusive access to secure areas of the chip's flash memory
  - OTA : In Field Secure Code/Data updates

  NOTE

  Refer Security Reference Manual and Firmware Reference Manual for the details on OTA.

  Also, HSE_B runs NXP firmware independently from the main chip processor cores and can implement advanced security and monitoring functions.
- Supports a secure debugger interface.
- Provides boot modes:

[7] EVITA was a European research project existing from 2008 to 2011. There is no standard or a specification to test compliancy. However, HSE_B meets the common expectations often described in the industry as EVITA Full and exceeds them in these cases:
- HSE_B supports up to AES-256 instead of AES-128
- HSE_B supports up to ECC-521 instead of ECC-256
- HSE_B supports SHA-2/Miyaguchi-Preneel instead of Whirlpool

  - Trusted and secure boot support
  - Handshake that supports BAF
- Monitors operating conditions to maximize tampering resistance.
- Includes basic debugger restrictions (on and off via censorship mode).

AES_ACCEL

Includes AES_ACCEL, which provides an independent DMA-controlled crypto accelerator with security features. AES_ACCEL provides timeout counters. The user can transfer keys from HSE_B to the AES_ACCEL keystore, which can hold 80 keys. [8]

NOTE

See AES_ACCEL Subsystem Reference Manual for details.

When HSE_B detects a security failure, it moves the chip to a secure state. The chip secure states comply with the chip safe states. You enter a safe state as reset when there are security errors leading to reset. Moreover, if the chip stays in Run mode and the lifecycle moves to the IN_FIELD phase in reaction to errors, the safety application can continue running without glitches.

35.3 Security information

Chip security feature details are published in the HSE firmware reference manual and HSE service interface manual. Contact your NXP sales representative for more information.

35.4 Glossary

AES	Advanced encryption standard
BAF	Boot assist flash
ECC	Elliptic curve cryptography
EVITA	E-safety vehicle intrusion protected applications
NVM	Nonvolatile memory
OTP	One-time programmable

[8] Applicable for S32K388/S32K389 only.

Chapter 36
Hardware Security Engine (HSE)

36.1 HSE subsystem

HSE is a security subsystem. It runs security functions for applications having stringent confidentiality and authenticity requirements. HSE has the following objectives:

- Isolating security-sensitive information (for example, secret keys) from the application (the host)
- Transferring the cryptographic operations from application cores and processing them
- Accelerating cryptographic operations with dedicated coprocessors
- Enforcing security measures on the application, during runtime and system startup

The HSE subsystem is the only master that is unconditionally released from reset after POR. It then releases the CPU subsystems in the host from reset, with the opportunity to apply certain checks beforehand (secure system startup). Based on certain conditions, HSE can also trigger interrupts and reset signals to the host during runtime.

36.1.1 CPU subsystem

The HSE CPU subsystems process the security functions and control system resources to provide security services to the application domain (the host).

36.1.2 Cryptographic accelerators

The HSE subsystem supports the following cryptographic accelerators:

- An AES engine supporting all standard key sizes (128, 192, 256 bits) and various complex ciphering modes (ECB, CBC, CTR, OFB, CFB, CCM, CMAC)
- A hash engine that supports standard several primitives: MD5, SHA-1, SHA-224, SHA-256
- PKC which accelerates RSA and ECC operations (key generation, signature generation, signature verification, ciphering)
  - RSA (1024, 2048, 3072, 4096-bit)
  - ECC over prime numbers
    - BN P256, P638
    - ANSI X9P192 to X9P512
    - Brainpool P160 to P512
    - Sec P128 to P521
    - TU Darmstadt prime Curve 1 to 35
    - Ed25519

The HSE subsystem supports several other cryptographic primitives through the software. See HSE firmware for more information.

36.1.3 Random number generator (RNG)

The RNG in this chip consists of a TRNG and a DRBG. Both are designed to be compliant to the highest strength in security as specified in

- BSI AIS20/31
- NIST SP800-90a,b,c

The TRNG function provides a seed for the DRBG, while the DRBG is available to the host via dedicated random number generation services.

36.1.4 Timers

The HSE subsystem features:

- An independent dedicated system timer
- HSE_STM (apart from chip timer resources), that allows recurring autonomous functions such as runtime memory verification checks
- A watchdog timer to reset the HSE subsystem in case of unexpected runtime failure

36.1.5 Memory resources

See Configuration_GPR memory map for more information on configuration controls related to HSE memory resources.

36.1.5.1 Secure RAM

The Secure area sizes are enforced by the Memory controllers and updated to appropriate value by HSE before any Host core release. This ensures that these secure areas are never exposed to any other core than HSE.

Secure sizes for each K3 derivative are described in the HSE Firmware Refence Manual.

Secure RAM refers to RAM area that the HSE subsystem access exclusively.

36.1.5.2 Secure flash

Secure flash refers to nonvolatile memory that the HSE subsystem accesses exclusively.

36.2 HSE interface

36.2.1 Messaging unit (MU)

The HSE subsystem has two messaging units:

- HSE_MU0
- HSE_MU1

See the "Messaging Unit (MU)" chapter for more information.

36.2.1.1 Overview

MU is the communication interface between the host and the HSE subsystem. The host uses MU to trigger service requests and to receive service responses. The HSE firmware uses MU to receive service requests, return service responses, and provide a general status of the HSE subsystem.

Each of the two MU instances available in the HSE subsystem has:

- Two sides:
  - MUA: Only the HSE subsystem accesses it.
  - MUB: The host accesses it.

    The registers on one side have corresponding registers on the other side.
- A set of 32-bit readable and writable transmit registers (MUB_TRn), which the host uses to transfer the address of the service descriptor to the HSE firmware. The HSE firmware retrieves the address of the service descriptor in the corresponding registers MUA_RRn.
- A set of 32-bit read-only receive registers (MUB_RRn), which the host uses to retrieve the response to the service requests. The HSE firmware provides the response to service requests in the corresponding registers MUA_TRn.

- Control and status registers to manage MU and the access to transmit and receive registers.

The advantages of using the MU to manage the HSE service requests and responses are:

- Hardware mechanisms are in place on the transmit registers to avoid service request overrun.
- Interrupt signals are available to allow asynchronous management of the requests (avoiding active waiting loops).
- Freedom from interference between different application cores. You can configure each MU instance with specific access restrictions that can be used, for example, to isolate the requests that different masters make (in different MU instances). You configure such access controls using XRDC.

36.3 Debug

36.3.1 HSE subsystem debug

The debugging of the HSE subsystem and associated firmware is restricted to NXP engineering teams.

36.3.2 Host debug

The host debug is either open or protected, depending on the device Lifecycle state. See the ‘Life cycle’ section in ‘Device Configuration Module’ chapter for details on device lifecycle advancement and decoding. See the Debug chapter for details on host/application core debug.

The debug protection consists of locking the debugger access through the JTAG interface until the HSE firmware authenticates the debugger. This authentication is based on ADK/P, a 16-byte region within UTEST used for application core debug. This location will be used by SBAF to run the debug authorization feature. SBAF will use this value to derive the application expected response register and HSE expected response register. See UTEST memory map in the DCF clients file attached to this document.

The authentication method can be:

- Static: In this case, ADK/P is a password which the debugger provides in plain form.
- Dynamic: In this case, ADK/P is a cryptographic key which the debugger uses to calculate a cryptographic response to a random challenge.

36.4 HSE firmware

Factory supplied firmware that runs in the HSE subsystem controls HSE functionality. It essentially serves the host with a set of security services as described in Table 223.

Table 223. Summary of firmware security services

Service	Summary
Administration	Install, configure, update, and test the HSE firmware
Key management	Available for the application to manage different sets of keys that the HSE firmware manages, for example, cryptographic services
Cryptographic	Provide the application with cryptographic primitives that high level security tasks use in the application
Random number	Generate random streams that can be used in various security protocols
Memory verification	Allow the application to verify different memory areas at startup (after reset) and during runtime
Monotonic counter	Provide the application with a set of monotonic counters that can be read and only incremented
Secure time	Allow the configuration of a secure tick to be signaled to the application

Table 224 provides an overview of services and features that the HSE firmware supports.

Table 224. HSE firmware features

Service	Category	Feature
Cryptography	Ciphers	AES: ECB, CBC, CFB, OFB, CTR, XTS[1] RSAES: PKCS1-v1_5, OAEP
	Message Authentication Code (MAC)	AES: CMAC, XCBC-MAC[1], HMAC[1], and GMAC[1]
	Hashing	SHA1 SHA224, SHA256, SHA384, SHA512 SHA3_224[1], SHA3_256[1], SHA3_384[1], SHA3_512[1] MD5 Miyaguchi-Preneel Compression
	Authenticated ciphers	AES: CCM, GCM[1]
	Digital signature generation and verification	RSASSA_PSS RSASSA_PKCS1-v1_5 ECDSA – ECC over GF(P) with all prime standard curve supported EdDSA - Ed25519 pre-hashed curve
Key management	Max key sizes	AES: 256 bits RSA: 4096 bits ECC: 521 bits DH: 4096 bits
	Key generation	Permanent and ephemeral RSA and ECC key pair generation
	Key import or export	Plain or encrypted form, with optional authentication tag. SHE key update protocol
	Key derivation	NIST 800-108, PBKDF2, and so on
	Key exchange	ECDH and Classic DH
	Certificate handling	Key Installation from X.509 and CVC certificates Certificate installation for Root of Trust establishment.
Boot and memory verification	Authentication schema	AES CMAC, XCBC-MAC RSA & ECC signatures
	Verification flow	Before application startup (strict secure boot) In parallel to the application startup On the application demand
	Sanctions	No startup (strict secure boot)

Table continues on the next page...

Table 224. HSE firmware features (continued)

Service	Category	Feature
		Chip reset Key usage restrictions
Monotonic counter	Counter management	Incrementing and reading volatile and non-volatile counters
Random number	Deterministic random bit generation	Based on a True Random Number AIS31 Class P2 high and FIPS 140-2 compliant
Secure time	Secure tick	Application interrupts at configurable frequency
Administration services	HSE administration	Firmware installation and update Subsystem configuration and testing

1. Software implementation of cryptographic primitives.

See documents in Table 225 for more information about how to install, configure, and use the HSE firmware that NXP provides.

Table 225. References

Document number	Document title	Description
HSEFWRM	HSE Firmware Reference Manual	Contains details about how to install, configure, and use the HSE firmware.
HSESIRM	HSE Service Interface Reference Manual	Security firmware API reference for non-AUTOSAR users.

36.5 Configuration_GPR register descriptions

36.5.1 Configuration_GPR memory map

This section describes the chip configurations that only the HSE core manages. These constitute control of peripherals, secure size configurations for SRAM and flash memory, flash memory program or erase control, and so on.

NOTE

Write accesses to configuration registers apart from 32-bit accesses might result in unpredictable chip behavior, therefore must not be done.

Configuration_GPR base address: 4039_C000h

Offset	Register	Width (In bits)	Access	Reset value
1Ch	General Purpose Configuration 0 (CONFIG_REG0)	32	R	0000_0000h
34h	General Purpose Configuration 6 (CONFIG_REG6)	32	R	0000_0035h
38h	Configuration RAM Protected Region (CONFIG_RAMPR)	32	R	See section
3Ch	Configuration Code Flash Memory Active Block (CONFIG_CFPRL)	32	R	See section
40h	Configuration Code Flash Memory Passive Block (CONFIG_CFPRH)	32	R	See section

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
44h	Configuration Data Flash Memory Protected Region (CONFIG_DFPR)	32	R	See section
50h	Configuration Program and Erase Lock (CONFIG_PE_LOCK)	32	R	0000_0000h
54h	Configuration RAM Protected Region Alternate (CONFIG_RAMPR_ALT)	32	R	See section
58h	Configuration Code Flash Memory Active Block Alternate (CONFIG_CFPRL_ALT)	32	R	See section
5Ch	Configuration Code Flash Memory Passive Block Alternate (CONFIG_CFPRH_ALT)	32	R	See section
60h	Configuration Data Flash Memory Protected Region Alternate (CONFIG_DFPR_ALT)	32	R	See section
64h	Configuration REG_GPR (CONFIG_REG_GPR)	32	RW	A000_0003h

36.5.2 General Purpose Configuration 0 (CONFIG_REG0)

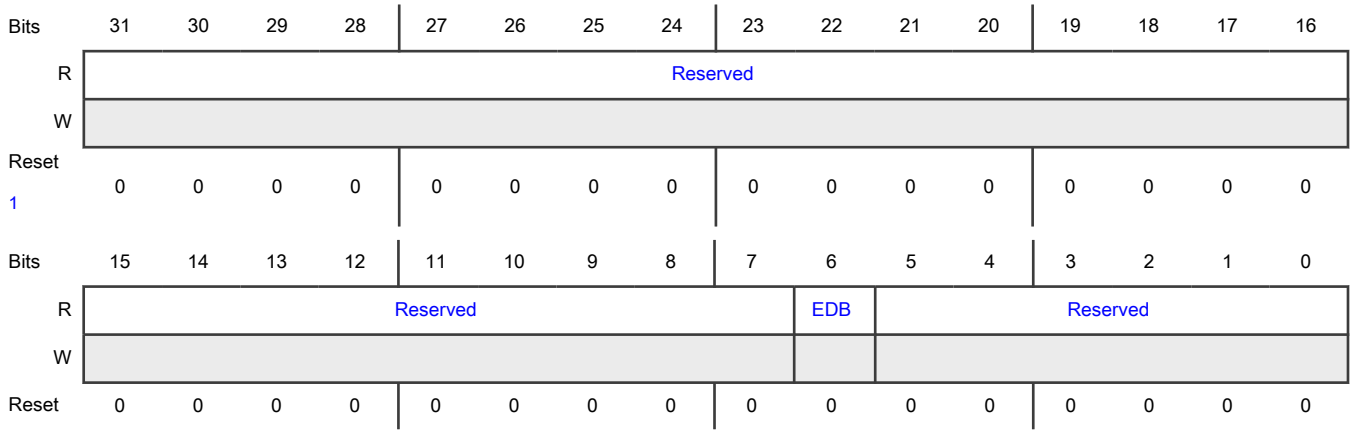
Offset

Register	Offset
CONFIG_REG0	1Ch

Function

The EDB field resets on destructive or POR reset events and functional reset has no impact on it.

Diagram



1. If export_control=1, the reset value of EDB is 1. If export_control=0, the reset value is 0.

Fields

Field	Function
31-7 —	Reserved
6 EDB	Hardware Debugger Attached This is a sticky field that clears on destructive reset or POR. 0b - Debugger not connected 1b - Debugger connected
5-0 —	Reserved

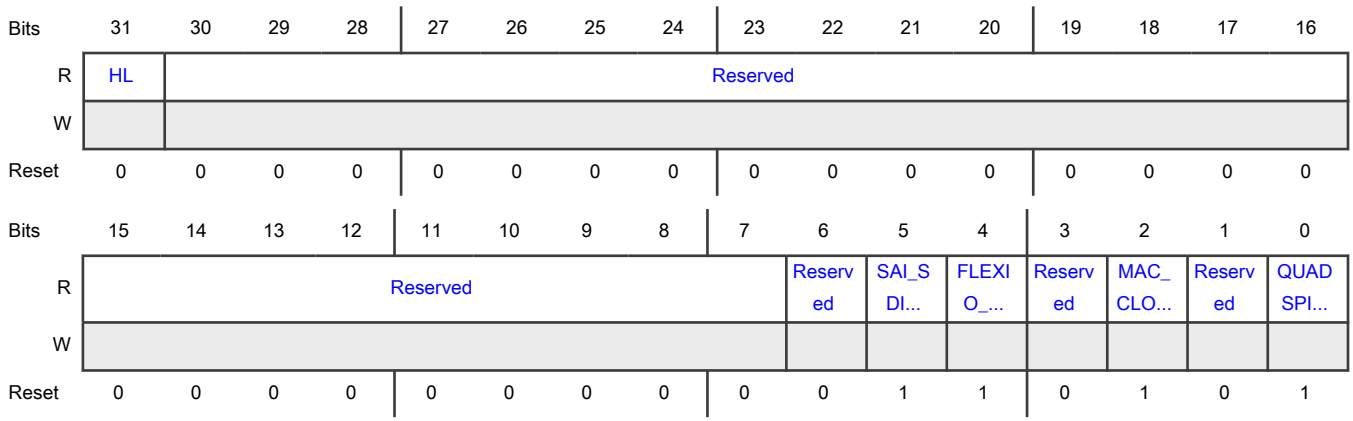
36.5.3 General Purpose Configuration 6 (CONFIG_REG6)

Offset

Register	Offset
CONFIG_REG6	34h

Function
Resets on functional reset.

Diagram



Fields

Field	Function
31 HL	Hard Lock This is a sticky field. If you write 1 to it, it remains 1 until next reset or POR.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - You can write to this register 1b - Register is locked for any write
30-7 —	Reserved
6 —	Reserved
5 SAI_SDID_PCTL	SAI0 and SAI1 clock gating Clock to SAI peripheral is on or off. 0b - Clock is off (gated) 1b - Clock is on
4 FLEXIO_CLOCK_GATE	FlexIO Clock Gating Clock to FlexIO peripheral is on or off. 0b - Clock is off (gated) 1b - Clock is on
3 —	Reserved
2 MAC_CLOCK_GATE	Ethernet Clock Gating Clock to Ethernet peripheral is on or off. 0b - Clock is off (gated) 1b - Clock is on
1 —	Reserved
0 QUADSPI_SDID_PCTL	QuadSPI Clock Gating Clock to QuadSPI peripheral is on or off. 0b - Clock is off (gated) 1b - Clock is on

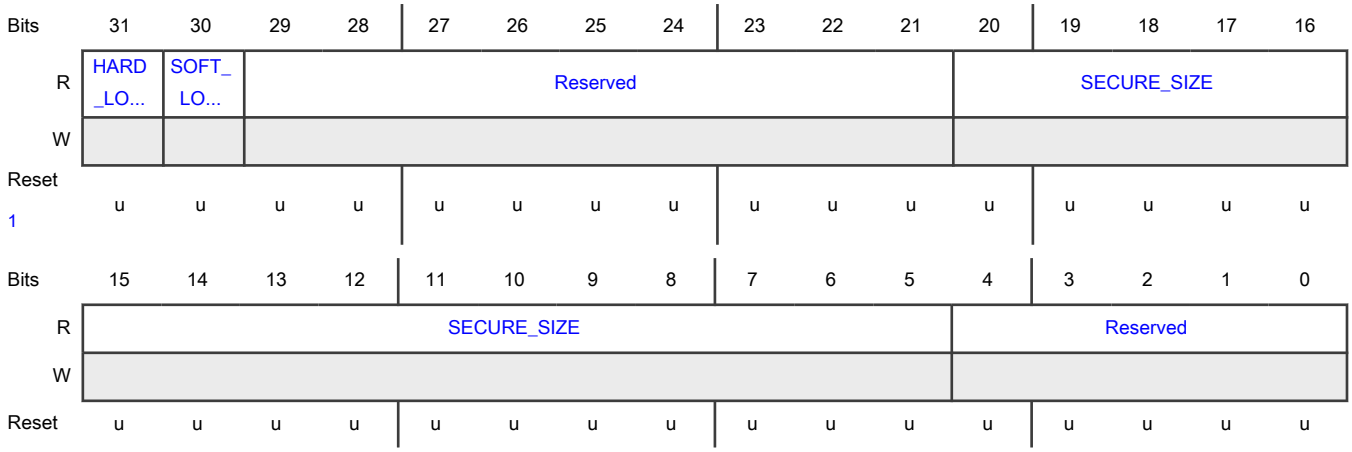
36.5.4 Configuration RAM Protected Region (CONFIG_RAMPR)

Offset

Register	Offset
CONFIG_RAMPR	38h

Function
Resets on functional reset.

Diagram



1. The default reset value of this register varies on the part basis, depending on NXP factory configurations.

Fields

Field	Function
31 HARD_LOCK	Hard Lock This is a sticky field. If you write 1 to it, it remains 1 until next reset or POR. 0b - Write access to this register is allowed 1b - Write access to this register is not allowed until next functional reset
30 SOFT_LOCK	Soft Lock 0b - Write access to this register is allowed 1b - Write access to this register is not allowed
29-21 —	Reserved
20-5 SECURE_SIZE	Secure Size Secure size region (in bytes) for PRAM1. This is 32-byte-aligned to ensure alignment with cache lines.
4-0	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	

36.5.5 Configuration Code Flash Memory Active Block (CONFIG_CFPRL)

Offset

Register	Offset
CONFIG_CFPRL	3Ch

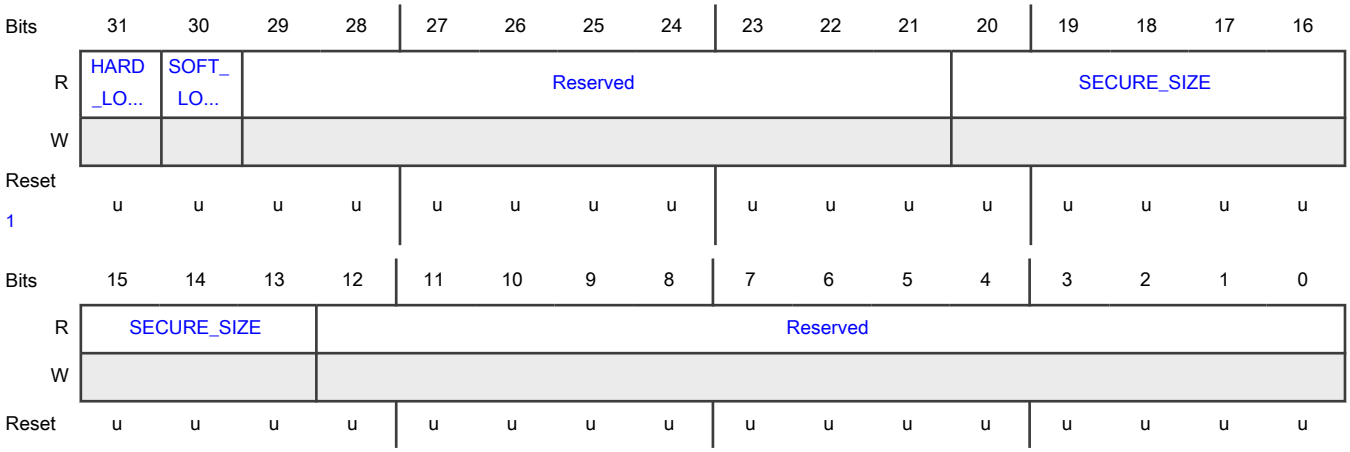
Function

Resets on functional reset.

NOTE

The secure size will go to default or reset value on assertion of any DCF violation from HSE.

Diagram



1. The default reset value of this register varies on the part basis, depending on NXP factory configurations.

Fields

Field	Function
31 HARD_LOCK	Hard Lock This is a sticky field. If you write 1 to it, it remains 1 until next reset or POR. 0b - Write access to this register is allowed 1b - Write access to this register is not allowed until next functional reset
30	Soft Lock

Table continues on the next page...

Table continued from the previous page...

Field	Function
SOFT_LOCK	0b - Write access to this register is allowed 1b - Write access to this register is not allowed
29-21 —	Reserved
20-13 SECURE_SIZE	Secure Size Flash memory active block secure size in bytes to align to 8 KB (sector) aligned write.
12-0 —	Reserved

36.5.6 Configuration Code Flash Memory Passive Block (CONFIG_CFPRH)

Offset

Register	Offset
CONFIG_CFPRH	40h

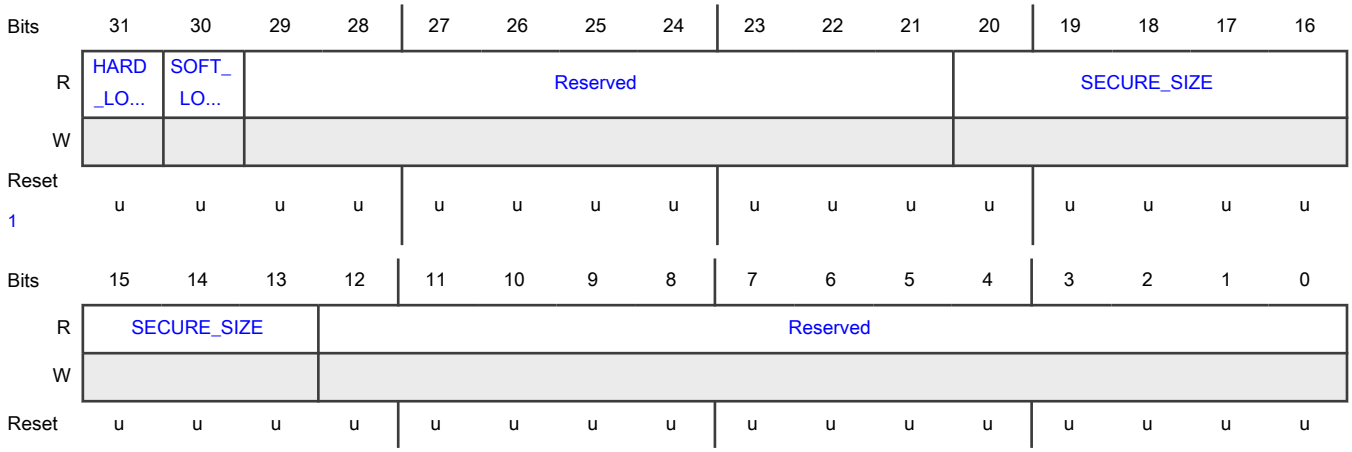
Function

Resets on functional reset.

NOTE

The secure size will go to default or reset value on assertion of any DCF violation from HSE.

Diagram



1. The default reset value of this register varies on the part basis, depending on NXP factory configurations.

Fields

Field	Function
31 HARD_LOCK	Hard Lock This is a sticky field. If you write 1 to it, it remains 1 until next reset or POR. 0b - Write access to this register is allowed 1b - Write access to this register is not allowed until next functional reset
30 SOFT_LOCK	Soft Lock 0b - Write access to this register is allowed 1b - Write access to this register is not allowed
29-21 —	Reserved
20-13 SECURE_SIZE	Secure Size Secure size region (in bytes) for flash memory passive block for alignment with 8 KB (sector) aligned writes.
12-0 —	Reserved

36.5.7 Configuration Data Flash Memory Protected Region (CONFIG_DFPR)

Offset

Register	Offset
CONFIG_DFPR	44h

Function

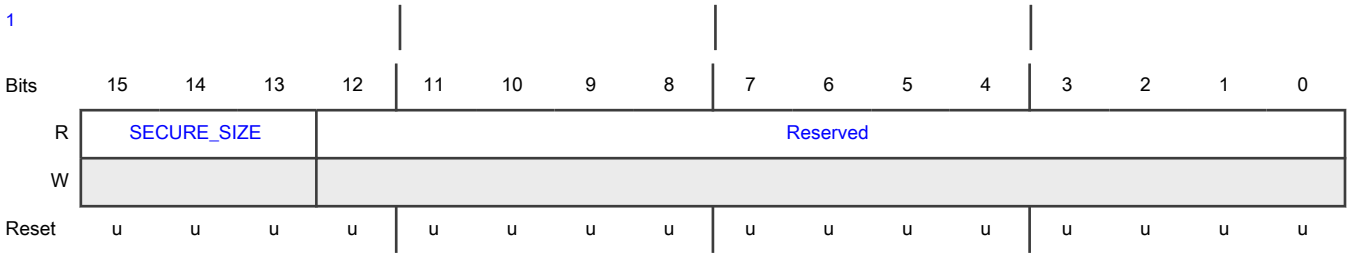
Resets on functional reset.

NOTE

The secure size will go to default or reset value on assertion of any DCF violation from HSE.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	HARD_LO...	SOFT_LO...	Reserved										SECURE_SIZE			
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u



1. The default reset value of this register varies on the part basis, depending on NXP factory configurations.

Fields

Field	Function
31 HARD_LOCK	Hard Lock This is a sticky field. If you write 1 to it, it remains 1 until next reset or POR. 0b - Write access to this register is allowed 1b - Write access to this register is not allowed until next functional reset
30 SOFT_LOCK	Soft Lock 0b - Write access to this register is allowed 1b - Write access to this register is not allowed
29-21 —	Reserved
20-13 SECURE_SIZE	Secure Size Secure size region (in bytes) for data flash memory aligned to 8KB (sector) aligned writes.
12-0 —	Reserved

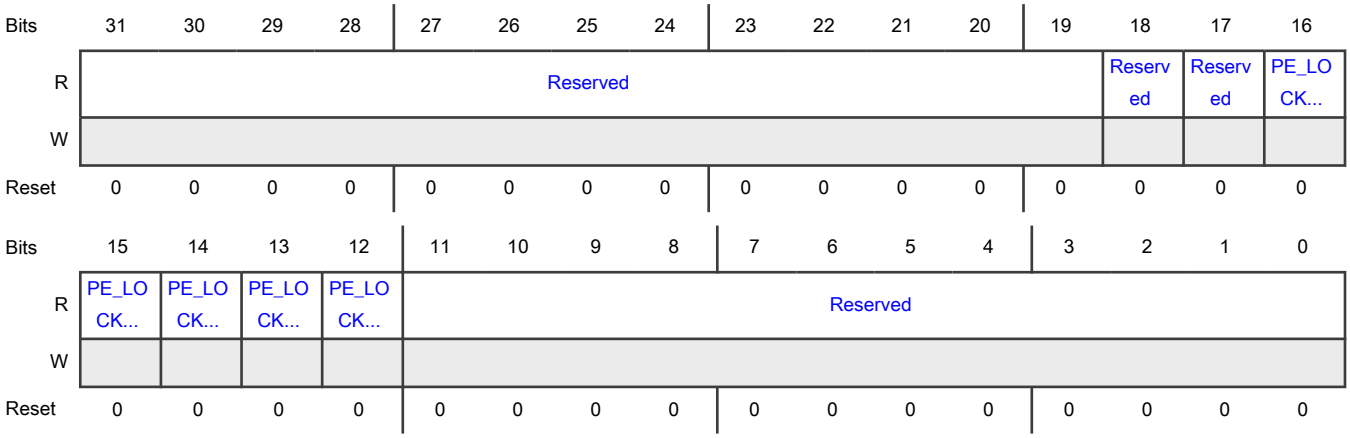
36.5.8 Configuration Program and Erase Lock (CONFIG_PE_LOCK)

Offset

Register	Offset
CONFIG_PE_LOCK	50h

Function
Resets on functional reset.

Diagram



Fields

Field	Function
31-19 —	Reserved
18 —	Reserved
17 —	Reserved
16 PE_LOCK_BLOCK_4	Program/Erase Lock for Block 4 0b - Block 4 is available for program and erase operations 1b - Block 4 is locked and unavailable for program and erase operations
15 PE_LOCK_BLOCK_3	Program/Erase Lock for Block 3 0b - Block 3 is available for program and erase operations 1b - Block 3 is locked and unavailable for program and erase operations
14 PE_LOCK_BLOCK_2	Program/Erase Lock for Block 2 0b - Block 2 is available for program and erase operations 1b - Block 2 is locked and unavailable for program and erase operations
13 PE_LOCK_BLOCK_1	Program/Erase Lock for Block 1 0b - Block 1 is available for program and erase operations 1b - Block 1 is locked and unavailable for program and erase operations
12 PE_LOCK_BLOCK_0	Program/Erase Lock for Block 0 0b - Block 0 is available for program and erase operations 1b - Block 0 is locked and unavailable for program and erase operations

Table continues on the next page...

Table continued from the previous page...

Field	Function
11-0 —	Reserved

36.5.9 Configuration RAM Protected Region Alternate (CONFIG_RAMPR_ALT)

Offset

Register	Offset
CONFIG_RAMPR_ALT	54h

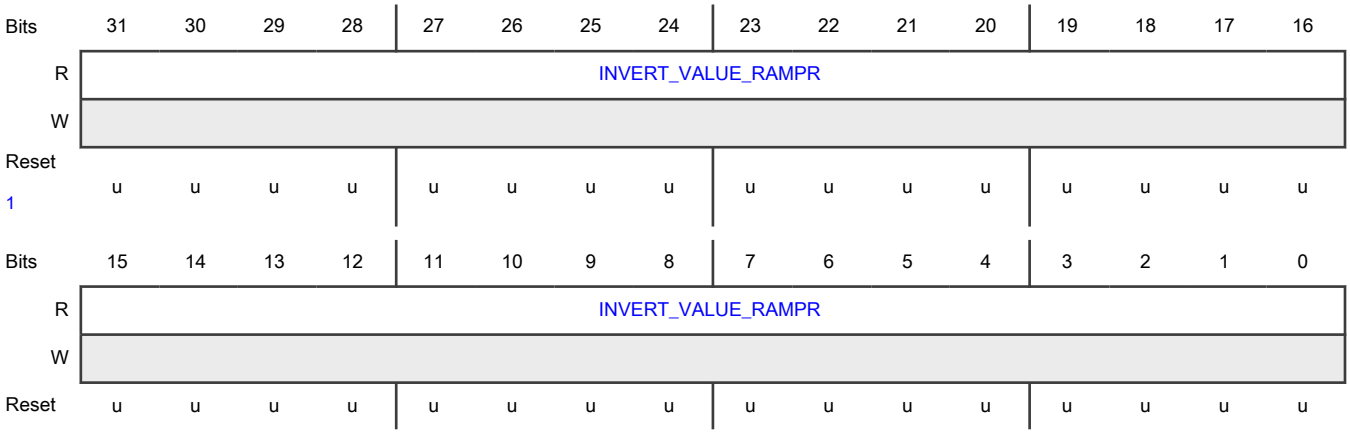
Function

Resets on functional reset.

NOTE

The secure size will go to default or reset value on assertion of any DCF violation from HSE.

Diagram



1. The default reset value of this register varies on the part basis, depending on NXP factory configurations.

Fields

Field	Function
31-0 INVERT_VALUE_RAMPR	Invert Value DFPR Write inverted value of register CONFIG_RAMPR to CONFIG_RAMPR_ALT.

36.5.10 Configuration Code Flash Memory Active Block Alternate (CONFIG_CFPRL_ALT)

Offset

Register	Offset
CONFIG_CFPRL_ALT	58h

Function

Resets on functional reset.

NOTE

The secure size will go to default or reset value on assertion of any DCF violation from HSE.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	INVERT_VALUE_CFPRAB															
W																
Reset [1]	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	INVERT_VALUE_CFPRAB															
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

1. The default reset value of this register varies on the part basis, depending on NXP factory configurations.

Fields

Field	Function
31-0 INVERT_VALUE_CFPRAB	Invert Value CFPRAB Write inverted value of register CONFIG_CFPRL to CONFIG_CFPRL_ALT.

36.5.11 Configuration Code Flash Memory Passive Block Alternate (CONFIG_CFPRH_ALT)

Offset

Register	Offset
CONFIG_CFPRH_ALT	5Ch

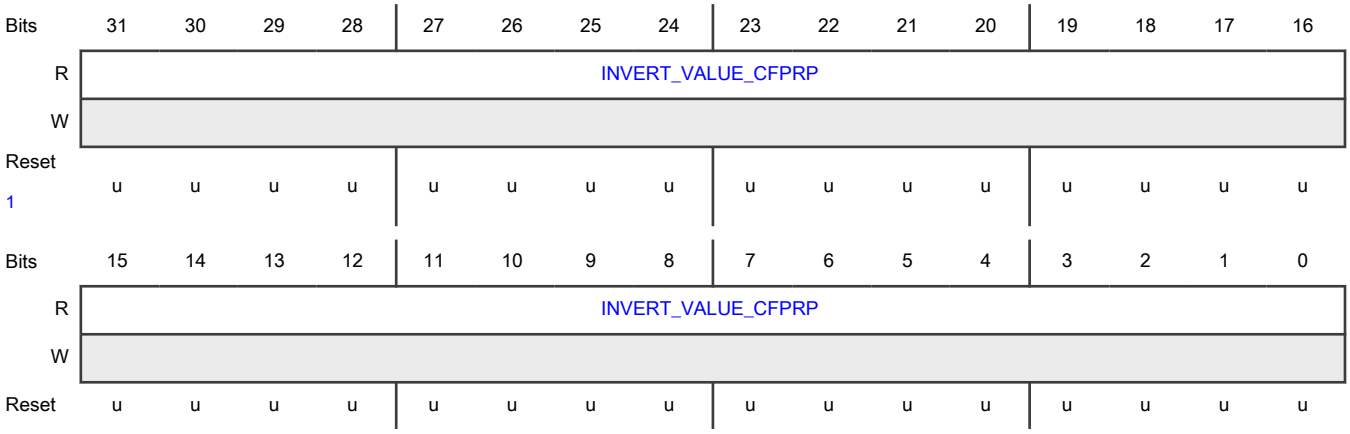
Function

Resets on functional reset.

NOTE

The secure size will go to default or reset value on assertion of any DCF violation from HSE.

Diagram



1. The default reset value of this register varies on the part basis, depending on NXP factory configurations.

Fields

Field	Function
31-0 INVERT_VALUE_CFPRP	Invert Value CFPRP Write inverted value of register CONFIG_CFPRH to CONFIG_CFPRH_ALT.

36.5.12 Configuration Data Flash Memory Protected Region Alternate (CONFIG_DFPR_ALT)

Offset

Register	Offset
CONFIG_DFPR_ALT	60h

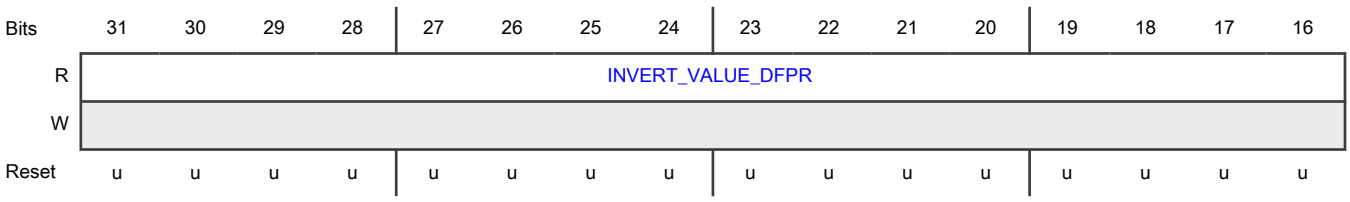
Function

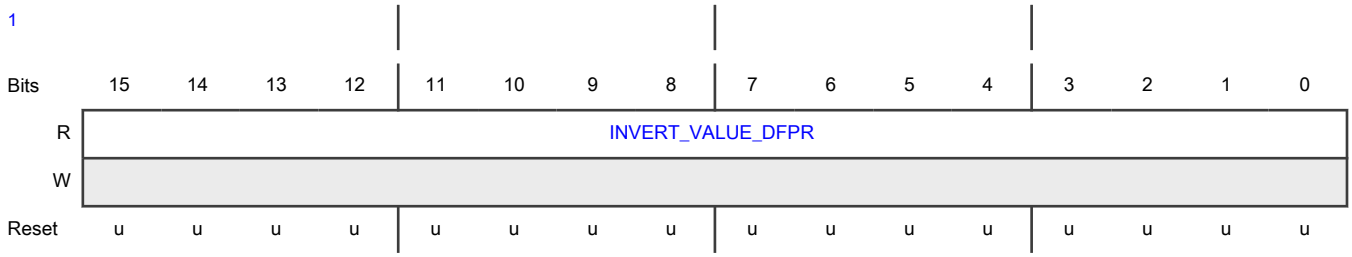
Resets on functional reset.

NOTE

The secure size will go to default or reset value on assertion of any DCF violation from HSE.

Diagram





1. The default reset value of this register varies on the part basis, depending on NXP factory configurations.

Fields

Field	Function
31-0 INVERT_VALUE_DFPR	Invert Value DFPR Write inverted value of register CONFIG_DFPR to CONFIG_DFPR_ALT.

36.5.13 Configuration REG_GPR (CONFIG_REG_GPR)

Offset

Register	Offset
CONFIG_REG_GPR	64h

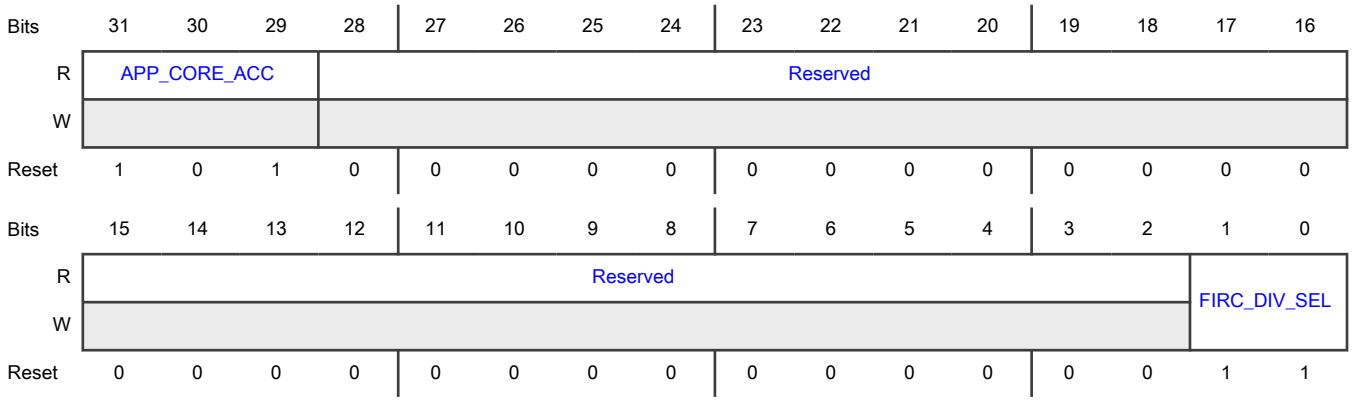
Function

Resets on functional reset.

NOTE

This register can be changed by application core when HSE is not installed.

Diagram



Fields

Field	Function
31-29 APP_CORE_ACC	APP_CORE_ACC **NOTE** While writing to this register, APP_CORE_ACC is RO and should not be changed from 0b101. 101b - Application core can write the [FIRC_DIV_SEL] field. The APP_CORE_ACC must be 101b to have access to FIRC_DIV_SEL field All other values - No access to application core
28-2 —	Reserved
1-0 FIRC_DIV_SEL	FIRC Divider Indicates this chip's FIRC clock division factor. 00b - Divided by 2 01b - Divided by 2 10b - Divided by 16 11b - Undivided

36.6 Glossary

ADK/P	Application debug key/password
AES	Advanced encryption standard
CBC	Cipher block chaining
CCM	Counter with CBC MAC (Cipher block chaining message authentication code)
CFB	Cipher feedback mode
CMAC	Cipher-based message authentication code
CTR	Counter-based block cipher mode
CVC	Card verifiable certificates
Classic DH	Classical Diffie–Hellman, a key exchange method
DRBG	Deterministic random bit generator
DH	Diffie Hellman, a key exchange method
ECB	Electronic code book
ECC	Elliptic curve cryptography
ECDSA	Elliptic curve digital signature algorithm
ECDH	Elliptic-curve Diffie–Hellman, a key exchange method
EdDSA	Edwards-curve digital signature algorithm
FIPS	Federal information processing standards

MUA	Messaging unit A interface
MUB	Messaging unit B interface
MD5	Message-Digest Algorithm 5
NIST	National institute of standards and technology
OAEP	Optimal asymmetric encryption padding
OFB	Output feedback based block cipher mode
GCM	Galois/counter mode
GMAC	Galois message authentication code
HMAC	Keyed-hash message authentication code
PKC	Public key cryptographic engine
PKCS1	Public-key cryptography standards. PKCS provides the basic definitions of, and recommendations for implementing the RSA algorithm.
POR	Power-on reset
RNG	Random number generator
RSA	Rivest–Shamir–Adleman (a public key cryptosystem)
RSASSA_PSS	RSA Signature Scheme with Appendix - Probabilistic Signature Scheme
SHE	Secure hardware extension
SHA	Secure Hash Algorithm
TRNG	True random number generator
XCBC-MAC	Extended ciphertext block chaining MAC
XTS	XEX (XOR encrypt XOR) based tweaked-codebook mode with ciphertext stealing

Chapter 37
Device Configuration Format (DCF) records

37.1 Overview

DCF configures certain registers of this chip during system boot while the reset signal asserts. An individual DCF record points to an internal register in the chip and the data to be written to that register.

UTEST DCF—The UTEST DCF clients are present within the UTEST region of the flash and programmed during production testing. You may write the other records and program them at the same time the application code is programmed in the flash memory. See the DCF client file attached to this document for the description on the UTEST DCF records.

System boot is a complex process that requires you to initialize the chip properly before releasing reset. Before using the chip, the user writes the application specific code into the flash memory. The user can also update the DCF records from their initial settings as per application requirements, in case if needed.

After power is supplied to an appropriately configured chip, PMC controls the chip, and after the system power supplies reach predefined levels, PMC signals MC_RGM to start the boot sequence. During this sequence, MC_RGM enables DCM to read the chip configuration records and then write the configuration information to the specified registers.

37.2 DCF clients

These are 32-bit wide hardware registers inside a module that receive and store data from a DCF record. This stored data is used to initialize registers and configure features.

DCF clients:

- Are assigned a default value before any DCF records are written.
- May have special writing constraints, such as:
  - Write once.
  - Change from 1 to 0 only.
  - Change from 0 to 1 only.
- May not implement all 32 bits.

37.2.1 Safety features of DCF clients

Depending on the DCF client's role in the chip, the client may be equipped with a safety feature or a combination of these features.

37.2.1.1 Parity

If a DCF client implements parity checking, the client receives a parity bit in addition to its data in the DCF record. During chip operation, the client continuously monitors whether the data it stores matches the parity. The parity scheme used is even parity. So, the number of 1s in the WDATA and parity field needs to be even. For example, if the WDATA field has the value 0000_0001h, the value of the parity field needs to be 1 so that the total number of 1s is even. It also reports errors to DCM in case of discrepancies.

37.2.1.2 Triple voting

DCF clients that use triple voting have three copies of the register. DCM writes to all the three registers in a single write cycle. During chip operation, the DCF client continuously monitors whether all these three copies match. In case of discrepancies, the client reports errors to DCM. The chip uses the majority result, so single errors do not affect the chip's operation. If case of single error, since the chip uses the majority result, there will be no impact to the chip's behavior.

37.2.2 DCF client modification rules

Depending on its role in the chip, a DCF client may implement one or a combination of modification rules. If a modification rule is in effect, the order in which DCF records are placed in the record list may be important.

37.2.2.1 Write once

A DCF client using the write once rule can only be written with a single DCF record. The records that are appended later in the list are ignored and do not change the value of the client.

37.2.2.2 Write 0 only

A field in a DCF client can only be changed from 1 to 0. Therefore, if the value of a field in the previous DCF record is 0, an attempt by a later record to write 1 to it is ignored.

37.2.2.3 Write 1 only

A field in a DCF client can only be changed from 0 to 1. Therefore, if the value of a field in the previous DCF record is 1, an attempt by a later record to write 0 to the field is ignored.

37.3 DCF record structure

A DCF record is a double-word (64-bit) entry that consists of the following:

- Control word—This provides information to locate the corresponding DCF client internal to the chip (pointer to the location of a register internal to the chip).
- Data word—This contains the data to be written to the DCF client.

DCF records select the target DCF client using a 30-bit field in the DCF record that consists of a 15-bit chip select field and a 15-bit address field. All modules that include DCF clients are assigned a chip select during chip definition. The address field is only relevant for address decoding within that module and may not necessarily relate to the address of a register that is visible to you.

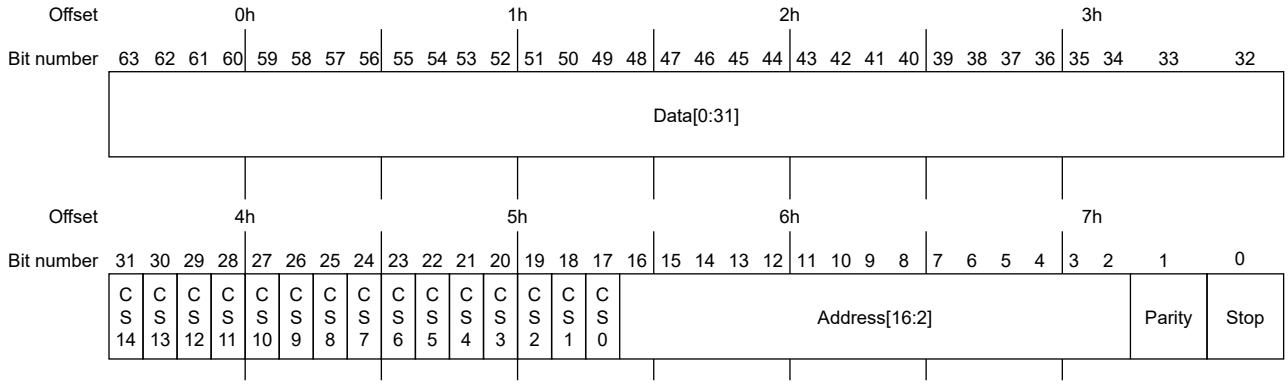


Figure 159. DCF record structure

Table 226. DCF record field descriptions

Field	Name	Description
0-31	Data[0:31]	Provides the data that is to be written to the DCF client.
32-46	CS*n*	Indicates chip select *n*. The value 1 is written to a chip select per DCF record to select the target module for the DCF client. The value 0 must be written to all other chip selects.

Table continues on the next page...

Table 226. DCF record field descriptions (continued)

Field	Name	Description
47-61	Address[16:2]	Contains the address of the DCF client within the selected module. Address decoding for DCF clients may not match the standard software address map decoding. For details, see the DCF client addresses provided with each module.
62	Parity	Indicates parity for the DCF record.
63	Stop	Indicates the end of the list for DCF records. 0 – Not the end 1 – End The erased state of flash memory is FFFF_FFFF_FFFF_FFFFh. Therefore, the list ends with the first unprogrammed double word. This location can be programmed with a new record to extend the list.

37.4 DCF records sequence

An individual DCF record contains information to locate the corresponding DCF client internal to the chip (control word) and the data to be written to that client (data word).

DCF records appear as contiguous series of entries programmed at a specific address within UTEST flash memory, and must present the following pattern:

- The first DCF record must be a start record. This record must be placed at the beginning of a DCF area in UTEST flash memory to indicate to the chip that the specified records must be processed.
- DCF records containing configuration data must immediately follow the start record with no blank records in between. An unprogrammed record is interpreted as a stop record and no DCF records following that are processed. This allows you to program the records in several sessions, appending new records at the end of the list each time.
- DCF stop record with bit set indicates the end of configuration records. It is not recommended to set STOP bit in last DCF record programmed during production because that prevents appending additional DCFs records. The UTEST flash memory location following the last DCF record programmed at the factory is an unprogrammed location, which has FFFF_FFFFh as its content. Thus, the stop bit location in this unprogrammed flash memory location is logic 1, signifying that this is the last DCF record and it is not to be acted upon.

Table 227 shows the record that DCM recognizes as a start record.

Table 227. DCF start record

0h (0:31)	4h (32:63)
05AA_55AFh	0000_0000h

The factory sets the DCF start record at the beginning of the UTEST flash memory area.

Table 228 shows the record that DCM recognizes as a stop record.

Table 228. DCF stop record

0:31	32:62	63
Ignored	Ignored	1

The DCF records that you supply must be added in a contiguous manner immediately following the factory-written DCF records. You must never have an unprogrammed record in the series of DCF records because that is interpreted as a stop record.

Table 229 shows the series of DCF records when *n* data records are stored in the UTEST flash memory.

Table 229. Series of DCF records in UTEST flash memory

Record type	Address offset	Data			
Start	0h	05AA_55AFh			
	4h	0000_0000h			STOP = 0
Data	8h	WDATA[31:0]			
	Ch	CS[14:0]	ADDR[16:2]	Parity	STOP = 0
	10h	WDATA[31:0]			
	14h	CS[14:0]	ADDR[16:2]	Parity	STOP = 0
	...	...			
Stop	8(*n*-1) + 0h	Reserved			
	8(*n*-1) + 4h	Reserved			1
Ignored	8*n* + 0h				
	8*n* + 4h				

More than one DCF records can write to the same DCF client. In this case, the later record usually overrides a DCF client value defined by a previous record. However, not all DCF clients allow overwrites; this depends on individual implementation of DCF clients.

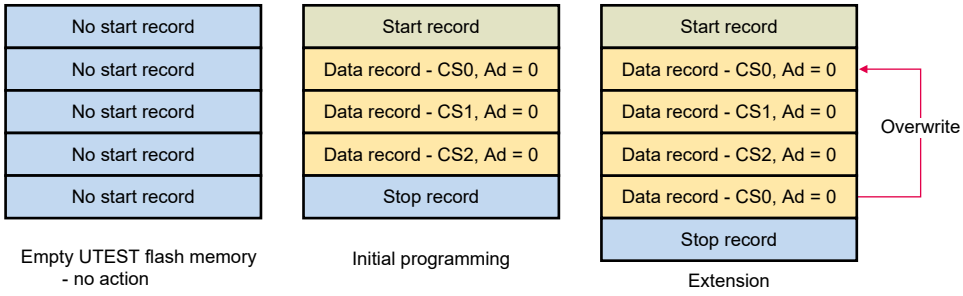


Figure 160. Appending DCF records

37.5 Chip configuration records

The DCF clients table contains information on DCF clients available in the chip. See the DCF clients file attached to this document.

The next table shows an example of how the information in this chapter is integrated with the attached DCF clients file.

Table 230. Integration of DCF information

Type	Data(n) assuming Quad page program	Data	Comment
Reset pad dedicated control DCF client (dcf_client_reset_pad_dedicated - column D in the "Utest DCF Clients" sheet)	Data word (determined base on the DCF record description in the "Utest DCF Client Register Bits" sheet)	0000_0001h	Data to enable pad as dedicated reset pad
	Control word (without parity) (selected from column C in the "Utest DCF Clients" sheet)	0010_0008h	Chip Select is 3 and address is 8, 0010_0000h+ 8h

37.6 Glossary

UTEST User test. Refers to UTEST region of the flash.

Chapter 38
Device Configuration Module General-Purpose Registers (DCM_GPR)

38.1 DCM controlled features and availability in product family

Based on the chip features described in ‘Feature comparison’ section in ‘Introduction’ chapter, there are some features which are present only in specific parts in the S32K3xx product family. The following table summarizes the corresponding DCM register fields along with the parts wherein the corresponding register fields are available. In rest of the parts within the product family, the corresponding fields are reserved.

Table 231. DCM controlled features and availability in product family

Register field	Register field abbreviation	Register field description	Parts wherein this field is available
DCMROD3[1]	CM7_1_LOCKUP	See Register section for details	S32K389, S32K388, S32K338, S32K328, S32K324, S32K322
DCMROD3[3]	CM7_RCCU1_ALARM	See Register section for details	S32K389, S32K388, S32K358, S32K348, S32K344, S32K342, S32K341,
DCMROD3[4]	CM7_RCCU2_ALARM	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K344, S32K342, S32K341,
DCMROD3[5]	TCM_GSKT_ALARM	See Register section for details	S32K389, S32K388, S32K344, S32K324, S32K314, S32K342, S32K341, S32K322, S32K312, S32K311
DCMROD3[6]	DMA_SYS_GSKT_ALARM	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328, S32K344, S32K324, S32K314, S32K342, S32K341, S32K322
DCMROD3[7]	DMA_PERIPH_GSKT_ALARM	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328, S32K344, S32K324, S32K314, S32K342, S32K341, S32K322
DCMROD3[9]	DMA_AXBS_ALARM	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328, S32K344, S32K324, S32K314, S32K342, S32K341, S32K322
DCMROD3[10]	SDHC_GSKT_ALARM	uSDHC IASHB Gasket Alarm Status. Read this bit to identify the reason of fault in case of FCCU NCF 1.	S32K358, S32K356, S32K348, S32K338, S32K328

Table continues on the next page...

Table 231. DCM controlled features and availability in product family (continued)

Register field	Register field abbreviation	Register field description	Parts wherein this field is available
DCMROD3[12]	QSPI_GSKT_ALARM	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328, S32K344, S32K324, S32K314, S32K342, S32K341, S32K322
DCMROD3[14]	AIPS2_GSKT_ALARM	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328, S32K344, S32K324, S32K314, S32K342, S32K341, S32K322
DCMROD3[17]	TCM_AXBS_ALARM	See Register section for details	S32K389, S32K388, S32K344, S32K324, S32K314, S32K342, S32K341, S32K322
DCMROD3[18]	MAC_GSKT_ALARM	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328, S32K344, S32K324, S32K314, S32K342, S32K341, S32K322
DCMROD3[19]	PERIPH_AXBS_ALARM	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328, S32K344, S32K324, S32K314, S32K342, S32K341, S32K322
DCMROD3[20]	PF3_CODE_ECC_ERR	The errors are reported from the FMU and are connected to FCCU NCFs. These are also connected to ERM. Read this bit to identify the reason of fault in case of FCCU NCF 3.	S32K388, S32K358, S32K356, S32K338
DCMROD3[21]	PF3_DATA_ECC_ERR	The errors are reported from the FMU and are connected to FCCU NCFs. These are also connected to ERM. Read this bit to identify the reason of fault in case of FCCU NCF 3.	S32K388, S32K358, S32K356, S32K338
DCMROD3[23]	PRAM2_ECC_ERR	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328
DCMROD3[24]	PRAM1_ECC_ERR	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328, S32K344, S32K324, S32K314
DCMROD3[27]	CM7_1_DCDATA_ECC_ERR	See Register section for details	S32K389, S32K388, S32K338, S32K328, S32K324, S32K322

Table continues on the next page...

Table 231. DCM controlled features and availability in product family (continued)

Register field	Register field abbreviation	Register field description	Parts wherein this field is available
DCMROD3[29]	CM7_1_DCTAG_ECC_ERR	See Register section for details	S32K389, S32K388, S32K338, S32K328, S32K324, S32K322
DCMROD3[31]	CM7_1_ICDATA_ECC_ERR	See Register section for details	S32K389, S32K388, S32K338, S32K328, S32K324, S32K322
DCMROD4[1]	CM7_1_ICTAG_ECC_ERR	See Register section for details	S32K389, S32K388, S32K338, S32K328, S32K324, S32K322
DCMROD4[5]	CM7_1_ITCM_ECC_ERR	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328, S32K344, S32K324, S32K342, S32K341, S32K322
DCMROD4[6]	CM7_1_DTCM0_ECC_ERR	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328, S32K344, S32K324, S32K342, S32K341, S32K322
DCMROD4[7]	CM7_1_DTCM1_ECC_ERR	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328, S32K344, S32K324, S32K342, S32K341, S32K322
DCMROD4[10]	PRAM1_FCCU_ALARM	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328, S32K344, S32K324, S32K314
DCMROD4[16]	PF2_CODE_ECC_ERR	The errors are reported from the FMU and are connected to FCCU NCFs. These are also connected to ERM. Read this bit to identify the reason of fault in case of FCCU NCF 3.	S32K388, S32K338, S32K328, S32K324, S32K322
DCMROD4[17]	PF2_DATA_ECC_ERR	The errors are reported from the FMU and are connected to FCCU NCFs. These are also connected to ERM. Read this bit to identify the reason of fault in case of FCCU NCF 3.	S32K388, S32K338, S32K328, S32K324, S32K322
DCMROD4[23]	PRAM2_FCCU_ALARM	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328
DCMROD4[28]	SDHC_RDATA_EDC_ERR	Integrity(EDC) error on uSDHC read data for safety. Read this bit to identify the reason of fault in case of FCCU NCF 1.	S32K358, S32K356, S32K348, S32K338, S32K328
DCMROD4[31]	CM7_2_LOCKUP	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K338

Table continues on the next page...

Table 231. DCM controlled features and availability in product family (continued)

Register field	Register field abbreviation	Register field description	Parts wherein this field is available
DCMROD5[14]	TCM_RDATA_EDC_ERR	Specifies whether an integrity error is reported on the TCM read data for safety. Read this field to identify the reason for a fault in case of FCCU NCF 1.	S32K344, S32K324, S32K314, S32K342, S32K341, S32K322
DCMROD5[15]	MAC_RDATA_EDC_ERR	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328, S32K344, S32K324, S32K314, S32K342, S32K341, S32K322
DCMROD5[18]	CM7_1_AHBP_RDATA_EDC_ERR	See Register section for details	S32K389, S32K388, S32K338, S32K328, S32K324, S32K322
DCMROD5[19]	CM7_1_AHBM_RDATA_EDC_ERR	See Register section for details	S32K389, S32K388, S32K338, S32K328, S32K324, S32K322
DCMROD5[23]	CM7_2_AHBP_RDATA_EDC_ERR	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K338
DCMROD5[24]	CM7_2_AHBM_RDATA_EDC_ERR	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K338
DCMROD5[25]	CM7_2_DCDATA_ECC_ERR	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K338
DCMROD5[26]	CM7_2_DCTAG_ECC_ERR	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K338
DCMROD5[27]	CM7_2_ICDATA_ECC_ERR	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K338
DCMROD5[28]	CM7_2_ICTAG_ECC_ERR	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K338
DCMROD5[29]	CM7_2_ITCM_ECC_ERR	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K338
DCMROD5[30]	CM7_2_DTCM0_ECC_ERR	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K338
DCMROD5[31]	CM7_2_DTCM1_ECC_ERR	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K338
DCMROD6[3]	QSPI_FLASHA_ECC_ERR	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328
DCMROD6[26]	AIPS0_GSKT_ALARM	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328

Table continues on the next page...

Table 231. DCM controlled features and availability in product family (continued)

Register field	Register field abbreviation	Register field description	Parts wherein this field is available
DCMROD6[30]	TCM_PRAM_AXBS_ALARM	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328
DCMROD7[0]	CM7_0_AHBM_ALARM	See Register section for details	S32K389, S32K388
DCMROD7[1]	CM7_1_AHBM_ALARM	See Register section for details	S32K389, S32K388
DCMROD7[2]	CM7_2_AHBM_ALARM	See Register section for details	S32K389, S32K388
DCMROD7[3]	CM7_0_AHBP_ALARM	See Register section for details	S32K389, S32K388
DCMROD7[4]	CM7_1_AHBP_ALARM	See Register section for details	S32K389, S32K388
DCMROD7[5]	CM7_2_AHBP_ALARM	See Register section for details	S32K389, S32K388
DCMROD7[11]	VDD1P1_GNG2_ERR	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328
DCMROD7[12]	VDD2P5_GNG2_ERR	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328
DCMROD7[28]	CM7_0_AHBS_ALARM	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328
DCMROD7[29]	CM7_1_AHBS_ALARM	See Register section for details	S32K389, S32K388, S32K338, S32K328
DCMROD7[30]	CM7_2_AHBS_ALARM	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K338
DCMROD8[0]	PRAM0_GSKT_ALARM	PRAM0 IAHB Gasket monitor alarm status. Read this bit to identify the reason of fault in case of FCCU NCF 1.	S32K358, S32K356, S32K348, S32K338, S32K328
DCMROD8[1]	PRAM1_GSKT_ALARM	PRAM1 IAHB Gasket monitor alarm status. Read this bit to identify the reason of fault in case of FCCU NCF 1.	S32K358, S32K356, S32K348, S32K338, S32K328
DCMROD8[2]	PRAM2_TCM_GSKT_ALARM	PRAM2_TCM IAHB Gasket monitor alarm status. Read this bit to identify the reason of fault in case of FCCU NCF 1.	S32K358, S32K356, S32K348, S32K338, S32K328
DCMROD8[3]	PRAM2_GSKT_ALARM	PRAM2 IAHB Gasket monitor alarm status. Read this bit to identify the reason of fault in case of FCCU NCF 1.	S32K358, S32K356, S32K348, S32K338, S32K328
DCMROD8[4]	CM7_3_LOCKUP	See Register section for details	S32K389, S32K388

Table continues on the next page...

Table 231. DCM controlled features and availability in product family (continued)

Register field	Register field abbreviation	Register field description	Parts wherein this field is available
DCMROD8[5]	CM7_2_RCCU1_ALARM	See Register section for details	S32K389, S32K388
DCMROD8[6]	CM7_2_RCCU2_ALARM	See Register section for details	S32K389, S32K388
DCMROD8[7]	PERIPH_AXBS_S3_GSKT_ALARM	See Register section for details	S32K389, S32K388
DCMROD8[9]	MAC2_GSKT_ALARM	See Register section for details	S32K389, S32K388
DCMROD8[10]	MAC2_RDATA_EDC_ERR	See Register section for details	S32K389, S32K388
DCMROD8[11]	CM7_3_AHBP_RDATA_EDC_ERR	See Register section for details	S32K389, S32K388
DCMROD8[12]	CM7_3_AHBM_RDATA_EDC_ERR	See Register section for details	S32K389, S32K388
DCMROD8[13]	HSE_AES_ACCEL_AXBS_ALARM	See Register section for details	S32K389, S32K388
DCMROD8[14]	CM7_3_AHBS_ALARM	See Register section for details	S32K389, S32K388
DCMROD8[16]	ACE_RESULT_RDATA_EDC_ERR	See Register section for details	S32K389, S32K388
DCMROD8[17]	ACE_FEED_RDATA_EDC_ERR	See Register section for details	S32K389, S32K388
DCMROD8[18]	AES_ACCEL_AXBS_ALARM	See Register section for details	S32K389, S32K388
DCMROD8[19]	AES_ACCEL_GSKT_ALARM	See Register section for details	S32K389, S32K388
DCMROD8[21]	CM7_3_AHBM_ALARM	See Register section for details	S32K389, S32K388
DCMROD8[22]	CM7_3_AHBP_ALARM	See Register section for details	S32K389, S32K388
DCMROD8[23]	CM7_3_DCDATA_ECC_ERR	See Register section for details	S32K389, S32K388
DCMROD8[24]	CM7_3_DCTAG_ECC_ERR	See Register section for details	S32K389, S32K388
DCMROD8[25]	CM7_3_ICDATA_ECC_ERR	See Register section for details	S32K389, S32K388
DCMROD8[26]	CM7_3_ICTAG_ECC_ERR	See Register section for details	S32K389, S32K388
DCMROD8[27]	CM7_3_ITCM_ECC_ERR	See Register section for details	S32K389, S32K388
DCMROD8[28]	CM7_3_DTCM0_ECC_ERR	See Register section for details	S32K389, S32K388
DCMROD8[29]	CM7_3_DTCM1_ECC_ERR	See Register section for details	S32K389, S32K388
DCMROD9[0]	AES_FEED_DMA_TCD_ECC_ERR	See Register section for details	S32K389, S32K388
DCMROD9[1]	AES_FEED_DMA_TCD_ADDR_ECC_ERR	See Register section for details	S32K389, S32K388
DCMROD9[2]	AES_RESULT_DMA_TCD_ECC_ERR	See Register section for details	S32K389, S32K388

Table continues on the next page...

Table 231. DCM controlled features and availability in product family (continued)

Register field	Register field abbreviation	Register field description	Parts wherein this field is available
DCMROD9[3]	AES_RESULT_DMA_TCD_ADDR_ECC_ERR	See Register section for details	S32K389, S32K388
DCMROD9[4]	AES_KP_CRC_SAFETY_ERR	See Register section for details	S32K389, S32K388
DCMROD9[5]	AES_FEED_DID_SAFETY_ERR	See Register section for details	S32K389, S32K388
DCMROD9[6]	AES_RESULT_DID_SAFETY_ERR	See Register section for details	S32K389, S32K388
DCMROD9[7]	PF1_0_CODE_ECC_ERR	See Register section for details	S32K389
DCMROD9[8]	PF1_0_DATA_ECC_ERR	See Register section for details	S32K389
DCMROD9[9]	PF1_1_CODE_ECC_ERR	See Register section for details	S32K389
DCMROD9[10]	PF1_1_DATA_ECC_ERR	See Register section for details	S32K389
DCMROD9[11]	FLASH1_EDC_ERR	See Register section for details	S32K389
DCMROD9[12]	FLASH1_ADDR_ENC_ERR	See Register section for details	S32K389
DCMROD9[13]	FLASH1_REF_ERR	See Register section for details	S32K389
DCMROD9[14]	FLASH1_RST_ERR	See Register section for details	S32K389
DCMROD9[16]	FLASH1_ECC_ERR	See Register section for details	S32K389
DCMROD9[17]	PRAM3_ECC_ERR	See Register section for details	S32K389
DCMROD9[18]	PRAM3_FCCU_ALARM	See Register section for details	S32K389
DCMROD9[19]	PF0_0_CMP_ALARM	See Register section for details	S32K389
DCMROD9[20]	PF0_1_CMP_ALARM	See Register section for details	S32K389
DCMROD9[21]	PF1_0_CMP_ALARM	See Register section for details	S32K389
DCMROD9[22]	PF1_1_CMP_ALARM	See Register section for details	S32K389
DCMROD9[23]	PF0_0_CHK_CMP_ALARM	See Register section for details	S32K389
DCMROD9[24]	PF0_1_CHK_CMP_ALARM	See Register section for details	S32K389
DCMROD9[25]	PF1_0_CHK_CMP_ALARM	See Register section for details	S32K389
DCMROD9[26]	PF1_1_CHK_CMP_ALARM	See Register section for details	S32K389
DCMROF1[0]	MAC_MDC_CHID_0	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328, S32K344, S32K324, S32K314, S32K342, S32K341, S32K322
DCMROF1[1]	MAC_MDC_CHID_1	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328, S32K344, S32K324, S32K314, S32K342, S32K341, S32K322

Table continues on the next page...

Table 231. DCM controlled features and availability in product family (continued)

Register field	Register field abbreviation	Register field description	Parts wherein this field is available
DCMROF1[2]	MAC_MDC_CHID_2	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328
DCMROF1[3]	MAC1_MDC_CHID_0	See Register section for details	S32K389, S32K388, S32K358
DCMROF1[4]	MAC1_MDC_CHID_1	See Register section for details	S32K389, S32K388, S32K358
DCMROF1[5]	MAC1_MDC_CHID_2	See Register section for details	S32K389, S32K388, S32K358
DCMROF1[16]	AES_FEED_DID_ERR_PRIV	See Register section for details	S32K389, S32K388
DCMROF1[17]	AES_FEED_DID_ERR_NS	See Register section for details	S32K389, S32K388
DCMROF1[18]	AES_FEED_DID_ERR_DID	See Register section for details	S32K389, S32K388
DCMROF1[19]			
DCMROF1[20]			
DCMROF1[21]			
DCMROF1[24]	AES_RESULT_DID_ERR_PRIV	See Register section for details	S32K389, S32K388
DCMROF1[25]	AES_RESULT_DID_ERR_NS	See Register section for details	S32K389, S32K388
DCMROF1[26]	AES_RESULT_DID_ERR_DID	See Register section for details	S32K389, S32K388
DCMROF1[27]			
DCMROF1[28]			
DCMROF1[29]			
DCMROF19[29]	LOCKSTEP_EN	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K344, S32K342, S32K341,
DCMROF20[1]	LMAUTO_DIS	Specifies whether the PMC last-mile automatic crossover from the boot regulation feature is supported for the chip.	S32K344, S32K324, S32K314, S32K342, S32K341, S32K322
DCMROF20[3]	DMA_AXBS_IAHB_BYP	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328, S32K344, S32K324, S32K314, S32K342, S32K341, S32K322
DCMROF20[5]	QSPI_IAHB_BYP	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328, S32K344, S32K324, S32K314, S32K342, S32K341, S32K322
DCMRWD3[1]	CM7_1_LOCKUP_EN	See Register section for details	S32K389, S32K388, S32K338, S32K328, S32K324, S32K322

Table continues on the next page...

Table 231. DCM controlled features and availability in product family (continued)

Register field	Register field abbreviation	Register field description	Parts wherein this field is available
DCMRWD3[3]	CM7_RCCU1_ALARM_EN	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K344, S32K342, S32K341,
DCMRWD3[4]	CM7_RCCU2_ALARM_EN	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K344, S32K342, S32K341,
DCMRWD3[5]	TCM_GSKT_ALARM_EN	See Register section for details	S32K389, S32K388, S32K344, S32K324, S32K314, S32K342, S32K341, S32K322, S32K312, S32K311
DCMRWD3[6]	DMA_SYS_GSKT_ALARM_EN	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328, S32K344, S32K324, S32K314, S32K342, S32K341, S32K322
DCMRWD3[7]	DMA_PERIPH_GSKT_ALARM_EN	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328, S32K344, S32K324, S32K314, S32K342, S32K341, S32K322
DCMRWD3[9]	DMA_AXBS_ALARM_EN	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328, S32K344, S32K324, S32K314, S32K342, S32K341, S32K322
DCMRWD3[10]	SDHC_GSKT_ALARM_EN	Enable bit for enabling the fault monitoring at FCCU NCF 1 for the fault: uSDHC IAHB gasket alarm.	S32K358, S32K356, S32K348, S32K338, S32K328
DCMRWD3[12]	QSPI_GSKT_ALARM_EN	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328, S32K344, S32K324, S32K314, S32K342, S32K341, S32K322
DCMRWD3[14]	AIPS2_GSKT_ALARM_EN	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328, S32K344, S32K324, S32K314, S32K342, S32K341, S32K322
DCMRWD3[17]	TCM_AXBS_ALARM_EN	See Register section for details	S32K389, S32K388, S32K344, S32K324, S32K314, S32K342, S32K341, S32K322

Table continues on the next page...

Table 231. DCM controlled features and availability in product family (continued)

Register field	Register field abbreviation	Register field description	Parts wherein this field is available
DCMRWD3[18]	MAC_GSKT_ALARM_EN	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328, S32K344, S32K324, S32K314, S32K342, S32K341, S32K322
DCMRWD3[19]	PERIPH_AXBS_ALARM_EN	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328, S32K344, S32K324, S32K314, S32K342, S32K341, S32K322
DCMRWD3[20]	PF3_CODE_ECC_ERR_EN	Enable bit for enabling the fault monitoring at FCCU NCF 3 for the fault: Flash3 code ECC uncorrectable error.	S32K388, S32K358, S32K356, S32K338
DCMRWD3[21]	PF3_DATA_ECC_ERR_EN	Enable bit for enabling the fault monitoring at FCCU NCF 3 for the fault: Flash3 data ECC uncorrectable error.	S32K388, S32K358, S32K356, S32K338
DCMRWD3[23]	PRAM2_ECC_ERR_EN	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328
DCMRWD3[24]	PRAM1_ECC_ERR_EN	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328, S32K344, S32K324, S32K314
DCMRWD3[27]	CM7_1_DCDATA_ECC_ERR_EN	See Register section for details	S32K389, S32K388, S32K338, S32K328, S32K324, S32K322
DCMRWD3[29]	CM7_1_DCTAG_ECC_ERR_EN	See Register section for details	S32K389, S32K388, S32K338, S32K328, S32K324, S32K322
DCMRWD3[31]	CM7_1_ICDATA_ECC_ERR_EN	See Register section for details	S32K389, S32K388, S32K338, S32K328, S32K324, S32K322
DCMRWD4[1]	CM7_1_ICTAG_ECC_ERR_EN	See Register section for details	S32K389, S32K388, S32K338, S32K328, S32K324, S32K322
DCMRWD4[5]	CM7_1_ITCM_ECC_ERR_EN	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328, S32K344, S32K324, S32K342, S32K341, S32K322
DCMRWD4[6]	CM7_1_DTCM0_ECC_ERR_EN	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328, S32K344, S32K324, S32K342, S32K341, S32K322

Table continues on the next page...

Table 231. DCM controlled features and availability in product family (continued)

Register field	Register field abbreviation	Register field description	Parts wherein this field is available
DCMRWD4[7]	CM7_1_DTCM1_ECC_ERR_EN	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328, S32K344, S32K324, S32K342, S32K341, S32K322
DCMRWD4[10]	PRAM1_FCCU_ALARM_EN	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328, S32K344, S32K324, S32K314
DCMRWD4[16]	PF2_CODE_ECC_ERR_EN	Enable bit for enabling the fault monitoring at FCCU NCF 3 for the fault: Flash2 code ECC uncorrectable error.	S32K388, S32K338, S32K328, S32K324, S32K322
DCMRWD4[17]	PF2_DATA_ECC_ERR_EN	Enable bit for enabling the fault monitoring at FCCU NCF 3 for the fault: Flash2 data ECC uncorrectable error.	S32K388, S32K338, S32K328, S32K324, S32K322
DCMRWD4[23]	PRAM2_FCCU_ALARM_EN	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328
DCMRWD4[28]	SDHC_RDATA_EDC_ERR_EN	Enable bit for enabling the fault monitoring at FCCU NCF 1 for the fault: Integrity (EDC) error on uSDHC read data for safety.	S32K358, S32K356, S32K348, S32K338, S32K328
DCMRWD4[31]	CM7_2_LOCKUP_EN	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K338
DCMRWD5[14]	TCM_RDATA_EDC_ERR_EN	Specifies whether an integrity error is reported on the TCM read data. The field enables fault monitoring at FCCU NCF 1, if there is an integrity error on the TCM read data, for safety.	S32K344, S32K324, S32K314, S32K342, S32K341, S32K322
DCMRWD5[15]	MAC_RDATA_EDC_ERR_EN	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328, S32K344, S32K324, S32K314, S32K342, S32K341, S32K322
DCMRWD5[18]	CM7_1_AHBP_RDATA_EDC_ERR_EN	See Register section for details	S32K389, S32K388, S32K338, S32K328, S32K324, S32K322
DCMRWD5[19]	CM7_1_AHBM_RDATA_EDC_ERR_EN	See Register section for details	S32K389, S32K388, S32K338, S32K328, S32K324, S32K322

Table continues on the next page...

Table 231. DCM controlled features and availability in product family (continued)

Register field	Register field abbreviation	Register field description	Parts wherein this field is available
DCMRWD5[23]	CM7_2_AHBP_RDATA_EDC_ERR_EN	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K338
DCMRWD5[24]	CM7_2_AHBM_RDATA_EDC_ERR_EN	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K338
DCMRWD5[25]	CM7_2_DCDATA_ECC_ERR_EN	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K338
DCMRWD5[26]	CM7_2_DCTAG_ECC_ERR_EN	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K338
DCMRWD5[27]	CM7_2_ICDATA_ECC_ERR_EN	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K338
DCMRWD5[28]	CM7_2_ICTAG_ECC_ERR_EN	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K338
DCMRWD5[29]	CM7_2_ITCM_ECC_ERR_EN	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K338
DCMRWD5[30]	CM7_2_DTCM0_ECC_ERR_EN	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K338
DCMRWD5[31]	CM7_2_DTCM1_ECC_ERR_EN	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K338
DCMRWD6[6]	eMIOS2_DBG_DIS_CM7_0	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328, S32K344, S32K324, S32K314
DCMRWD6[9]	SWT1_DBG_DIS_CM7_0	See Register section for details	S32K389, S32K388, S32K338, S32K328, S32K324, S32K322
DCMRWD6[11]	STM1_DBG_DIS_CM7_0	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328, S32K344, S32K324, S32K314, S32K342, S32K341, S32K322
DCMRWD6[14]	PIT2_DBG_DIS_CM7_0	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328, S32K344, S32K324, S32K314, S32K342, S32K341, S32K322
DCMRWD6[19]	LPSPI4_DBG_DIS_CM7_0	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328, S32K344, S32K324, S32K314
DCMRWD6[20]	LPSPI5_DBG_DIS_CM7_0	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338,

Table continues on the next page...

Table 231. DCM controlled features and availability in product family (continued)

Register field	Register field abbreviation	Register field description	Parts wherein this field is available
			S32K328, S32K344, S32K324, S32K314
DCMRWD6[27]	FLEXCAN3_DBG_DIS_CM7_0	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328, S32K344, S32K324, S32K314, S32K342, S32K341, S32K322, S32K312
DCMRWD6[28]	FLEXCAN4_DBG_DIS_CM7_0	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328, S32K344, S32K324, S32K314, S32K312
DCMRWD6[29]	FLEXCAN5_DBG_DIS_CM7_0	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328, S32K344, S32K324, S32K314, S32K312
DCMRWD6[30]	SAI0_DBG_DIS_CM7_0	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328, S32K344, S32K324, S32K314, S32K342, S32K341, S32K322
DCMRWD6[31]	SAI1_DBG_DIS_CM7_0	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328, S32K344, S32K324, S32K314, S32K342, S32K341, S32K322
DCMRWD7[1]	FLEXCAN6_DBG_DIS_CM7_0	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328
DCMRWD7[2]	FLEXCAN7_DBG_DIS_CM7_0	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328
DCMRWD7[3]	STM2_DBG_DIS_CM7_0	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328
DCMRWD7[4]	SWT2_DBG_DIS_CM7_0	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K338
DCMRWD7[21]	SWT3_DBG_DIS_CM7_0	See Register section for details	S32K389, S32K388
DCMRWD7[22]	STM3_DBG_DIS_CM7_0	See Register section for details	S32K389, S32K388
DCMRWD7[23]	PIT3_DBG_DIS_CM7_0	See Register section for details	S32K389, S32K388
DCMRWD7[24]	FLEXCAN8_DBG_DIS_CM7_0	See Register section for details	S32K389
DCMRWD7[25]	FLEXCAN9_DBG_DIS_CM7_0	See Register section for details	S32K389

Table continues on the next page...

Table 231. DCM controlled features and availability in product family (continued)

Register field	Register field abbreviation	Register field description	Parts wherein this field is available
DCMRWD7[26]	FLEXCAN10_DBG_DIS_CM7_0	See Register section for details	S32K389
DCMRWD7[27]	FLEXCAN11_DBG_DIS_CM7_0	See Register section for details	S32K389
DCMRWD8[0]	EDMA_DBG_DIS_CM7_1	See Register section for details	S32K389, S32K388, S32K338, S32K328, S32K324, S32K322
DCMRWD8[1]	FCCU_DBG_DIS_CM7_1	See Register section for details	S32K389, S32K388, S32K338, S32K328, S32K324, S32K322
DCMRWD8[2]	LCU0_DBG_DIS_CM7_1	See Register section for details	S32K389, S32K388, S32K338, S32K328, S32K324, S32K322
DCMRWD8[3]	LCU1_DBG_DIS_CM7_1	See Register section for details	S32K389, S32K388, S32K338, S32K328, S32K324, S32K322
DCMRWD8[4]	eMIOS0_DBG_DIS_CM7_1	See Register section for details	S32K389, S32K388, S32K338, S32K328, S32K324, S32K322
DCMRWD8[5]	eMIOS1_DBG_DIS_CM7_1	See Register section for details	S32K389, S32K388, S32K338, S32K328, S32K324, S32K322
DCMRWD8[6]	eMIOS2_DBG_DIS_CM7_1	See Register section for details	S32K389, S32K388, S32K338, S32K328, S32K324
DCMRWD8[7]	RTC_DBG_DIS_CM7_1	See Register section for details	S32K389, S32K388, S32K338, S32K328, S32K324, S32K322
DCMRWD8[8]	SWT0_DBG_DIS_CM7_1	See Register section for details	S32K389, S32K388, S32K338, S32K328, S32K324, S32K322
DCMRWD8[9]	SWT1_DBG_DIS_CM7_1	See Register section for details	S32K389, S32K388, S32K338, S32K328, S32K324, S32K322
DCMRWD8[10]	STM0_DBG_DIS_CM7_1	See Register section for details	S32K389, S32K388, S32K338, S32K328, S32K324, S32K322
DCMRWD8[11]	STM1_DBG_DIS_CM7_1	See Register section for details	S32K389, S32K388, S32K338, S32K328, S32K324, S32K322
DCMRWD8[12]	PIT0_DBG_DIS_CM7_1	See Register section for details	S32K389, S32K388, S32K338, S32K328, S32K324, S32K322
DCMRWD8[13]	PIT1_DBG_DIS_CM7_1	See Register section for details	S32K389, S32K388, S32K338, S32K328, S32K324, S32K322
DCMRWD8[14]	PIT2_DBG_DIS_CM7_1	See Register section for details	S32K389, S32K388, S32K338, S32K328, S32K324, S32K322
DCMRWD8[15]	LPSPI0_DBG_DIS_CM7_1	See Register section for details	S32K389, S32K388, S32K338, S32K328, S32K324, S32K322
DCMRWD8[16]	LPSPI1_DBG_DIS_CM7_1	See Register section for details	S32K389, S32K388, S32K338, S32K328, S32K324, S32K322

Table continues on the next page...

Table 231. DCM controlled features and availability in product family (continued)

Register field	Register field abbreviation	Register field description	Parts wherein this field is available
DCMRWD8[17]	LPSPI2_DBG_DIS_CM7_1	See Register section for details	S32K389, S32K388, S32K338, S32K328, S32K324, S32K322
DCMRWD8[18]	LPSPI3_DBG_DIS_CM7_1	See Register section for details	S32K389, S32K388, S32K338, S32K328, S32K324, S32K322
DCMRWD8[19]	LPSPI4_DBG_DIS_CM7_1	See Register section for details	S32K389, S32K388, S32K338, S32K328, S32K324
DCMRWD8[20]	LPSPI5_DBG_DIS_CM7_1	See Register section for details	S32K389, S32K388, S32K338, S32K328, S32K324
DCMRWD8[21]	LPI2C0_DBG_DIS_CM7_1	See Register section for details	S32K389, S32K388, S32K338, S32K328, S32K324, S32K322
DCMRWD8[22]	LPI2C1_DBG_DIS_CM7_1	See Register section for details	S32K389, S32K388, S32K338, S32K328, S32K324, S32K322
DCMRWD8[23]	FLEXIO_DBG_DIS_CM7_1	See Register section for details	S32K389, S32K388, S32K338, S32K328, S32K324, S32K322
DCMRWD8[24]	FLEXCAN0_DBG_DIS_CM7_1	See Register section for details	S32K389, S32K388, S32K338, S32K328, S32K324, S32K322
DCMRWD8[25]	FLEXCAN1_DBG_DIS_CM7_1	See Register section for details	S32K389, S32K388, S32K338, S32K328, S32K324, S32K322
DCMRWD8[26]	FLEXCAN2_DBG_DIS_CM7_1	See Register section for details	S32K389, S32K388, S32K338, S32K328, S32K324, S32K322
DCMRWD8[27]	FLEXCAN3_DBG_DIS_CM7_1	See Register section for details	S32K389, S32K388, S32K338, S32K328, S32K324, S32K322
DCMRWD8[28]	FLEXCAN4_DBG_DIS_CM7_1	See Register section for details	S32K389, S32K388, S32K338, S32K328, S32K324
DCMRWD8[29]	FLEXCAN5_DBG_DIS_CM7_1	See Register section for details	S32K389, S32K388, S32K338, S32K328, S32K324
DCMRWD8[30]	SAI0_DBG_DIS_CM7_1	See Register section for details	S32K389, S32K388, S32K338, S32K328, S32K324, S32K322
DCMRWD8[31]	SAI1_DBG_DIS_CM7_1	See Register section for details	S32K389, S32K388, S32K338, S32K328, S32K324, S32K322
DCMRWD9[1]	FLEXCAN6_DBG_DIS_CM7_1	See Register section for details	S32K389, S32K388, S32K338, S32K328
DCMRWD9[2]	FLEXCAN7_DBG_DIS_CM7_1	See Register section for details	S32K389, S32K388, S32K338, S32K328
DCMRWD9[3]	STM2_DBG_DIS_CM7_1	See Register section for details	S32K389, S32K388, S32K338, S32K328
DCMRWD9[4]	SWT2_DBG_DIS_CM7_1	See Register section for details	S32K389, S32K388, S32K338
DCMRWD9[21]	SWT3_DBG_DIS_CM7_1	See Register section for details	S32K389, S32K388

Table continues on the next page...

Table 231. DCM controlled features and availability in product family (continued)

Register field	Register field abbreviation	Register field description	Parts wherein this field is available
DCMRWD9[22]	STM3_DBG_DIS_CM7_1	See Register section for details	S32K389, S32K388
DCMRWD9[23]	PIT3_DBG_DIS_CM7_1	See Register section for details	S32K389, S32K388
DCMRWD9[24]	FLEXCAN8_DBG_DIS_CM7_1	See Register section for details	S32K389
DCMRWD9[25]	FLEXCAN9_DBG_DIS_CM7_1	See Register section for details	S32K389
DCMRWD9[26]	FLEXCAN10_DBG_DIS_CM7_1	See Register section for details	S32K389
DCMRWD9[27]	FLEXCAN11_DBG_DIS_CM7_1	See Register section for details	S32K389
DCMRWD12[0]	EDMA_DBG_DIS_CM7_2	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K338
DCMRWD12[1]	FCCU_DBG_DIS_CM7_2	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K338
DCMRWD12[2]	LCU0_DBG_DIS_CM7_2	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K338
DCMRWD12[3]	LCU1_DBG_DIS_CM7_2	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K338
DCMRWD12[4]	eMIOS0_DBG_DIS_CM7_2	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K338
DCMRWD12[5]	eMIOS1_DBG_DIS_CM7_2	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K338
DCMRWD12[6]	eMIOS2_DBG_DIS_CM7_2	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K338
DCMRWD12[7]	RTC_DBG_DIS_CM7_2	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K338
DCMRWD12[8]	SWT0_DBG_DIS_CM7_2	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K338
DCMRWD12[9]	SWT1_DBG_DIS_CM7_2	See Register section for details	S32K389, S32K388, S32K338
DCMRWD12[10]	STM0_DBG_DIS_CM7_2	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K338
DCMRWD12[11]	STM1_DBG_DIS_CM7_2	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K338
DCMRWD12[12]	PIT0_DBG_DIS_CM7_2	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K338
DCMRWD12[13]	PIT1_DBG_DIS_CM7_2	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K338
DCMRWD12[14]	PIT2_DBG_DIS_CM7_2	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K338

Table continues on the next page...

Table 231. DCM controlled features and availability in product family (continued)

Register field	Register field abbreviation	Register field description	Parts wherein this field is available
DCMRWD12[15]	LPSPI0_DBG_DIS_CM7_2	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K338
DCMRWD12[16]	LPSPI1_DBG_DIS_CM7_2	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K338
DCMRWD12[17]	LPSPI2_DBG_DIS_CM7_2	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K338
DCMRWD12[18]	LPSPI3_DBG_DIS_CM7_2	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K338
DCMRWD12[19]	LPSPI4_DBG_DIS_CM7_2	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K338
DCMRWD12[20]	LPSPI5_DBG_DIS_CM7_2	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K338
DCMRWD12[21]	LPI2C0_DBG_DIS_CM7_2	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K338
DCMRWD12[22]	LPI2C1_DBG_DIS_CM7_2	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K338
DCMRWD12[23]	FLEXIO_DBG_DIS_CM7_2	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K338
DCMRWD12[24]	FLEXCAN0_DBG_DIS_CM7_2	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K338
DCMRWD12[25]	FLEXCAN1_DBG_DIS_CM7_2	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K338
DCMRWD12[26]	FLEXCAN2_DBG_DIS_CM7_2	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K338
DCMRWD12[27]	FLEXCAN3_DBG_DIS_CM7_2	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K338
DCMRWD12[28]	FLEXCAN4_DBG_DIS_CM7_2	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K338
DCMRWD12[29]	FLEXCAN5_DBG_DIS_CM7_2	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K338
DCMRWD12[30]	SAI0_DBG_DIS_CM7_2	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K338
DCMRWD12[31]	SAI1_DBG_DIS_CM7_2	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K338
DCMRWD13[1]	FLEXCAN6_DBG_DIS_CM7_2	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K338
DCMRWD13[2]	FLEXCAN7_DBG_DIS_CM7_2	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K338

Table continues on the next page...

Table 231. DCM controlled features and availability in product family (continued)

Register field	Register field abbreviation	Register field description	Parts wherein this field is available
DCMRWD13[3]	STM2_DBG_DIS_CM7_2	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K338
DCMRWD13[4]	SWT2_DBG_DIS_CM7_2	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K338
DCMRWD13[21]	SWT3_DBG_DIS_CM7_2	See Register section for details	S32K389, S32K388
DCMRWD13[22]	STM3_DBG_DIS_CM7_2	See Register section for details	S32K389, S32K388
DCMRWD13[23]	PIT3_DBG_DIS_CM7_2	See Register section for details	S32K389, S32K388
DCMRWD13[24]	FLEXCAN8_DBG_DIS_CM7_2	See Register section for details	S32K389
DCMRWD13[25]	FLEXCAN9_DBG_DIS_CM7_2	See Register section for details	S32K389
DCMRWD13[26]	FLEXCAN10_DBG_DIS_CM7_2	See Register section for details	S32K389
DCMRWD13[27]	FLEXCAN11_DBG_DIS_CM7_2	See Register section for details	S32K389
DCMRWD14[3]	QSPI_FLASHA_ECC_ERR_EN	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328
DCMRWD14[26]	AIPS0_GSKT_ALARM_EN	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328
DCMRWD14[30]	TCM_PRAM_AXBS_ALARM_EN	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328
DCMRWD15[0]	CM7_0_AHBM_ALARM_EN	See Register section for details	S32K389, S32K388
DCMRWD15[1]	CM7_1_AHBM_ALARM_EN	See Register section for details	S32K389, S32K388
DCMRWD15[2]	CM7_2_AHBM_ALARM_EN	See Register section for details	S32K389, S32K388
DCMRWD15[3]	CM7_0_AHBP_ALARM_EN	See Register section for details	S32K389, S32K388
DCMRWD15[4]	CM7_1_AHBP_ALARM_EN	See Register section for details	S32K389, S32K388
DCMRWD15[5]	CM7_2_AHBP_ALARM_EN	See Register section for details	S32K389, S32K388
DCMRWD15[11]	VDD1P1_GNG2_ERR_EN	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328
DCMRWD15[12]	VDD2P5_GNG2_ERR_EN	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328
DCMRWD15[28]	CM7_0_AHBS_ALARM_EN	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328

Table continues on the next page...

Table 231. DCM controlled features and availability in product family (continued)

Register field	Register field abbreviation	Register field description	Parts wherein this field is available
DCMRWD15[29]	CM7_1_AHBS_ALARM_EN	See Register section for details	S32K389, S32K388, S32K338, S32K328
DCMRWD15[30]	CM7_2_AHBS_ALARM_EN	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K338
DCMRWD16[0]	PRAM0_GSKT_ALARM_EN	Enables bit for enabling the fault monitoring at FCCU NCF 1 for the fault: PRAM0 IAHB Gasket monitor alarm.	S32K358, S32K356, S32K348, S32K338, S32K328
DCMRWD16[1]	PRAM1_GSKT_ALARM_EN	Enables bit for enabling the fault monitoring at FCCU NCF 1 for the fault: PRAM1 IAHB Gasket monitor alarm.	S32K358, S32K356, S32K348, S32K338, S32K328
DCMRWD16[2]	PRAM2_TCM_GSKT_ALARM_EN	Enables bit for enabling the fault monitoring at FCCU NCF 1 for the fault: PRAM2_TCM IAHB Gasket monitor alarm.	S32K358, S32K356, S32K348, S32K338, S32K328
DCMRWD16[3]	PRAM2_GSKT_ALARM_EN	Enables bit for enabling the fault monitoring at FCCU NCF 1 for the fault: PRAM2 IAHB Gasket monitor alarm.	S32K358, S32K356, S32K348, S32K338, S32K328
DCMRWD16[4]	CM7_3_LOCKUP_EN	See Register section for details	S32K389, S32K388
DCMRWD16[5]	CM7_2_RCCU1_ALARM_EN	See Register section for details	S32K389, S32K388
DCMRWD16[6]	CM7_2_RCCU2_ALARM_EN	See Register section for details	S32K389, S32K388
DCMRWD16[7]	PERIPH_AXBS_S3_GSKT_ALARM_EN	See Register section for details	S32K389, S32K388
DCMRWD16[9]	MAC2_GSKT_ALARM_EN	See Register section for details	S32K389, S32K388
DCMRWD16[10]	MAC2_RDATA_EDC_ERR_EN	See Register section for details	S32K389, S32K388
DCMRWD16[11]	CM7_3_AHBP_RDATA_EDC_ERR_EN	See Register section for details	S32K389, S32K388
DCMRWD16[12]	CM7_3_AHBM_RDATA_EDC_ERR_EN	See Register section for details	S32K389, S32K388
DCMRWD16[13]	HSE_AES_ACCEL_AXBS_ALARM_EN	See Register section for details	S32K389, S32K388
DCMRWD16[14]	CM7_3_AHBS_ALARM_EN	See Register section for details	S32K389, S32K388
DCMRWD16[16]	ACE_RESULT_RDATA_EDC_ERR_EN	See Register section for details	S32K389, S32K388
DCMRWD16[17]	ACE_FEED_RDATA_EDC_ERR_EN	See Register section for details	S32K389, S32K388

Table continues on the next page...

Table 231. DCM controlled features and availability in product family (continued)

Register field	Register field abbreviation	Register field description	Parts wherein this field is available
DCMRWD16[18]	AES_ACCEL_AXBS_ALARM_EN	See Register section for details	S32K389, S32K388
DCMRWD16[19]	AES_ACCEL_GSKT_ALARM_EN	See Register section for details	S32K389, S32K388
DCMRWD16[21]	CM7_3_AHBM_ALARM_EN	See Register section for details	S32K389, S32K388
DCMRWD16[22]	CM7_3_AHBP_ALARM_EN	See Register section for details	S32K389, S32K388
DCMRWD16[23]	CM7_3_DCDATA_ECC_ERR_EN	See Register section for details	S32K389, S32K388
DCMRWD16[24]	CM7_3_DCTAG_ECC_ERR_EN	See Register section for details	S32K389, S32K388
DCMRWD16[25]	CM7_3_ICDATA_ECC_ERR_EN	See Register section for details	S32K389, S32K388
DCMRWD16[26]	CM7_3_ICTAG_ECC_ERR_EN	See Register section for details	S32K389, S32K388
DCMRWD16[27]	CM7_3_ITCM_ECC_ERR_EN	See Register section for details	S32K389, S32K388
DCMRWD16[28]	CM7_3_DTCM0_ECC_ERR_EN	See Register section for details	S32K389, S32K388
DCMRWD16[29]	CM7_3_DTCM1_ECC_ERR_EN	See Register section for details	S32K389, S32K388
DCMRWD17[0]	AES_FEED_DMA_TCD_ECC_ERR_EN	See Register section for details	S32K389, S32K388
DCMRWD17[1]	AES_FEED_DMA_TCD_ADDR_ECC_ERR_EN	See Register section for details	S32K389, S32K388
DCMRWD17[2]	AES_RESULT_DMA_TCD_ECC_ERR_EN	See Register section for details	S32K389, S32K388
DCMRWD17[3]	AES_RESULT_DMA_TCD_ADDR_ECC_ERR_EN	See Register section for details	S32K389, S32K388
DCMRWD17[4]	AES_KP_CRC_SAFETY_ERR_EN	See Register section for details	S32K389, S32K388
DCMRWD17[5]	AES_FEED_DID_SAFETY_ERR_EN	See Register section for details	S32K389, S32K388
DCMRWD17[6]	AES_RESULT_DID_SAFETY_ERR_EN	See Register section for details	S32K389, S32K388
DCMRWD17[7]	PF1_0_CODE_ECC_ERR_EN	See Register section for details	S32K389
DCMRWD17[8]	PF1_0_DATA_ECC_ERR_EN	See Register section for details	S32K389
DCMRWD17[9]	PF1_1_CODE_ECC_ERR_EN	See Register section for details	S32K389
DCMRWD17[10]	PF1_1_DATA_ECC_ERR_EN	See Register section for details	S32K389
DCMRWD17[11]	FLASH1_EDC_ERR_EN	See Register section for details	S32K389

Table continues on the next page...

Table 231. DCM controlled features and availability in product family (continued)

Register field	Register field abbreviation	Register field description	Parts wherein this field is available
DCMRWD17[12]	FLASH1_ADDR_ENC_ERR_EN	See Register section for details	S32K389
DCMRWD17[13]	FLASH1_REF_ERR_EN	See Register section for details	S32K389
DCMRWD17[14]	FLASH1_RST_ERR_EN	See Register section for details	S32K389
DCMRWD17[16]	FLASH1_ECC_ERR_EN	See Register section for details	S32K389
DCMRWD17[17]	PRAM3_ECC_ERR_EN	See Register section for details	S32K389
DCMRWD17[18]	PRAM3_FCCU_ALARM_EN	See Register section for details	S32K389
DCMRWD17[19]	PF0_0_CMP_EN	See Register section for details	S32K389
DCMRWD17[20]	PF0_1_CMP_EN	See Register section for details	S32K389
DCMRWD17[21]	PF1_0_CMP_EN	See Register section for details	S32K389
DCMRWD17[22]	PF1_1_CMP_EN	See Register section for details	S32K389
DCMRWD17[23]	PF0_0_CHK_CMP_EN	See Register section for details	S32K389
DCMRWD17[24]	PF0_1_CHK_CMP_EN	See Register section for details	S32K389
DCMRWD17[25]	PF1_0_CHK_CMP_EN	See Register section for details	S32K389
DCMRWD17[26]	PF1_1_CHK_CMP_EN	See Register section for details	S32K389
DCMRWD19[0]	EDMA_DBG_DIS_CM7_3	See Register section for details	S32K389, S32K388
DCMRWD19[1]	FCCU_DBG_DIS_CM7_3	See Register section for details	S32K389, S32K388
DCMRWD19[2]	LCU0_DBG_DIS_CM7_3	See Register section for details	S32K389, S32K388
DCMRWD19[3]	LCU1_DBG_DIS_CM7_3	See Register section for details	S32K389, S32K388
DCMRWD19[4]	eMIOS0_DBG_DIS_CM7_3	See Register section for details	S32K389, S32K388
DCMRWD19[5]	eMIOS1_DBG_DIS_CM7_3	See Register section for details	S32K389, S32K388
DCMRWD19[6]	eMIOS2_DBG_DIS_CM7_3	See Register section for details	S32K389, S32K388
DCMRWD19[7]	RTC_DBG_DIS_CM7_3	See Register section for details	S32K389, S32K388
DCMRWD19[8]	SWT0_DBG_DIS_CM7_3	See Register section for details	S32K389, S32K388
DCMRWD19[9]	SWT1_DBG_DIS_CM7_3	See Register section for details	S32K389, S32K388
DCMRWD19[10]	STM0_DBG_DIS_CM7_3	See Register section for details	S32K389, S32K388
DCMRWD19[11]	STM1_DBG_DIS_CM7_3	See Register section for details	S32K389, S32K388
DCMRWD19[12]	PIT0_DBG_DIS_CM7_3	See Register section for details	S32K389, S32K388
DCMRWD19[13]	PIT1_DBG_DIS_CM7_3	See Register section for details	S32K389, S32K388
DCMRWD19[14]	PIT2_DBG_DIS_CM7_3	See Register section for details	S32K389, S32K388
DCMRWD19[15]	LPSPI0_DBG_DIS_CM7_3	See Register section for details	S32K389, S32K388
DCMRWD19[16]	LPSPI1_DBG_DIS_CM7_3	See Register section for details	S32K389, S32K388

Table continues on the next page...

Table 231. DCM controlled features and availability in product family (continued)

Register field	Register field abbreviation	Register field description	Parts wherein this field is available
DCMRWD19[17]	LPSPI2_DBG_DIS_CM7_3	See Register section for details	S32K389, S32K388
DCMRWD19[18]	LPSPI3_DBG_DIS_CM7_3	See Register section for details	S32K389, S32K388
DCMRWD19[19]	LPSPI4_DBG_DIS_CM7_3	See Register section for details	S32K389, S32K388
DCMRWD19[20]	LPSPI5_DBG_DIS_CM7_3	See Register section for details	S32K389, S32K388
DCMRWD19[21]	LPI2C0_DBG_DIS_CM7_3	See Register section for details	S32K389, S32K388
DCMRWD19[22]	LPI2C1_DBG_DIS_CM7_3	See Register section for details	S32K389, S32K388
DCMRWD19[23]	FLEXIO_DBG_DIS_CM7_3	See Register section for details	S32K389, S32K388
DCMRWD19[24]	FLEXCAN0_DBG_DIS_CM7_3	See Register section for details	S32K389, S32K388
DCMRWD19[25]	FLEXCAN1_DBG_DIS_CM7_3	See Register section for details	S32K389, S32K388
DCMRWD19[26]	FLEXCAN2_DBG_DIS_CM7_3	See Register section for details	S32K389, S32K388
DCMRWD19[27]	FLEXCAN3_DBG_DIS_CM7_3	See Register section for details	S32K389, S32K388
DCMRWD19[28]	FLEXCAN4_DBG_DIS_CM7_3	See Register section for details	S32K389, S32K388
DCMRWD19[29]	FLEXCAN5_DBG_DIS_CM7_3	See Register section for details	S32K389, S32K388
DCMRWD19[30]	SAI0_DBG_DIS_CM7_3	See Register section for details	S32K389, S32K388
DCMRWD19[31]	SAI1_DBG_DIS_CM7_3	See Register section for details	S32K389, S32K388
DCMRWD20[1]	FLEXCAN6_DBG_DIS_CM7_3	See Register section for details	S32K389, S32K388
DCMRWD20[2]	FLEXCAN7_DBG_DIS_CM7_3	See Register section for details	S32K389, S32K388
DCMRWD20[3]	STM2_DBG_DIS_CM7_3	See Register section for details	S32K389, S32K388
DCMRWD20[4]	SWT2_DBG_DIS_CM7_3	See Register section for details	S32K389, S32K388
DCMRWD20[21]	SWT3_DBG_DIS_CM7_3	See Register section for details	S32K389, S32K388
DCMRWD20[22]	STM3_DBG_DIS_CM7_3	See Register section for details	S32K389, S32K388
DCMRWD20[23]	PIT3_DBG_DIS_CM7_3	See Register section for details	S32K389, S32K388
DCMRWD20[24]	FLEXCAN8_DBG_DIS_CM7_3	See Register section for details	S32K389
DCMRWD20[25]	FLEXCAN9_DBG_DIS_CM7_3	See Register section for details	S32K389
DCMRWD20[26]	FLEXCAN10_DBG_DIS_CM7_3	See Register section for details	S32K389
DCMRWD20[27]	FLEXCAN11_DBG_DIS_CM7_3	See Register section for details	S32K389
DCMRWF1[0]	CAN_TIMESTAMP_SEL	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328, S32K344, S32K324, S32K314, S32K342, S32K341, S32K322

Table continues on the next page...

Table 231. DCM controlled features and availability in product family (continued)

Register field	Register field abbreviation	Register field description	Parts wherein this field is available
DCMRWF1[1]	CAN_TIMESTAMP_EN	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328, S32K344, S32K324, S32K314, S32K342, S32K341, S32K322
DCMRWF1[6]	MAC_CONF_SEL	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328
DCMRWF1[7]	MAC_CONF_SEL	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328, S32K344, S32K324, S32K314, S32K342, S32K341, S32K322
DCMRWF1[15]	VDD_HV_B_IO_CTRL_LATCH	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328, S32K344, S32K324, S32K314, S32K342, S32K341, S32K322
DCMRWF1[19]	MAC_SB_END_CTRL	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328
DCMRWF1[26]	VDD_HV_B_VLT_DVDR_EN	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328, S32K344, S32K324, S32K314, S32K342, S32K341, S32K322
DCMRWF1[27]	VDD_1_5_VLT_DVDR_EN	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328, S32K344, S32K324, S32K314, S32K342, S32K341, S32K322
DCMRWF2[8]	PMOS_CTRL_GPIO_DATA	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328
DCMRWF2[11]	SAI_MCLK2_SEL	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328
DCMRWF2[12]	SUPPLY2_MON_EN	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328
DCMRWF2[13]	SUPPLY2_MON_SEL	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328
DCMRWF2[14]			

Table continues on the next page...

Table 231. DCM controlled features and availability in product family (continued)

Register field	Register field abbreviation	Register field description	Parts wherein this field is available
DCMRWF2[15]			
DCMRWF2[17]	VIRT_WRAP_IPSYNC_BYPASS	See Register section for details	S32K389, S32K388
DCMRWF2[21]	PGOOD_POLARITY	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328
DCMRWF2[24]	PLL1_LOL_RST_EN	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328
DCMRWF2[25]	WKPU0_SRC_SELECT	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328
DCMRWF2[26]	WKPU14_SRC_SELECT	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328
DCMRWF2[27]	WKPU15_SRC_SELECT	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328
DCMRWF2[28]	WKPU18_SRC_SELECT	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328
DCMRWF2[29]	WKPU27_SRC_SELECT	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328
DCMRWF2[30]	WKPU45_SRC_SELECT	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328
DCMRWF2[31]	WKPU8_SRC_SELECT	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328
DCMRWF3[13]	MAC_RX_CLK_MUX_BYPASS	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328
DCMRWF3[14]	MAC_RX_CLK_MUX_BYPASS	See Register section for details	S32K358, S32K356, S32K348, S32K338, S32K328
DCMRWF3[15]	MAC_TX_CLK_MUX_BYPASS	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328
DCMRWF4[0]	MUX_MODE_EN_ADC1_S18	Controls the selection of GPIOs to drive ADC1 standard channel 18th.	S32K311

Table continues on the next page...

Table 231. DCM controlled features and availability in product family (continued)

Register field	Register field abbreviation	Register field description	Parts wherein this field is available
DCMRWF4[5]	MUX_MODE_EN_ADC1_S22	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328, S32K344, S32K324, S32K314, S32K342, S32K341, S32K322, S32K312
DCMRWF4[6]	MUX_MODE_EN_ADC1_S23	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328, S32K344, S32K324, S32K314, S32K342, S32K341, S32K322, S32K312
DCMRWF4[7]	MUX_MODE_EN_ADC0_S12	Controls the selection of GPIOs to drive ADC_0 standard channel 12.	S32K311
DCMRWF4[8]	MUX_MODE_EN_ADC0_S13	Controls the selection of GPIOs to drive ADC_0 standard channel 13.	S32K311
DCMRWF4[9]	MUX_MODE_EN_ADC2_S8	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328, S32K344, S32K324, S32K314
DCMRWF4[10]	MUX_MODE_EN_ADC2_S9	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K348, S32K338, S32K328, S32K344, S32K324, S32K314
DCMRWF4[11]	MUX_MODE_EN_ADC0_S14	Controls the selection of GPIOs to drive ADC_0 standard channel 14.	S32K311
DCMRWF4[12]	MUX_MODE_EN_ADC0_S17	Controls the selection of GPIOs to drive ADC_0 standard channel 17.	S32K311
DCMRWF4[18]	CM7_1_CPUWAIT	See Register section for details	S32K389, S32K388, S32K338, S32K328, S32K324, S32K322
DCMRWF4[19]	CM7_2_CPUWAIT	See Register section for details	S32K389, S32K388, S32K358, S32K356, S32K338
DCMRWF4[20]	CM7_3_CPUWAIT	See Register section for details	S32K389, S32K388
DCMRWF4[23]	MAC_SB_END_CTRL	See Register section for details	S32K389, S32K388
DCMRWF4[24]	MAC2_CONF_SEL	See Register section for details	S32K389, S32K388
DCMRWF4[25]			
DCMRWF4[26]	MAC_RMII_CLK_MUX_BYPASS	See Register section for details	S32K389, S32K388

Table continues on the next page...

Table 231. DCM controlled features and availability in product family (continued)

Register field	Register field abbreviation	Register field description	Parts wherein this field is available
DCMRWF4[27]	MAC2_RMII_CLK_MUX_BYPASS	See Register section for details	S32K389, S32K388
DCMRWF4[28]	MUX_MODE_EN_ADC0_P2	Controls the selection of GPIOs to drive ADC0 precision channel 2nd.	S32K311
DCMRWF4[29]	MAC2_RX_CLK_MUX_BYPASS	See Register section for details	S32K389, S32K388
DCMRWF4[30]			
DCMRWF4[31]	MAC2_TX_CLK_MUX_BYPASS	See Register section for details	S32K389, S32K388

38.2 DCM_GPR register descriptions

Before you start to work with the GPR register take care about the following:

- Do not modify the reserved locations, registers, or reserved bits with respect to their default configurations. Chip behavior is not guaranteed in case of such writes.
- The writes to the DCM Read Write registers (DCMRWPx, DCMRWDx, and DCMRWFx) are synchronized and take up to 4 cycles of CORE_CLK, which means that the register configuration is effective at least 4 CORE_CLK after its write.
- The DCMROXn registers are sticky in nature. These registers latch the previous state and retain values across standby mode. Therefore, reading these registers might indicate a previously latched value. To read these sticky status registers after a reset event, a standby mode exit or any update in the signals which they latch, it is recommended to first clear the corresponding fields by writing 1 to the fields. This operation clears the previously latched status, and the fields get updated with the new status correctly.
- You must access DCM after at least 9 AIPS_SLOW_CLK cycles of writing to MC_RGM.ERCTRL because the configuration of MC_RGM.ERCTRL takes several cycles to be effective.

NOTE

The register section is based on the configuration of S32K389. For other chips, see Table 231 for more details.

38.2.1 DCM_GPR memory map

DCM_GPR base address: 402A_C000h

Offset	Register	Width (In bits)	Access	Reset value
200h	Read-Only GPR On Destructive Reset 1 (DCMROD1)	32	RW	0000_0000h
208h	Read-Only GPR On Destructive Reset 3 (DCMROD3)	32	RW	0000_0000h
20Ch	Read-Only GPR On Destructive Reset 4 (DCMROD4)	32	RW	0000_0000h
210h	Read-Only GPR On Destructive Reset 5 (DCMROD5)	32	RW	0000_0000h
214h	Read-Only GPR On Destructive Reset 6 (DCMROD6)	32	RW	0000_0000h
218h	Read-Only GPR On Destructive Reset 7 (DCMROD7)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
21Ch	Read-Only GPR On Destructive Reset Register (DCMROD8)	32	RW	0000_0000h
220h	Read-Only GPR On Destructive Reset 9 (DCMROD9)	32	RW	0000_0000h
300h	Read-Only GPR On Functional Reset 1 (DCMROF1)	32	RW	0000_0000h
304h	Read-Only GPR On Functional Reset 2 (DCMROF2)	32	RW	0000_0000h
308h	Read-Only GPR On Functional Reset 3 (DCMROF3)	32	RW	0000_0000h
30Ch	Read-Only GPR On Functional Reset 4 (DCMROF4)	32	RW	0000_0000h
310h	Read-Only GPR On Functional Reset 5 (DCMROF5)	32	RW	0000_0000h
314h	Read-Only GPR On Functional Reset 6 (DCMROF6)	32	RW	0000_0000h
318h	Read-Only GPR On Functional Reset 7 (DCMROF7)	32	RW	0000_0000h
31Ch	Read-Only GPR On Functional Reset 8 (DCMROF8)	32	RW	0000_0000h
320h	Read-Only GPR On Functional Reset 9 (DCMROF9)	32	RW	0000_0000h
324h	Read-Only GPR On Functional Reset 10 (DCMROF10)	32	RW	0000_0000h
328h	Read-Only GPR On Functional Reset 11 (DCMROF11)	32	RW	0000_0000h
32Ch	Read-Only GPR On Functional Reset 12 (DCMROF12)	32	RW	0000_0000h
330h	Read-Only GPR On Functional Reset 13 (DCMROF13)	32	RW	0000_0000h
334h	Read-Only GPR On Functional Reset 14 (DCMROF14)	32	RW	0000_0000h
338h	Read-Only GPR On Functional Reset 15 (DCMROF15)	32	RW	0000_0000h
33Ch	Read-Only GPR On Functional Reset 16 (DCMROF16)	32	RW	0000_0000h
340h	Read-Only GPR On Functional Reset 17 (DCMROF17)	32	RW	0000_0000h
348h	Read-Only GPR On Functional Reset 19 (DCMROF19)	32	R	4000_0000h
34Ch	Read-Only GPR On Functional Reset 20 (DCMROF20)	32	R	See section
350h	Read-Only GPR On Functional Reset 21 (DCMROF21)	32	R	0000_0000h
400h	Read Write GPR On POR 1 (DCMRWP1)	32	RW	0000_0400h
408h	Read Write GPR On POR 3 (DCMRWP3)	32	RW	0000_0000h
504h	Read Write GPR On Destructive Reset 2 (DCMRWD2)	32	RW	0000_0000h
508h	Read Write GPR On Destructive Reset 3 (DCMRWD3)	32	RW	FFFF_FBFFh
50Ch	Read Write GPR On Destructive Reset 4 (DCMRWD4)	32	RW	EFFF_FFFFh
510h	Read Write GPR On Destructive Reset 5 (DCMRWD5)	32	RW	FFFF_BFFFh
514h	Read Write GPR On Destructive Reset 6 (DCMRWD6)	32	RW	0000_0000h
518h	Read Write GPR On Destructive Reset 7 (DCMRWD7)	32	RW	0000_0000h
51Ch	Read Write GPR On Destructive Reset 8 (DCMRWD8)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
520h	Read Write GPR On Destructive Reset 9 (DCMRWD9)	32	RW	0000_0000h
52Ch	Read Write GPR On Destructive Reset 12 (DCMRWD12)	32	RW	0000_0000h
530h	Read Write GPR On Destructive Reset 13 (DCMRWD13)	32	RW	0000_0000h
534h	Read Write GPR On Destructive Reset 14 (DCMRWD14)	32	RW	4400_0008h
538h	Read Write GPR On Destructive Reset 15 (DCMRWD15)	32	RW	7000_183Fh
53Ch	Read Write GPR On Destructive Reset 16 (DCMRWD16)	32	RW	3FEF_7EF0h
540h	Read Write GPR On Destructive Reset 17 (DCMRWD17)	32	RW	07FF_FFFFh
548h	Read Write GPR On Destructive Reset 19 (DCMRWD19)	32	RW	0000_0000h
54Ch	Read Write GPR On Destructive Reset 20 (DCMRWD20)	32	RW	0000_0000h
600h	Read Write GPR On Functional Reset 1 (DCMRWF1)	32	RW	0000_0000h
604h	Read Write GPR On Functional Reset 2 (DCMRWF2)	32	RW	0000_0000h
608h	Read Write GPR On Functional Reset 3 (DCMRWF3)	32	RW	0000_0000h
60Ch	Read Write GPR On Functional Reset 4 (DCMRWF4)	32	RW	0000_0000h
610h	Read Write GPR On Functional Reset 5 (DCMRWF5)	32	RW	See section
700h	Read-Only GPR On PMCPOR Reset 1 (DCMROPP1)	32	RW	0000_0000h
704h	Read-Only GPR On PMCPOR Reset 2 (DCMROPP2)	32	RW	0000_0000h
708h	Read-Only GPR On PMCPOR Reset 3 (DCMROPP3)	32	RW	0000_0000h
70Ch	Read-Only GPR On PMCPOR Reset 4 (DCMROPP4)	32	RW	0000_0000h

38.2.2 Read-Only GPR On Destructive Reset 1 (DCMROD1)

Offset

Register	Offset
DCMROD1	200h

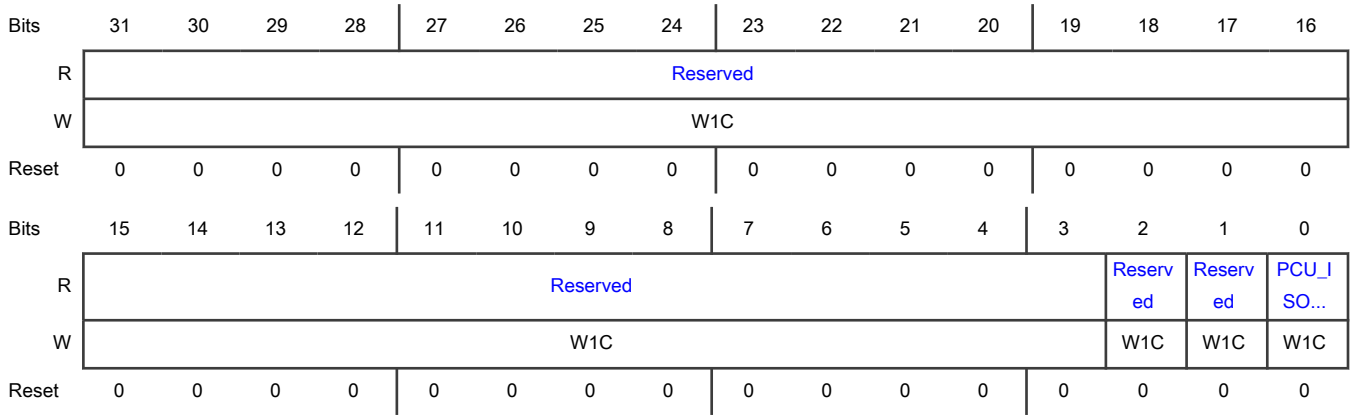
Function

Contains information related to:

- Key response ready status.
- DCF violation from HSE_B.
- PCU input isolation status.

This register resets after destructive reset 1.

Diagram



Fields

Field	Function
31-3 —	Reserved
2 —	Reserved
1 —	Reserved
0 PCU_ISO_STATUS	PCU Input Isolation Status On Previous Standby Entry Specifies whether input isolation was enabled in the previous standby entry. 0b - No 1b - Yes

38.2.3 Read-Only GPR On Destructive Reset 3 (DCMROD3)

Offset

Register	Offset
DCMROD3	208h

Function

Contains information related to:

- ECC and life cycle errors.
- AXBS, RCCU, and gasket alarms.

This register resets after destructive reset 3.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CM7_1_I...	CM7_0_I...	CM7_1_D...	CM7_0_D...	CM7_1_D...	CM7_0_D...	PRAM0_E...	PRAM1_E...	PRAM2_E...	LC_ERR	Reserved		PERIPH_...	MAC_GSK...	TCM_AXB...	DATA_ED...
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C		W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ADDR_ED...	AIPS2_G...	AIPS1_G...	QSPI_GS...	HSE_GSK...	Reserved	DMA_AXB...	SYS_AXB...	DMA_PER...	DMA_SYS...	TCM_GSK...	CM7_RCC...	CM7_RCC...	HSE_LOC...	CM7_1_L...	CM7_0_L...
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 CM7_1_ICDATA_ECC_ERR	Cortex-M7_1 I-cache Multi-Bit ECC Error Specifies whether the Cortex-M7_1 core's I-cache data memory detected a multi-bit ECC error. Read this field to identify the reason for a fault in case of FCCU noncritical fault (NCF) 2. 0b - No 1b - Yes
30 CM7_0_ICDATA_ECC_ERR	Cortex-M7_0 I-cache Data ECC Error Specifies whether the Cortex-M7_0 core's I-cache data memory detected a multi-bit ECC error. Read this field to identify the reason for a fault in case of FCCU NCF 2. 0b - No 1b - Yes
29 CM7_1_DCTAG_ECC_ERR	Cortex-M7_1 D-cache Tag ECC Error Specifies whether the Cortex-M7_1 core's D-cache tag memory detected a multi-bit ECC error. Read this field to identify the reason for a fault in case of FCCU NCF 2. 0b - No 1b - Yes
28 CM7_0_DCTAG_ECC_ERR	Cortex-M7_0 D-cache Tag ECC Error Specifies whether the Cortex-M7_0 core's D-cache tag memory detected a multi-bit ECC error. Read this field to identify the reason for a fault in case of FCCU NCF 2. 0b - No 1b - Yes
27	Cortex-M7_1 D-cache Data Memory ECC Error

Table continues on the next page...

Table continued from the previous page...

Field	Function
CM7_1_DCDATA_ECC_ERR	Specifies whether the Cortex-M7_1 core's D-cache data memory detected a multi-bit ECC error. Read this field to identify the reason for a fault in case of FCCU NCF 2. 0b - No 1b - Yes
26 CM7_0_DCDATA_ECC_ERR	Cortex-M7_0 D-cache Data Memory ECC Error Specifies whether the Cortex-M7_0 core's D-cache data memory detected a multi-bit ECC error. Read this field to identify the reason for a fault in case of FCCU NCF 2. 0b - No 1b - Yes
25 PRAM0_ECC_ERR	Multi-Bit ECC Error From PRAM0 Specifies whether a multi-bit ECC error occurred from PRAM0. Read this field to identify the reason for a fault in case of FCCU NCF 2. 0b - No 1b - Yes
24 PRAM1_ECC_ERR	Multi-Bit ECC Error From PRAM1 Specifies whether a multi-bit ECC error occurred from PRAM1. Read this field to identify the reason for a fault in case of FCCU NCF 2. 0b - No 1b - Yes
23 PRAM2_ECC_ERR	Multi bit ECC error from SRAM2. Read this bit to identify the reason for a fault in case of FCCU NCF 2. 0b - No multi-bit ECC error. 1b - Multi-bit ECC error.
22 LC_ERR	Error In Life Cycle Scanning Specifies whether an error occurred during life-cycle scanning. Read this bit to identify the reason of fault in case of FCCU NCF 3. 0b - No error while lifecycle scanning. 1b - Error while lifecycle scanning
21-20 —	Reserved
19 PERIPH_AXBS_ALARM	Peripheral AXBS_Lite Safety Alarm Status Specifies whether peripheral AXBS_Lite reported a safety alarm.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Read this field to identify the reason for a fault in case of FCCU NCF 1. 0b - No 1b - Yes
18 MAC_GSKT_ALARM	MAC IAHB Gasket Alarm Status Specifies whether the MAC IAHB gasket reported an alarm. Read this field to identify the reason for a fault in case of FCCU NCF 1. 0b - No 1b - Yes
17 TCM_AXBS_ALARM	TCM AHB Splitter Safety Alarm Status Specifies whether the TCM AHB splitter reported a safety alarm. Read this field to identify the reason for a fault in case of FCCU NCF 1. 0b - No 1b - Yes
16 DATA_EDC_ERR	Data EDC Error Specifies whether an integrity error occurred on address for safety. Read this field to identify the reason for a fault in case of FCCU NCF 1. 0b - No 1b - Yes
15 ADDR_EDC_ERR	Address EDC Error Status Specifies whether an integrity error occurred on address for safety. Read this field to identify the reason for a fault in case of FCCU NCF 1. 0b - No 1b - Yes
14 AIPS2_GSKT_ALARM	AIPS2 IAHB Gasket Alarm Status. Read this bit to identify the reason for a fault in case of FCCU NCF 1. 0b - No alarm indicated by AIPS2 IAHB gasket. 1b - Alarm indicated by AIPS2 IAHB gasket.
13 AIPS1_GSKT_ALARM	AIPS1 IAHB Gasket Alarm Status. Read this bit to identify the reason for a fault in case of FCCU NCF 1. 0b - No alarm indicated by AIPS1 IAHB gasket. 1b - Alarm indicated by AIPS1 IAHB gasket.
12 QSPI_GSKT_ALARM	QuadSPI IAHB Gasket Alarm Status Specifies whether the QuadSPI IAHB gasket reported an alarm. Read this field to identify the reason for a fault in case of FCCU NCF 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No 1b - Yes
11 HSE_GSKT_ALARM	HSE IAHB Gasket Alarm Status Specifies whether the HSE_B IAHB gasket reported an alarm. Read this field to identify the reason for a fault in case of FCCU NCF 1. 0b - No 1b - Yes
10 —	Reserved
9 DMA_AXBS_ALARM	eDMA AXBS_Lite Safety Alarm Status Specifies whether eDMA AXBS_Lite reported a safety alarm. Read this field to identify the reason for a fault in case of FCCU NCF 1. 0b - No 1b - Yes
8 SYS_AXBS_ALARM	System AXBS Safety Alarm Status Specifies whether the system AXBS indicated a safety alarm. Read this field to identify the reason for a fault in case of FCCU NCF 1. 0b - No 1b - Yes
7 DMA_PERIPH_GSKT_ALARM	eDMA Peripheral Gasket Alarm Status Specifies whether the eDMA peripheral AXBS IAHB gasket reported a safety alarm. Read this field to identify the reason for a fault in case of FCCU NCF 1. 0b - No 1b - Yes
6 DMA_SYS_GSKT_ALARM	eDMA System Gasket Alarm Status Specifies whether the IAHB gasket safety alarm, from the eDMA system AXBS IAHB gasket, reported a safety alarm. Read this field to identify the reason for a fault in case of FCCU NCF 1. 0b - No 1b - Yes
5	TCM IAHB Gasket Monitor Alarm Status

Table continues on the next page...

Table continued from the previous page...

Field	Function
TCM_GSKT_ALARM	Specifies whether the TCM IAHB gasket reported an alarm. If the value of this field is 1, the gasket reports a monitor alarm. Read this field to identify the reason for a fault in case of FCCU NCF 1. 0b - No 1b - Yes
4 CM7_RCCU2_ALARM	Cortex-M7 Core Redundant Lockstep Error Status Specifies whether RCCU reported a lockstep alarm. Read this field to identify the reason for a fault in case of FCCU NCF 0. 0b - No 1b - Yes
3 CM7_RCCU1_ALARM	Cortex-M7 Core Lockstep Error Status Specifies whether RCCU reported a lockstep alarm. Read this field to identify the reason for a fault in case of FCCU NCF 0. 0b - No 1b - Yes
2 HSE_LOCKUP	HSE_B Core Lockup Status Specifies whether the HSE_B core is in the Lockup state. Read this field to identify the reason for a fault in case of FCCU NCF 0. 0b - No 1b - Yes
1 CM7_1_LOCKUP	Cortex-M7_1 Core Lockup Status Specifies whether the Cortex_M7_1 core is in the Lockup state. Read this field to identify the reason for a fault in case of FCCU NCF 0. 0b - No 1b - Yes
0 CM7_0_LOCKUP	Cortex-M7_0 Core Lockup Status Specifies whether the Cortex-M7_0 core is in the Lockup state. Read this field to identify the reason for a fault in case of FCCU NCF 0. 0b - No 1b - Yes

38.2.4 Read-Only GPR On Destructive Reset 4 (DCMROD4)

Offset

Register	Offset
DCMROD4	20Ch

Function

Contains information related to:

- Accidental partial test activation errors.
- Go/No-go indicator supply statuses.
- Flash memory errors.
- Alarm statuses.

This register resets after destructive reset 4.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CM7_2_L...	TEST_AC...	TEST_AC...	Reserved	VDD2P5_...	VDD1P1_...	FLASH_E...	Reserved	PRAM2_F...	FLASH_S...	FLASH_R...	FLASH_R...	FLASH_A...	FLASH_E...	Reserved	Reserved
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PF1_DAT...	PF1_COD...	PF0_DAT...	PF0_COD...	HSE_RAM...	PRAM1_F...	PRAM0_F...	DMA_TCD...	CM7_1_D...	CM7_1_D...	CM7_1_I...	CM7_0_D...	CM7_0_D...	CM7_0_I...	CM7_1_I...	CM7_0_I...
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 CM7_2_LOCKUP	CM7_2 Core Lockup Status Read this bit to identify the reason of fault in case of FCCU NCF 0. 0b - CM7_2 core not in lockup state. 1b - CM7_2 core in lockup state.
30 TEST_ACTIVATION_1_ERR	Accidental Partial Test Activation 1 Error Specifies whether partial test 1 is activated accidentally. Read this field to identify the reason for a fault in case of FCCU NCF 5. 0b - No

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Yes
29 TEST_ACTIVATION_0_ERR	Accidental Partial Test Activation 0 Error Specifies whether partial test 0 is activated accidentally. Read this field to identify the reason for a fault in case of FCCU NCF 5. 0b - No 1b - Yes
28 —	Reserved
27 VDD2P5_GNG_ERR	Go/No-go Indicator For VDD_HV_FLA Specifies whether the VDD_HV_FLA (double-bond) supply going to XOSC and PLL is clean. Read this field to identify the reason for a fault in case of FCCU NCF 4. If this field = 1, the "go" indication specifies a clean supply, and if this field = 0, the "no-go" indication specifies an unclean supply with a fault in the double-bond connection or its routing within the chip. 0b - Yes 1b - No
26 VDD1P1_GNG_ERR	Go/No-go Indicator For VDD1PD1 Specifies whether the VDD1PD1 (double-bond) supply going to PLL is clean. Read this field to identify the reason for a fault in case of FCCU NCF 4. If this field = 1, the "go" indication specifies a clean supply, and if this field = 0, the "no-go" indication specifies an unclean supply with a fault in the double-bond connection or its routing within the chip. 0b - Yes 1b - No
25 FLASH_ECC_ERR	ECC Error From Flash Controller This alarm Specifies that the flash controller detected an error in the address ECC manipulation logic through EDC. Read this bit to identify the reason for a fault in case of FCCU NCF 3. 0b - No ECC error from flash controller. 1b - ECC error from flash controller.
24 —	Reserved
23 PRAM2_FCCU_ALARM	Status of PRAM2 safety alarm. This alarm is set on faulty SRAM2 read or read-modify error. Read this bit to identify the reason for a fault in case of FCCU NCF 2. 0b - No safety alarm indicated by PRAM2. 1b - Safety alarm indicated by PRAM2.

Table continues on the next page...

Table continued from the previous page...

Field	Function
22 FLASH_SCAN_ERR	Flash Memory Scan Error Status Specifies whether the flash memory encountered an error during the DCM flash scanning process because of invalid data. Read this field to identify the reason for a fault in case of FCCU NCF 3. 0b - No 1b - Yes
21 FLASH_RST_ERR	Flash Reset Error Status Specifies whether the flash memory encountered a flash memory reset error during its reset reads. Read this field to identify the reason for a fault in case of FCCU NCF 3. 0b - No 1b - Yes
20 FLASH_REF_ERR	Flash Memory Reference Error Status Specifies whether the flash memory encountered a reference current loss or read voltage error during previous read(s). Read this field to identify the reason for a fault in case of FCCU NCF 3. 0b - No 1b - Yes
19 FLASH_ADDR_ENC_ERR	Flash Memory Address Encode Error Status Specifies whether FMU reported an address encode error in the flash memory. During address decoding, if multiple or no address line is selected, FMU reports an address encode error. Read this field to identify the reason for a fault in case of FCCU NCF 3. 0b - No 1b - Yes
18 FLASH_EDC_ERR	Flash Memory EDC Error Status Specifies whether FMU reported an integrity error after an ECC correction error in the flash memory. Read this field to identify the reason for a fault in case of FCCU NCF 3. 0b - No 1b - Yes
17 —	Reserved
16 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
15 PF1_DATA_ECC_ERR	Program Flash Memory 1 Data ECC Error Status Specifies whether FMU reported uncorrectable errors in the flash memory controller port 1 data memory. These errors are connected to FCCU NCFs and to ERM. See the "Error Reporting Module (ERM)" chapter for memory errors and mapping onto ERM channels. Read this field to identify the reason for a fault in case of FCCU NCF 3. 0b - No 1b - Yes
14 PF1_CODE_ECC_ERR	Program Flash Memory 1 Code ECC Error Status Specifies whether FMU reported uncorrectable errors in the flash memory controller port 1 code memory. These errors are connected to FCCU NCFs and to ERM. See the "Error Reporting Module (ERM)" chapter for memory errors and mapping onto ERM channels. Read this field to identify the reason for a fault in case of FCCU NCF 3. 0b - No 1b - Yes
13 PF0_DATA_ECC_ERR	Program Flash Memory 0 Data ECC Error Status Specifies whether FMU reported uncorrectable errors in the flash memory controller port 0 data memory. These errors are connected to FCCU NCFs and to ERM. See the "Error Reporting Module (ERM)" chapter for memory errors and mapping onto ERM channels. Read this field to identify the reason for a fault in case of FCCU NCF 3. 0b - No 1b - Yes
12 PF0_CODE_ECC_ERR	Program Flash Memory 0 Code ECC Error Status Specifies whether FMU reported uncorrectable errors in the flash memory controller port 0 code memory. These errors are connected to FCCU NCFs and to ERM. See the "Error Reporting Module (ERM)" chapter for memory errors and mapping onto ERM channels. Read this field to identify the reason for a fault in case of FCCU NCF 3. 0b - No 1b - Yes
11 HSE_RAM_ECC_ERR	HSE_B RAM Uncorrectable ECC Status Specifies whether HSE_B RAM reported an uncorrectable ECC error. Read this field to identify the reason for a fault in case of FCCU NCF 2. 0b - No 1b - Yes

Table continues on the next page...

Table continued from the previous page...

Field	Function
10 PRAM1_FCCU_ALARM	PRAM1 FCCU Alarm Status Specifies whether PRAM1 reported a safety alarm. This field specifies the status of the PRAM1 safety alarm, whether the alarm is set on faulty PRAM1 read or read-modify error. Read this field to identify the reason for a fault in case of FCCU NCF 2. 0b - No 1b - Yes
9 PRAM0_FCCU_ALARM	PRAM0 FCCU Alarm Status Specifies whether PRAM0 reported a safety alarm. This field specifies the status of the PRAM0 safety alarm, whether the alarm is set on faulty PRAM0 read or read-modify error. Read this field to identify the reason for a fault in case of FCCU NCF 2. 0b - No 1b - Yes
8 DMA_TCD_RAM_ECC_ERR	eDMA TCD RAM ECC Error Specifies whether the eDMA_TCD memory detected an uncorrectable ECC error. This uncorrectable ECC error consists of a multi-bit data ECC error and an address ECC error. Read this field to identify the reason for a fault in case of FCCU NCF 2. 0b - No 1b - Yes
7 CM7_1_DTCM1_ECC_ERR	Cortex-M7_1 DTCM 1 ECC Error Specifies whether the Cortex-M7_1 core's DTCM block 1 detected an uncorrectable ECC error. This uncorrectable ECC error consists of a multi-bit data ECC error and an address ECC error. The Cortex-M7_1 core's DTCM consists of two physical blocks. Read this field to identify the reason for a fault in case of FCCU NCF 2. **NOTE** The Cortex-M7_1 core's DTCM 1 does not support address ECC errors in S32K324, S32K344, and S32K314. 0b - No 1b - Yes
6 CM7_1_DTCM0_ECC_ERR	Cortex-M7_1 DTCM 0 ECC Error Specifies whether the Cortex-M7_1 core's DTCM block 0 detected an uncorrectable ECC error. This uncorrectable ECC error consists of a multi-bit data ECC error and an address ECC error. The Cortex-M7_1 core's DTCM consists of two physical blocks. Read this field to identify the reason for a fault in case of FCCU NCF 2.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE The Cortex-M7_1 core's DTCM 0 does not support address ECC errors in S32K324, S32K344, and S32K314. 0b - No 1b - Yes
5 CM7_1_ITCM_ECC_ERR	Cortex-M7_1 ITCM ECC Error Specifies whether the Cortex-M7_1 core's ITCM detected an uncorrectable ECC error. This uncorrectable ECC error consists of a multi-bit data ECC error and an address ECC error. Read this field to identify the reason for a fault in case of FCCU NCF 2. **NOTE** The Cortex-M7_1 core's ITCM does not support address ECC errors in S32K324, S32K344, and S32K314. 0b - No 1b - Yes
4 CM7_0_DTCM1_ECC_ERR	Cortex-M7_0 DTCM 1 ECC Error Specifies whether the Cortex-M7_0 core's DTCM block 1 detected an uncorrectable ECC error. This uncorrectable ECC error consists of a multi-bit data ECC error and an address ECC error. The Cortex-M7_0 core's DTCM consists of two physical blocks. Read this field to identify the reason for a fault in case of FCCU NCF 2. **NOTE** The Cortex-M7_0 core's DTCM 1 does not support address ECC errors in S32K324, S32K344, and S32K314. 0b - No 1b - Yes
3 CM7_0_DTCM0_ECC_ERR	Cortex-M7_0 DTCM 0 ECC Error Specifies whether the Cortex-M7_0 core's DTCM block 0 detected an uncorrectable ECC error. This uncorrectable ECC error consists of a multi-bit data ECC error and an address ECC error. The Cortex-M7_0 core's DTCM consists of two physical blocks. Read this field to identify the reason for a fault in case of FCCU NCF 2. **NOTE** The Cortex-M7_0 core's DTCM 0 does not support address ECC errors in S32K324, S32K344, and S32K314. 0b - No 1b - Yes

Table continues on the next page...

Table continued from the previous page...

Field	Function
2 CM7_0_ITCM_ECC_ERR	Cortex-M7_0 ITCM ECC Error Specifies whether the Cortex-M7_0 core's ITCM detected an uncorrectable ECC error. This uncorrectable ECC error consists of a multi-bit data ECC error and an address ECC error. Read this field to identify the reason of fault in case of FCCU NCF 2. **NOTE** The Cortex-M7_0 core's ITCM does not support address ECC errors in S32K324, S32K344, and S32K314. 0b - No 1b - Yes
1 CM7_1_ICTAG_ECC_ERR	Cortex-M7_1 I-cache Tag ECC Error Specifies whether the Cortex-M7_1 core's I-cache tag memory detected a multi-bit ECC error. Read this field to identify the reason for a fault in case of FCCU NCF 2. 0b - No 1b - Yes
0 CM7_0_ICTAG_ECC_ERR	Cortex-M7_0 I-cache Tag ECC Error Specifies whether the Cortex-M7_0 core's I-cache tag memory detected a multi-bit ECC error. Read this field to identify the reason for a fault in case of FCCU NCF 2. 0b - No 1b - Yes

38.2.5 Read-Only GPR On Destructive Reset 5 (DCMROD5)

Offset

Register	Offset
DCMROD5	210h

Function

Contains information related to:

- ECC and EDC errors.
- Activation and bus errors.

This register resets after destructive reset 5.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CM7_2_D...	CM7_2_D...	CM7_2_I...	CM7_2_I...	CM7_2_I...	CM7_2_D...	CM7_2_D...	CM7_2_A...	CM7_2_A...	HSE_RDA...	CM7_0_A...	CM7_0_A...	CM7_1_A...	CM7_1_A...	DMA_RDA...	Reserved
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MAC_RDA...	Reserved	DEBUG_A...	MCT_BUS...	STCU_BI...	MBIST_A...	STCU_NCF	SW_NCF_3	SW_NCF_2	SW_NCF_1	SW_NCF_0	INTM_3_...	INTM_2_...	INTM_1_...	INTM_0_...	Reserved
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 CM7_2_DTCM1_ECC_ERR	Cortex-M7_2 DTCM 1 ECC Error Specifies whether the Cortex-M7_2 core's DTCM block 1 detected an uncorrectable ECC error. This uncorrectable ECC error consists of a multi-bit data ECC error and an address ECC error. The Cortex-M7_2 core's DTCM consists of two physical blocks. Read this field to identify the reason for a fault in case of FCCU NCF 2. **NOTE** The Cortex-M7_2 core's DTCM 1 does not support address ECC errors in S32K324, S32K344, and S32K314. 0b - No 1b - Yes
30 CM7_2_DTCM0_ECC_ERR	Cortex-M7_2 DTCM 0 ECC Error Specifies whether the Cortex-M7_2 core's DTCM block 0 detected an uncorrectable ECC error. This uncorrectable ECC error consists of a multi-bit data ECC error and an address ECC error. The Cortex-M7_2 core's DTCM consists of two physical blocks. Read this field to identify the reason for a fault in case of FCCU NCF 2. **NOTE** The Cortex-M7_2 core's DTCM 0 does not support address ECC errors in S32K324, S32K344, and S32K314. 0b - No 1b - Yes
29 CM7_2_ITCM_ECC_ERR	Cortex-M7_2 ITCM ECC Error Specifies whether the Cortex-M7_2 core's ITCM detected an uncorrectable ECC error. This uncorrectable ECC error consists of a multi-bit data ECC error and an address ECC error. Read this field to identify the reason for a fault in case of FCCU NCF 2.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE The Cortex-M7_2 core's ITCM does not support address ECC errors in S32K324, S32K344, and S32K314. 0b - No 1b - Yes
28 CM7_2_ICTAG_ECC_ERR	Cortex-M7_2 I-cache Tag ECC Error Specifies whether the Cortex-M7_2 core's I-cache tag memory reported a multi-bit ECC error. Read this field to identify the reason for a fault in case of FCCU NCF 2. 0b - No 1b - Yes
27 CM7_2_ICDATA_ECC_ERR	Cortex-M7_2 I-cache Data ECC Error Specifies whether the Cortex-M7_2 core's I-cache data memory reported a multi-bit ECC error. Read this field to identify the reason for a fault in case of FCCU NCF 2. 0b - No 1b - Yes
26 CM7_2_DCTAG_ECC_ERR	Cortex-M7_2 D-cache Tag ECC Error Specifies whether the Cortex-M7_2 core's D-cache tag memory detected a multi-bit ECC error. Read this field to identify the reason for a fault in case of FCCU NCF 2. 0b - No 1b - Yes
25 CM7_2_DCDATA_ECC_ERR	Cortex-M7_2 D-cache Data ECC Error Specifies whether the Cortex-M7_2 core's D-cache data memory detected a multi-bit ECC error. Read this field to identify the reason for a fault in case of FCCU NCF 2. 0b - No 1b - Yes
24 CM7_2_AHBM_RDATA_EDC_ERR	Cortex-M7_2 AHBM Read Data EDC Error Specifies whether an integrity error is reported on the Cortex-M7_2 core's main read data for safety. Read this field to identify the reason for a fault in case of FCCU NCF 1. 0b - No 1b - Yes
23	Cortex-M7_2 AHBP Read Data EDC Error Specifies whether an integrity error is reported on the Cortex-M7_2 core's peripheral read data for safety.

Table continues on the next page...

Table continued from the previous page...

Field	Function
CM7_2_AHBP_RDATA_EDC_ERR	Read this field to identify the reason for a fault in case of FCCU NCF 1. 0b - No 1b - Yes
22 HSE_RDATA_EDC_ERR	HSE_B Read Data EDC Error Specifies whether an integrity error is reported on the HSE_B read data for safety. Read this field to identify the reason for a fault in case of FCCU NCF 1. 0b - No 1b - Yes
21 CM7_0_AHBM_RDATA_EDC_ERR	Cortex-M7_0 AHBM Read Data EDC Error Specifies whether an integrity error is reported on the Cortex-M7_0 core's main read data for safety. Read this field to identify the reason for a fault in case of FCCU NCF 1. 0b - No 1b - Yes
20 CM7_0_AHBP_RDATA_EDC_ERR	Cortex-M7_0 AHBP Read Data EDC Error Specifies whether an integrity error is reported on the Cortex-M7_0 core's peripheral read data for safety. Read this field to identify the reason for a fault in case of FCCU NCF 1. 0b - No 1b - Yes
19 CM7_1_AHBM_RDATA_EDC_ERR	Cortex-M7_1 AHBM Read Data EDC Error Specifies whether an integrity error is reported on the Cortex-M7_1 core's main read data for safety. Read this field to identify the reason for a fault in case of FCCU NCF 1. 0b - No 1b - Yes
18 CM7_1_AHBP_RDATA_EDC_ERR	Cortex-M7_1 AHBP Read Data EDC Error Specifies whether an integrity error is reported on the Cortex-M7_1 core's peripheral read data for safety. Read this field to identify the reason for a fault in case of FCCU NCF 1. 0b - No 1b - Yes
17 DMA_RDATA_EDC_ERR	eDMA Read Data EDC Error Specifies whether an integrity error is reported on the eDMA read data for safety. Read this field to identify the reason for a fault in case of FCCU NCF 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No 1b - Yes
16 —	Reserved
15 MAC_RDATA_EDC_ERR	MAC Read Data EDC Error Specifies whether an integrity error is reported on the MAC read data for safety. Read this field to identify the reason for a fault in case of FCCU NCF 1. 0b - No 1b - Yes
14 —	Reserved
13 DEBUG_ACTIVATION_ERR	Debug Activation Error Specifies whether an unintended debug is activated. This field monitors unintended debug activation. It displays 1 as its value when the core is in halted state with the application debug or debugger request not enabled. Read this field to identify the reason for a fault in case of FCCU NCF 5. 0b - No 1b - Yes
12 MCT_BUS_ERR	MCT Bus Error Fault reported due to illegal access on MBIST Master Controller (MCT). This fault is reported via a transfer error indication to the system. Read this bit to identify the reason of fault in case of FCCU NCF 5. 0b - No transfer error indicated from MCT. 1b - Transfer error indicated from MCT.
11 STCU_BIST_USER_CF	STCU2 BIST User Critical Fault (CF) Specifies whether LBIST or MBIST is enabled accidentally when the fault condition is detected in Run mode. Read this field to identify the reason for a fault in case of FCCU NCF 5. 0b - No 1b - Yes
10 MBIST_ACTIVATION_ERR	MBIST Activation Error Specifies whether an accidental backdoor access is enabled on memories. DCMRWD5[MBIST_ACTIVATION_ERR_EN] needs to be disabled on FCCU when performing a fault injection.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Read this field to identify the reason for a fault in case of FCCU NCF 5. 0b - No 1b - Yes
9 STCU_NCF	STCU2 NCF Result Error Specifies whether STCU2 NCF, which is a BIST result error, is reported. Read this field to identify the reason for a fault in case of FCCU NCF 5. 0b - No 1b - Yes
8 SW_NCF_3	Software NCF3 Status Specifies whether DCMRWF1[FCCU_SW_NCF3] is enabled. Read this field to identify the reason for a fault in case of FCCU NCF 7. 0b - No 1b - Yes
7 SW_NCF_2	Software NCF2 Status Specifies whether DCMRWF1[FCCU_SW_NCF2] is enabled. Read this field to identify the reason for a fault in case of FCCU NCF 7. 0b - No 1b - Yes
6 SW_NCF_1	Software NCF1 Status Specifies whether DCMRWF1[FCCU_SW_NCF1] is enabled. Read this field to identify the reason for a fault in case of FCCU NCF 7. 0b - No 1b - Yes
5 SW_NCF_0	Software NCF 0 Status Specifies whether DCMRWF1[FCCU_SW_NCF0] is enabled. Read this field to identify the reason for a fault in case of FCCU NCF 7. 0b - No 1b - Yes
4 INTM_3_ERR	INTM_3 Error Specifies whether INTM_3 reported an error. The reported error is recorded in INTM.INTM_STATUS3. See the "Functional description" section of the "Interrupt Monitor (INTM)" chapter for details.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Read this field to identify the reason for a fault in case of FCCU NCF 6. 0b - No 1b - Yes
3 INTM_2_ERR	INTM_2 Error Specifies whether INTM_2 reported an error. The reported error is recorded in INTM.INTM_STATUS2. See the "Functional description" section of the "Interrupt Monitor (INTM)" chapter for details. Read this field to identify the reason for a fault in case of FCCU NCF 6. 0b - No 1b - Yes
2 INTM_1_ERR	INTM_1 Error Specifies whether INTM_1 reported an error. The reported error is recorded in INTM.INTM_STATUS1. See the "Functional description" section of the "Interrupt Monitor (INTM)" chapter for details. Read this field to identify the reason for a fault in case of FCCU NCF 6. 0b - No 1b - Yes
1 INTM_0_ERR	INTM_0 Error Specifies whether INTM_0 reported an error. The reported error is recorded in INTM.INTM_STATUS0. See the "Functional description" section of the "Interrupt Monitor (INTM)" chapter for details. Read this field to identify the reason for a fault in case of FCCU NCF 6. 0b - No 1b - Yes
0 —	Reserved

38.2.6 Read-Only GPR On Destructive Reset 6 (DCMROD6)

Offset

Register	Offset
DCMROD6	214h

Function

Contains information related to:

- Safety, RCCU, and AXI alarm statuses.
- Core memory errors.
- CF and NCF errors.
- ECC and EDC errors.

This register resets after destructive reset 6.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved	TCM_PRA...	Reserved	Reserved	Reserved	AIPS0_G...	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30 TCM_PRAM_AXBS_ALARM	Status of TCM_PRAM AXBS_Lite safety alarm. Read this bit to identify the reason of fault in case of FCCU NCF 1. 0b - No safety alarm indicated by TCM_PRAM AXBS_Lite. 1b - Safety alarm indicated by TCM_PRAM AXBS_Lite.
29 —	Reserved
28 —	Reserved
27 —	Reserved
26	AIPS0 IAHB Gasket Alarm Status. Read this bit to identify the reason of fault in case of FCCU NCF 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
AIPS0_GSKT_ALARM	0b - No alarm indicated by AIPS0 IAHB gasket. 1b - Alarm indicated by AIPS0 IAHB gasket.
25 —	Reserved
24 —	Reserved
23 —	Reserved
22 —	Reserved
21 —	Reserved
20 —	Reserved
19 —	Reserved
18 —	Reserved
17 —	Reserved
16 —	Reserved
15 —	Reserved
14 —	Reserved
13 —	Reserved
12	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
11 —	Reserved
10 —	Reserved
9 —	Reserved
8 —	Reserved
7 —	Reserved
6 —	Reserved
5 —	Reserved
4 —	Reserved
3 —	Reserved
2 —	Reserved
1 —	Reserved
0 —	Reserved

38.2.7 Read-Only GPR On Destructive Reset 7 (DCMROD7)

Offset

Register	Offset
DCMROD7	218h

Function

Contains information related to:

- LVDS pad receiver fault statuses.
- AXBS, AHBP, and AHBM alarm statuses.

This field resets after destructive reset 7.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved	CM7_2_A...	CM7_1_A...	CM7_0_A...	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	Reserved	Reserved	VDD2P5_...	VDD1P1_...	Reserved	Reserved	Reserved	Reserved	Reserved	CM7_2_A...	CM7_1_A...	CM7_0_A...	CM7_2_A...	CM7_1_A...	CM7_0_A...
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30 CM7_2_AHBS_ALARM	CM7_2 AHBS interface IAHB Gasket monitor alarm status. Read this bit to identify the reason of fault in case of FCCU NCF 1. 0b - No alarm reported from CM7_0 AHBS interface IAHB gasket. 1b - Monitor alarm reported from CM7_0 AHBS interface IAHB gasket.
29 CM7_1_AHBS_ALARM	CM7_1 AHBS interface IAHB Gasket monitor alarm status. Read this bit to identify the reason of fault in case of FCCU NCF 1. 0b - No alarm reported from CM7_0 AHBS interface IAHB gasket. 1b - Monitor alarm reported from CM7_0 AHBS interface IAHB gasket.

Table continues on the next page...

Table continued from the previous page...

Field	Function
28 CM7_0_AHBS_ALARM	CM7_0 AHBS interface IAHB Gasket monitor alarm status. Read this bit to identify the reason of fault in case of FCCU NCF 1. 0b - No alarm reported from CM7_0 AHBS interface IAHB gasket. 1b - Monitor alarm reported from CM7_0 AHBS interface IAHB gasket.
27 —	Reserved
26 —	Reserved
25 —	Reserved
24 —	Reserved
23 —	Reserved
22 —	Reserved
21 —	Reserved
20 —	Reserved
19 —	Reserved
18 —	Reserved
17 —	Reserved
16 —	Reserved
15 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
14 —	Reserved
13 —	Reserved
12 VDD2P5_GNG2_ERR	Go/Nogo indicator status for VDD_HV_FLA (triple bond) going to FXOSC and PLL. Read this bit to identify the reason of fault in case of FCCU NCF4. 0b - Go indication referring to the supply being clean. 1b - No go indication referring to the supply being unclean and a fault in double bond connection or its routing within the chip.
11 VDD1P1_GNG2_ERR	Go/Nogo indicator status for VDD_HV_FLA (triple bond) going to FXOSC and PLL. Read this bit to identify the reason of fault in case of FCCU NCF4. 0b - Go indication referring to the supply being clean. 1b - No go indication referring to the supply being unclean and a fault in double bond connection or its routing within the chip.
10 —	Reserved
9 —	Reserved
8 —	Reserved
7 —	Reserved
6 —	Reserved
5 CM7_2_AHBP_ALARM	Cortex-M7_2 AHBP Alarm Status Specifies the Cortex-M7_2 AHBP interface IAHB gasket monitor alarm status, showing whether or not the gasket reported a monitor alarm. Read this field to identify the reason for a fault in case of FCCU NCF 1. 0b - No 1b - Yes
4	Cortex-M7_1 AHBP Alarm Status

Table continues on the next page...

Table continued from the previous page...

Field	Function
CM7_1_AHBP_ALARM	Specifies the Cortex-M7_1 AHBP interface IAHB gasket monitor alarm status, showing whether or not the gasket reported a monitor alarm. Read this field to identify the reason for a fault in case of FCCU NCF 1. 0b - No 1b - Yes
3 CM7_0_AHBP_ALARM	Cortex-M7_0 AHBP Alarm Status Specifies the Cortex-M7_0 AHBP interface IAHB gasket monitor alarm status, showing whether or not the gasket reported a monitor alarm. Read this field to identify the reason for a fault in case of FCCU NCF 1. 0b - No 1b - Yes
2 CM7_2_AHBM_ALARM	Cortex-M7_2 AHBM Alarm Status Specifies the Cortex-M7_2 AHBM interface IAHB gasket monitor alarm status, showing whether or not the gasket reported a monitor alarm. Read this field to identify the reason for a fault in case of FCCU NCF 1. 0b - No 1b - Yes
1 CM7_1_AHBM_ALARM	Cortex-M7_1 AHBM Alarm Status Specifies the Cortex-M7_1 AHBM interface IAHB gasket monitor alarm status, showing whether or not the gasket reported a monitor alarm. Read this field to identify the reason for a fault in case of FCCU NCF 1. 0b - No 1b - Yes
0 CM7_0_AHBM_ALARM	Cortex-M7_0 AHBM Alarm Status Specifies the Cortex-M7_0 AHBM interface IAHB gasket monitor alarm status, showing whether or not the gasket reported a monitor alarm. Read this field to identify the reason for a fault in case of FCCU NCF 1. 0b - No 1b - Yes

38.2.8 Read-Only GPR On Destructive Reset Register (DCMROD8)

Offset

Register	Offset
DCMROD8	21Ch

Function

This is a read only general purpose register which captures the states and gets reset on destructive reset. Writing 1 to a bit in this register, clears the bit.

NOTE

If the bit signal gets enabled again or is always enabled, the bit gets configured even after writing 1 to clear the bit.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved	Reserved	CM7_3_D...	CM7_3_D...	CM7_3_I...	CM7_3_I...	CM7_3_I...	CM7_3_D...	CM7_3_D...	CM7_3_A...	CM7_3_A...	Reserved	AES_ACC...	AES_ACC...	ACE_FEE...	ACE_RES...
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	CM7_3_A...	HSE_AES...	CM7_3_A...	CM7_3_A...	MAC2_RD...	MAC2_GS...	Reserved	PERIPH_...	CM7_2_R...	CM7_2_R...	CM7_3_L...	Reserved			
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30 —	Reserved
29 CM7_3_DTCM1_ECC_ERR	Status of Uncorrectable ECC error from CM7_3 Data TCM memory block 1. This uncorrectable ECC error consists of multi-bit data ECC error and address ECC error. The CM7_3 Data TCM physically consists of two blocks. Read this bit to identify the reason of fault in case of FCCU NCF 2. 0b - Uncorrectable ECC error detection not enabled at FCCU 1b - Uncorrectable ECC error detection enabled at FCCU
28	Status of Uncorrectable ECC error from CM7_3 Data TCM memory block 0. This uncorrectable ECC error consists of multi-bit data ECC error and address ECC error. The CM7_3 Data TCM physically consists of two blocks. Read this bit to identify the reason of fault in case of FCCU NCF 2.

Table continues on the next page...

Table continued from the previous page...

Field	Function
CM7_3_DTCM0_ECC_ERR	0b - Uncorrectable ECC error detection not enabled at FCCU 1b - Uncorrectable ECC error detection enabled at FCCU
27 CM7_3_ITCM_ECC_ERR	Status of Uncorrectable ECC error from CM7_3 Instruction TCM memory. This uncorrectable ECC error consists of multi-bit data ECC error and address ECC error. Read this bit to identify the reason of fault in case of FCCU NCF 2. 0b - No uncorrectable ECC error detected 1b - Uncorrectable ECC error detected
26 CM7_3_ICTAG_ECC_ERR	Status of Multi bit ECC error from CM7_3 ICache tag memory. Read this bit to identify the reason of fault in case of FCCU NCF 2. 0b - No multi-bit ECC error reported 1b - Multi-bit ECC error reported
25 CM7_3_ICDATA_ECC_ERR	Status of Multi bit ECC error from CM7_3 ICache data memory. Read this bit to identify the reason of fault in case of FCCU NCF 2. 0b - No multi-bit ECC error reported 1b - Multi-bit ECC error reported
24 CM7_3_DCTAG_ECC_ERR	Status of Multi bit ECC error from CM7_3 DCache tag memory. Read this bit to identify the reason of fault in case of FCCU NCF 2. 0b - No multi-bit ECC error reported 1b - Multi-bit ECC error reported
23 CM7_3_DCDATA_ECC_ERR	Status of Multi bit ECC error from CM7_3 DCache data memory. Read this bit to identify the reason of fault in case of FCCU NCF 2. 0b - No multi-bit ECC error reported 1b - Multi-bit ECC error reported
22 CM7_3_AHBP_ALARM	Status of CM7_3 AHBP interface IAHB Gasket monitor alarm. Read this bit to identify the reason of fault in case of FCCU NCF 1. 0b - No alarm reported 1b - Monitor alarm reported
21 CM7_3_AHBM_ALARM	Status of CM7_3 AHBM interface IAHB Gasket monitor alarm. Read this bit to identify the reason of fault in case of FCCU NCF 1. 0b - No alarm reported 1b - Monitor alarm reported
20 —	Reserved
19	AES ACCEL IAHB Gasket monitor alarm status. Read this bit to identify the reason of fault in case of FCCU NCF 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
AES_ACCEL_GSKT_ALARM	0b - No alarm reported 1b - Monitor alarm reported
18 AES_ACCEL_AXBS_ALARM	AES_ACCEL AXBS_Lite safety alarm status. Read this bit to identify the reason of fault in case of FCCU NCF 1. 0b - No safety alarm indicated 1b - Safety alarm indicated
17 ACE_FEED_RDATA_EDC_ERR	Status of Integrity error on ACE ACCEL FEED DMA master port read data for safety. Read this bit to identify the reason of fault in case of FCCU NCF 1. 0b - No integrity error reported 1b - Integrity error reported
16 ACE_RESULT_RDATA_EDC_ERR	Status of Integrity error on ACE ACCEL RESULT DMA master port read data for safety. Read this bit to identify the reason of fault in case of FCCU NCF 1. 0b - No integrity error reported 1b - Integrity error reported
15 —	Reserved
14 CM7_3_AHBS_ALARM	CM7_3 AHBS interface IAHB Gasket monitor alarm status. Read this bit to identify the reason of fault in case of FCCU NCF 1. 0b - No alarm reported 1b - Monitor alarm reported
13 HSE_AES_ACCEL_AXBS_ALARM	HSE_AES_ACCEL AXBS_Lite safety alarm status. Read this bit to identify the reason of fault in case of FCCU NCF 1. 0b - No safety alarm indicated 1b - Safety alarm indicated
12 CM7_3_AHBM_RDATA_EDC_ERR	Status of Integrity error on CM7_3 main read data for safety. Read this bit to identify the reason of fault in case of FCCU NCF 1. 0b - No integrity error reported 1b - Integrity error reported
11 CM7_3_AHBP_RDATA_EDC_ERR	Status of Integrity error on CM7_3 peripheral read data for safety. Read this bit to identify the reason of fault in case of FCCU NCF 1. 0b - No integrity error reported 1b - Integrity error reported
10	Status of Integrity(EDC) error on MAC2 read data for safety. Read this bit to identify the reason of fault in case of FCCU NCF 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
MAC2_RDATA_EDC_ERR	0b - No integrity error reported 1b - Integrity error reported
9 MAC2_GSKT_ALARM	MAC2 IAHB gasket alarm status. Read this bit to identify the reason of fault in case of FCCU NCF 1. 0b - No alarm indicated 1b - Alarm indicated
8 —	Reserved
7 PERIPH_AXBS_S3_GSKT_ALARM	Peripheral AXBS bridge S3 IAHB gasket alarm status. Read this bit to identify the reason of fault in case of FCCU NCF 1. 0b - No alarm reported 1b - Monitor alarm reported
6 CM7_2_RCCU2_ALARM	Cortex M7 cores (CM7_2 and CM7_2_checker core) redundant lockstep error status. Read this bit to identify the reason of fault in case of FCCU NCF 0. 0b - No Error reported. 1b - Error reported
5 CM7_2_RCCU1_ALARM	Cortex M7 cores (CM7_2 and CM7_2_checker core) lockstep error status. Read this bit to identify the reason of fault in case of FCCU NCF 0. 0b - No Error reported. 1b - Error reported
4 CM7_3_LOCKUP	CM7_3 core lockup status. Read this bit to identify the reason of fault in case of FCCU NCF 0. 0b - Not in lockup state 1b - In lockup state
3-0 —	Reserved

38.2.9 Read-Only GPR On Destructive Reset 9 (DCMROD9)

Offset

Register	Offset
DCMROD9	220h

Function

This is a read only general purpose register which captures the states and gets reset on destructive reset. Writing 1 to a bit in this register, clears the bit.

NOTE

If the bit signal gets enabled again or is always enabled, the bit gets configured even after writing 1 to clear the bit.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved					PF1_1_C...	PF1_0_C...	PF0_1_C...	PF0_0_C...	PF1_1_C...	PF1_0_C...	PF0_1_C...	PF0_0_C...	PRAM3_F...	PRAM3_E...	FLASH1_...
W	W1C					W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	FLASH1_...	FLASH1_...	FLASH1_...	FLASH1_...	PF1_1_D...	PF1_1_C...	PF1_0_D...	PF1_0_C...	AES_RES...	AES_FEE...	AES_KP_...	AES_RES...	AES_RES...	AES_FEE...	AES_FEE...
W		W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-27 —	Reserved
26 PF1_1_CHK_CMP_ALARM	Status of PFLASH3 checker (redundant) safety comparator alarm. Read this bit to identify the reason of fault in case of FCCU NCF 3. 0b - No comparator error indicated by PFLASH3. 1b - Comparator error indicated by PFLASH3.
25 PF1_0_CHK_CMP_ALARM	Status of PFLASH2 checker (redundant) safety comparator alarm. Read this bit to identify the reason of fault in case of FCCU NCF 3. 0b - No comparator error indicated by PFLASH2. 1b - Comparator error indicated by PFLASH2.
24 PF0_1_CHK_CMP_ALARM	Status of PFLASH1 checker (redundant) safety comparator alarm. Read this bit to identify the reason of fault in case of FCCU NCF 3. 0b - No comparator error indicated by PFLASH1. 1b - Comparator error indicated by PFLASH1.
23 PF0_0_CHK_CMP_ALARM	Status of PFLASH0 checker (redundant) safety comparator alarm. Read this bit to identify the reason of fault in case of FCCU NCF 3. 0b - No comparator error indicated by PFLASH0. 1b - Comparator error indicated by PFLASH0.
22	Status of PFLASH3 safety comparator alarm. Read this bit to identify the reason of fault in case of FCCU NCF 3.

Table continues on the next page...

Table continued from the previous page...

Field	Function
PF1_1_CMP_ALARM	0b - No comparator error indicated by PFLASH3. 1b - Comparator error indicated by PFLASH3.
21 PF1_0_CMP_ALARM	Status of PFLASH2 safety comparator alarm. Read this bit to identify the reason of fault in case of FCCU NCF 3. 0b - No comparator error indicated by PFLASH2. 1b - Comparator error indicated by PFLASH2.
20 PF0_1_CMP_ALARM	Status of PFLASH1 safety comparator alarm. Read this bit to identify the reason of fault in case of FCCU NCF 3. 0b - No comparator error indicated by PFLASH1. 1b - Comparator error indicated by PFLASH1.
19 PF0_0_CMP_ALARM	Status of PFLASH0 safety comparator alarm. Read this bit to identify the reason of fault in case of FCCU NCF 3. 0b - No comparator error indicated by PFLASH0. 1b - Comparator error indicated by PFLASH0.
18 PRAM3_FCCU_ALARM	PRAM3 FCCU Alarm Status Status of PRAM3 safety alarm. This alarm is set on faulty SRAM3 read or read modify error. Read this bit to identify the reason of fault in case of FCCU NCF 2. 0b - No safety alarm indicated by PRAM3. 1b - Safety alarm indicated by PRAM3.
17 PRAM3_ECC_ERR	PRAM3 Multi-bit ECC Error Status Multi-bit ECC error from SRAM3. Read this bit to identify the reason of fault in case of FCCU NCF 2. 0b - No multi-bit ECC error. 1b - Multi-bit ECC error.
16 FLASH1_ECC_ERR	ECC Error From Flash Controller1 This alarm indicates that the flash controller1 detected an error in the address ECC manipulation logic through EDC. Read this bit to identify the reason of fault in case of FCCU NCF 3. 0b - No ECC error from flash controller1. 1b - ECC error from flash controller1.
15 —	Reserved
14 FLASH1_RST_ERR	Flash1 Reset Error Status This error indication is set when flash1 encounters errors during its reset reads. Read this bit to identify the reason of fault in case of FCCU NCF 3.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No flash1 reset error indicated. 1b - Flash1 reset error indicated.
13 FLASH1_REF_ERR	Flash1 Reference Error Flash1 reference current loss or read voltage error while prevoius read. Read this bit to identify the reason of fault in case of FCCU NCF 3. 0b - No reference current loss or read voltage error while previous read. 1b - Reference current loss or read voltage error while previous read.
12 FLASH1_ADDR_ENC_ERR	Flash1 Address Encode Error In address decoding, if multiple or no address line is selected, FMU reports address encode error. Read this bit to identify the reason of fault in case of FCCU NCF 3. 0b - No address encode error in flash1. 1b - Address enocde error in flash1.
11 FLASH1_EDC_ERR	Flash1 EDC Error Status of flash1 ECC correction error through EDC reported by FMU. Read this bit to identify the reason of fault in case of FCCU NCF 3. 0b - No EDC after ECC error reported in flash1. 1b - EDC after ECC error reported in flash1.
10 PF1_1_DATA_ECC_ERR	Flash3 Data ECC Uncorrectable Error The errors are reported from the FMU and are connected to FCCU NCFs. These are also connected to ERM. See ERM chapter for the memory errors and mapping onto ERM channels. Read this bit to identify the reason of fault in case of FCCU NCF 3. The path is from FMU to PFLASH controller to ERM to FCCU. 0b - No uncorrectable error reported in flash controller port 3 data memory by FMU. 1b - Uncorrectable error reported in flash controller port 3 data memory by FMU.
9 PF1_1_CODE_ECC_ERR	Flash3 Code ECC Uncorrectable Error The errors are reported from the FMU and are connected to FCCU NCFs. These are also connected to ERM. See ERM chapter for the memory errors and mapping onto ERM channels. Read this bit to identify the reason of fault in case of FCCU NCF 3. The path is from FMU to PFLASH controller to ERM to FCCU. 0b - No uncorrectable error reported in flash controller port 3 code memory by FMU. 1b - Uncorrectable error reported in flash controller port 3 code memory by FMU.
8 PF1_0_DATA_ECC_ERR	Flash2 Data ECC Uncorrectable Error The errors are reported from the FMU and are connected to FCCU NCFs. These are also connected to ERM. See ERM chapter for the memory errors and mapping onto ERM channels. Read this bit to identify

Table continues on the next page...

Table continued from the previous page...

Field	Function
	the reason of fault in case of FCCU NCF 3. The path is from FMU to PFLASH controller to ERM to FCCU. 0b - No uncorrectable error reported in flash controller port 2 data memory by FMU. 1b - Uncorrectable error reported in flash controller port 2 data memory by FMU.
7 PF1_0_CODE_ECC_ERR	Flash2 Code ECC Uncorrectable Error The errors are reported from the FMU and are connected to FCCU NCFs. These are also connected to ERM. See ERM chapter for the memory errors and mapping onto ERM channels. Read this bit to identify the reason of fault in case of FCCU NCF 3. The path is from FMU to PFLASH controller to ERM to FCCU. 0b - No uncorrectable error reported in flash controller port 2 code memory by FMU. 1b - Uncorrectable error reported in flash controller port 2 code memory by FMU.
6 AES_RESULT_DID_SAFETY_ERR	AES RESULT DMA DID error status Read this bit to identify the reason of fault in case of FCCU NCF 2. 0b - AES RESULT DMA DID error not reported. 1b - AES RESULT DMA DID error reported.
5 AES_FEED_DID_SAFETY_ERR	AES FEED DMA DID Error Status Read this bit to identify the reason of fault in case of FCCU NCF 2. 0b - AES FEED DMA DID error not reported. 1b - AES FEED DMA DID error reported.
4 AES_KP_CRC_SAFETY_ERR	AES Key Property CRC Safety Error status Read this bit to identify the reason of fault in case of FCCU NCF 2. 0b - AES key-property CRC safety error not reported. 1b - AES key-property CRC safety error reported.
3 AES_RESULT_DMA_TCD_ADDR_ECC_ERR	AES ACCEL RESULT DMA_TCD Address ECC Error Status Read this bit to identify the reason of fault in case of FCCU NCF 2. 0b - No address error reported in AES ACCEL RESULT DMA_TCD memory. 1b - Address error reported in AES ACCEL RESULT DMA_TCD memory.
2 AES_RESULT_DMA_TCD_ECC_ERR	AES ACCEL RESULT DMA_TCD memory uncorrectable ECC error status. Read this bit to identify the reason of fault in case of FCCU NCF 2. 0b - No uncorrectable error reported 1b - Uncorrectable error reported
1	AES ACCEL FEED DMA TCD Address ECC Error Status Read this bit to identify the reason of fault in case of FCCU NCF 2.

Table continues on the next page...

Table continued from the previous page...

Field	Function
AES_FEED_DMA_TCD_ADDR_ECC_ERR	0b - No address error reported in AES ACCEL FEED DMA_TCD memory. 1b - Address error reported in AES ACCEL FEED DMA_TCD memory.
0 AES_FEED_DMA_TCD_ECC_ERR	AES ACCEL FEED DMA_TCD memory uncorrectable ECC error status. Read this bit to identify the reason of fault in case of FCCU NCF 2. 0b - No uncorrectable error reported 1b - Uncorrectable error reported

38.2.10 Read-Only GPR On Functional Reset 1 (DCMROF1)

Offset

Register	Offset
DCMROF1	300h

Function

Specifies current transfer channel ID.

This register resets after functional reset 1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved		AES_RESULT_DID_ERR_DID				AES_RES...	AES_RES...	Reserved		AES_FEED_DID_ERR_DID				AES_FEE...	AES_FEE...
W	W1C		W1C				W1C	W1C	W1C		W1C				W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved		Reserved				Reserved	Reserved	Reserved		MAC1_MD...	MAC1_MD...	MAC1_MD...	MAC_MDC...	MAC_MDC...	MAC_MDC...
W	W1C		W1C				W1C	W1C	W1C		W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
29-26 AES_RESULT_DID_ERR_DID	Indicates DID[3:0] value when the AES result DID error was reported.
25 AES_RESULT_DID_ERR_NS	Indicates whether the non-secure attribute was set or not when the AES result DID error was reported. 0b - Not set 1b - Set
24 AES_RESULT_DID_ERR_PRIV	Indicates whether the privilege attribute was set or not when the AES result DID error was reported. 0b - Not set 1b - Set
23-22 —	Reserved
21-18 AES_FEED_DID_ERR_DID	Indicates DID[3:0] value when the AES feed DID error was reported.
17 AES_FEED_DID_ERR_NS	Indicates whether the non-secure attribute was set or not when the AES feed DID error was reported. 0b - Not set 1b - Set
16 AES_FEED_DID_ERR_PRIV	Indicates whether the privilege attribute was set or not when the AES feed DID error was reported. 0b - Not set 1b - Set
15-14 —	Reserved
13-10 —	Reserved
9 —	Reserved
8 —	Reserved
7-6 —	Reserved
5	MAC eDMA Channel ID2 Status

Table continues on the next page...

Table continued from the previous page...

Field	Function
MAC1_MDC_CHID_2	Specifies whether channel ID2 is the current transfer channel ID. 0b - No 1b - Yes
4 MAC1_MDC_CHID_1	MAC eDMA Channel ID1 Status Specifies whether channel ID1 is the current transfer channel ID. 0b - No 1b - Yes
3 MAC1_MDC_CHID_0	MAC eDMA Channel ID0 Status Specifies whether channel ID0 is the current transfer channel ID. 0b - No 1b - Yes
2 MAC_MDC_CHID_2	MAC eDMA Channel ID2 Status Specifies whether channel ID2 is the current transfer channel ID. 0b - No 1b - Yes
1 MAC_MDC_CHID_1	MAC eDMA Channel ID1 Status Specifies whether channel ID1 is the current transfer channel ID. 0b - No 1b - Yes
0 MAC_MDC_CHID_0	MAC eDMA Channel ID0 Status Specifies whether channel ID0 is the current transfer channel ID. 0b - No 1b - Yes

38.2.11 Read-Only GPR On Functional Reset 2 (DCMROF2)

Offset

Register	Offset
DCMROF2	304h

Function

Specifies the SDID0 contents that DCM scans from the flash memory.

This register resets after functional reset 2.

NOTE

See the DCF clients file attached to this document for more information. The Utest section contains chip-configurable ID information, captured as status on read-only GPR in functional reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DCF_SDID0															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DCF_SDID0															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 DCF_SDID0	DCF Client SDID 0 Configuration

38.2.12 Read-Only GPR On Functional Reset 3 (DCMROF3)

Offset

Register	Offset
DCMROF3	308h

Function

Specifies the SDID1 contents that DCM scans from the flash memory.

This register resets after functional reset 3.

NOTE

See the DCF clients file attached to this document for more information. The Utest section contains chip-configurable ID information, captured as status on read-only GPR in functional reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DCF_SDID1															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DCF_SDID1															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 DCF_SDID1	DCF Client SDID 1 Configuration

38.2.13 Read-Only GPR On Functional Reset 4 (DCMROF4)

Offset

Register	Offset
DCMROF4	30Ch

Function

Specifies the SDID2 contents that DCM scans from the flash memory.

This register resets after functional reset 4.

NOTE

See the DCF clients file attached to this document for more information. The Utest section contains chip-configurable ID information, captured as status on read-only GPR in functional reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DCF_SDID2															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DCF_SDID2															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 DCF_SDID2	DCF Client SDID 2 Configuration

38.2.14 Read-Only GPR On Functional Reset 5 (DCMROF5)

Offset

Register	Offset
DCMROF5	310h

Function

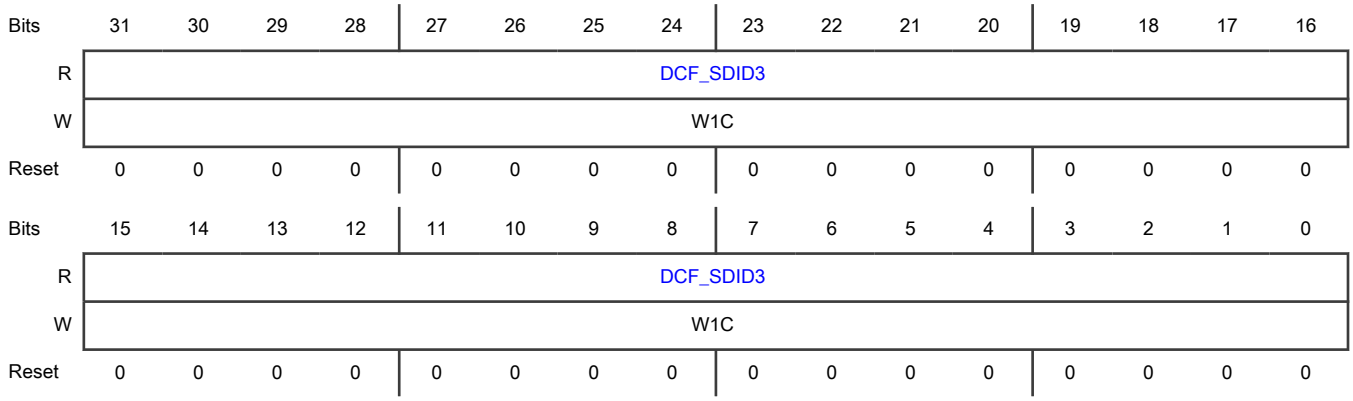
Specifies the SDID3 contents that DCM scans from the flash memory.

This register resets after functional reset 5.

NOTE

See the DCF clients file attached to this document for more information. The Utest section contains chip-configurable ID information, captured as status on read-only GPR in functional reset.

Diagram



Fields

Field	Function
31-0 DCF_SDID3	DCF Client SDID 3 Configuration

38.2.15 Read-Only GPR On Functional Reset 6 (DCMROF6)

Offset

Register	Offset
DCMROF6	314h

Function

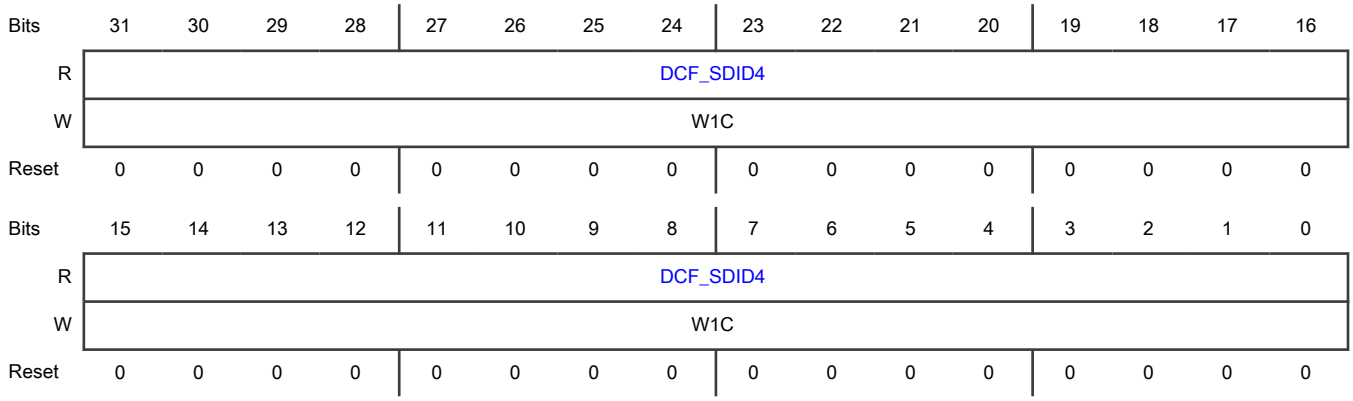
Specifies the SDID4 contents that DCM scans from the flash memory.

This register resets after functional reset 6.

NOTE

See the DCF clients file attached to this document for more information. The Utest section contains chip-configurable ID information, captured as status on read-only GPR in functional reset.

Diagram



Fields

Field	Function
31-0 DCF_SDID4	DCF Client SDID 4 Configuration

38.2.16 Read-Only GPR On Functional Reset 7 (DCMROF7)

Offset

Register	Offset
DCMROF7	318h

Function

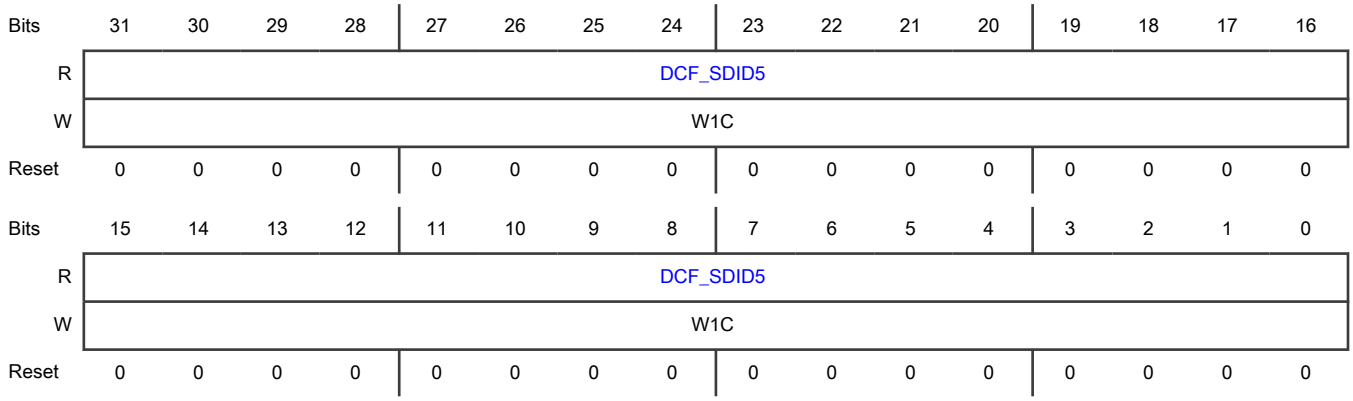
Specifies the SDID5 contents that DCM scans from the flash memory.

This register resets after functional reset 7.

NOTE

See the DCF clients file attached to this document for more information. The Utest section contains chip-configurable ID information, captured as status on read-only GPR in functional reset.

Diagram



Fields

Field	Function
31-0 DCF_SDID5	DCF Client SDID 5 Configuration

38.2.17 Read-Only GPR On Functional Reset 8 (DCMROF8)

Offset

Register	Offset
DCMROF8	31Ch

Function

Specifies the SDID6 contents that DCM scans from the flash memory.

This register resets after functional reset 8.

NOTE

See the DCF clients file attached to this document for more information. The Utest section contains chip-configurable ID information, captured as status on read-only GPR in functional reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DCF_SDID6															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DCF_SDID6															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 DCF_SDID6	DCF Client SDID 6 Configuration

38.2.18 Read-Only GPR On Functional Reset 9 (DCMROF9)

Offset

Register	Offset
DCMROF9	320h

Function

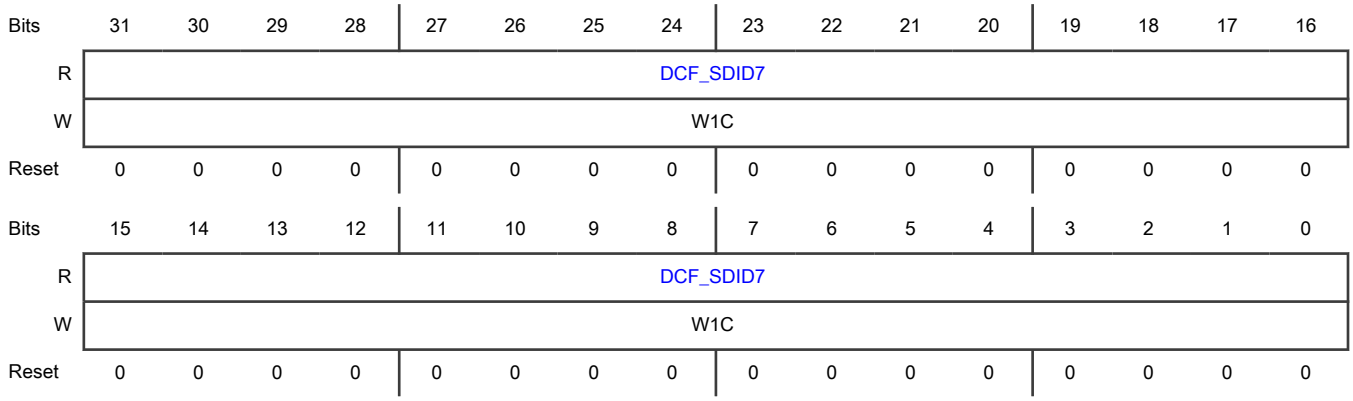
Specifies the SDID7 contents that DCM scans from the flash memory.

This register resets after functional reset 9.

NOTE

See the DCF clients file attached to this document for more information. The Utest section contains chip-configurable ID information, captured as status on read-only GPR in functional reset.

Diagram



Fields

Field	Function
31-0 DCF_SDID7	DCF Client SDID 7 Configuration

38.2.19 Read-Only GPR On Functional Reset 10 (DCMROF10)

Offset

Register	Offset
DCMROF10	324h

Function

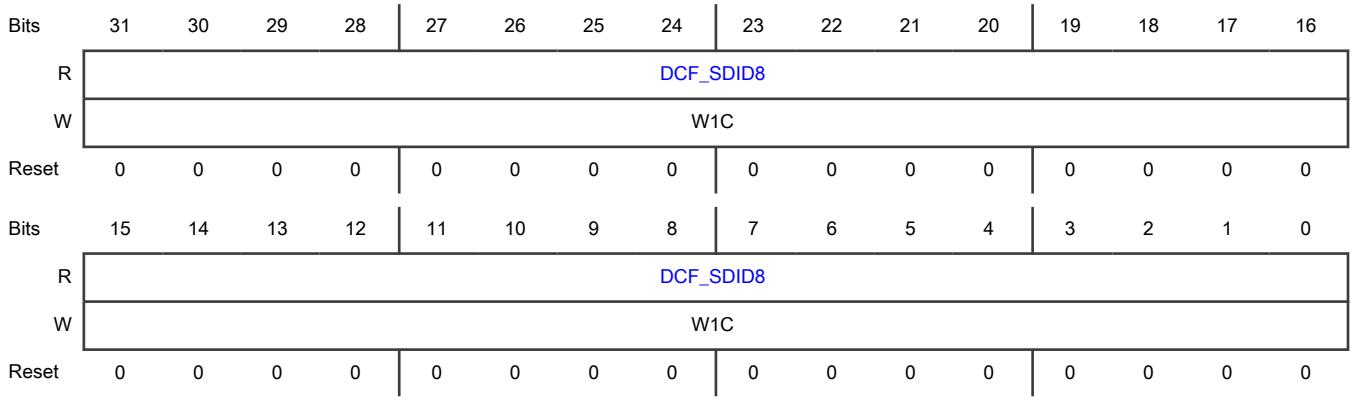
Specifies the SDID8 contents that DCM scans from the flash memory.

This register resets after functional reset 10.

NOTE

See the DCF clients file attached to this document for more information. The Utest section contains chip-configurable ID information, captured as status on read-only GPR in functional reset.

Diagram



Fields

Field	Function
31-0 DCF_SDID8	DCF Client SDID 8 Configuration

38.2.20 Read-Only GPR On Functional Reset 11 (DCMROF11)

Offset

Register	Offset
DCMROF11	328h

Function

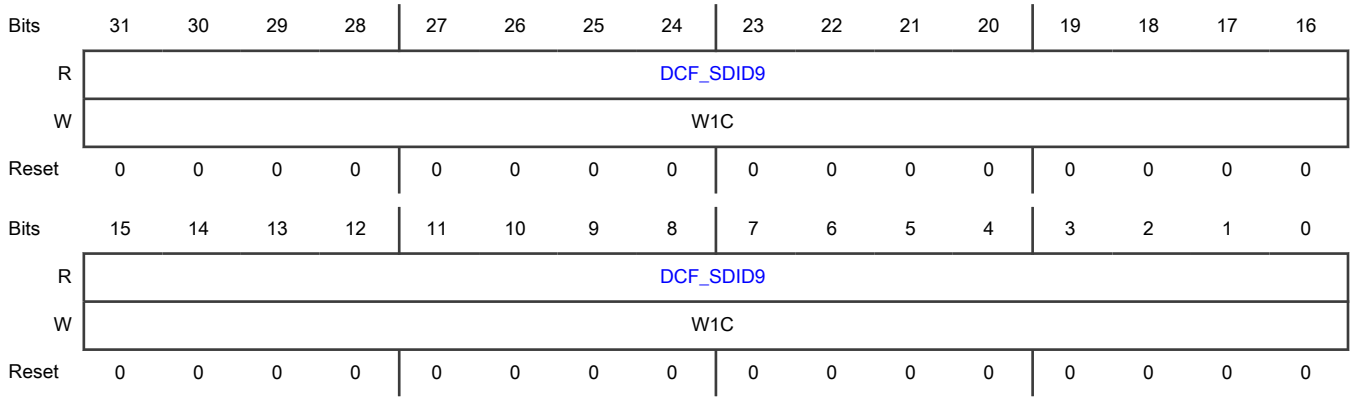
Specifies the SDID9 contents that DCM scans from the flash memory.

This register resets after functional reset 11.

NOTE

See the DCF clients file attached to this document for more information. The Utest section contains chip-configurable ID information, captured as status on read-only GPR in functional reset.

Diagram



Fields

Field	Function
31-0 DCF_SDID9	DCF Client SDID 9 Configuration

38.2.21 Read-Only GPR On Functional Reset 12 (DCMROF12)

Offset

Register	Offset
DCMROF12	32Ch

Function

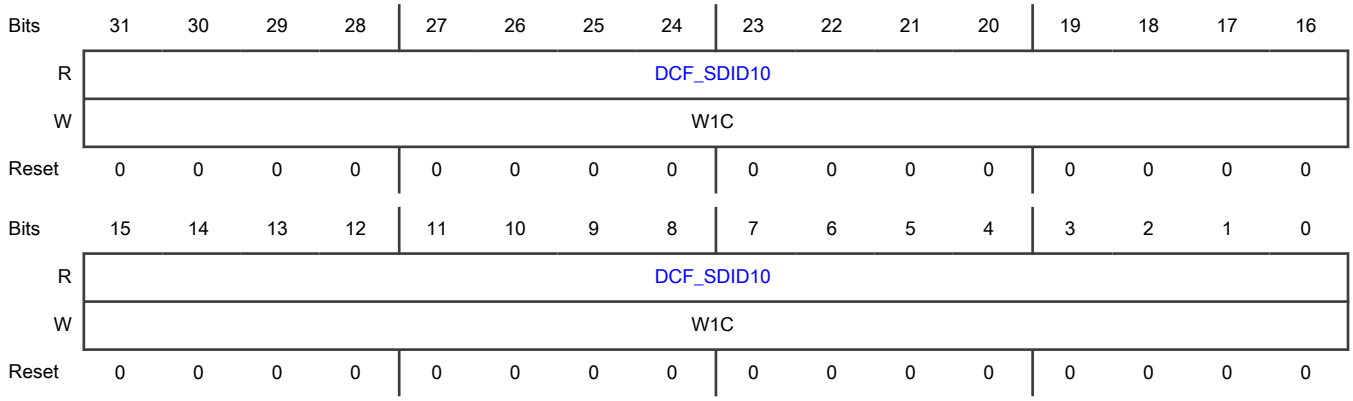
Specifies the SDID10 contents that DCM scans from the flash memory.

This register resets after functional reset 12.

NOTE

See the DCF clients file attached to this document for more information. The Utest section contains chip-configurable ID information, captured as status on read-only GPR in functional reset.

Diagram



Fields

Field	Function
31-0 DCF_SDID10	DCF Client SDID 10 Configuration

38.2.22 Read-Only GPR On Functional Reset 13 (DCMROF13)

Offset

Register	Offset
DCMROF13	330h

Function

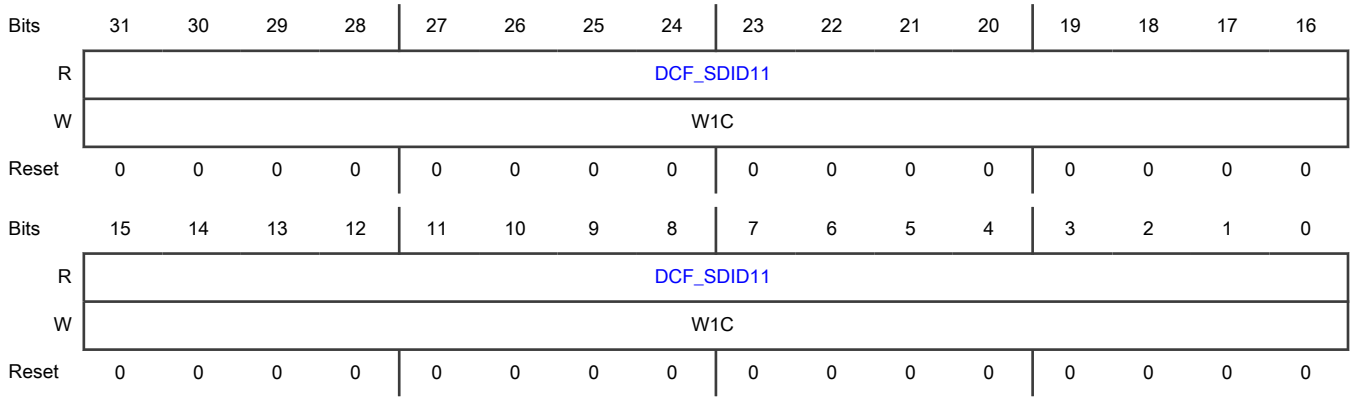
Specifies the SDID11 contents that DCM scans from the flash memory.

This register resets after functional reset 13.

NOTE

See the DCF clients file attached to this document for more information. The Utest section contains chip-configurable ID information, captured as status on read-only GPR in functional reset.

Diagram



Fields

Field	Function
31-0 DCF_SDID11	DCF Client SDID 11 Configuration

38.2.23 Read-Only GPR On Functional Reset 14 (DCMROF14)

Offset

Register	Offset
DCMROF14	334h

Function

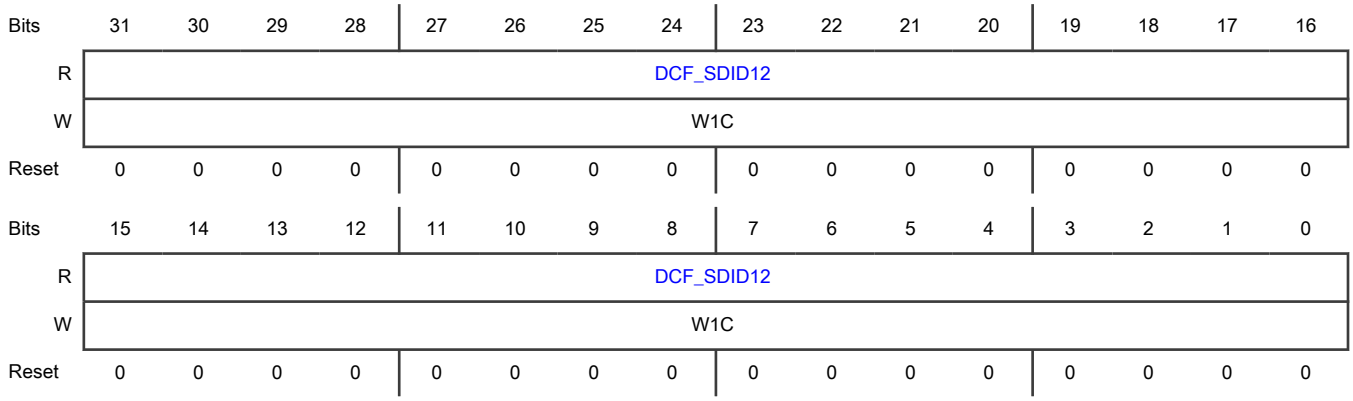
Specifies the SDID12 contents that DCM scans from flash memory.

This register resets after functional reset 14.

NOTE

See the DCF clients file attached to this document for more information. The Utest section contains chip-configurable ID information, captured as status on read-only GPR in functional reset.

Diagram



Fields

Field	Function
31-0 DCF_SDID12	DCF Client SDID 12 Configuration

38.2.24 Read-Only GPR On Functional Reset 15 (DCMROF15)

Offset

Register	Offset
DCMROF15	338h

Function

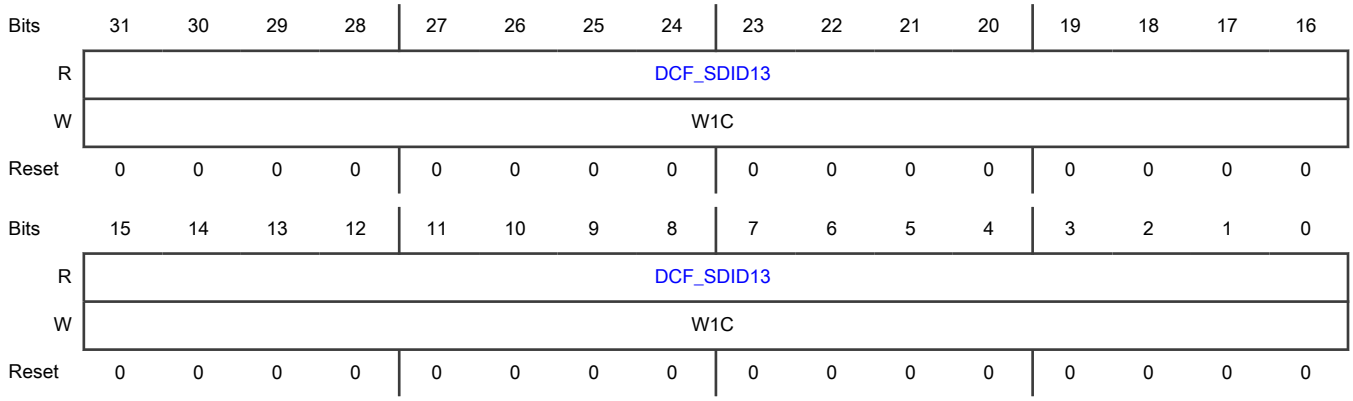
Specifies the SDID13 contents that DCM scans from the flash memory.

This register resets after functional reset 15.

NOTE

See the DCF clients file attached to this document for more information. The Utest section contains chip-configurable ID information, captured as status on read-only GPR in functional reset.

Diagram



Fields

Field	Function
31-0 DCF_SDID13	DCF Client SDID 13 Configuration

38.2.25 Read-Only GPR On Functional Reset 16 (DCMROF16)

Offset

Register	Offset
DCMROF16	33Ch

Function

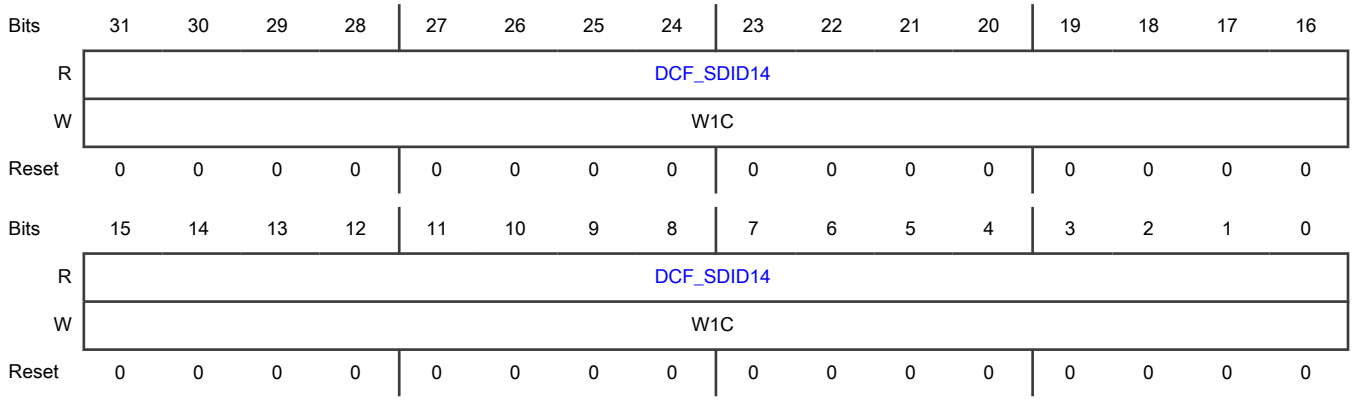
Specifies the SDID14 contents that DCM scans from the flash memory.

This register resets after functional reset 16.

NOTE

See the DCF clients file attached to this document for more information. The Utest section contains chip-configurable ID information, captured as status on read-only GPR in functional reset.

Diagram



Fields

Field	Function
31-0 DCF_SDID14	DCF Client SDID 14 Configuration

38.2.26 Read-Only GPR On Functional Reset 17 (DCMROF17)

Offset

Register	Offset
DCMROF17	340h

Function

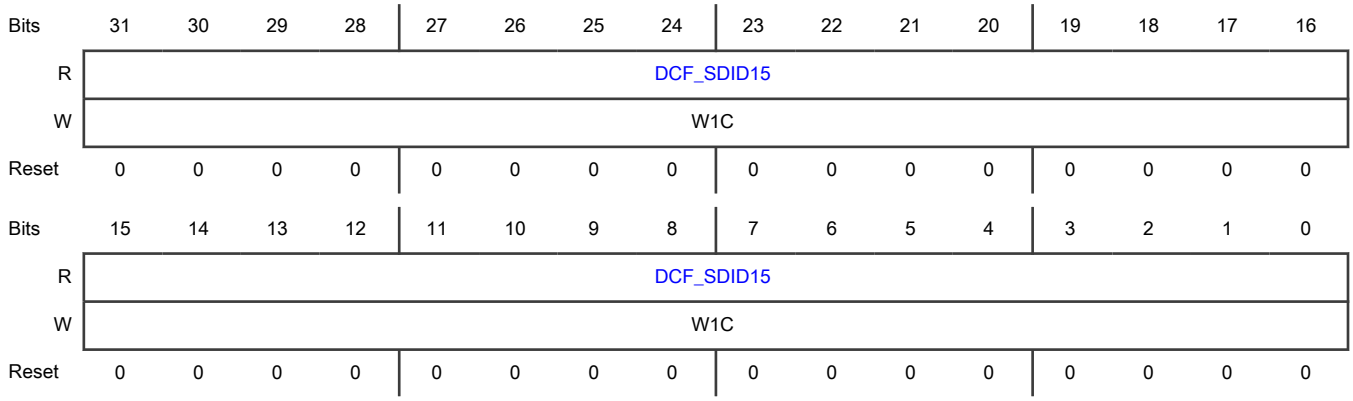
Specifies the SDID15 contents that DCM scans from the flash memory.

This register resets after functional reset 17.

NOTE

See the DCF clients file attached to this document for more information. The Utest section contains chip-configurable ID information, captured as status on read-only GPR in functional reset.

Diagram



Fields

Field	Function
31-0 DCF_SDID15	DCF Client SDID 15 Configuration

38.2.27 Read-Only GPR On Functional Reset 19 (DCMROF19)

Offset

Register	Offset
DCMROF19	348h

Function

Contains information related to:

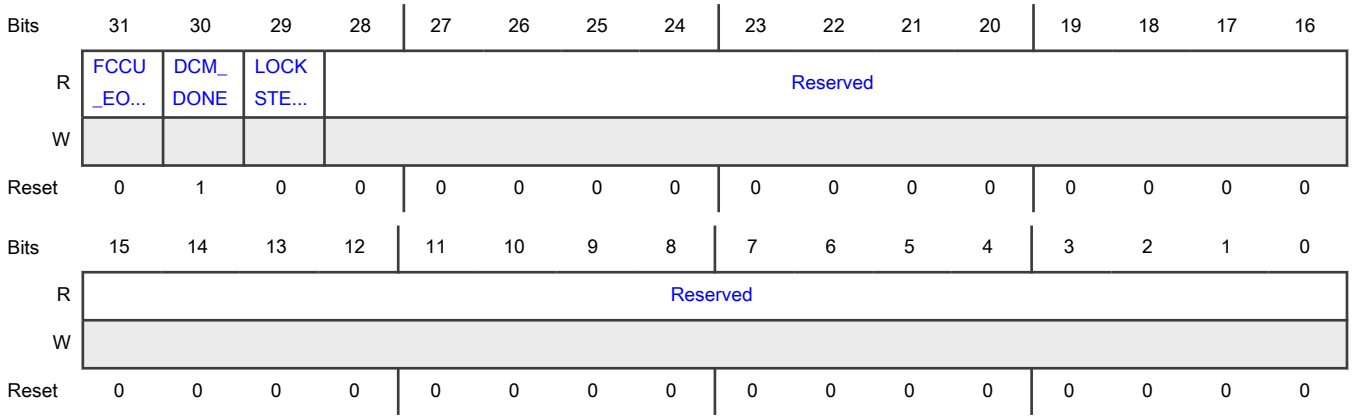
- FCCU EOUT status.
- Flash memory scanning status.
- Lockstep enable.

This register resets after functional reset 19.

NOTE

The reset value is undefined on reset and is loaded from the flash memory contents at the end of the reset sequence.

Diagram



Fields

Field	Function
31 FCCU_EOUT_DEDICATED	FCCU EOUT Status Specifies the status of FCCU_EOUT pins on GPIO_2 and GPIO_3 as configured in the DCF record, UTEST_MISC[FCCU_EOUT_DEDICATED]. See the DCF clients file attached to this document for more information. **NOTE** If the pads are dedicated to FCCU error output, these must not be programmed as input. 0b - General purpose, supporting all functions 1b - Dedicated EOUT pins
30 DCM_DONE	Flash Memory Scanning Status Specifies the status of flash memory scanning by DCM. 0b - Incomplete 1b - Complete
29 LOCKSTEP_EN	Lockstep Enable Specifies the current chip operation mode as configured in the DCF record, UTEST_MISC[LOCKSTEP_EN]. See the DCF clients file attached to this document for more information. 0b - Decoupled operation of Cortex-M7_0 and Cortex-M7_1 1b - Lockstep operation of Cortex-M7_0 and Cortex-M7_1
28-0 —	Reserved

38.2.28 Read-Only GPR On Functional Reset 20 (DCMROF20)

Offset

Register	Offset
DCMROF20	34Ch

Function

Specifies the information of chip destructive reset escalation support for destructive reset sources as configured in the DCF record, DEST_RST_ESC[13:0]. See the DCF clients file attached to this document for the mapping of corresponding destructive reset events.

This register resets after functional reset 20.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DCF_DEST_RST_ESC														Reserved	Reserved
W																
Reset	u[1]	u	u	u	u	u	u	u	u	u	u	u	u	u	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved				Reserved					AIPS_IA...	QSPI_IA...	Reserved	DMA_AXB...	Reserved	Reserved	POR_WDG...
W																
Reset	0	0	0	0	0	0	0	0	0	u[2]	u[1]	0	u[2]	0	0	u[1]

1. The reset value of this register is dependent on the DCF client's default value.
2. The reset value of this register is dependent on DCF client default value.

Fields

Field	Function
31-18 DCF_DEST_RST_ESC	DCF Destructive Reset Escalation Enables the destructive reset escalation feature for the corresponding destructive reset event. 00_0000_0000_0000b - Disables 00_0000_0000_0001b - Enables
17 —	Reserved
16 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-12 —	Reserved
11-7 —	Reserved
6 AIPS_IAHB_BYP	Status of AIPS1/2 IAHB gasket as configured in DCF record, UTEST_MISC[AIPS_IAHB_BYP]. 0b - Register wall enabled. 1b - Register wall bypassed.
5 QSPI_IAHB_BYP	QuadSPI IAHB Bypass Status Specifies the status of the QuadSPI IAHB gasket as configured in the DCF record, UTEST_MISC[QSPI_IAHB_BYP]. See the DCF clients file attached to this document for more information. 0b - Register wall enabled 1b - Register wall bypassed
4 —	Reserved
3 DMA_AXBS_IAHB_BYP	Status of DMA AXBS IAHB gasket as configured in DCF record, UTEST_MISC[DMA_AXBS_IAHB_BYP]. 0b - Register wall enabled. 1b - Register wall bypassed.
2 —	Reserved
1 —	Reserved
0 POR_WDG_EN	POR Watchdog (POR_WDG) Status Specifies the status of POR_WDG as configured in the DCF record, UTEST_MISC[POR_WDG_EN]. The POR_WDG is enabled by default. See the DCF clients file attached to this document for more information. 0b - Disabled 1b - Enabled

38.2.29 Read-Only GPR On Functional Reset 21 (DCMROF21)

Offset

Register	Offset
DCMROF21	350h

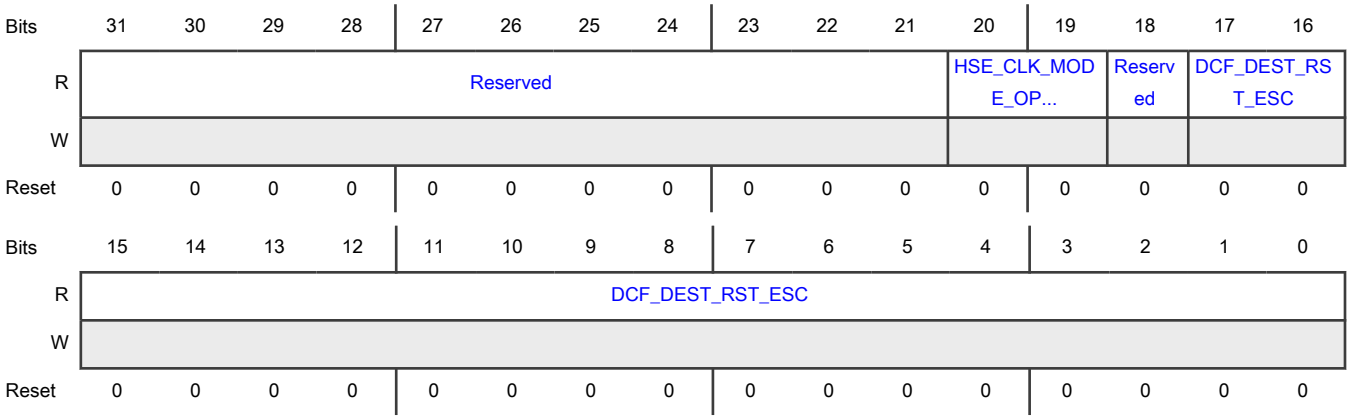
Function

Specifies the information of chip destructive reset escalation support for destructive reset sources as configured in the DCF record, DEST_RST_ESC[31:14].

This register resets after functional reset 21.

See the DCF clients file attached to this document for the mapping of corresponding destructive reset events.

Diagram



Fields

Field	Function
31-21 —	Reserved
20-19 HSE_CLK_MODE_OPTION	HSE_B Clock Mode Option Specifies the applicable clocking options. 00b - Ratio of 1:2 in between HSE subsystem programming interface clock (AIPS_SLOW_CLK) and HSE module clock (HSE_CLK). HSE_IAHB gasket enabled. See the ‘Clocking Overview’ section in the 'Clocking' chapter for the clock frequency ratios and the gasket configurations. 01b - Ratio of 1:2 in between HSE subsystem programming interface clock (AIPS_SLOW_CLK) and HSE module clock (HSE_CLK). HSE_IAHB gasket bypass. See the ‘Clocking Overview’ section in the 'Clocking' chapter for the clock frequency ratios and the gasket configurations. 1xb - Ratio of 1:4 in between HSE subsystem programming interface clock (AIPS_SLOW_CLK) and HSE module clock (HSE_CLK). HSE_IAHB gasket enabled. See the ‘Clocking Overview’ section in the 'Clocking' chapter for the clock frequency ratios and the gasket configurations.

Table continues on the next page...

Table continued from the previous page...

Field	Function
18 —	Reserved
17-0 DCF_DEST_RST_ESC	DCF Destructive Reset Escalation Enables the destructive reset escalation feature for the corresponding destructive reset event. 00_0000_0000_0000_0000b - Disables 00_0000_0000_0000_0001b - Enables

38.2.30 Read Write GPR On POR 1 (DCMRWP1)

Offset

Register	Offset
DCMRWP1	400h

Function

Contains information related to:

- Voltage dividers.
- Supply voltage monitoring.
- Ethernet modes.
- Software NCFs.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved								SBAF_RE...	SBAF_RE...	Reserved		SYS_REC_COUNTER			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved	DEST_RESET_COUNT				POR_WDOG_TRIM		STANBDY...	Reserved	Reserved	Reserved		CLKOUT_...	Reserved		Reserved
Reset	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23 SBAF_REC_DIS_DRST	Disable Recovery Mode On Destructive Reset This bit is reset by default and Secure BAF allows recovery mode sequence if Application issues > 8 destructive reset. Application can set this bit to disable recovery mode when Application issues > 8 destructive reset. 0 - Recovery mode is enabled on greater than 8 destructive resets. 1 - Recovery mode is disabled on greater than 8 destructive resets.
22 SBAF_REC_DIS_FRST	Disable Recovery Mode On Functional Reset This bit is reset by default and Secure BAF allows recovery mode sequence if Application issues > 8 functional reset. Application can set this bit to disable recovery mode when Application issues > 8 functional reset. 0 - Recovery mode is enabled on greater than 8 functional resets. 1 - Recovery mode is disabled on greater than 8 functional resets.
21 —	Reserved
20-16 SYS_REC_COUNTER	System Recovery Counter System recovery counter stored by sBAF and HSE FW
15 —	Reserved
14-11 DEST_RESET_COUNT	Destructive Reset Counts This bit is used by sBAF internally to preserve destructive reset counts.
10-9 POR_WDOG_TRIM	POR_WDG Trim Specifies the trims for the POR_WDG timeout value. 00b - POR_WDG timeout = 06.25 ms 01b - POR_WDG timeout = 12.50 ms 10b - POR_WDG timeout = 25.00 ms 11b - POR_WDG timeout = 50.00 ms
8 STANBDY_PWDOG_DIS	Standby POR_WDG Disable Disables the standby entry and exit monitoring window of POR_WDG.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Enables 1b - Disables
7 —	Reserved
6 —	Reserved
5-4 —	Reserved
3 CLKOUT_STANDBY	Clockout Standby Expose Over Functional And Destructive Reset Specifies whether the CLKOUT_STANDBY function is available during functional or destructive reset on PTA12. 0b - No 1b - Yes
2-1 —	Reserved
0 —	Reserved

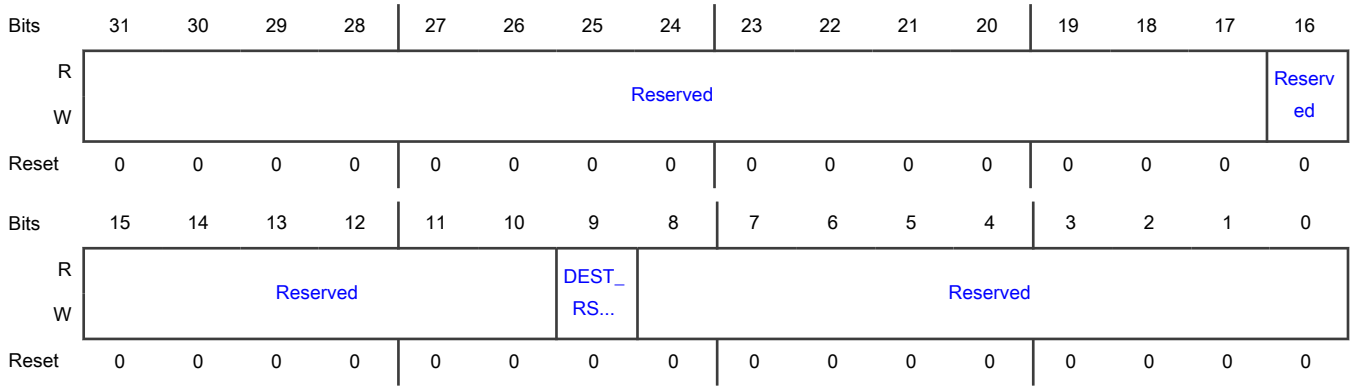
38.2.31 Read Write GPR On POR 3 (DCMRWP3)

Offset

Register	Offset
DCMRWP3	408h

Function
Resets after POR 3.

Diagram



Fields

Field	Function
31-17 —	Reserved
16 —	Reserved
15-10 —	Reserved
9 DEST_RST9_AS_IPI	Destructive Reset 9 Configures a destructive reset to interrupt. 0b - Destructive reset 1b - PLL LOL interrupt
8-0 —	Reserved

38.2.32 Read Write GPR On Destructive Reset 2 (DCMRWD2)

Offset

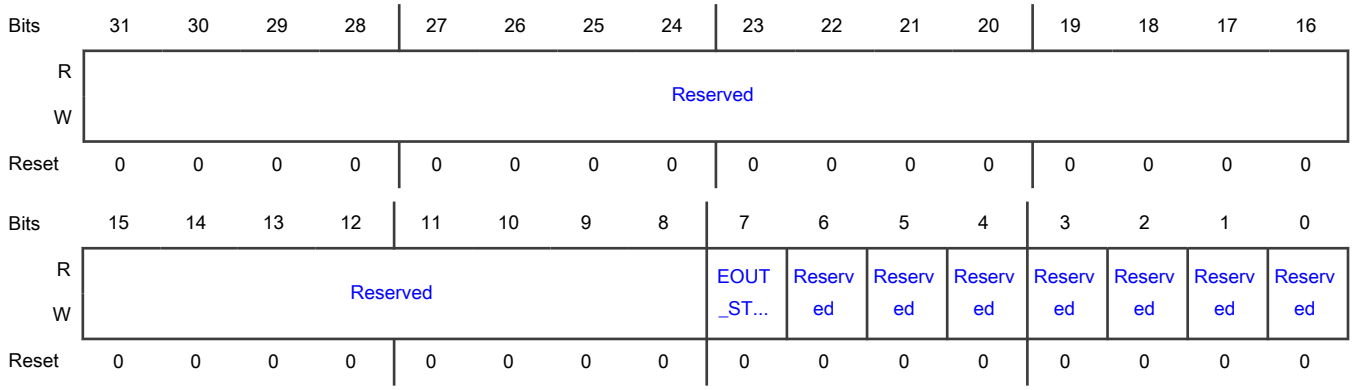
Register	Offset
DCMRWD2	504h

Function

Controls the EOUT state during self-test.

This register resets after destructive reset 2.

Diagram



Fields

Field	Function
31-8 —	Reserved
7 EOUT_STAT_DUR_STEST	Controls the EOUT state during self-test If this field = 0, the EOUT state changes to high impedance post self-test when the chip is under reset, and if this field = 1, the EOUT state remains in Fault state until this field becomes 0. DCMROF19[FCCU_EOUT_DEDICATED] is required for the feature. DCMROF19[FCCU_EOUT_DEDICATED] = 1 when you are using this feature. 0b - High impedance 1b - Fault state
6 —	Reserved
5 —	Reserved
4 —	Reserved
3 —	Reserved
2 —	Reserved
1 —	Reserved
0 —	Reserved

38.2.33 Read Write GPR On Destructive Reset 3 (DCMRWD3)

Offset

Register	Offset
DCMRWD3	508h

Function

Includes fault disable fields and resets after destructive reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved	CM7_0_I...	Reserved	CM7_0_D...	Reserved	CM7_0_D...	PRAM0_E...	PRAM1_E...	PRAM2_E...	LC_ERR_...	Reserved		PERIPH_...	MAC_GSK...	TCM_AXB...	DATA_ED...
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ADDR_ED...	AIPS2_G...	AIPS1_G...	QSPI_GS...	HSE_GSK...	Reserved	DMA_AXB...	SYS_AXB...	DMA_PER...	DMA_SYS...	TCM_GSK...	CM7_RCC...	CM7_RCC...	Reserved	Reserved	CM7_0_L...
Reset	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31 —	Reserved
30 CM7_0_ICDATA_ECC_ERR_EN	Cortex-M7_0 I-cache ECC Error Enable Specifies whether the Cortex-M7_0 core's I-cache data memory detected a multi-bit ECC error. The field enables fault monitoring at FCCU NCF 2 if there is a multi-bit ECC error from the Cortex-M7_0 core's I-cache data memory. 0b - No 1b - Yes
29 —	Reserved
28 CM7_0_DCTAG_ECC_ERR_EN	Cortex-M7_0 D-cache Tag ECC Error Enable Specifies whether the Cortex-M7_0 core's D-cache tag memory detected a multi-bit ECC error. The field enables fault monitoring at FCCU NCF 2 if there is a multi-bit ECC error from the Cortex-M7_0 core's D-cache tag memory.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No 1b - Yes
27 —	Reserved
26 CM7_0_DCDATA_ECC_ERR_EN	Cortex-M7_0 D-cache Data ECC Error Enable Specifies whether the Cortex-M7_0 core's D-cache data memory detected a multi-bit ECC error. The field enables fault monitoring at FCCU NCF 2 if there is a multi-bit ECC error from the Cortex-M7_0 core's D-cache data memory. 0b - No 1b - Yes
25 PRAM0_ECC_ERR_EN	PRAM0 ECC Error Enable Specifies whether a multi-bit ECC error occurred from PRAM0. The field enables fault monitoring at FCCU NCF 2 if there is a multi-bit ECC error from PRAM0. 0b - No 1b - Yes
24 PRAM1_ECC_ERR_EN	PRAM1 ECC Error Enable Specifies whether a multi-bit ECC error occurred from PRAM1. The field enables fault monitoring at FCCU NCF 2 if there is a multi-bit ECC error from PRAM1. 0b - No 1b - Yes
23 PRAM2_ECC_ERR_EN	Enable bit for enabling the fault monitoring at FCCU NCF 2 for the fault: Multi bit ECC error from SRAM1. 0b - No multi-bit ECC error. 1b - Multi-bit ECC error.
22 LC_ERR_EN	Life Cycle Scanning Error Enable Specifies whether an error is encountered during life-cycle scanning. The field enables fault monitoring at FCCU NCF 3 if there is an error in life-cycle scanning. **NOTE** On any PoR or destructive reset event, since this field becomes 1, lifecycle error scanning is enabled. 0b - No 1b - Yes
21-20	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
19 PERIPH_AXBS_ALARM_EN	Peripheral AXBS Alarm Enable Specifies whether peripheral AXBS_Lite reported a safety alarm. The field enables fault monitoring at FCCU NCF 1 in case of a peripheral AXBS_Lite safety alarm. 0b - No 1b - Yes
18 MAC_GSKT_ALARM_EN	MAC Gasket Alarm Enable Specifies whether the MAC IAHB gasket reported an alarm. The field enables fault monitoring at FCCU NCF 1 in case of an MAC IAHB gasket alarm. 0b - No 1b - Yes
17 TCM_AXBS_ALARM_EN	TCM AXBS Alarm Enable Specifies whether the TCM AHB splitter reported a safety alarm. The field enables fault monitoring at FCCU NCF 1 in case of a TCM AHB splitter safety alarm. 0b - No 1b - Yes
16 DATA_EDC_ERR_EN	Data EDC Error Enable Specifies whether a data EDC error occurred. The field enables fault monitoring at FCCU NCF 1, in case of an integrity error on data, for safety. 0b - No 1b - Yes
15 ADDR_EDC_ERR_EN	Address EDC Error Enable Specifies whether an address integrity (EDC) error occurred. The field enables fault monitoring at FCCU NCF 1, in case of an integrity error on address, for safety. 0b - No 1b - Yes
14 AIPS2_GSKT_ALARM_EN	Enable bit for enabling the fault monitoring at FCCU NCF 1 for the fault: AIPS2 IAHB gasket alarm. 0b - No alarm indicated by AIPS2 IAHB gasket. 1b - Alarm indicated by AIPS2 IAHB gasket.
13 AIPS1_GSKT_ALARM_EN	Enable bit for enabling the fault monitoring at FCCU NCF 1 for the fault: AIPS1 IAHB gasket alarm. 0b - No alarm indicated by AIPS1 IAHB gasket.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Alarm indicated by AIPS1 IAHB gasket.
12 QSPI_GSKT_ALARM_EN	QuadSPI Gasket Alarm Enable Specifies whether the QuadSPI IAHB gasket reported an alarm. The field enables fault monitoring at FCCU NCF 1 in case of QuadSPI IAHB gasket alarm. 0b - No 1b - Yes
11 HSE_GSKT_ALARM_EN	HSE_B Gasket Alarm Enable Specifies whether the HSE_B IAHB gasket reported an alarm. The field enables fault monitoring at FCCU NCF 1 in case of HSE_B IAHB gasket alarm. 0b - No 1b - Yes
10 —	Reserved
9 DMA_AXBS_ALARM_EN	DMA AXBS Alarm Enable Specifies whether eDMA AXBS_Lite reported a safety alarm. The field enables fault monitoring at FCCU NCF 1 in case of an eDMA AXBS_Lite safety alarm. 0b - No 1b - Yes
8 SYS_AXBS_ALARM_EN	System AXBS Alarm Enable Specifies whether the system AXBS reported a safety alarm. The field enables fault monitoring at FCCU NCF 1 in case of a system AXBS safety alarm. 0b - No 1b - Yes
7 DMA_PERIPH_GSKT_ALARM_EN	TCM Gasket Alarm Enable Specifies whether the eDMA-peripheral AXBS IAHB gasket reported a safety alarm. The field enables fault monitoring at FCCU NCF 1 in case of IAHB gasket safety alarm from the eDMA peripheral AXBS IAHB gasket. 0b - No 1b - Yes
6 DMA_SYS_GSKT_ALARM_EN	DMA System Gasket Alarm Enable Specifies whether the eDMA-system AXBS IAHB gasket reported a safety alarm.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	The field enables fault monitoring at FCCU NCF 1 in case of IAHB gasket safety alarm from the eDMA-system AXBS IAHB gasket. 0b - No 1b - Yes
5 TCM_GSKT_ALARM_EN	TCM Gasket Alarm Enable Specifies whether the TCM IAHB gasket reported an alarm. If this field = 1, the gasket reports a monitor alarm. The field enables fault monitoring at FCCU NCF 1 in case of TCM IAHB gasket monitor alarm. 0b - No 1b - Yes
4 CM7_RCCU2_ALARM_EN	Cortex-M7 RCCU2 Alarm Enable Specifies whether a redundant RCCU reported a lockstep alarm. The field enables fault monitoring at FCCU NCF 0 in case of the Cortex-M7 core redundant lockstep. 0b - No 1b - Yes
3 CM7_RCCU1_ALARM_EN	Cortex-M7 RCCU1 Alarm Enable Specifies whether RCCU reported a lockstep alarm. The field enables fault monitoring at FCCU NCF 0 in case of the Cortex-M7 core lockstep. 0b - No 1b - Yes
2 —	Reserved
1 —	Reserved
0 CM7_0_LOCKUP_EN	Cortex-M7 Lockup Enable Specifies whether the Cortex-M7_0 core is in the Lockup state. The field enables fault monitoring at FCCU NCF 0 in case of the Cortex-M7_0 core lockup. 0b - No 1b - Yes

38.2.34 Read Write GPR On Destructive Reset 4 (DCMRWD4)

Offset

Register	Offset
DCMRWD4	50Ch

Function

Contains information related to:

- Test activation errors.
- Fault monitoring.
- DCM flash memory scanning.
- ECC errors.

This register resets after destructive reset 4.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CM7_2_L...	TEST_AC...	TEST_AC...	Reserved	VDD2P5_...	VDD1P1_...	Reserved	FLASH_A...	PRAM2_F...	FLASH_S...	FLASH_R...	FLASH_R...	FLASH_A...	FLASH_E...	Reserved	Reserved
W																
Reset	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PF1_DAT...	PF1_COD...	PF0_DAT...	PF0_COD...	Reserved	PRAM1_F...	PRAM0_F...	DMA_TCD...	CM7_1_D...	CM7_1_D...	CM7_1_I...	CM7_0_D...	CM7_0_D...	CM7_0_I...	Reserved	CM7_0_I...
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31 CM7_2_LOCKUP_EN	Enable bit for enabling the fault monitoring at FCCU NCF 0 for the fault: CM7_2 core lockup. 0b - CM7_2 core not in lockup state. 1b - CM7_2 core in lockup state.
30 TEST_ACTIVATION_1_ERR_EN	Test Activation 1 Error Enable Specifies whether a partial test is activated accidentally. The field enables fault monitoring at FCCU NCF 5 in case of an accidental partial test activation 1. 0b - No 1b - Yes
29	Test Activation 0 Error Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
TEST_ACTIVATION_0_ERR_EN	Specifies whether a partial test is activated accidentally. The field enables fault monitoring at FCCU NCF 5 in case of an accidental partial test activation 0. 0b - No 1b - Yes
28 —	Reserved
27 VDD2P5_GNG_ERR_EN	VDD2P5 Go/No-go Error Enable Specifies whether the VDD2P5 (double bond) supply is clean. The field enables fault monitoring at FCCU NCF 4 if there is a "go/no-go" indicator for VDD_HV_FLA (double bond) going to FXOSC and PLL. If the value of this field = 0, there is a "go" indication that the supply if clean; if it = 1, there is a "no-go" indication that the supply is unclean and there is a fault in double-bond connection or its routing within the chip. 0b - Clean 1b - Unclean
26 VDD1P1_GNG_ERR_EN	VDD1PD1 Go/No-go Error Enable Specifies whether the VDD1PD1 (double bond) supply is clean. The field enables fault monitoring at FCCU NCF 4 if there is a "go/no-go" indicator for VDD1PD1 (double bond) supply going to PLL. If the value of this field = 0, there is a "go" indication that the supply if clean; if it = 1, there is a "no-go" indication that the supply is unclean and there is a fault in double-bond connection or its routing within the chip. 0b - Clean 1b - Unclean
25 —	Reserved
24 FLASH_ACCESS_ERR_EN	Flash Memory Access Error Enable Specifies whether a transaction monitor mismatch error occurred from the flash memory controller. The field enables fault monitoring at FCCU NCF 3 in case of a transaction monitor mismatch error from the flash memory controller. The alarm indicates that the flash memory controller detected a transaction monitor mismatch when compared to the flash memory safety feedback output. The flash memory specifies where the reconstructed address is compared with the address that invoked the flash memory access. 0b - No 1b - Yes
23 PRAM2_FCCU_ALARM_EN	Enable bit for enabling the fault monitoring at FCCU NCF 2 for the fault: PRAM2 safety alarm. This alarm is set on faulty SRAM1 read or read-modify error.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No safety alarm indicated by PRAM2. 1b - Safety alarm indicated by PRAM2.
22 FLASH_SCAN_ERR_EN	Flash Memory Scanning Error Enable Specifies whether an error occurred during DCM flash memory scanning. The field enables fault monitoring at FCCU NCF 3 in case of an error during the DCM flash memory scanning process because of invalid data. **NOTE** On a POR or destructive reset event, because this field becomes 0, the field has no effect. A life-cycle error (in case it is present) is not disabled. 0b - No 1b - Yes
21 FLASH_RST_ERR_EN	Flash Memory Reset Error Enable Specifies whether a flash memory reset error occurred. The field enables fault monitoring at FCCU NCF 3 in case of a flash memory reset error. This error indication is set when the flash memory encounters errors during its reset reads. 0b - No 1b - Yes
20 FLASH_REF_ERR_EN	Flash Memory Reference Error Encode Specifies whether a reference current loss or read voltage error occurred during previous read(s). The field enables fault monitoring at FCCU NCF 3 if there is a flash memory reference current loss or read voltage error during previous read(s). 0b - No 1b - Yes
19 FLASH_ADDR_ENC_ERR_EN	Flash Memory Address Encode Error Enable Specifies whether an address encode error occurred in the flash memory. The field enables fault monitoring at FCCU NCF 3 if there is a flash memory address encode error. During address decoding, if multiple or no address line is selected, FMU reports an address encode error. 0b - No 1b - Yes
18 FLASH_EDC_ERR_EN	Flash Memory EDC Error Enable Specifies whether EDC after ECC error is reported in the flash memory. The field enables fault monitoring at FCCU NCF 3 in case of a flash memory ECC correction error via EDC, reported by FMU.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No 1b - Yes
17 —	Reserved
16 —	Reserved
15 PF1_DATA_ECC_ERR_EN	PF1 Data ECC Error Enable Specifies whether FMU reported an uncorrectable error in the flash memory controller port 1 data memory. The field enables fault monitoring at FCCU NCF 3 in case of Flash1 data ECC uncorrectable error. 0b - No 1b - Yes
14 PF1_CODE_ECC_ERR_EN	PF1 Code ECC Error Enable Specifies whether FMU reported an uncorrectable error in the flash memory controller port 1 code memory. The field enables fault monitoring at FCCU NCF 3 in case of Flash1 code ECC uncorrectable error. 0b - No 1b - Yes
13 PF0_DATA_ECC_ERR_EN	PF0 Data ECC Error Enable Specifies whether FMU reported an uncorrectable error in the flash memory controller port 0 data memory. The field enables fault monitoring at FCCU NCF 3 in case of Flash0 data ECC uncorrectable error. 0b - No 1b - Yes
12 PF0_CODE_ECC_ERR_EN	PF0 Code ECC Error Enable Specifies whether FMU reported an uncorrectable error in the flash memory controller port 0 code memory. The field enables fault monitoring at FCCU NCF 3 in case of Flash0 code ECC uncorrectable error. 0b - No 1b - Yes
11 —	Reserved
10 PRAM1_FCCU_ALARM_EN	PRAM1 FCCU Alarm Enable Specifies whether PRAM1 reported a safety alarm.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	The field enables fault monitoring at FCCU NCF 2 in case of PRAM1 safety alarm. This alarm is set on faulty SRAM1 read or read-modify error. 0b - No 1b - Yes
9 PRAM0_FCCU_ALARM_EN	PRAM0 FCCU Alarm Enable Specifies whether PRAM0 reported a safety alarm. The field enables fault monitoring at FCCU NCF 2 in case of PRAM0 safety alarm. This alarm is set to faulty SRAM0 read or read-modify error. 0b - No 1b - Yes
8 DMA_TCD_RAM_ECC_ERR_EN	eDMA TCD RAM ECC Error Enable Specifies whether uncorrectable ECC error detection is enabled at FCCU. The field enables fault monitoring at FCCU NCF 2 in case of an uncorrectable ECC error reported from the eDMA_TCD memory. This uncorrectable ECC error consists of a multi-bit data ECC error and an address ECC error. 0b - No 1b - Yes
7 CM7_1_DTCM1_ECC_ERR_EN	Cortex-M7_1 DTCM 1 ECC Error Enable Specifies whether uncorrectable ECC error detection is enabled at FCCU. The field enables fault monitoring at FCCU NCF 2 in case of an uncorrectable ECC error from the Cortex-M7_1 core's DTCM block 1. This uncorrectable ECC error consists of a multi-bit data ECC error and an address ECC error. The Cortex-M7_1 core's DTCM 1 consists of two physical blocks. **NOTE** The Cortex-M7_1 core's DTCM 1 does not support address ECC errors in S32K324, S32K344, and S32K314. 0b - No 1b - Yes
6 CM7_1_DTCM0_ECC_ERR_EN	Cortex-M7_1 DTCM 0 ECC Error Enable Specifies whether uncorrectable ECC error detection is enabled at FCCU. This uncorrectable ECC error consists of a multi-bit data ECC error and an address ECC error. The Cortex-M7_1 core's DTCM consists of two physical blocks. This field enables fault monitoring at FCCU NCF 2 in case of an uncorrectable ECC error from the Cortex-M7_1 core's DTCM block 0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE The Cortex-M7_1 core's DTCM 0 does not support address ECC errors in S32K324, S32K344, and S32K314. 0b - No 1b - Yes
5 CM7_1_ITCM_ECC_ERR_EN	Cortex-M7_1 ITCM ECC Error Enable Specifies whether uncorrectable ECC error detection is enabled at FCCU. The field enables fault monitoring at FCCU NCF 2 in case of an uncorrectable ECC error from the Cortex-M7_1 core's ITCM. This uncorrectable ECC error consists of a multi-bit data ECC error and an address ECC error. **NOTE** The Cortex-M7_1 core's ITCM does not support address ECC errors in S32K324, S32K344, and S32K314. 0b - No 1b - Yes
4 CM7_0_DTCM1_ECC_ERR_EN	Cortex-M7_0 DTCM 1 ECC Error Enable Specifies whether uncorrectable ECC error detection is enabled at FCCU. The field enables fault monitoring at FCCU NCF 2 in case of an uncorrectable ECC error from the Cortex-M7_0 core's DTCM block 1. This uncorrectable ECC error consists of a multi-bit data ECC error and an address ECC error. The Cortex-M7_0 core's DTCM consists of two physical blocks. **NOTE** The Cortex-M7_0 core's DTCM 1 does not support address ECC errors in S32K324, S32K344, and S32K314. 0b - No 1b - Yes
3 CM7_0_DTCM0_ECC_ERR_EN	Cortex-M7_0 DTCM 0 ECC Error Enable Specifies whether uncorrectable ECC error detection is enabled at FCCU. The field enables fault monitoring at FCCU NCF 2 in case of an uncorrectable ECC error from the Cortex-M7_0 core's DTCM block 0. This uncorrectable ECC error consists of a multi-bit data ECC error and an address ECC error. The Cortex-M7_0 core's DTCM consists of two physical blocks. **NOTE** The Cortex-M7_0 core's DTCM 0 does not support address ECC errors in S32K324, S32K344, and S32K314. 0b - No 1b - Yes

Table continues on the next page...

Table continued from the previous page...

Field	Function
2 CM7_0_ITCM_ECC_ERR_EN	Cortex-M7 ITCM ECC Error Enable Specifies whether uncorrectable ECC error detection is enabled at FCCU. The field enables fault monitoring at FCCU NCF 2 in case of an uncorrectable ECC error from the Cortex-M7_0 core's ITCM. This uncorrectable ECC error consists of a multi-bit data ECC error and an address ECC error. **NOTE** The Cortex-M7_0 core's ITCM does not support address ECC errors in S32K324, S32K344, and S32K314. 0b - No 1b - Yes
1 —	Reserved
0 CM7_0_ICTAG_ECC_ERR_EN	Cortex-M7_0 I-cache Tag ECC Error Enable Specifies whether the Cortex-M7_0 core's I-cache tag memory detected a multi-bit ECC error. The field enables fault monitoring at FCCU NCF 2 if there is a multi-bit ECC error from the Cortex-M7_0 core's I-cache tag memory. 0b - No 1b - Yes

38.2.35 Read Write GPR On Destructive Reset 5 (DCMRWD5)

Offset

Register	Offset
DCMRWD5	510h

Function

Contains information related to:

- Uncorrectable ECC error detection.
- I-cache and D-cache ECC errors.
- Fault monitoring.

This register resets after destructive reset 5.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CM7_2_D...	CM7_2_D...	CM7_2_I...	CM7_2_I...	CM7_2_I...	CM7_2_D...	CM7_2_D...	CM7_2_A...	CM7_2_A...	Reserved	CM7_0_A...	CM7_0_A...	Reserved	Reserved	DMA_RDA...	Reserved
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MAC_RDA...	Reserved	DEBUG_A...	MCT_BUS...	STCU_BI...	MBIST_A...	STCU_NC...	SW_NCF_...	SW_NCF_...	SW_NCF_...	SW_NCF_...	INTM_3_...	INTM_2_...	INTM_1_...	INTM_0_...	Reserved
W																
Reset	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31 CM7_2_DTCM1_ECC_ERR_EN	Cortex-M7_2 DTCM 1 ECC Error Enable Specifies whether uncorrectable ECC error detection is enabled at FCCU from the Cortex-M7_2 core's DTCM block 1. The field enables fault monitoring at FCCU NCF 2 in case of an uncorrectable ECC error from the Cortex-M7_2 core's DTCM block 1. This uncorrectable ECC error consists of a multi-bit data ECC error and an address ECC error. The Cortex-M7_2 core's DTCM consists of two physical blocks. **NOTE** The Cortex-M7_2 core's DTCM 1 does not support address ECC errors in S32K324, S32K344, and S32K314. 0b - No 1b - Yes
30 CM7_2_DTCM0_ECC_ERR_EN	Cortex-M7_2 DTCM 0 ECC Error Enable Specifies whether uncorrectable ECC error detection is enabled at FCCU from the Cortex-M7_2 core's DTCM block 0. The field enables fault monitoring at FCCU NCF 2 in case of an uncorrectable ECC error from the Cortex-M7_2 core's DTCM block 0. This uncorrectable ECC error consists of a multi-bit data ECC error and an address ECC error. The Cortex-M7_2 core's DTCM consists of two physical blocks. **NOTE** The Cortex-M7_2 core's DTCM 0 does not support address ECC errors in S32K324, S32K344, and S32K314. 0b - No 1b - Yes
29 CM7_2_ITCM_ECC_ERR_EN	Cortex-M7_2 ITCM ECC Error Enable Specifies whether uncorrectable ECC error detection is enabled at FCCU from the Cortex-M7_2 core's ITCM.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	The field enables fault monitoring at FCCU NCF 2 in case of an uncorrectable ECC error from the Cortex-M7_2 core's ITCM. This uncorrectable ECC error consists of a multi-bit data ECC error and an address ECC error. **NOTE** The Cortex-M7_2 core's ITCM does not support address ECC errors in S32K324, S32K344, and S32K314. 0b - No 1b - Yes
28 CM7_2_ICTAG_ECC_ERR_EN	Cortex-M7_2 I-cache Tag ECC Error Enable Specifies whether the Cortex-M7_2 core's I-cache tag memory detected a multi-bit error. The field enables fault monitoring at FCCU NCF 2 if there is a multi-bit ECC error from the Cortex-M7_2 core's I-cache tag memory. 0b - No 1b - Yes
27 CM7_2_ICDATA_ECC_ERR_EN	Cortex-M7_2 I-cache Data ECC Error Enable Specifies whether the Cortex-M7_2 core's I-cache data memory detected a multi-bit ECC error. The field enables fault monitoring at FCCU NCF 2 if there is a multi-bit ECC error from the Cortex-M7_2 core's I-cache data memory. 0b - No 1b - Yes
26 CM7_2_DCTAG_ECC_ERR_EN	Cortex-M7_2 D-cache Tag ECC Error Enable Specifies whether the Cortex-M7_2 core's D-cache tag memory detected a multi-bit ECC error. The field enables fault monitoring at FCCU NCF 2 if there is a multi-bit ECC error from the Cortex-M7_2 core's D-cache tag memory. 0b - No 1b - Yes
25 CM7_2_DCDATA_ECC_ERR_EN	Cortex-M7_2 D-cache Data ECC Error Enable Specifies whether the Cortex-M7_2 core's D-cache data memory detected a multi-bit ECC error. The field enables fault monitoring at FCCU NCF 2 if there is a multi-bit ECC error from the Cortex-M7_2 core's D-cache data memory. 0b - No 1b - Yes
24	Cortex-M7_2 AHBM Read Data EDC Error Enable Specifies whether an integrity error is reported on the Cortex-M7_2 core's main read data.

Table continues on the next page...

Table continued from the previous page...

Field	Function
CM7_2_AHBM_RDATA_EDC_ERR_EN	The field enables fault monitoring at FCCU NCF 1, if there is an integrity error on the Cortex-M7_2 core's main read data, for safety. 0b - No 1b - Yes
23 CM7_2_AHBP_RDATA_EDC_ERR_EN	Cortex-M7_2 AHBP Read Data EDC Error Enable Specifies whether an integrity error is reported on the Cortex-M7_2 core's peripheral read data. The field enables fault monitoring at FCCU NCF 1, if there is an integrity error on the Cortex-M7_2 core's peripheral read data, for safety. 0b - No 1b - Yes
22 —	Reserved
21 CM7_0_AHBM_RDATA_EDC_ERR_EN	Cortex-M7_0 AHBM Read Data EDC Error Enable Specifies whether an integrity error is reported on the Cortex-M7_0 core's main read data. The field enables fault monitoring at FCCU NCF 1, if there is an integrity error on the Cortex-M7_0 core's main read data, for safety. 0b - No 1b - Yes
20 CM7_0_AHBP_RDATA_EDC_ERR_EN	Cortex-M7_0 AHBP Read Data EDC Error Enable Specifies whether an integrity error is reported on the Cortex-M7_0 core's peripheral read data. The field enables fault monitoring at FCCU NCF 1, if there is an integrity error on the Cortex-M7_0 core's peripheral read data, for safety. 0b - No 1b - Yes
19 —	Reserved
18 —	Reserved
17 DMA_RDATA_EDC_ERR_EN	eDMA Read Data EDC Error Enable Specifies whether an integrity error is reported on the eDMA read data. The field enables fault monitoring at FCCU NCF 1, if there is an integrity error on the eDMA read data, for safety.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No 1b - Yes
16 —	Reserved
15 MAC_RDATA_EDC_ERR_EN	MAC Read Data EDC Error Enable Specifies whether an integrity error is reported on the MAC read data. The field enables fault monitoring at FCCU NCF 1, if there is an integrity error on the MAC read data, for safety. 0b - No 1b - Yes
14 —	Reserved
13 DEBUG_ACTIVATION_ERR_EN	Debug Activation Error Enable Specifies whether unintended debug is activated. The field enables fault monitoring at FCCU NCF 5 for monitoring of unintended debug activation. The value of this field is 1 when the core is in the Halted state with application debug not enabled or debugger request not enabled. **NOTE** While the debugger is connected, DEBUG_ACTIVATION_ERR_EN must be 0 to disable debug activation error monitoring because the debugger is intentionally connected to the chip. 0b - No 1b - Yes
12 MCT_BUS_ERR_EN	MCT Bus Error Enable Enable bit for enabling the fault monitoring at FCCU NCF 5 for the fault: Fault reported due to illegal access on MBIST Master Controller (MCT). This fault is reported via a transfer error indication to the system. 0b - No transfer error indicated from MCT. 1b - Transfer error indicated from MCT.
11 STCU_BIST_USER_CF_EN	STCU2 BIST User CF Enable Specifies whether LBIST or MBIST is enabled accidentally (a fault condition is detected in Run mode). The field enables fault monitoring at FCCU NCF 5 if LBIST or MBIST is enabled accidentally. 0b - No 1b - Yes

Table continues on the next page...

Table continued from the previous page...

Field	Function
10 MBIST_ACTIVATION_ERR_EN	MBIST Activation Error Enable Specifies whether accidental backdoor is enabled on memories. The field enables fault monitoring at FCCU NCF 5 in case of an accidental backdoor access on memories. This monitor needs to be disabled on FCCU when performing a fault injection. 0b - No 1b - Yes
9 STCU_NCF_EN	STCU2 NCF Enable Enables fault monitoring at FCCU NCF 5 for STCU2 NCF, that is, BIST result error. 0b - Disables 1b - Enables
8 SW_NCF_3_EN	Software NCF 3 Enable Enables fault monitoring at FCCU NCF 7 for software NCF3 (DCMRWF1[FCCU_SW_NCF3]). 0b - Disables 1b - Enables
7 SW_NCF_2_EN	Software NCF 2 Enable Enables fault monitoring at FCCU NCF 7 for software NCF2 (DCMRWF1[FCCU_SW_NCF2]). 0b - Disables 1b - Enables
6 SW_NCF_1_EN	Software NCF 1 Enable Enables fault monitoring at FCCU NCF 7 for software NCF1 (DCMRWF1[FCCU_SW_NCF1]). 0b - Disables 1b - Enables
5 SW_NCF_0_EN	Software NCF 0 Enable Enables fault monitoring at FCCU NCF 7 for software NCF0 (DCMRWF1[FCCU_SW_NCF0]). 0b - Disables 1b - Enables
4 INTM_3_ERR_EN	INTM 3 Error Enable Specifies whether interrupt monitor 3 reported an error. The field enables fault monitoring at FCCU NCF 6 if INTM reports an interrupt monitor 3 error. This error is also reported in INTM.INTM_STATUS3. See the "Functional description" section in the INTM chapter for details. 0b - No

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Yes
3 INTM_2_ERR_EN	INTM 2 Error Enable Specifies whether interrupt monitor 2 reported an error. The field enables fault monitoring at FCCU NCF 6 if INTM reports an interrupt monitor 2 error. This error is also reported in INTM.INTM_STATUS2. See the "Functional description" section in the INTM chapter for details. 0b - No 1b - Yes
2 INTM_1_ERR_EN	INTM 1 Error Enable Specifies whether interrupt monitor 1 reported an error. The field enables fault monitoring at FCCU NCF 6 if INTM reports an interrupt monitor 1 error. This error is also reported in INTM.INTM_STATUS1. See the "Functional description" section in the INTM chapter for details. 0b - No 1b - Yes
1 INTM_0_ERR_EN	INTM 0 Error Enable Specifies whether interrupt monitor 0 reported an error. The field enables fault monitoring at FCCU NCF 6 if INTM reports an interrupt monitor 0 error. This error is also reported in INTM.INTM_STATUS0. See the "Functional description" section in the INTM chapter for details. 0b - No 1b - Yes
0 —	Reserved

38.2.36 Read Write GPR On Destructive Reset 6 (DCMRWD6)

Offset

Register	Offset
DCMRWD6	514h

Function

Contains information related to module debug disable.

This register resets after destructive reset 6.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SAI1_DB...	SAI0_DB...	FLEXCAN...	FLEXCAN...	FLEXCAN...	FLEXCAN...	FLEXCAN...	FLEXCAN...	FLEXIO_...	LPI2C1_...	LPI2C0_...	LPSPI5_...	LPSPI4_...	LPSPI3_...	LPSPI2_...	LPSPI1_...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LPSPI0_...	PIT2_DB...	PIT1_DB...	PIT0_DB...	STM1_DB...	STM0_DB...	Reserved	SWT0_DB...	RTC_DBG...	EMIOS2_...	EMIOS1_...	EMIOS0_...	LCU1_DB...	LCU0_DB...	FCCU_DB...	EDMA_DB...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SAI1_DBG_DIS_CM7_0	SAI1 debug disable bit for CM7_0. Set this bit 1 to disable the debug of IP. 0b - SAI1 enters debug mode when CM7_0 enters debug mode. 1b - SAI1 remains functional and is not impacted when CM7_0 enters debug mode.
30 SAI0_DBG_DIS_CM7_0	SAI0 debug disable bit for CM7_0. Set this bit 1 to disable the debug of IP. 0b - SAI0 enters debug mode when CM7_0 enters debug mode. 1b - SAI0 remains functional and is not impacted when CM7_0 enters debug mode.
29 FLEXCAN5_DBG_DIS_CM7_0	FlexCAN_5 Debug Disable Cortex-M7_0 Specifies whether FlexCAN_5 enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
28 FLEXCAN4_DBG_DIS_CM7_0	FlexCAN_4 Debug Disable Cortex-M7_0 Specifies whether FlexCAN_4 enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
27 FLEXCAN3_DBG_DIS_CM7_0	FlexCAN_3 Debug Disable Cortex-M7_0 Specifies whether FlexCAN_3 enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Remains functional and unimpacted
26 FLEXCAN2_DBG_DIS_CM7_0	FlexCAN_2 Debug Disable Cortex-M7_0 Specifies whether FlexCAN_2 enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
25 FLEXCAN1_DBG_DIS_CM7_0	FlexCAN_1 Debug Disable Cortex-M7_0 Specifies whether FlexCAN_1 enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
24 FLEXCAN0_DBG_DIS_CM7_0	FlexCAN_0 Debug Disable Cortex-M7_0 Specifies whether FlexCAN_0 enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
23 FLEXIO_DBG_DIS_CM7_0	FlexIO Debug Disable Cortex-M7_0 Specifies whether FlexIO enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
22 LPI2C1_DBG_DIS_CM7_0	LPI2C_1 Debug Disable Cortex-M7_0 Specifies whether LPI2C_1 enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
21	LPI2C_0 Debug Disable Cortex-M7_0 Specifies whether LPI2C_0 enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode.

Table continues on the next page...

Table continued from the previous page...

Field	Function
LPI2C0_DBG_DIS_CM7_0	Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
20 LPSPI5_DBG_DIS_CM7_0	LPSPI_5 Debug Disable Cortex-M7_0 Specifies whether LPSPI_5 enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
19 LPSPI4_DBG_DIS_CM7_0	LPSPI_4 Debug Disable Cortex-M7_0 Specifies whether LPSPI_4 enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
18 LPSPI3_DBG_DIS_CM7_0	LPSPI_3 Debug Disable Cortex-M7_0 Specifies whether LPSPI_3 enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
17 LPSPI2_DBG_DIS_CM7_0	LPSPI_2 Debug Disable Cortex-M7_0 Specifies whether LPSPI_2 enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
16 LPSPI1_DBG_DIS_CM7_0	LPSPI_1 Debug Disable Cortex-M7_0 Specifies whether LPSPI_1 enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted

Table continues on the next page...

Table continued from the previous page...

Field	Function
15 LPSPI0_DBG_DIS_CM7_0	LPSPI_0 Debug Disable Cortex-M7_0 Specifies whether LPSPI_0 enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
14 PIT2_DBG_DIS_CM7_0	PIT_2 Debug Disable Cortex-M7_0 Specifies whether PIT_2 enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
13 PIT1_DBG_DIS_CM7_0	PIT_1 Debug Disable Cortex-M7_0 Specifies whether PIT_1 enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
12 PIT0_DBG_DIS_CM7_0	PIT_0 Debug Disable Cortex-M7_0 Specifies whether PIT_0 enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
11 STM1_DBG_DIS_CM7_0	STM_1 Debug Disable Cortex-M7_0 Specifies whether STM_1 enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
10 STM0_DBG_DIS_CM7_0	STM_0 Debug Disable Cortex-M7_0 Specifies whether STM_0 enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode. Write 1 to this field to disable debug.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Enters Debug mode 1b - Remains functional and unimpacted
9 —	Reserved
8 SWT0_DBG_DIS_CM7_0	SWT_0 Debug Disable Cortex-M7_0 Specifies whether SWT_0 enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
7 RTC_DBG_DIS_CM7_0	RTC Debug Disable Cortex-M7_0 Specifies whether RTC enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
6 EMIOS2_DBG_DIS_CM7_0	EMIOS2 debug disable bit for CM7_0. Set this bit 1 to disable the debug of IP. 0b - eMIOS2 enters debug mode when CM7_0 enters debug mode. 1b - eMIOS2 remains functional and is not impacted when CM7_0 enters debug mode.
5 EMIOS1_DBG_DIS_CM7_0	EMIOS1 debug disable bit for CM7_0. Set this bit 1 to disable the debug of IP. 0b - eMIOS1 enters debug mode when CM7_0 enters debug mode. 1b - eMIOS1 remains functional and is not impacted when CM7_0 enters debug mode.
4 EMIOS0_DBG_DIS_CM7_0	eMIOS_0 Debug Disable Cortex-M7_0 Specifies whether eMIOS_0 enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
3 LCU1_DBG_DIS_CM7_0	LCU_1 Debug Disable Cortex-M7_0 Specifies whether LCU_1 enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode. Write 1 to this field to disable debug.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Enters Debug mode 1b - Remains functional and unimpacted
2 LCU0_DBG_DIS_CM7_0	LCU_0 Debug Disable Cortex-M7_0 Specifies whether LCU_0 enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
1 FCCU_DBG_DIS_CM7_0	FCCU Debug Disable Cortex-M7_0 Specifies whether FCCU enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
0 EDMA_DBG_DIS_CM7_0	eDMA Debug Disable Cortex-M7_0 Specifies whether eDMA enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted

38.2.37 Read Write GPR On Destructive Reset 7 (DCMRWD7)

Offset

Register	Offset
DCMRWD7	518h

Function

Contains information related to module debug disable.

This register resets after destructive reset 7.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved				FLEX CAN...	FLEX CAN...	FLEX CAN...	FLEX CAN...	PIT3_ DB...	STM3_ DB...	SWT3 _DB...	Reserv ed	Reserv ed	Reserv ed	Reserv ed	Reserv ed
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserv ed	Reserv ed	Reserv ed	Reserv ed	Reserv ed	Reserv ed	Reserv ed	Reserv ed	Reserv ed	Reserv ed	Reserv ed	SWT2 _DB...	STM2_ DB...	FLEX CAN...	FLEX CAN...	Reserv ed
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved
27 FLEXCAN11_DBG_DIS_CM7_0	FLEXCAN11 Debug Disable Cortex-M7_0 FLEXCAN11 debug disable bit for CM7_0. Set this bit 1 to disable the debug of module. 0b - FLEXCAN11 enters debug mode when CM7_0 enters debug mode. 1b - FLEXCAN11 remains functional and is not impacted when CM7_0 enters debug mode.
26 FLEXCAN10_DBG_DIS_CM7_0	FLEXCAN10 Debug Disable Cortex-M7_0 FLEXCAN10 debug disable bit for CM7_0. Set this bit 1 to disable the debug of module. 0b - FLEXCAN10 enters debug mode when CM7_0 enters debug mode. 1b - FLEXCAN10 remains functional and is not impacted when CM7_0 enters debug mode.
25 FLEXCAN9_DBG_DIS_CM7_0	FLEXCAN9 Debug Disable Cortex-M7_0 FLEXCAN9 debug disable bit for CM7_0. Set this bit 1 to disable the debug of module. 0b - FLEXCAN9 enters debug mode when CM7_0 enters debug mode. 1b - FLEXCAN9 remains functional and is not impacted when CM7_0 enters debug mode.
24 FLEXCAN8_DBG_DIS_CM7_0	FLEXCAN8 Debug Disable Cortex-M7_0 FLEXCAN8 debug disable bit for CM7_0. Set this bit 1 to disable the debug of module. 0b - FLEXCAN8 enters debug mode when CM7_0 enters debug mode. 1b - FLEXCAN8 remains functional and is not impacted when CM7_0 enters debug mode.
23 PIT3_DBG_DIS_CM7_0	PIT3 Debug Disable Cortex-M7_0 Specifies whether PIT3 enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode. Write 1 to this field to disable debug.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Enters Debug mode 1b - Remains functional and unimpacted
22 STM3_DBG_DIS_CM7_0	STM3 Debug Disable Cortex-M7_0 Specifies whether STM3 enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
21 SWT3_DBG_DIS_CM7_0	SWT3 Debug Disable Cortex-M7_0 Specifies whether SWT3 enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
20 —	Reserved
19 —	Reserved
18 —	Reserved
17 —	Reserved
16-15 —	Reserved
14 —	Reserved
13 —	Reserved
12 —	Reserved
11	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
10 —	Reserved
9 —	Reserved
8 —	Reserved
7 —	Reserved
6 —	Reserved
5 —	Reserved
4 SWT2_DBG_DIS_CM7_0	SWT_2 Debug Disable Cortex-M7_0 Specifies whether SWT_2 enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
3 STM2_DBG_DIS_CM7_0	STM_2 Debug Disable Cortex-M7_0 Specifies whether STM_2 enters Debug mode or remains functional and unimpacted when the Cortex-M7_0 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
2 FLEXCAN7_DBG_DIS_CM7_0	FLEXCAN7 debug disable bit for CM7_0. Set this bit 1 to disable the debug of module. 0b - FLEXCAN7 enters debug mode when CM7_0 enters debug mode. 1b - FLEXCAN7 remains functional and is not impacted when CM7_0 enters debug mode.
1 FLEXCAN6_DBG_DIS_CM7_0	FLEXCAN6 debug disable bit for CM7_0. Set this bit 1 to disable the debug of module. 0b - FLEXCAN6 enters debug mode when CM7_0 enters debug mode. 1b - FLEXCAN6 remains functional and is not impacted when CM7_0 enters debug mode.

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 —	Reserved

38.2.38 Read Write GPR On Destructive Reset 8 (DCMRWD8)

Offset

Register	Offset
DCMRWD8	51Ch

Function

Provides module debug disable information.

This register resets after destructive reset 8.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SAI1_DB...	SAI0_DB...	FLEXCAN...	FLEXCAN...	FLEXCAN...	FLEXCAN...	FLEXCAN...	FLEXCAN...	FLEXIO_...	LPI2C1_...	LPI2C0_...	LPSPI5_...	LPSPI4_...	LPSPI3_...	LPSPI2_...	LPSPI1_...
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	LPSPI0_...	PIT2_DB...	PIT1_DB...	PIT0_DB...	STM1_DB...	STM0_DB...	SWT1_DB...	SWT0_DB...	RTC_DBG...	EMIOS2_...	EMIOS1_...	EMIOS0_...	LCU1_DB...	LCU0_DB...	FCCU_DB...	EDMA_DB...
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SAI1_DBG_DIS_CM7_1	SAI1 debug disable bit for CM7_1. Set this bit 1 to disable the debug of IP. 0b - SAI1 enters debug mode when CM7_1 enters debug mode. 1b - SAI1 remains functional and is not impacted when CM7_1 enters debug mode.
30 SAI0_DBG_DIS_CM7_1	SAI0 debug disable bit for CM7_1. Set this bit 1 to disable the debug of IP. 0b - SAI0 enters debug mode when CM7_1 enters debug mode. 1b - SAI0 remains functional and is not impacted when CM7_1 enters debug mode.
29	FlexCAN_5 Debug Disable Cortex-M7_1

Table continues on the next page...

Table continued from the previous page...

Field	Function
FLEXCAN5_DBG_DIS_CM7_1	Specifies whether FlexCAN_5 enters Debug mode or remains functional and unimpacted when the Cortex-M7_1 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
28 FLEXCAN4_DBG_DIS_CM7_1	FlexCAN_4 Debug Disable Cortex-M7_1 Specifies whether FlexCAN_4 enters Debug mode or remains functional and unimpacted when the Cortex-M7_1 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
27 FLEXCAN3_DBG_DIS_CM7_1	FlexCAN_3 Debug Disable Cortex-M7_1 Specifies whether FlexCAN_3 enters Debug mode or remains functional and unimpacted when the Cortex-M7_1 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
26 FLEXCAN2_DBG_DIS_CM7_1	FlexCAN_2 Debug Disable Cortex-M7_1 Specifies whether FlexCAN_2 enters Debug mode or remains functional and unimpacted when the Cortex-M7_1 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
25 FLEXCAN1_DBG_DIS_CM7_1	FlexCAN_1 Debug Disable Cortex-M7_1 Specifies whether FlexCAN_1 enters Debug mode or remains functional and unimpacted when the Cortex-M7_1 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
24 FLEXCAN0_DBG_DIS_CM7_1	FlexCAN_0 Debug Disable Cortex-M7_1 Specifies whether FlexCAN_0 enters Debug mode or remains functional and unimpacted when the Cortex-M7_1 core enters Debug mode. Write 1 to this field to disable debug.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Enters Debug mode 1b - Remains functional and unimpacted
23 FLEXIO_DBG_DIS_CM7_1	FlexIO Debug Disable Cortex-M7_1 Specifies whether FlexIO enters Debug mode or remains functional and unimpacted when the Cortex-M7_1 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
22 LPI2C1_DBG_DIS_CM7_1	LPI2C_1 Debug Disable Cortex-M7_1 Specifies whether LPI2C_1 enters Debug mode or remains functional and unimpacted when the Cortex-M7_1 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
21 LPI2C0_DBG_DIS_CM7_1	LPI2C_0 Debug Disable Cortex-M7_1 Specifies whether LPI2C_0 enters Debug mode or remains functional and unimpacted when the Cortex-M7_1 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
20 LPSPI5_DBG_DIS_CM7_1	LPSPI_5 Debug Disable Cortex-M7_1 Specifies whether LPSPI_5 enters Debug mode or remains functional and unimpacted when the Cortex-M7_1 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
19 LPSPI4_DBG_DIS_CM7_1	LPSPI_4 Debug Disable Cortex-M7_1 Specifies whether LPSPI_4 enters Debug mode or remains functional and unimpacted when the Cortex-M7_1 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
18	LPSPI_3 Debug Disable Cortex-M7_1

Table continues on the next page...

Table continued from the previous page...

Field	Function
LPSPI3_DBG_DIS_CM7_1	Specifies whether LPSPI_3 enters Debug mode or remains functional and unimpacted when the Cortex-M7_1 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
17 LPSPI2_DBG_DIS_CM7_1	LPSPI_2 Debug Disable Cortex-M7_1 Specifies whether LPSPI_2 enters Debug mode or remains functional and unimpacted when the Cortex-M7_1 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
16 LPSPI1_DBG_DIS_CM7_1	LPSPI_1 Debug Disable Cortex-M7_1 Specifies whether LPSPI_1 enters Debug mode or remains functional and unimpacted when the Cortex-M7_1 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
15 LPSPI0_DBG_DIS_CM7_1	LPSPI_0 Debug Disable Cortex-M7_1 Specifies whether LPSPI_0 enters Debug mode or remains functional and unimpacted when the Cortex-M7_1 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
14 PIT2_DBG_DIS_CM7_1	PIT_2 Debug Disable Cortex-M7_1 Specifies whether PIT_2 enters Debug mode or remains functional and unimpacted when the Cortex-M7_1 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
13 PIT1_DBG_DIS_CM7_1	PIT_1 Debug Disable Cortex-M7_1 Specifies whether PIT_1 enters Debug mode or remains functional and unimpacted when the Cortex-M7_1 core enters Debug mode. Write 1 to this field to disable debug.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Enters Debug mode 1b - Remains functional and unimpacted
12 PIT0_DBG_DIS_CM7_1	PIT_0 Debug Disable Cortex-M7_1 Specifies whether PIT_0 enters Debug mode or remains functional and unimpacted when the Cortex-M7_1 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
11 STM1_DBG_DIS_CM7_1	STM_1 Debug Disable Cortex-M7_1 Specifies whether STM_1 enters Debug mode or remains functional and unimpacted when the Cortex-M7_1 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
10 STM0_DBG_DIS_CM7_1	STM_0 Debug Disable Cortex-M7_1 Specifies whether STM_0 enters Debug mode or remains functional and unimpacted when the Cortex-M7_1 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
9 SWT1_DBG_DIS_CM7_1	SWT_1 Debug Disable Cortex-M7_1 Specifies whether SWT_1 enters Debug mode or remains functional and unimpacted when the Cortex-M7_1 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
8 SWT0_DBG_DIS_CM7_1	SWT_0 Debug Disable Cortex-M7_1 Specifies whether SWT_0 enters Debug mode or remains functional and unimpacted when the Cortex-M7_1 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
7	RTC Debug Disable Cortex-M7_1

Table continues on the next page...

Table continued from the previous page...

Field	Function
RTC_DBG_DIS_CM7_1	Specifies whether RTC enters Debug mode or remains functional and unimpacted when the Cortex-M7_1 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
6 EMIOS2_DBG_DIS_CM7_1	EMIOS2 debug disable bit for CM7_1. Set this bit 1 to disable the debug of IP. 0b - EMIOS2 enters debug mode when CM7_1 enters debug mode. 1b - EMIOS2 remains functional and is not impacted when CM7_1 enters debug mode.
5 EMIOS1_DBG_DIS_CM7_1	EMIOS1 debug disable bit for CM7_1. Set this bit 1 to disable the debug of IP. 0b - EMIOS1 enters debug mode when CM7_1 enters debug mode. 1b - EMIOS1 remains functional and is not impacted when CM7_1 enters debug mode.
4 EMIOS0_DBG_DIS_CM7_1	eMIOS_0 Debug Disable Cortex-M7_1 Specifies whether eMIOS_0 enters Debug mode or remains functional and unimpacted when the Cortex-M7_1 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
3 LCU1_DBG_DIS_CM7_1	LCU_1 Debug Disable Cortex-M7_1 Specifies whether LCU_1 enters Debug mode or remains functional and unimpacted when the Cortex-M7_1 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
2 LCU0_DBG_DIS_CM7_1	LCU_0 Debug Disable Cortex-M7_1 Specifies whether LCU_0 enters Debug mode or remains functional and unimpacted when the Cortex-M7_1 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
1 FCCU_DBG_DIS_CM7_1	FCCU Debug Disable Cortex-M7_1 Specifies whether FCCU enters Debug mode or remains functional and unimpacted when the Cortex-M7_1 core enters Debug mode. Write 1 to this field to disable debug.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Enters Debug mode 1b - Remains functional and unimpacted
0 EDMA_DBG_DIS_CM7_1	eDMA Debug Disable Cortex-M7_1 Specifies whether eDMA enters Debug mode or remains functional and unimpacted when the Cortex-M7_1 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted

38.2.39 Read Write GPR On Destructive Reset 9 (DCMRWD9)

Offset

Register	Offset
DCMRWD9	520h

Function

Provides module debug disable information.

This register resets after destructive reset 9.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved				FLEXCAN...	FLEXCAN...	FLEXCAN...	FLEXCAN...	PIT3_DB...	STM3_DB...	SWT3_DB...	Reserved				Reserved
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved					Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
27 FLEXCAN11_DBG_DIS_CM7_1	FLEXCAN11 Debug Disable Cortex-M7_1 FLEXCAN11 debug disable bit for CM7_1. Set this bit 1 to disable the debug of module. 0b - FLEXCAN11 enters debug mode when CM7_1 enters debug mode. 1b - FLEXCAN11 remains functional and is not impacted when CM7_1 enters debug mode.
26 FLEXCAN10_DBG_DIS_CM7_1	FLEXCAN10 Debug Disable Cortex-M7_1 FLEXCAN10 debug disable bit for CM7_1. Set this bit 1 to disable the debug of module. 0b - FLEXCAN10 enters debug mode when CM7_1 enters debug mode. 1b - FLEXCAN10 remains functional and is not impacted when CM7_1 enters debug mode.
25 FLEXCAN9_DBG_DIS_CM7_1	FLEXCAN9 Debug Disable Cortex-M7_1 FLEXCAN9 debug disable bit for CM7_1. Set this bit 1 to disable the debug of module. 0b - FLEXCAN9 enters debug mode when CM7_1 enters debug mode. 1b - FLEXCAN9 remains functional and is not impacted when CM7_1 enters debug mode.
24 FLEXCAN8_DBG_DIS_CM7_1	FLEXCAN8 Debug Disable Cortex-M7_1 FLEXCAN8 debug disable bit for CM7_1. Set this bit 1 to disable the debug of module. 0b - FLEXCAN8 enters debug mode when CM7_1 enters debug mode. 1b - FLEXCAN8 remains functional and is not impacted when CM7_1 enters debug mode.
23 PIT3_DBG_DIS_CM7_1	PIT3 Debug Disable Cortex-M7_1 Specifies whether PIT3 enters Debug mode or remains functional and unimpacted when the Cortex-M7_1 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
22 STM3_DBG_DIS_CM7_1	STM3 Debug Disable Cortex-M7_1 Specifies whether STM3 enters Debug mode or remains functional and unimpacted when the Cortex-M7_1 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
21 SWT3_DBG_DIS_CM7_1	SWT3 Debug Disable Cortex-M7_1 Specifies whether SWT3 enters Debug mode or remains functional and unimpacted when the Cortex-M7_1 core enters Debug mode. Write 1 to this field to disable debug.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Enters Debug mode 1b - Remains functional and unimpacted
20-17 —	Reserved
16-15 —	Reserved
14 —	Reserved
13 —	Reserved
12 —	Reserved
11 —	Reserved
10-6 —	Reserved
5 —	Reserved
4 —	Reserved
3 —	Reserved
2 —	Reserved
1 —	Reserved
0 —	Reserved

38.2.40 Read Write GPR On Destructive Reset 12 (DCMRWD12)

Offset

Register	Offset
DCMRWD12	52Ch

Function

Provides module debug disable information.

This register resets after destructive reset 12.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SAI1_DB...	SAI0_DB...	FLEXCAN...	FLEXCAN...	FLEXCAN...	FLEXCAN...	FLEXCAN...	FLEXCAN...	FLEXIO_...	LPI2C1_...	LPI2C0_...	LPSPI5_...	LPSPI4_...	LPSPI3_...	LPSPI2_...	LPSPI1_...
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	LPSPI0_...	PIT2_DB...	PIT1_DB...	PIT0_DB...	STM1_DB...	STM0_DB...	Reserved	SWT0_DB...	RTC_DBG...	eMIOS2_...	eMIOS1_...	eMIOS0_...	LCU1_DB...	LCU0_DB...	FCCU_DB...	EDMA_DB...
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SAI1_DBG_DIS_CM7_2	SAI1 debug disable bit for CM7_2. Set this bit 1 to disable the debug of module. 0b - SAI1 enters debug mode when CM7_2 enters debug mode. 1b - SAI1 remains functional and is not impacted when CM7_2 enters debug mode.
30 SAI0_DBG_DIS_CM7_2	SAI0 debug disable bit for CM7_2. Set this bit 1 to disable the debug of module. 0b - SAI0 enters debug mode when CM7_2 enters debug mode. 1b - SAI0 remains functional and is not impacted when CM7_2 enters debug mode.
29 FLEXCAN5_DBG_DIS_CM7_2	FlexCAN5 Debug Disable For Cortex M7_2 Specifies whether FlexCAN5 enters Debug mode or remains functional and unimpacted when the Cortex-M7_2 core enters Debug mode. If this field = 0, FlexCAN5 enters Debug mode, and if this field = 1, FlexCAN5 remains functional. Write 1 to this field to disable debug. 0b - Enables Debug mode 1b - Disables Debug mode
28	FlexCAN4 Debug Disable For Cortex M7_2

Table continues on the next page...

Table continued from the previous page...

Field	Function
FLEXCAN4_DBG_DIS_CM7_2	Specifies whether FlexCAN4 enters Debug mode or remains functional and unimpacted when the Cortex-M7_2 core enters Debug mode. If this field = 0, FlexCAN4 enters Debug mode, and if this field = 1, FlexCAN4 remains functional. Write 1 to this field to disable debug. 0b - Enables Debug mode 1b - Disables Debug mode
27 FLEXCAN3_DBG_DIS_CM7_2	FlexCAN3 Debug Disable For Cortex M7_2 Specifies whether FlexCAN3 enters Debug mode or remains functional and unimpacted when the Cortex-M7_2 core enters Debug mode. If this field = 0, FlexCAN3 enters Debug mode, and if this field = 1, FlexCAN3 remains functional. Write 1 to this field to disable debug. 0b - Enables Debug mode 1b - Disables Debug mode
26 FLEXCAN2_DBG_DIS_CM7_2	FlexCAN2 Debug Disable For Cortex M7_2 Specifies whether FlexCAN2 enters Debug mode or remains functional and unimpacted when the Cortex-M7_2 core enters Debug mode. If this field = 0, FlexCAN2 enters Debug mode, and if this field = 1, FlexCAN2 remains functional. Write 1 to this field to disable debug. 0b - Enables Debug mode 1b - Disables Debug mode
25 FLEXCAN1_DBG_DIS_CM7_2	FlexCAN1 Debug Disable For Cortex M7_2 Specifies whether FlexCAN1 enters Debug mode or remains functional and unimpacted when the Cortex-M7_2 core enters Debug mode. If this field = 0, FlexCAN1 enters Debug mode, and if this field = 1, FlexCAN1 remains functional. Write 1 to this field to disable debug. 0b - Enables Debug mode 1b - Disables Debug mode
24 FLEXCAN0_DBG_DIS_CM7_2	FlexCAN0 Debug Disable For Cortex M7_2 Specifies whether FlexCAN0 enters Debug mode or remains functional and unimpacted when the Cortex-M7_2 core enters Debug mode. If this field = 0, FlexCAN0 enters Debug mode, and if this field = 1, FlexCAN0 remains functional. Write 1 to this field to disable debug. 0b - Enables Debug mode 1b - Disables Debug mode
23	FlexIO Debug Disable For Cortex M7_2

Table continues on the next page...

Table continued from the previous page...

Field	Function
FLEXIO_DBG_DIS_CM7_2	Specifies whether FlexIO enters Debug mode or remains functional and unimpacted when the Cortex-M7_2 core enters Debug mode. If this field = 0, FlexIO enters Debug mode, and if this field = 1, FlexIO remains functional. Write 1 to this field to disable debug. 0b - Enables Debug mode 1b - Disables Debug mode
22 LPI2C1_DBG_DIS_CM7_2	LPI2C1 Debug Disable For Cortex M7_2 Specifies whether LPI2C1 enters Debug mode or remains functional and unimpacted when the Cortex-M7_2 core enters Debug mode. If this field = 0, LPI2C1 enters Debug mode, and if this field = 1, LPI2C1 remains functional. Write 1 to this field to disable debug. 0b - Enables Debug mode 1b - Disables Debug mode
21 LPI2C0_DBG_DIS_CM7_2	LPI2C0 Debug Disable For Cortex M7_2 Specifies whether LPI2C0 enters Debug mode or remains functional and unimpacted when the Cortex-M7_2 core enters Debug mode. If this field = 0, LPI2C0 enters Debug mode, and if this field = 1, LPI2C0 remains functional. Write 1 to this field to disable debug. 0b - Enables Debug mode 1b - Disables Debug mode
20 LPSPI5_DBG_DIS_CM7_2	LPSPI5 Debug Disable For Cortex M7_2 Specifies whether LPSPI5 enters Debug mode or remains functional and unimpacted when the Cortex-M7_2 core enters Debug mode. If this field = 0, LPSPI5 enters Debug mode, and if this field = 1, LPSPI5 remains functional. Write 1 to this field to disable debug. 0b - Enables Debug mode 1b - Disables Debug mode
19 LPSPI4_DBG_DIS_CM7_2	LPSPI4 Debug Disable For Cortex M7_2 Specifies whether LPSPI4 enters Debug mode or remains functional and unimpacted when the Cortex-M7_2 core enters Debug mode. If this field = 0, LPSPI4 enters Debug mode, and if this field = 1, LPSPI4 remains functional. Write 1 to this field to disable debug. 0b - Enables Debug mode 1b - Disables Debug mode
18	LPSPI3 Debug Disable For Cortex M7_2

Table continues on the next page...

Table continued from the previous page...

Field	Function
LPSPI3_DBG_DIS_CM7_2	Specifies whether LPSPI3 enters Debug mode or remains functional and unimpacted when the Cortex-M7_2 core enters Debug mode. If this field = 0, LPSPI3 enters Debug mode, and if this field = 1, LPSPI3 remains functional. Write 1 to this field to disable debug. 0b - Enables Debug mode 1b - Disables Debug mode
17 LPSPI2_DBG_DIS_CM7_2	LPSPI2 Debug Disable For Cortex M7_2 Specifies whether LPSPI2 enters Debug mode or remains functional and unimpacted when the Cortex-M7_2 core enters Debug mode. If this field = 0, LPSPI2 enters Debug mode, and if this field = 1, LPSPI2 remains functional. Write 1 to this field to disable debug. 0b - Enables Debug mode 1b - Disables Debug mode
16 LPSPI1_DBG_DIS_CM7_2	LPSPI1 Debug Disable For Cortex M7_2 Specifies whether LPSPI1 enters Debug mode or remains functional and unimpacted when the Cortex-M7_2 core enters Debug mode. If this field = 0, LPSPI1 enters Debug mode, and if this field = 1, LPSPI1 remains functional. Write 1 to this field to disable debug. 0b - Enables Debug mode 1b - Disables Debug mode
15 LPSPI0_DBG_DIS_CM7_2	LPSPI0 Debug Disable For Cortex M7_2 Specifies whether LPSPI0 enters Debug mode or remains functional and unimpacted when the Cortex-M7_2 core enters Debug mode. If this field = 0, LPSPI0 enters Debug mode, and if this field = 1, LPSPI0 remains functional. Write 1 to this field to disable debug. 0b - Enables Debug mode 1b - Disables Debug mode
14 PIT2_DBG_DIS_CM7_2	PIT2 Debug Disable For Cortex M7_2 Specifies whether PIT2 enters Debug mode or remains functional and unimpacted when the Cortex-M7_2 core enters Debug mode. If this field = 0, PIT2 enters Debug mode, and if this field = 1, PIT2 remains functional. Write 1 to this field to disable debug. 0b - Enables Debug mode 1b - Disables Debug mode
13	PIT1 Debug Disable For Cortex M7_2

Table continues on the next page...

Table continued from the previous page...

Field	Function
PIT1_DBG_DIS_CM7_2	Specifies whether PIT1 enters Debug mode or remains functional and unimpacted when the Cortex-M7_2 core enters Debug mode. If this field = 0, PIT1 enters Debug mode, and if this field = 1, PIT1 remains functional. Write 1 to this field to disable debug. 0b - Enables Debug mode 1b - Disables Debug mode
12 PIT0_DBG_DIS_CM7_2	PIT0 Debug Disable For Cortex M7_2 Specifies whether PIT0 enters Debug mode or remains functional and unimpacted when the Cortex-M7_2 core enters Debug mode. If this field = 0, PIT0 enters Debug mode, and if this field = 1, PIT0 remains functional. Write 1 to this field to disable debug. 0b - Enables Debug mode 1b - Disables Debug mode
11 STM1_DBG_DIS_CM7_2	STM1 Debug Disable For Cortex M7_2 Specifies whether STM1 enters Debug mode or remains functional and unimpacted when the Cortex-M7_2 core enters Debug mode. If this field = 0, STM1 enters Debug mode, and if this field = 1, STM1 remains functional. Write 1 to this field to disable debug. 0b - Enables Debug mode 1b - Disables Debug mode
10 STM0_DBG_DIS_CM7_2	STM0 Debug Disable For Cortex M7_2 Specifies whether STM0 enters Debug mode or remains functional and unimpacted when the Cortex-M7_2 core enters Debug mode. If this field = 0, STM0 enters Debug mode, and if this field = 1, STM0 remains functional. Write 1 to this field to disable debug. 0b - Enables Debug mode 1b - Disables Debug mode
9 —	Reserved
8 SWT0_DBG_DIS_CM7_2	SWT0 Debug Disable For Cortex M7_2 Specifies whether SWT0 enters Debug mode or remains functional and unimpacted when the Cortex-M7_2 core enters Debug mode. If this field = 0, SWT0 enters Debug mode, and if this field = 1, SWT0 remains functional. Write 1 to this field to disable debug.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Enables Debug mode 1b - Disables Debug mode
7 RTC_DBG_DIS_CM7_2	RTC Debug Disable For Cortex M7_2 Specifies whether RTC enters Debug mode or remains functional and unimpacted when the Cortex-M7_2 core enters Debug mode. If this field = 0, RTC enters Debug mode, and if this field = 1, RTC remains functional. Write 1 to this field to disable debug. 0b - Enables Debug mode 1b - Disables Debug mode
6 eMIOS2_DBG_DIS_CM7_2	eMIOS2 debug disable bit for CM7_2. Set this bit 1 to disable the debug of module. 0b - eMIOS2 enters debug mode when CM7_2 enters debug mode. 1b - eMIOS2 remains functional and is not impacted when CM7_2 enters debug mode.
5 eMIOS1_DBG_DIS_CM7_2	eMIOS1 debug disable bit for CM7_2. Set this bit 1 to disable the debug of module. 0b - eMIOS1 enters debug mode when CM7_2 enters debug mode. 1b - eMIOS1 remains functional and is not impacted when CM7_2 enters debug mode.
4 eMIOS0_DBG_DIS_CM7_2	eMIOS0 Debug Disable For Cortex M7_2 Specifies whether eMIOS0 enters Debug mode or remains functional and unimpacted when the Cortex-M7_2 core enters Debug mode. If this field = 0, eMIOS0 enters Debug mode, and if this field = 1, eMIOS0 remains functional. Write 1 to this field to disable debug. 0b - Enables Debug mode 1b - Disables Debug mode
3 LCU1_DBG_DIS_CM7_2	LCU1 Debug Disable For Cortex M7_2 Specifies whether LCU1 enters Debug mode or remains functional and unimpacted when the Cortex-M7_2 core enters Debug mode. If this field = 0, LCU1 enters Debug mode, and if this field = 1, LCU1 remains functional. Write 1 to this field to disable debug. 0b - Enables Debug mode 1b - Disables Debug mode
2 LCU0_DBG_DIS_CM7_2	LCU0 Debug Disable For Cortex M7_2 Specifies whether LCU0 enters Debug mode or remains functional and unimpacted when the Cortex-M7_2 core enters Debug mode. If this field = 0, LCU0 enters Debug mode, and if this field = 1, LCU0 remains functional. Write 1 to this field to disable debug.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Enables Debug mode 1b - Disables Debug mode
1 FCCU_DBG_DIS_CM7_2	FCCU Debug Disable For Cortex M7_2 Specifies whether FCCU enters Debug mode or remains functional and unimpacted when the Cortex-M7_2 core enters Debug mode. If this field = 0, FCCU enters Debug mode, and if this field = 1, FCCU remains functional. Write 1 to this field to disable debug. 0b - Enables Debug mode 1b - Disables Debug mode
0 EDMA_DBG_DIS_CM7_2	eDMA Debug Disable For Cortex M7_2 Specifies whether eDMA enters Debug mode or remains functional and unimpacted when the Cortex-M7_2 core enters Debug mode. If this field = 0, eDMA enters Debug mode, and if this field = 1, eDMA remains functional. Write 1 to this field to disable debug. 0b - Enables Debug mode 1b - Disables Debug mode

38.2.41 Read Write GPR On Destructive Reset 13 (DCMRWD13)

Offset

Register	Offset
DCMRWD13	530h

Function

Provides module debug disable information.

This register resets after destructive reset 13.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved				FLEX CAN...	FLEX CAN...	FLEX CAN...	FLEX CAN...	PIT3_ DB...	STM3_ DB...	SWT3 _DB...	Reserv ed	Reserv ed	Reserv ed	Reserv ed	Reserv ed
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserv ed	Reserv ed	Reserv ed	Reserv ed	Reserv ed	Reserv ed	Reserv ed	Reserv ed	Reserv ed	Reserv ed	Reserv ed	SWT2 _DB...	STM2_ DB...	FLEX CAN...	FLEX CAN...	Reserv ed
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved
27 FLEXCAN11_DBG_DIS_CM7_2	FLEXCAN11 Debug Disable Cortex-M7_2 FLEXCAN11 debug disable bit for CM7_2. Set this bit 1 to disable the debug of module. 0b - FLEXCAN11 enters debug mode when CM7_2 enters debug mode. 1b - FLEXCAN11 remains functional and is not impacted when CM7_2 enters debug mode.
26 FLEXCAN10_DBG_DIS_CM7_2	FLEXCAN10 Debug Disable Cortex-M7_2 FLEXCAN10 debug disable bit for CM7_2. Set this bit 1 to disable the debug of module. 0b - FLEXCAN10 enters debug mode when CM7_2 enters debug mode. 1b - FLEXCAN10 remains functional and is not impacted when CM7_2 enters debug mode.
25 FLEXCAN9_DBG_DIS_CM7_2	FLEXCAN9 Debug Disable Cortex-M7_2 FLEXCAN9 debug disable bit for CM7_2. Set this bit 1 to disable the debug of module. 0b - FLEXCAN9 enters debug mode when CM7_2 enters debug mode. 1b - FLEXCAN9 remains functional and is not impacted when CM7_2 enters debug mode.
24 FLEXCAN8_DBG_DIS_CM7_2	FLEXCAN8 Debug Disable Cortex-M7_2 FLEXCAN8 debug disable bit for CM7_2. Set this bit 1 to disable the debug of module. 0b - FLEXCAN8 enters debug mode when CM7_2 enters debug mode. 1b - FLEXCAN8 remains functional and is not impacted when CM7_2 enters debug mode.
23 PIT3_DBG_DIS_CM7_2	PIT3 Debug Disable Cortex-M7_2 Specifies whether PIT3 enters Debug mode or remains functional and unimpacted when the Cortex-M7_2 core enters Debug mode. Write 1 to this field to disable debug.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Enters Debug mode 1b - Remains functional and unimpacted
22 STM3_DBG_DIS_CM7_2	STM3 Debug Disable Cortex-M7_2 Specifies whether STM3 enters Debug mode or remains functional and unimpacted when the Cortex-M7_2 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
21 SWT3_DBG_DIS_CM7_2	SWT3 Debug Disable Cortex-M7_2 Specifies whether SWT3 enters Debug mode or remains functional and unimpacted when the Cortex-M7_2 core enters Debug mode. Write 1 to this field to disable debug. 0b - Enters Debug mode 1b - Remains functional and unimpacted
20 —	Reserved
19 —	Reserved
18 —	Reserved
17 —	Reserved
16-15 —	Reserved
14 —	Reserved
13 —	Reserved
12 —	Reserved
11	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
10 —	Reserved
9 —	Reserved
8 —	Reserved
7 —	Reserved
6 —	Reserved
5 —	Reserved
4 SWT2_DBG_DIS_CM7_2	SWT2 Debug Disable For Cortex-M7_2 Specifies whether SWT2 enters Debug mode or remains functional and unimpacted when the Cortex-M7_2 core enters Debug mode. If this field = 0, SWT2 enters Debug mode, and if this field = 1, SWT2 remains functional. Write 1 to this field to disable debug. 0b - Enables Debug mode 1b - Disables Debug mode
3 STM2_DBG_DIS_CM7_2	STM2 Debug Disable For Cortex-M7_2 Specifies whether STM2 enters Debug mode or remains functional and unimpacted when the Cortex-M7_2 core enters Debug mode. If this field = 0, STM2 enters Debug mode, and if this field = 1, STM2 remains functional. Write 1 to this field to disable debug. 0b - Enables Debug mode 1b - Disables Debug mode
2 FLEXCAN7_DBG_DIS_CM7_2	FLEXCAN7 debug disable bit for CM7_2. Set this bit 1 to disable the debug of module. 0b - FLEXCAN7 enters debug mode when CM7_2 enters debug mode. 1b - FLEXCAN7 remains functional and is not impacted when CM7_2 enters debug mode.
1	FlexCAN6 Debug Disable For Cortex-M7_2

Table continues on the next page...

Table continued from the previous page...

Field	Function
FLEXCAN6_DBG_DIS_CM7_2	Specifies whether FlexCAN6 enters Debug mode or remains functional and unimpacted when the Cortex-M7_2 core enters Debug mode. If this field = 0, FlexCAN6 enters Debug mode, and if this field = 1, FlexCAN6 remains functional. Write 1 to this field to disable debug. 0b - Enables Debug mode 1b - Disables Debug mode
0 —	Reserved

38.2.42 Read Write GPR On Destructive Reset 14 (DCMRWD14)

Offset

Register	Offset
DCMRWD14	534h

Function

Provides module debug disable information.

This register resets after destructive reset 14.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved	TCM_PRA...	Reserved	Reserved	Reserved	AIPS0_G...	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
Reset	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

Fields

Field	Function
31 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
30 TCM_PRAM_AXBS_ALARM_EN	TCM_PRAM AXBS Alarm Enable Specifies whether TCM_PRAM AXBS_Lite reported a safety alarm. The field enables fault monitoring at FCCU NCF 1 for TCM_PRAM AXBS_Lite safety alarm. 0b - No 1b - Yes
29 —	Reserved
28 —	Reserved
27 —	Reserved
26 AIPS0_GSKT_ALARM_EN	AIPS0 Gasket Alarm Enable Specifies whether AIPS0 gasket reported a safety alarm. The field enables fault monitoring at FCCU NCF 1 for AIPS0 gasket safety alarm. 0b - No 1b - Yes
25 —	Reserved
24 —	Reserved
23 —	Reserved
22 —	Reserved
21 —	Reserved
20 —	Reserved
19 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
18 —	Reserved
17 —	Reserved
16 —	Reserved
15 —	Reserved
14 —	Reserved
13 —	Reserved
12 —	Reserved
11 —	Reserved
10 —	Reserved
9 —	Reserved
8 —	Reserved
7 —	Reserved
6 —	Reserved
5 —	Reserved
4	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
3 —	Reserved
2 —	Reserved
1 —	Reserved
0 —	Reserved

38.2.43 Read Write GPR On Destructive Reset 15 (DCMRWD15)

Offset

Register	Offset
DCMRWD15	538h

Function

Provides module debug disable information.

This register resets after destructive reset 15.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved	CM7_2_A...	Reserved	CM7_0_A...	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
Reset	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved	Reserved	Reserved	VDD2P5_...	VDD1P1_...	Reserved	Reserved	Reserved	Reserved	Reserved	CM7_2_A...	CM7_1_A...	CM7_0_A...	CM7_2_A...	CM7_1_A...	CM7_0_A...
Reset	0	0	0	1	1	0	0	0	0	0	1	1	1	1	1	1

Fields

Field	Function
31 —	Reserved
30 CM7_2_AHBS_ALARM_EN	Cortex-M7_2 AHBS Alarm Enable Specifies whether the Cortex-M7_2 AHBS interface IAHB gasket reported an alarm. Enables fault monitoring at FCCU NCF 1 in case of a Cortex-M7_2 AHBS interface IAHB Gasket monitor alarm. 0b - No 1b - Yes
29 —	Reserved
28 CM7_0_AHBS_ALARM_EN	Cortex-M7_0 AHBS Alarm Enable Specifies whether the Cortex-M7_0 AHBS interface IAHB gasket reported an alarm. Enables fault monitoring at FCCU NCF 1 in case of a Cortex-M7_0 AHBS interface IAHB Gasket monitor alarm. 0b - No 1b - Yes
27 —	Reserved
26 —	Reserved
25 —	Reserved
24 —	Reserved
23 —	Reserved
22 —	Reserved
21 —	Reserved
20 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
19 —	Reserved
18 —	Reserved
17 —	Reserved
16 —	Reserved
15 —	Reserved
14 —	Reserved
13 —	Reserved
12 VDD2P5_GNG2_ERR_EN	VDD2P5 Go Nogo Error Enable Enables bit for enabling the fault monitoring at FCCU NCF 4 for the fault: Go/Nogo for VDD_HV_FLA (triple bond) going to FXOSC and PLL. 0b - Go indication referring to the supply being clean. 1b - No go indication referring to the supply being unclean and a fault in double bond connection or its routing within the chip.
11 VDD1P1_GNG2_ERR_EN	VDD1P1 Go Nogo Error Enable Enable bit for enabling the fault monitoring at FCCU NCF 4 for the fault: Go/Nogo indicator for VDD1PD1 (triple bond) supply going to PLL. 0b - Go indication referring to the supply being clean. 1b - No go indication referring to the supply being unclean and a fault in double bond connection or its routing within the chip.
10 —	Reserved
9 —	Reserved
8	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
7 —	Reserved
6 —	Reserved
5 CM7_2_AHBP_ALARM_EN	Cortex-M7_2 AHBP Alarm Enable Specifies whether the Cortex-M7_2 AHBP interface IAHB gasket reported an alarm. If this field = 1, the gasket reports a monitor alarm. The field enables fault monitoring at FCCU NCF 1 in case of a Cortex-M7_2 AHBP interface IAHB gasket monitor alarm. 0b - No 1b - Yes
4 CM7_1_AHBP_ALARM_EN	Cortex-M7_1 AHBP Alarm Enable Specifies whether the Cortex-M7_1 AHBP interface IAHB gasket reported an alarm. If this field = 1, the gasket reports a monitor alarm. The field enables fault monitoring at FCCU NCF 1 in case of a Cortex-M7_1 AHBP interface IAHB gasket monitor alarm. 0b - No 1b - Yes
3 CM7_0_AHBP_ALARM_EN	Cortex-M7_0 AHBP Alarm Enable Specifies whether the Cortex-M7_0 AHBP interface IAHB gasket reported an alarm. If this field = 1, the gasket reports a monitor alarm. The field enables fault monitoring at FCCU NCF 1 in case of a Cortex-M7_0 AHBP interface IAHB gasket monitor alarm. 0b - No 1b - Yes
2 CM7_2_AHBM_ALARM_EN	Cortex-M7_2 AHBM Alarm Enable Specifies whether the Cortex-M7_2 AHBM interface IAHB gasket reported an alarm. If this field = 1, the gasket reports a monitor alarm. The field enables fault monitoring at FCCU NCF 1 in case of a Cortex-M7_2 AHBM interface IAHB gasket monitor alarm. 0b - No 1b - Yes
1	Cortex-M7_1 AHBM Alarm Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
CM7_1_AHBM_ALARM_EN	Specifies whether the Cortex-M7_1 AHBM interface IAHB gasket reported an alarm. If this field = 1, the gasket reports a monitor alarm. The field enables fault monitoring at FCCU NCF 1 in case of a Cortex-M7_1 AHBM interface IAHB gasket monitor alarm. 0b - No 1b - Yes
0 CM7_0_AHBM_ALARM_EN	Cortex-M7_0 AHBM Alarm Enable Specifies whether the Cortex-M7_0 AHBM interface IAHB gasket reported an alarm. If this field = 1, the gasket reports a monitor alarm. The field enables fault monitoring at FCCU NCF 1 in case of a Cortex-M7_0 AHBM interface IAHB gasket monitor alarm. 0b - No 1b - Yes

38.2.44 Read Write GPR On Destructive Reset 16 (DCMRWD16)

Offset

Register	Offset
DCMRWD16	53Ch

Function

This is a readable and writable general purpose register which gets reset on destructive reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved	Reserved	CM7_3_D...	CM7_3_D...	CM7_3_I...	CM7_3_I...	CM7_3_I...	CM7_3_D...	CM7_3_D...	CM7_3_A...	CM7_3_A...	Reserved	AES_ACC...	AES_ACC...	ACE_FEE...	ACE_RES...
Reset	0	0	1	1	1	1	1	1	1	1	1	0	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved	CM7_3_A...	HSE_AES...	CM7_3_A...	CM7_3_A...	MAC2_RD...	MAC2_GS...	Reserved	PERIPH_...	CM7_2_R...	CM7_2_R...	CM7_3_L...	Reserved			
Reset	0	1	1	1	1	1	1	0	1	1	1	1	0	0	0	0

Fields

Field	Function
31 —	Reserved
30 —	Reserved
29 CM7_3_DTCM1_ECC_ERR_EN	Enables bit for enabling the fault monitoring at FCCU NCF 2 for the fault: Uncorrectable ECC error from CM7_3 Data TCM memory block 1. This uncorrectable ECC error consists of multi-bit data ECC error and address ECC error. The CM7_3 Data TCM physically consists of two blocks. 0b - Uncorrectable ECC error detection at FCCU not enabled. 1b - Uncorrectable ECC error detection enabled at FCCU.
28 CM7_3_DTCM0_ECC_ERR_EN	Enables bit for enabling the fault monitoring at FCCU NCF 2 for the fault: Uncorrectable ECC error from CM7_3 Data TCM memory block 0. This uncorrectable ECC error consists of multi-bit data ECC error and address ECC error. The CM7_3 Data TCM physically consists of two blocks. 0b - Uncorrectable ECC error detection at FCCU not enabled. 1b - Uncorrectable ECC error detection enabled at FCCU.
27 CM7_3_ITCM_ECC_ERR_EN	Enables bit for enabling the fault monitoring at FCCU NCF 2 for the fault: Uncorrectable ECC error from CM7_3 Instruction TCM memory. This uncorrectable ECC error consists of multi-bit data ECC error and address ECC error. 0b - Uncorrectable ECC error detection at FCCU not enabled. 1b - Uncorrectable ECC error detection enabled at FCCU.
26 CM7_3_ICTAG_ECC_ERR_EN	Enables bit for enabling the fault monitoring at FCCU NCF 2 for the fault: Multi bit ECC error from CM7_3 ICache tag memory. 0b - No multi-bit ECC error. 1b - Multi-bit ECC error.
25 CM7_3_ICDATA_ECC_ERR_EN	Enables bit for enabling the fault monitoring at FCCU NCF 2 for the fault: Multi bit ECC error from CM7_3 ICache data memory 0b - No multi-bit ECC error. 1b - Multi-bit ECC error.
24 CM7_3_DCTAG_ECC_ERR_EN	Enables bit for enabling the fault monitoring at FCCU NCF 2 for the fault: Multi bit ECC error from CM7_3 DCache tag memory 0b - No multi-bit ECC error. 1b - Multi-bit ECC error.
23 CM7_3_DCDATA_ECC_ERR_EN	Enables bit for enabling the fault monitoring at FCCU NCF 2 for the fault: Multi bit ECC error from CM7_3 DCache data memory. 0b - No multi-bit ECC error. 1b - Multi-bit ECC error.

Table continues on the next page...

Table continued from the previous page...

Field	Function
22 CM7_3_AHBP_ALARM_EN	Enables bit for enabling the fault monitoring at FCCU NCF 1 for the fault: CM7_3 AHBP interface IAHB Gasket monitor alarm. 0b - No alarm reported from CM7_3 AHBP interface IAHB gasket. 1b - Monitor alarm reported from CM7_3 AHBP interface IAHB gasket.
21 CM7_3_AHBM_ALARM_EN	Enables bit for enabling the fault monitoring at FCCU NCF 1 for the fault: CM7_3 AHBM interface IAHB Gasket monitor alarm. 0b - No alarm reported from CM7_3 AHBP interface IAHB gasket. 1b - Monitor alarm reported from CM7_3 AHBP interface IAHB gasket.
20 —	Reserved
19 AES_ACCEL_GSKT_ALARM_EN	Enables bit for enabling the fault monitoring at FCCU NCF 1 for the fault: AES ACCEL IAHB Gasket monitor alarm. 0b - No alarm reported from AES ACCEL IAHB gasket. 1b - Monitor alarm reported from AES ACCEL IAHB gasket.
18 AES_ACCEL_AXBS_ALARM_EN	Enables bit for enabling the fault monitoring at FCCU NCF 1 for the fault: AES_ACCEL AXBS_Lite safety alarm. 0b - No safety alarm indicated by AES_ACCEL AXBS_Lite. 1b - Safety alarm indicated by AES_ACCEL AXBS_Lite.
17 ACE_FEED_RDATA_EDC_ERR_EN	Enables bit for enabling the fault monitoring at FCCU NCF 1 for the fault: Integrity error on ACE ACCEL FEED DMA master port read data for safety. 0b - No integrity error reported on ACE ACCEL FEED DMA master port read data. 1b - Integrity error reported on ACE ACCEL FEED DMA master port read data.
16 ACE_RESULT_RDATA_EDC_ERR_EN	Enables bit for enabling the fault monitoring at FCCU NCF 1 for the fault: Integrity error on ACE ACCEL RESULT DMA master port read data for safety. 0b - No integrity error reported on ACE ACCEL RESULT DMA master port read data. 1b - Integrity error reported on ACE ACCEL RESULT DMA master port read data.
15 —	Reserved
14 CM7_3_AHBS_ALARM_EN	Enables bit for enabling the fault monitoring at FCCU NCF 1 for the fault: CM7_3 AHBS interface IAHB Gasket monitor alarm. 0b - No alarm reported from CM7_3 AHBS interface IAHB gasket. 1b - Monitor alarm reported from CM7_3 AHBS interface IAHB gasket.
13	Enables bit for enabling the fault monitoring at FCCU NCF 1 for the fault: HSE_AES_ACCEL AXBS_Lite safety alarm.

Table continues on the next page...

Table continued from the previous page...

Field	Function
HSE_AES_ACCEL_AXBS_ALARM_EN	0b - No safety alarm indicated by HSE_AES_ACCEL AXBS_Lite. 1b - Safety alarm indicated by HSE_AES_ACCEL AXBS_Lite.
12 CM7_3_AHBM_RDATA_EDC_ERR_EN	Enables bit for enabling the fault monitoring at FCCU NCF 1 for the fault: Integrity error on CM7_3 main read data for safety. 0b - No integrity error reported on CM7_3 main read data. 1b - Integrity error reported on CM7_3 main read data.
11 CM7_3_AHBP_RDATA_EDC_ERR_EN	Enables bit for enabling the fault monitoring at FCCU NCF 1 for the fault: Integrity error on CM7_3 peripheral read data for safety. 0b - No integrity error reported on CM7_3 peripheral read data. 1b - Integrity error reported on CM7_3 peripheral read data.
10 MAC2_RDATA_EDC_ERR_EN	Enables bit for enabling the fault monitoring at FCCU NCF 1 for the fault: Integrity(EDC) error on MAC2 read data for safety. 0b - No integrity error reported on MAC2 read data. 1b - Integrity error reported on MAC2 read data.
9 MAC2_GSKT_ALARM_EN	Enables bit for enabling the fault monitoring at FCCU NCF 1 for the fault: MAC2 IAHB gasket alarm. 0b - No alarm indicated by MAC2 IAHB gasket. 1b - Alarm indicated by MAC2 IAHB gasket.
8 —	Reserved
7 PERIPH_AXBS_S3_GSKT_ALARM_EN	Enables bit for enabling the fault monitoring at FCCU NCF 1 for the fault: Peripheral AXBS bridge S3 IAHB gasket alarm. 0b - No alarm indicated by Peripheral AXBS bridge S3 IAHB gasket alarm. 1b - Alarm indicated by Peripheral AXBS bridge S3 IAHB gasket alarm.
6 CM7_2_RCCU2_ALARM_EN	Enables bit for enabling the fault monitoring at FCCU NCF 0 for the fault: Cortex M7 cores (CM7_2 and CM7_2_checker core) redundant lockstep error. 0b - No lockstep alarm reported by redundant RCCU. 1b - Lockstep alarm reported by redundant RCCU.
5 CM7_2_RCCU1_ALARM_EN	Enables bit for enabling the fault monitoring at FCCU NCF 0 for the fault: Cortex M7 cores (CM7_2 and CM7_2_checker core) lockstep error. 0b - No lockstep alarm reported by RCCU. 1b - Lockstep alarm reported by RCCU.
4	Enables bit for enabling the fault monitoring at FCCU NCF 0 for the fault: CM7_3 core lockup. 0b - CM7_3 core not in lockup state.

Table continues on the next page...

Table continued from the previous page...

Field	Function
CM7_3_LOCKUP_EN	1b - CM7_3 core in lockup state.
3-0 —	Reserved

38.2.45 Read Write GPR On Destructive Reset 17 (DCMRWD17)

Offset

Register	Offset
DCMRWD17	540h

Function
This is a readable and writable general purpose register which gets reset on destructive reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved					PF1_1_C...	PF1_0_C...	PF0_1_C...	PF0_0_C...	PF1_1_C...	PF1_0_C...	PF0_1_C...	PF0_0_C...	PRAM3_F...	PRAM3_E...	FLASH1_...
Reset	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved	FLASH1_...	FLASH1_...	FLASH1_...	FLASH1_...	PF1_1_D...	PF1_1_C...	PF1_0_D...	PF1_0_C...	AES_RES...	AES_FEE...	AES_KP_...	AES_RES...	AES_RES...	AES_FEE...	AES_FEE...
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-27 —	Reserved
26 PF1_1_CHK_CMP_EN	Enable bit for enabling the fault monitoring at FCCU NCF 3 for the fault: PFLASH3 checker(redundant) safety comparator. 0b - No comparator error by PFLASH3. 1b - Comparator error by PFLASH3.

Table continues on the next page...

Table continued from the previous page...

Field	Function
25 PF1_0_CHK_CMP_EN	Enable bit for enabling the fault monitoring at FCCU NCF 3 for the fault: PFLASH2 checker(redundant) safety comparator. 0b - No comparator error by PFLASH2. 1b - Comparator error by PFLASH2.
24 PF0_1_CHK_CMP_EN	Enable bit for enabling the fault monitoring at FCCU NCF 3 for the fault: PFLASH1 checker(redundant) safety comparator. 0b - No comparator error by PFLASH1. 1b - Comparator error by PFLASH1.
23 PF0_0_CHK_CMP_EN	Enable bit for enabling the fault monitoring at FCCU NCF 3 for the fault: PFLASH0 checker(redundant) safety comparator. 0b - No comparator error by PFLASH0. 1b - Comparator error by PFLASH0.
22 PF1_1_CMP_EN	Enable bit for enabling the fault monitoring at FCCU NCF 3 for the fault: PFLASH3 safety comparator. 0b - No comparator error by PFLASH3. 1b - Comparator error by PFLASH3.
21 PF1_0_CMP_EN	Enable bit for enabling the fault monitoring at FCCU NCF 3 for the fault: PFLASH2 safety comparator. 0b - No comparator error by PFLASH2. 1b - Comparator error by PFLASH2.
20 PF0_1_CMP_EN	Enable bit for enabling the fault monitoring at FCCU NCF 3 for the fault: PFLASH1 safety comparator. 0b - No comparator error by PFLASH1. 1b - Comparator error by PFLASH1.
19 PF0_0_CMP_EN	Enable bit for enabling the fault monitoring at FCCU NCF 3 for the fault: PFLASH0 safety comparator. 0b - No comparator error by PFLASH0. 1b - Comparator error by PFLASH0.
18 PRAM3_FCCU_ALARM_EN	Enable bit for enabling the fault monitoring at FCCU NCF 2 for the fault: PRAM3 safety alarm. This alarm is set on faulty SRAM3 read or read modify error. 0b - No safety alarm indicated by PRAM3. 1b - Safety alarm indicated by PRAM3.
17 PRAM3_ECC_ERR_EN	Enable bit for enabling the fault monitoring at FCCU NCF 2 for the fault: Multi bit ECC error from SRAM3. 0b - No multi-bit ECC error. 1b - Multi-bit ECC error.
16	Enable bit for enabling the fault monitoring at FCCU NCF 3 for the fault: ECC error from Flash Controller1. This alarm indicates that the flash controller1 detected an error in the address ECC manipulation logic through EDC.

Table continues on the next page...

Table continued from the previous page...

Field	Function
FLASH1_ECC_ERR_EN	0b - No ECC error from flash controller1. 1b - ECC error from flash controller1.
15 —	Reserved
14 FLASH1_RST_ERR_EN	Enable bit for enabling the fault monitoring at FCCU NCF 3 for the fault: Flash1 reset error . This error indication is set when flash1 encounters errors during its reset reads. 0b - No flash1 reset error indicated. 1b - Flash1 reset error indicated.
13 FLASH1_REF_ERR_EN	Enable bit for enabling the fault monitoring at FCCU NCF 3 for the fault: Flash1 reference current loss or read voltage error while prevoius read. 0b - No reference current loss or read voltage error while previous read. 1b - Reference current loss or read voltage error while previous read.
12 FLASH1_ADDR_ENC_ERR_EN	Enable bit for enabling the fault monitoring at FCCU NCF 3 for the fault: Flash1 address encode error. In address decoding, if multiple or no address line is selected, FMU reports address encode error. 0b - No address encode error in flash1. 1b - Address encode error in flash1.
11 FLASH1_EDC_ERR_EN	Enable bit for enabling the fault monitoring at FCCU NCF 3 for the fault: Flash1 ECC correction error through EDC reported by FMU. 0b - No EDC after ECC error reported in flash1. 1b - EDC after ECC error reported in flash1.
10 PF1_1_DATA_ECC_ERR_EN	Enable bit for enabling the fault monitoring at FCCU NCF 3 for the fault: Flash3 data ECC uncorrectable error. The path is from FMU to PFLASH controller to ERM to FCCU. 0b - No uncorrectable error reported in flash controller port 3 data memory by FMU. 1b - Uncorrectable error reported in flash controller port 3 data memory by FMU.
9 PF1_1_CODE_ECC_ERR_EN	Enable bit for enabling the fault monitoring at FCCU NCF 3 for the fault: Flash3 code ECC uncorrectable error. The path is from FMU to PFLASH controller to ERM to FCCU. 0b - No uncorrectable error reported in flash controller port 3 code memory by FMU. 1b - Uncorrectable error reported in flash controller port 3 code memory by FMU.
8 PF1_0_DATA_ECC_ERR_EN	Enable bit for enabling the fault monitoring at FCCU NCF 3 for the fault: Flash2 data ECC uncorrectable error. The path is from FMU to PFLASH controller to ERM to FCCU. 0b - No uncorrectable error reported in flash controller port 2 data memory by FMU. 1b - Uncorrectable error reported in flash controller port 2 data memory by FMU.
7	Enable bit for enabling the fault monitoring at FCCU NCF 3 for the fault: Flash2 code ECC uncorrectable error. The path is from FMU to PFLASH controller to ERM to FCCU.

Table continues on the next page...

Table continued from the previous page...

Field	Function
PF1_0_CODE_ECC_ERR_EN	0b - No uncorrectable error reported in flash controller port 2 code memory by FMU. 1b - Uncorrectable error reported in flash controller port 2 code memory by FMU.
6 AES_RESULT_DID_SAFETY_ERR_EN	Enables bit for enabling the fault monitoring at FCCU NCF 2 for the fault: AES RESULT DMA DID error. 0b - AES RESULT DMA DID error not enabled. 1b - AES RESULT DMA DID error enabled.
5 AES_FEED_DID_SAFETY_ERR_EN	Enables bit for enabling the fault monitoring at FCCU NCF 2 for the fault: AES FEED DMA DID error. 0b - AES FEED DMA DID error not enabled. 1b - AES FEED DMA DID error enabled.
4 AES_KP_CRC_SAFETY_ERR_EN	Enables bit for enabling the fault monitoring at FCCU NCF 2 for the fault: AES Key Property CRC Safety Error. 0b - AES key-property CRC safety error not enabled. 1b - AES key-property CRC safety error enabled.
3 AES_RESULT_DMA_TCD_ADDR_ECC_ERR_EN	Enables bit for enabling the fault monitoring at FCCU NCF 2 for the fault: AES ACCEL RESULT DMA_TCD address ECC error. 0b - No address error reported in AES ACCEL RESULT DMA_TCD memory. 1b - Address error reported in AES ACCEL RESULT DMA_TCD memory.
2 AES_RESULT_DMA_TCD_ECC_ERR_EN	Enables bit for enabling the fault monitoring at FCCU NCF 2 for the fault: AES ACCEL RESULT DMA_TCD memory uncorrectable ECC error. 0b - No uncorrectable error reported in AES ACCEL RESULT DMA_TCD memory. 1b - Uncorrectable error reported in AES ACCEL RESULT DMA_TCD memory.
1 AES_FEED_DMA_TCD_ADDR_ECC_ERR_EN	Enables bit for enabling the fault monitoring at FCCU NCF 2 for the fault: AES ACCEL FEED DMA_TCD address ECC error. 0b - No address error reported in AES ACCEL FEED DMA_TCD memory. 1b - Address error reported in AES ACCEL FEED DMA_TCD memory.
0 AES_FEED_DMA_TCD_ECC_ERR_EN	Enables bit for enabling the fault monitoring at FCCU NCF 2 for the fault: AES ACCEL FEED DMA_TCD memory uncorrectable ECC error. 0b - No uncorrectable error reported in AES ACCEL FEED DMA_TCD memory. 1b - Uncorrectable error reported in AES ACCEL FEED DMA_TCD memory.

38.2.46 Read Write GPR On Destructive Reset 19 (DCMRWD19)

Offset

Register	Offset
DCMRWD19	548h

Function

This is a readable and writable general purpose register which gets reset on destructive reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SAI1_DB...	SAI0_DB...	FLEXCAN...	FLEXCAN...	FLEXCAN...	FLEXCAN...	FLEXCAN...	FLEXCAN...	FLEXIO_...	LPI2C1_...	LPI2C0_...	LPSPI5_...	LPSPI4_...	LPSPI3_...	LPSPI2_...	LPSPI1_...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LPSPI0_...	PIT2_DB...	PIT1_DB...	PIT0_DB...	STM1_DB...	STM0_DB...	SWT1_DB...	SWT0_DB...	RTC_DBG...	eMIOS2_...	eMIOS1_...	eMIOS0_...	LCU1_DB...	LCU0_DB...	FCCU_DB...	EDMA_DB...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SAI1_DBG_DIS_CM7_3	Specifies whether SAI1 enters Debug mode or remains functional and unimpacted when the Cortex-M7_3 core enters Debug mode. Write 1 to this field to disable debug. 0b - SAI1 enters debug mode when CM7_3 enters debug mode. 1b - SAI1 remains functional and is not impacted when CM7_3 enters debug mode.
30 SAI0_DBG_DIS_CM7_3	Specifies whether SAI0 enters Debug mode or remains functional and unimpacted when the Cortex-M7_3 core enters Debug mode. Write 1 to this field to disable debug. 0b - SAI0 enters debug mode when CM7_3 enters debug mode. 1b - SAI0 remains functional and is not impacted when CM7_3 enters debug mode.
29 FLEXCAN5_DBG_DIS_CM7_3	FlexCAN5 Debug Disable For Cortex-M7_3 Specifies whether FlexCAN5 enters Debug mode or remains functional and unimpacted when the Cortex-M7_3 core enters Debug mode. Write 1 to this field to disable debug. 0b - FlexCAN5 enters debug mode when CM7_3 enters debug mode.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - FlexCAN5 remains functional and is not impacted when CM7_3 enters debug mode.
28 FLEXCAN4_DBG_DIS_CM7_3	FlexCAN4 Debug Disable For Cortex-M7_3 Specifies whether FlexCAN4 enters Debug mode or remains functional and unimpacted when the Cortex-M7_3 core enters Debug mode. Write 1 to this field to disable debug. 0b - FlexCAN4 enters debug mode when CM7_3 enters debug mode. 1b - FlexCAN4 remains functional and is not impacted when CM7_3 enters debug mode.
27 FLEXCAN3_DBG_DIS_CM7_3	FlexCAN3 Debug Disable For Cortex-M7_3 Specifies whether FlexCAN3 enters Debug mode or remains functional and unimpacted when the Cortex-M7_3 core enters Debug mode. Write 1 to this field to disable debug. 0b - FlexCAN3 enters debug mode when CM7_3 enters debug mode. 1b - FlexCAN3 remains functional and is not impacted when CM7_3 enters debug mode.
26 FLEXCAN2_DBG_DIS_CM7_3	FlexCAN2 Debug Disable For Cortex-M7_3 Specifies whether FlexCAN2 enters Debug mode or remains functional and unimpacted when the Cortex-M7_3 core enters Debug mode. Write 1 to this field to disable debug. 0b - FlexCAN2 enters debug mode when CM7_3 enters debug mode. 1b - FlexCAN2 remains functional and is not impacted when CM7_3 enters debug mode.
25 FLEXCAN1_DBG_DIS_CM7_3	FlexCAN1 Debug Disable For Cortex-M7_3 Specifies whether FlexCAN1 enters Debug mode or remains functional and unimpacted when the Cortex-M7_3 core enters Debug mode. Write 1 to this field to disable debug. 0b - FlexCAN1 enters debug mode when CM7_3 enters debug mode. 1b - FlexCAN1 remains functional and is not impacted when CM7_3 enters debug mode.
24 FLEXCAN0_DBG_DIS_CM7_3	FlexCAN0 Debug Disable For Cortex-M7_3 Specifies whether FlexCAN0 enters Debug mode or remains functional and unimpacted when the Cortex-M7_3 core enters Debug mode. Write 1 to this field to disable debug. 0b - FlexCAN0 enters debug mode when CM7_3 enters debug mode. 1b - FlexCAN0 remains functional and is not impacted when CM7_3 enters debug mode.
23	FlexIO Debug Disable For Cortex-M7_3 Specifies whether FlexIO enters Debug mode or remains functional and unimpacted when the Cortex-M7_3 core enters Debug mode.

Table continues on the next page...

Table continued from the previous page...

Field	Function
FLEXIO_DBG_DIS_CM7_3	Write 1 to this field to disable debug. 0b - FlexIO enters debug mode when CM7_3 enters debug mode. 1b - FlexIO remains functional and is not impacted when CM7_3 enters debug mode.
22 LPI2C1_DBG_DIS_CM7_3	LPI2C1 Debug Disable For Cortex-M7_3 Specifies whether LPI2C1 enters Debug mode or remains functional and unimpacted when the Cortex-M7_3 core enters Debug mode. Write 1 to this field to disable debug. 0b - LPI2C1 enters debug mode when CM7_3 enters debug mode. 1b - LPI2C1 remains functional and is not impacted when CM7_3 enters debug mode.
21 LPI2C0_DBG_DIS_CM7_3	LPI2C0 Debug Disable For Cortex-M7_3 Specifies whether LPI2C0 enters Debug mode or remains functional and unimpacted when the Cortex-M7_3 core enters Debug mode. Write 1 to this field to disable debug. 0b - LPI2C0 enters debug mode when CM7_3 enters debug mode. 1b - LPI2C0 remains functional and is not impacted when CM7_3 enters debug mode.
20 LPSPI5_DBG_DIS_CM7_3	LPSPI5 Debug Disable For Cortex-M7_3 Specifies whether LPSPI5 enters Debug mode or remains functional and unimpacted when the Cortex-M7_3 core enters Debug mode. Write 1 to this field to disable debug. 0b - LPSPI5 enters debug mode when CM7_3 enters debug mode. 1b - LPSPI5 remains functional and is not impacted when CM7_3 enters debug mode.
19 LPSPI4_DBG_DIS_CM7_3	LPSPI4 Debug Disable For Cortex-M7_3 Specifies whether LPSPI4 enters Debug mode or remains functional and unimpacted when the Cortex-M7_3 core enters Debug mode. Write 1 to this field to disable debug. 0b - LPSPI4 enters debug mode when CM7_3 enters debug mode. 1b - LPSPI4 remains functional and is not impacted when CM7_3 enters debug mode.
18 LPSPI3_DBG_DIS_CM7_3	LPSPI3 Debug Disable For Cortex-M7_3 Specifies whether LPSPI3 enters Debug mode or remains functional and unimpacted when the Cortex-M7_3 core enters Debug mode. Write 1 to this field to disable debug. 0b - LPSPI3 enters debug mode when CM7_3 enters debug mode. 1b - LPSPI3 remains functional and is not impacted when CM7_3 enters debug mode.

Table continues on the next page...

Table continued from the previous page...

Field	Function
17 LPSPI2_DBG_DIS_CM7_3	LPSPI2 Debug Disable For Cortex-M7_3 Specifies whether LPSPI2 enters Debug mode or remains functional and unimpacted when the Cortex-M7_3 core enters Debug mode. Write 1 to this field to disable debug. 0b - LPSPI2 enters debug mode when CM7_3 enters debug mode. 1b - LPSPI2 remains functional and is not impacted when CM7_3 enters debug mode.
16 LPSPI1_DBG_DIS_CM7_3	LPSPI1 Debug Disable For Cortex-M7_3 Specifies whether LPSPI1 enters Debug mode or remains functional and unimpacted when the Cortex-M7_3 core enters Debug mode. Write 1 to this field to disable debug. 0b - LPSPI1 enters debug mode when CM7_3 enters debug mode. 1b - LPSPI1 remains functional and is not impacted when CM7_3 enters debug mode.
15 LPSPI0_DBG_DIS_CM7_3	LPSPI0 Debug Disable For Cortex-M7_3 Specifies whether LPSPI0 enters Debug mode or remains functional and unimpacted when the Cortex-M7_3 core enters Debug mode. Write 1 to this field to disable debug. 0b - LPSPI0 enters debug mode when CM7_3 enters debug mode. 1b - LPSPI0 remains functional and is not impacted when CM7_3 enters debug mode.
14 PIT2_DBG_DIS_CM7_3	PIT2 Debug Disable For Cortex-M7_3 Specifies whether PIT2 enters Debug mode or remains functional and unimpacted when the Cortex-M7_3 core enters Debug mode. Write 1 to this field to disable debug. 0b - PIT2 enters debug mode when CM7_3 enters debug mode. 1b - PIT2 remains functional and is not impacted when CM7_3 enters debug mode.
13 PIT1_DBG_DIS_CM7_3	PIT1 Debug Disable For Cortex-M7_3 Specifies whether PIT1 enters Debug mode or remains functional and unimpacted when the Cortex-M7_3 core enters Debug mode. Write 1 to this field to disable debug. 0b - PIT1 enters debug mode when CM7_3 enters debug mode. 1b - PIT1 remains functional and is not impacted when CM7_3 enters debug mode.
12 PIT0_DBG_DIS_CM7_3	PIT0 Debug Disable For Cortex-M7_3 Specifies whether PIT0 enters Debug mode or remains functional and unimpacted when the Cortex-M7_3 core enters Debug mode. Write 1 to this field to disable debug.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - PIT0 enters debug mode when CM7_3 enters debug mode. 1b - PIT0 remains functional and is not impacted when CM7_3 enters debug mode.
11 STM1_DBG_DIS_CM7_3	STM1 Debug Disable For Cortex-M7_3 Specifies whether STM1 enters Debug mode or remains functional and unimpacted when the Cortex-M7_3 core enters Debug mode. Write 1 to this field to disable debug. 0b - STM1 enters debug mode when CM7_3 enters debug mode. 1b - STM1 remains functional and is not impacted when CM7_3 enters debug mode.
10 STM0_DBG_DIS_CM7_3	STM0 Debug Disable For Cortex-M7_3 Specifies whether STM0 enters Debug mode or remains functional and unimpacted when the Cortex-M7_3 core enters Debug mode. Write 1 to this field to disable debug. 0b - STM0 enters debug mode when CM7_3 enters debug mode. 1b - STM0 remains functional and is not impacted when CM7_3 enters debug mode.
9 SWT1_DBG_DIS_CM7_3	SWT1 Debug Disable For Cortex-M7_3 Specifies whether SWT1 enters Debug mode or remains functional and unimpacted when the Cortex-M7_3 core enters Debug mode. Write 1 to this field to disable debug. 0b - SWT1 enters debug mode when CM7_3 enters debug mode. 1b - SWT1 remains functional and is not impacted when CM7_3 enters debug mode.
8 SWT0_DBG_DIS_CM7_3	SWT0 Debug Disable For Cortex-M7_3 Specifies whether SWT0 enters Debug mode or remains functional and unimpacted when the Cortex-M7_3 core enters Debug mode. Write 1 to this field to disable debug. 0b - SWT0 enters debug mode when CM7_3 enters debug mode. 1b - SWT0 remains functional and is not impacted when CM7_3 enters debug mode.
7 RTC_DBG_DIS_CM7_3	RTC Debug Disable For Cortex-M7_3 Specifies whether RTC enters Debug mode or remains functional and unimpacted when the Cortex-M7_3 core enters Debug mode. Write 1 to this field to disable debug. 0b - RTC enters debug mode when CM7_3 enters debug mode. 1b - RTC remains functional and is not impacted when CM7_3 enters debug mode.

Table continues on the next page...

Table continued from the previous page...

Field	Function
6 eMIOS2_DBG_DIS_CM7_3	Specifies whether eMIOS2 enters Debug mode or remains functional and unimpacted when the Cortex-M7_3 core enters Debug mode. Write 1 to this field to disable debug. 0b - eMIOS2 enters debug mode when CM7_3 enters debug mode. 1b - eMIOS2 remains functional and is not impacted when CM7_3 enters debug mode.
5 eMIOS1_DBG_DIS_CM7_3	Specifies whether eMIOS1 enters Debug mode or remains functional and unimpacted when the Cortex-M7_3 core enters Debug mode. Write 1 to this field to disable debug. 0b - eMIOS1 enters debug mode when CM7_3 enters debug mode. 1b - eMIOS1 remains functional and is not impacted when CM7_3 enters debug mode.
4 eMIOS0_DBG_DIS_CM7_3	eMIOS0 Debug Disable For Cortex-M7_3 Specifies whether eMIOS0 enters Debug mode or remains functional and unimpacted when the Cortex-M7_3 core enters Debug mode. Write 1 to this field to disable debug. 0b - eMIOS0 enters debug mode when CM7_3 enters debug mode. 1b - eMIOS0 remains functional and is not impacted when CM7_3 enters debug mode.
3 LCU1_DBG_DIS_CM7_3	LCU1 Debug Disable For Cortex-M7_3 Specifies whether LCU1 enters Debug mode or remains functional and unimpacted when the Cortex-M7_3 core enters Debug mode. If this field = 0, LCU1 enters Debug mode, and if this field = 1, LCU1 remains functional. Write 1 to this field to disable debug. 0b - LCU1 enters debug mode when CM7_3 enters debug mode. 1b - LCU1 remains functional and is not impacted when CM7_3 enters debug mode.
2 LCU0_DBG_DIS_CM7_3	LCU0 Debug Disable For Cortex-M7_3 Specifies whether LCU0 enters Debug mode or remains functional and unimpacted when the Cortex-M7_3 core enters Debug mode. If this field = 0, LCU0 enters Debug mode, and if this field = 1, LCU0 remains functional. Write 1 to this field to disable debug. 0b - LCU0 enters debug mode when CM7_3 enters debug mode. 1b - LCU0 remains functional and is not impacted when CM7_3 enters debug mode.
1 FCCU_DBG_DIS_CM7_3	FCCU Debug Disable For Cortex-M7_3 Specifies whether FCCU enters Debug mode or remains functional and unimpacted when the Cortex-M7_3 core enters Debug mode. If this field = 0, FCCU enters Debug mode, and if this field = 1, FCCU remains functional. Write 1 to this field to disable debug.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - FCCU enters debug mode when CM7_3 enters debug mode. 1b - FCCU remains functional and is not impacted when CM7_3 enters debug mode.
0 EDMA_DBG_DIS_CM7_3	EDMA Debug Disable For Cortex-M7_3 Specifies whether eDMA enters Debug mode or remains functional and unimpacted when the Cortex-M7_3 core enters Debug mode. If this field = 0, eDMA enters Debug mode, and if this field = 1, eDMA remains functional. Write 1 to this field to disable debug. 0b - EDMA enters debug mode when CM7_3 enters debug mode. 1b - EDMA remains functional and is not impacted when CM7_3 enters debug mode.

38.2.47 Read Write GPR On Destructive Reset 20 (DCMRWD20)

Offset

Register	Offset
DCMRWD20	54Ch

Function
This is a readable and writable general purpose register which gets reset on destructive reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved				FLEXCAN...	FLEXCAN...	FLEXCAN...	FLEXCAN...	PIT3_DB...	STM3_DB...	SWT3_DB...	Reserved				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SWT2_DB...	STM2_DB...	FLEXCAN...	FLEXCAN...	Reserved
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
27 FLEXCAN11_DBG_DIS_CM7_3	FLEXCAN11 Debug Disable Cortex-M7_3 FLEXCAN11 debug disable bit for CM7_3. Set this bit 1 to disable the debug of module. 0b - FLEXCAN11 enters debug mode when CM7_3 enters debug mode. 1b - FLEXCAN11 remains functional and is not impacted when CM7_3 enters debug mode.
26 FLEXCAN10_DBG_DIS_CM7_3	FLEXCAN10 Debug Disable Cortex-M7_3 FLEXCAN10 debug disable bit for CM7_3. Set this bit 1 to disable the debug of module. 0b - FLEXCAN10 enters debug mode when CM7_3 enters debug mode. 1b - FLEXCAN10 remains functional and is not impacted when CM7_3 enters debug mode.
25 FLEXCAN9_DBG_DIS_CM7_3	FLEXCAN9 Debug Disable Cortex-M7_3 FLEXCAN9 debug disable bit for CM7_3. Set this bit 1 to disable the debug of module. 0b - FLEXCAN9 enters debug mode when CM7_3 enters debug mode. 1b - FLEXCAN9 remains functional and is not impacted when CM7_3 enters debug mode.
24 FLEXCAN8_DBG_DIS_CM7_3	FLEXCAN8 Debug Disable Cortex-M7_3 FLEXCAN8 debug disable bit for CM7_3. Set this bit 1 to disable the debug of module. 0b - FLEXCAN8 enters debug mode when CM7_3 enters debug mode. 1b - FLEXCAN8 remains functional and is not impacted when CM7_3 enters debug mode.
23 PIT3_DBG_DIS_CM7_3	PIT3 Debug Disable Cortex-M7_3 Specifies whether PIT3 enters Debug mode or remains functional and unimpacted when the Cortex-M7_3 core enters Debug mode. Write 1 to this field to disable debug. 0b - PIT3 enters debug mode when CM7_3 enters debug mode. 1b - PIT3 remains functional and is not impacted when CM7_3 enters debug mode.
22 STM3_DBG_DIS_CM7_3	STM3 Debug Disable Cortex-M7_3 Specifies whether STM3 enters Debug mode or remains functional and unimpacted when the Cortex-M7_3 core enters Debug mode. Write 1 to this field to disable debug. 0b - STM3 enters debug mode when CM7_3 enters debug mode. 1b - STM3 remains functional and is not impacted when CM7_3 enters debug mode.
21 SWT3_DBG_DIS_CM7_3	SWT3 Debug Disable Cortex-M7_3 Specifies whether SWT3 enters Debug mode or remains functional and unimpacted when the Cortex-M7_3 core enters Debug mode. Write 1 to this field to disable debug.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - SWT3 enters debug mode when CM7_3 enters debug mode. 1b - SWT3 remains functional and is not impacted when CM7_3 enters debug mode.
20-5 —	Reserved
4 SWT2_DBG_DIS_CM7_3	SWT2 Debug Disable For Cortex-M7_3 Specifies whether SWT2 enters Debug mode or remains functional and unimpacted when the Cortex-M7_3 core enters Debug mode. Write 1 to this field to disable debug. 0b - SWT2 enters debug mode when CM7_3 enters debug mode. 1b - SWT2 remains functional and is not impacted when CM7_3 enters debug mode.
3 STM2_DBG_DIS_CM7_3	STM2 Debug Disable For Cortex-M7_3 Specifies whether STM2 enters Debug mode or remains functional and unimpacted when the Cortex-M7_3 core enters Debug mode. Write 1 to this field to disable debug. 0b - STM2 enters debug mode when CM7_3 enters debug mode. 1b - STM2 remains functional and is not impacted when CM7_3 enters debug mode.
2 FLEXCAN7_DBG_DIS_CM7_3	Specifies whether FLEXCAN7 enters Debug mode or remains functional and unimpacted when the Cortex-M7_3 core enters Debug mode. Write 1 to this field to disable debug. 0b - FLEXCAN7 enters debug mode when CM7_3 enters debug mode. 1b - FLEXCAN7 remains functional and is not impacted when CM7_3 enters debug mode.
1 FLEXCAN6_DBG_DIS_CM7_3	FlexCAN6 Debug Disable For Cortex-M7_3 Specifies whether FlexCAN6 enters Debug mode or remains functional and unimpacted when the Cortex-M7_3 core enters Debug mode. Write 1 to this field to disable debug. 0b - FlexCAN6 enters debug mode when CM7_3 enters debug mode. 1b - FlexCAN6 remains functional and is not impacted when CM7_3 enters debug mode.
0 —	Reserved

38.2.48 Read Write GPR On Functional Reset 1 (DCMRWF1)

Offset

Register	Offset
DCMRWF1	600h

Function

Contains information related to:

- Voltage dividers, LFAST clocks, and supply voltage monitoring.
- I/O configurations.

This field resets after functional reset 1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MAC_TX_...	Reserved			VDD_1_5...	VDD_HV_...	VDD_HV_...	VSS_LV_...	SUPPLY_MON_SEL			SUPPLY_...	MAC_SB_...	PMIC_PG...	Reserved	STANDBY...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	VDD_HV_...	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	MAC_CONF_SEL		FCCU_SW...	FCCU_SW...	FCCU_SW...	FCCU_SW...	CAN_TIM...	CAN_TIM...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 MAC_TX_RMII_CLK_LPBCK_EN	MAC_TX_RMII_CLK Loopback Enable Enables the MAC_TX_RMII_CLK loopback. 0b - Disables 1b - Enables
30-28 —	Reserved
27 VDD_1_5_VLT_DVDR_EN	VDD1P5 Voltage Divider Enable Enables the VDD1P5 2:1 divider for voltage measurement using the supply voltage that ADC monitors. 0b - Disables 1b - Enables
26	VDD_HV_B Voltage Divider Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
VDD_HV_B_VLT_DVDR_EN	Enables the VDD_HV_B 2:1 divider for voltage measurement by using the supply voltage that ADC monitors. 0b - Disables 1b - Enables
25 VDD_HV_A_VLT_DVDR_EN	VDD_HV_A Voltage Divider Enable Enables the VDD_HV_A 2:1 divider for voltage measurement by using supply voltage that ADC monitors. 0b - Disables 1b - Enables
24 VSS_LV_ANMUX_EN	VSS_LV Monitoring Enable Enables VSS_LV monitoring. **NOTE** You must write 1 to this field (with DCMRWF1[SUPPLY_MON_EN] = 1b0 for VSS_LV monitoring by ADC0). 0b - Disables 1b - Enables
23-21 SUPPLY_MON_SEL	Supply Monitoring Select Selects the source of voltage that ADC uses for supply monitoring. **NOTE** • The SUPPLY_MON_SEL configurations are effective only when SUPPLY_MON_EN is 1. • When SUPPLY_MON_EN is 0 and VSS_LV_ANMUX_EN is 1, VSS_LV is monitored. 000b - VDD_HV_A_DIV 001b - VDD_HV_B_DIV 010b - VDD_1.5_DIV 011b - VDD_2.5_OSC 100b - VDD1.1_PD1_HOT_POINT 101b - VDD1.1_PD1_COLD_POINT 110b - VDD1.1_PLL 111b - VDD1.1_PD0
20 SUPPLY_MON_EN	Supply Monitoring Enable Enables the ADC supply voltage monitoring. 0b - Disables 1b - Enables

Table continues on the next page...

Table continued from the previous page...

Field	Function
19 MAC_SB_END_CTRL	MAC Sideband Data Endianness Control 0b - The MAC sideband data is transferred in little-endian mode. 1b - The MAC sideband data is transferred in big-endian mode.
18 PMIC_PGOOD_HNDSHK_BYP	Controls the PMIC_PGOOD handshake with the external Power Management IC (PMIC) while standby exit. 0b - The standby exit proceeds only when the active edge (as per DCMRWF2[PGOOD_POLARITY] configuration) is detected on the PGOOD signal. 1b - The PMIC_PGOOD handshake with the PMIC is bypassed.
17 —	Reserved
16 STANDBY_IO_CONFIG	Standby I/O Configuration Controls the IO state in the standby mode. This bit needs to be written both at the standby entry as well as standby exit as per the below description. 0b - Must be written as 0 before IO configurations are done in standby entry sequence. 1b - Must be written as 1 after IO configurations are done on standby exit.
15 VDD_HV_B_IO_CTRL_LATCH	VDD_HV_B I/O Control Latch Controls the IO controls (SRC, DSE, PKE, PUS, PUE, IBE and OBE) latching in low frequency RUN mode to reduce power consumption on VDD_HV_B domain pins. The pad output path is functional as usual. **NOTE** This field must remain 0, except in FIRC 3 MHz and FIRC 187.5 kHz operation modes. 0b - VDD_HV_B domain pins function as normal. 1b - The IO controls of VDD_HV_B domain pins are latched.
14 —	Reserved
13 —	Reserved
12 —	Reserved
11 —	Reserved
10	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
9 —	Reserved
8 —	Reserved
7-6 MAC_CONF_SEL	Selects between MII and RMII mode of ethernet. 00b - MII mode 01b - RGMII mode 10b - RMII mode 11b - Reserved
5 FCCU_SW_NCF3	FCCU Software NCF 3 Specifies whether NCF 3 to FCCU is generated. For the exact FCCU slot, see the "Fault Collection and Control Unit (FCCU)" chapter. 0b - Not generated 1b - Generated
4 FCCU_SW_NCF2	FCCU Software NCF 2 Specifies whether NCF 2 to FCCU is generated. For the exact FCCU slot, see the "Fault Collection and Control Unit (FCCU)" chapter. 0b - Not generated 1b - Generated
3 FCCU_SW_NCF1	FCCU Software NCF 1 Specifies whether NCF 1 to FCCU is generated. For the exact FCCU slot, see the "Fault Collection and Control Unit (FCCU)" chapter. 0b - Not generated 1b - Generated
2 FCCU_SW_NCF0	FCCU Software NCF 0 Specifies whether NCF0 to FCCU is generated. For the exact FCCU slot, see the "Fault Collection and Control Unit (FCCU)" chapter. 0b - Not generated 1b - Generated
1 CAN_TIMESTAMP_EN	FlexCAN Timestamp Enable Enables the FlexCAN timestamping feature.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disables 1b - Enables
0 CAN_TIMESTAMP_SEL	FlexCAN Timestamp Select Selects either EMAC or STM for FlexCAN timestamping. 0b - EMAC 1b - STM0

38.2.49 Read Write GPR On Functional Reset 2 (DCMRWF2)

Offset

Register	Offset
DCMRWF2	604h

Function

Contains information related to:

- WKPU source select.
- PGOOD polarity.
- HSE_B gasket bypass.
- Bypass standby exit.

This register resets after functional reset 2.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	WKPU8_S...	WKPU45_...	WKPU27_...	WKPU18_...	WKPU15_...	WKPU14_...	WKPU0_S...	PLL1_LO...	Reserved		PGOOD_P...	Reserved	Reserved	Reserved	VIRT_WR...	HSE_GSK...
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SUPPLY2_MON_SEL			SUPPLY2...	SAI_MCL...	MAC_TX_RX_CLK_M...		PMOS_CT...	Reserved		PMC_TRI...	FIRC_TR...	DCM_SCA...	MAC2_LO...	MAC2_TX...	Reserved
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 WKPU8_SRC_SELECT	WKPU[8] Source Select Controls and specifies the source of WKPU[8]. 0b - GPIO[34] 1b - GPIO[231]
30 WKPU45_SRC_SELECT	WKPU[45] Source Select Controls and specifies the source of WKPU[45]. 0b - GPIO[89] 1b - GPIO[217]
29 WKPU27_SRC_SELECT	WKPU[27] Source Select Controls and specifies the source of WKPU[27]. 0b - GPIO[130] 1b - GPIO[233]
28 WKPU18_SRC_SELECT	WKPU[18] Source Select Controls and specifies the source of WKPU[18]. 0b - GPIO[75] 1b - GPIO[235]
27 WKPU15_SRC_SELECT	Controls the source of WKPU[15]. 0b - GPIO[6] is used as source of WKPU[15]. 1b - GPIO[227] is used as source of WKPU[15].
26 WKPU14_SRC_SELECT	Controls the source of WKPU[14]. 0b - GPIO[49] is used as source of WKPU[14]. 1b - GPIO[229] is used as source of WKPU[14].
25 WKPU0_SRC_SELECT	Controls the source of WKPU[0]. 0b - GPIO[2] is used as source of WKPU[0]. 1b - GPIO[225] is used as source of WKPU[0].
24 PLL1_LOL_RST_EN	PLL1 LOL Reset Enable Controls the functional reset or interrupt behavior of the PLL1 LOL event. 0b - PLL1 LOL event results in an interrupt 1b - PLL1 LOL event results in a reset
23-22 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
21 PGOOD_POLARITY	PGOOD Signal Edge Polarity See the PMIC documentation before performing this configuration. **NOTE** PGOOD_POLARITY configuration is valid only if PMIC_PGOOD_HNDSHK_BYP is set as 1'0b. 0b - Rising egde. PGOOD signal is considered active while it sees a transitioning edge from active low state to active high state. 1b - Falling egde. PGOOD signal is considered active while it sees a transitioning edge from active high state to active low state.
20 —	Reserved
19 —	Reserved
18 —	Reserved
17 VIRT_WRAP_IPSYNC_BYPASS	VIRT_WRAPPER IPSYNC Bypass Bypasses the VIRT_WRAPPER IPSYNC. **NOTE** The VIRT_WRAPPER IPSYNC can alternatively also be bypassed using UTEST_MISC[12]. See UTEST client description for details. 0b - Enables 1b - Bypasses
16 HSE_GSKT_BYPASS	HSE_B Gasket Bypass Enables the HSE_B IAHB gasket behavior out of Standby mode. • If you write 0 to this field, the DCF client controls the HSE_B IAHB gasket bypass configuration. The system must continue to run on FIRC, and if intended to run on PLL, a functional reset must be asserted. • If you write 1 to this field, the HSE_B IAHB gasket is bypassed out of standby. 0b - Not bypassed 1b - Bypassed
15-13 SUPPLY2_MON_SEL	Supply 2 Monitoring Select Selects the source of voltage that ADC uses for supply monitoring.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE The SUPPLY2_MON_SEL configurations are effective only when SUPPLY2_MON_EN is 1. 000b - VDD2P5_PLL2 001b - VDD1P1_PLL2 010b - Reserved 011b - Reserved
12 SUPPLY2_MON_EN	Supply 2 Monitoring Enable Enables supply voltage that ADC monitors to makes use of another ANAMUX. 0b - Disables 1b - Enables
11 SAI_MCLK2_SEL	SAI MCLK2 Select Controls the SAI_MCLK2 clk source. 0b - FXOSC is the SAI_MCLK2 clock source. 1b - PLL_AUX_PHI1 is the SAI_MCLK2 clock source.
10-9 MAC_TX_RX_CLK_MUX_BYPASS	MAC TX RX CLK MUX BYPASS Bypasses the MAC_TX_CLK and MAC_RX_CLK sources depending on DCMRWF3[15:13] configuration. 00b - The MAC_RX_CLK in the configuration if DCMRWF3[14:13] is 2'b00 is derived from MC_CGM_MUX7. The MAC_TX_CLK in the configuration if DCMRWF3[15] is 1'b0 is derived from MC_CGM_MUX8. 01b - The MAC_RX_CLK in the configuration if DCMRWF3[14:13] is 2'b00 is derived from RMII_CLK DIV2. The MAC_TX_CLK in the configuration if DCMRWF3[15] is 1'b0 is derived from RMII_CLK DIV2. 10b - The MAC_RX_CLK in the configuration if DCMRWF3[14:13] is 2'b00 is derived from RMII_CLK DIV20. The MAC_TX_CLK in the configuration if DCMRWF3[15] is 1'b0 is derived from RMII_CLK DIV20. 11b - The MAC_RX_CLK in the configuration if DCMRWF3[14:13] is 2'b00 is inactive/disabled. The MAC_TX_CLK in the configuration if DCMRWF3[15] is 1'b0 is inactive/disabled.
8 PMOS_CTRL_GPIO_DATA	PMOS Control GPIO Data Controls the data-out from the PMOS_CTRL pad when PMC.CONFIG[LMSMPSEN] is 0. PMOS_CTRL_GPIO_DATA is not impacted if PMC.CONFIG[LMSMPSEN] is 1. 0b - Data is 0 1b - Data is 1
7-6 —	Reserved Contact NXP support for details.

Table continues on the next page...

Table continued from the previous page...

Field	Function
5 PMC_TRIM_RGM_DCF_BYP_STDBY_EXT	PMC Trim MC_RGM DCF Bypass Standby Exit Controls the bypassing of PMC trimming and MC_RGM loading on standby exit. 0b - Not bypassed 1b - Bypassed
4 FIRC_TRIM_BYP_STDBY_EXT	FIRC Trim Bypass Standby Exit Controls the bypassing of FIRC trimming on standby exit. 0b - Not bypassed 1b - Bypassed
3 DCM_SCAN_BYP_STDBY_EXT	DCM Scan Bypass Standby Exit Controls the bypassing of DCM scanning on standby exit. 0b - Not bypassed 1b - Bypassed
2 MAC2_LOOBPACK_CLK_SEL	Selects MAC2_LOOPBACK_CLK source. 0b - Reserved 1b - MAC_CLK_TX is selected.
1 MAC2_TX_RMII_CLK_LPBCK_EN	Enables the MAC2_TX_RMII_CLK loopback. 0b - Disabled 1b - Enabled
0 —	Reserved

38.2.50 Read Write GPR On Functional Reset 3 (DCMRWF3)

Offset

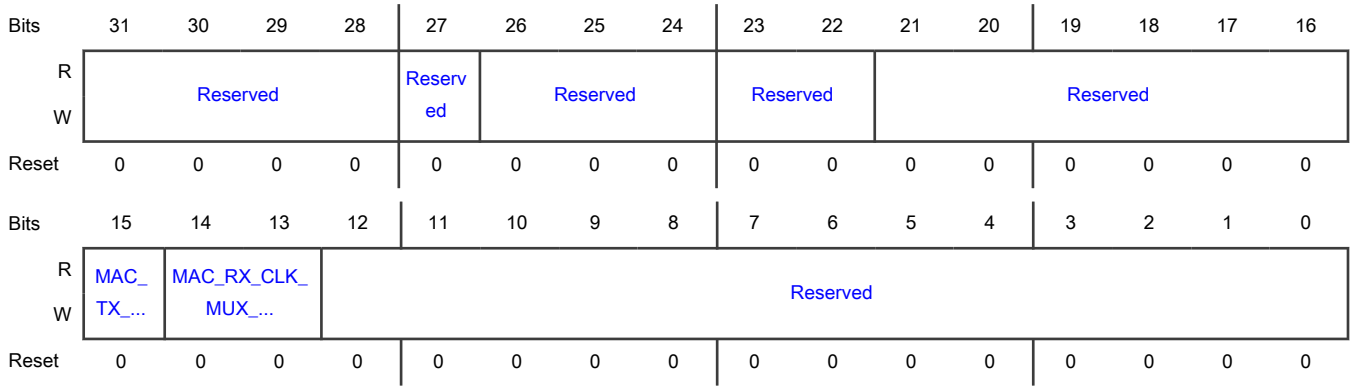
Register	Offset
DCMRWF3	608h

Function

Controls and specifies sources of WKPU.

This register resets after functional reset 3.

Diagram



Fields

Field	Function
31-28 —	Reserved
27 —	Reserved
26-24 —	Reserved
23-22 —	Reserved
21-16 —	Reserved
15 MAC_TX_CLK_MUX_BYPASS	MAC TX Clock MUX Bypass Bypasses the MAC_TX_CLK mux MC_CGM_MUX8 and the TX_CLK arrives without MC_CGM_MUX8 to the MAC in case of external clock source. **NOTE** The MAC_RX_CLK in this configuration is derived based on DCMRWF2[10:9] configuration. 0b - The MAC_TX_CLK is derived from MC_CGM_MUX8. 1b - The MAC_TX_CLK arrives directly from the RMII_TX_CLK pin.
14-13 MAC_RX_CLK_MUX_BYPASS	MAC RX Clock MUX Bypass Bypasses the MAC_RX_CLK mux MC_CGM_MUX7 and the RX_CLK arrives without MC_CGM_MUX7 to the MAC in case of external clock source.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE The MAC_RX_CLK in this configuration is derived based on DCMRWF2[10:9] configuration. 00b - The MAC_RX_CLK is derived from MC_CGM_MUX7. 01b - The MAC_RX_CLK arrives directly from the RX_CLK pin. 10b - The MAC_RX_CLK arrives directly from the RMII_TX_CLK pin. RESERVED for S32K388 and S32K389. 11b - Reserved
12-0 —	Reserved

38.2.51 Read Write GPR On Functional Reset 4 (DCMRWF4)

Offset

Register	Offset
DCMRWF4	60Ch

Function

Contains information related to:

- Mux mode enable.
- Input bypass.

This field resets after functional reset 4.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MAC2_TX...	MAC2_RX_CLK_MUX...		Reserved	MAC2_RM...	MAC_RMI...	MAC2_CONF_SEL		MAC_SB_...	MAC2_TX_RX_CLK_...		CM7_3_C...	CM7_2_C...	CM7_1_C...	CM7_0_C...	GLITCH_...
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	GLITCH_...	GLITCH_...	GLITCH_...	Reserved		MUX_MOD...	MUX_MOD...	Reserved		MUX_MOD...	MUX_MOD...	MUX_MOD...	MUX_MOD...	MUX_MOD...	MUX_MOD...	Reserved
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 MAC2_TX_CLK_MUX_BYPASS	MAC2 TX Clock Mux Bypass Bypasses the MAC2_TX_CLK mux MC_CGM_MUX16 and the TX_CLK arrives without MC_CGM_MUX16 to the MAC2 in case of external clock source. **NOTE** The MAC2_TX_CLK in this configuration is derived based on DCMRWF4[22:21] configuration. 0b - The MAC2_TX_CLK is derived from MC_CGM_MUX16. 1b - The MAC2_TX_CLK arrives directly from the RMII_TX_CLK pin.
30-29 MAC2_RX_CLK_MUX_BYPASS	MAC2 RX Clock Mux Bypass Bypasses the MAC2_RX_CLK mux MC_CGM_MUX15 and the RX_CLK arrives without MC_CGM_MUX15 to the MAC2 in case of external clock source. **NOTE** The MAC2_RX_CLK in this configuration is derived based on DCMRWF4[22:21] configuration. 00b - The MAC2_RX_CLK is derived from MC_CGM_MUX15. 01b - The MAC2_RX_CLK arrives directly from the MAC RX_CLK pin. 10b - The MAC2_RX_CLK arrives directly from the MAC2 RMII_TX_CLK pin. 11b - Reserved.
28 —	Reserved
27 MAC2_RMII_CLK_MUX_BYPASS	MAC2 RMII Clock Mux Bypass Bypasses MAC2_RMII_CLK mux MC_CGM_MUX16 and RMII_CLK arrives without MC_CGM_MUX16 to the MAC2 in case of external clock source. 0b - The MAC2_RMII_CLK is derived from MC_CGM_MUX16. 1b - The MAC2_RMII_CLK arrives directly from the RMII2_TX_CLK pin.
26 MAC_RMII_CLK_MUX_BYPASS	MAC RMII Clock Mux Bypass Bypasses MAC_RMII_CLK mux MC_CGM_MUX8 and RMII_CLK arrives without MC_CGM_MUX8 to the MAC in case of external clock source. 0b - The MAC_RMII_CLK is derived from MC_CGM_MUX8. 1b - The MAC_RMII_CLK arrives directly from the RMII_TX_CLK pin.
25-24 MAC2_CONF_SEL	MAC Configuration Selection Selects between MII and RMII mode of ethernet. 00b - MII mode

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01b - RGMII mode 10b - RMII mode 11b - Reserved. Device operation not guaranteed.
23 MAC_SB_END_CTRL	MAC Sideband Data Endianness Control 0b - Data is transferred in little-endian mode. 1b - Data is transferred in big-endian mode.
22-21 MAC2_TX_RX_CLK_MUX_BYPASS	MAC2 TX RX Clock MUX Bypass Bypasses the MAC2_TX_CLK and MAC2_RX_CLK sources depending on DCMRWF4[31:29] configuration. 00b - The MAC2_RX_CLK in the configuration if DCMRWF4[30:29] is 2'b00 is derived from MC_CGM_MUX7. The MAC2_TX_CLK in the configuration if DCMRWF4[31] is 1'b0 is derived from MC_CGM_MUX8. 01b - The MAC2_RX_CLK in the configuration if DCMRWF4[30:29] is 2'b00 is derived from RMII_CLK DIV2. The MAC2_TX_CLK in the configuration if DCMRWF4[31] is 1'b0 is derived from RMII_CLK DIV2. 10b - The MAC2_RX_CLK in the configuration if DCMRWF4[30:29] is 2'b00 is derived from RMII_CLK DIV20. The MAC2_TX_CLK in the configuration if DCMRWF4[31] is 1'b0 is derived from RMII_CLK DIV20. 11b - The MAC2_RX_CLK in the configuration if DCMRWF4[30:29] is 2'b00 is inactive/disabled. The MAC2_TX_CLK in the configuration if DCMRWF4[31] is 1'b0 is inactive/disabled.
20 CM7_3_CPUWAIT	Cortex-M7_3 CPU Wait Enables the configuration to place the Cortex-M7_3 core into CPU wait mode. 0b - Disables CPUWAIT 1b - Enables CPUWAIT
19 CM7_2_CPUWAIT	Cortex-M7_2 CPU Wait Enables the configuration to place the Cortex-M7_2 core into CPU wait mode. 0b - Disables CPUWAIT 1b - Enables CPUWAIT
18 CM7_1_CPUWAIT	Cortex-M7_1 CPU Wait Enables the configuration to place the Cortex-M7_1 core into CPU wait mode. 0b - Disables CPUWAIT 1b - Enables CPUWAIT
17	Cortex-M7_0 CPU Wait Enables the configuration to place the Cortex-M7_0 core in CPU wait mode.

Table continues on the next page...

Table continued from the previous page...

Field	Function
CM7_0_CPUWAIT	0b - Disables CPUWAIT 1b - Enables CPUWAIT
16 GLITCH_FIL_TRG_IN3_BYP	Glitch Filter TRGMUX Input 3 Bypass Selects whether to bypass or filter out the pulse. If this field = 0, it enables glitch filter on TRGMUX input 60, and if the field = 1, it bypasses glitch filter on TRGMUX input 60. 0b - Enables 1b - Bypasses
15 GLITCH_FIL_TRG_IN2_BYP	Glitch Filter TRGMUX Input 2 Bypass Selects whether to bypass or filter out the pulse. If this field = 0, it enables glitch filter on TRGMUX input 61, and if the field = 1, it bypasses glitch filter on TRGMUX input 61. 0b - Enables 1b - Bypasses
14 GLITCH_FIL_TRG_IN1_BYP	Glitch Filter TRGMUX Input 1 Bypass Selects whether to bypass or filter out the pulse. If this field = 0, it enables glitch filter on TRGMUX input 62, and if the field = 1, it bypasses glitch filter on TRGMUX input 62. 0b - Enables 1b - Bypasses
13 GLITCH_FIL_TRG_IN0_BYP	Glitch Filter TRGMUX Input 0 Bypass Selects whether to bypass or filter out the pulse. If this field = 0, it enables glitch filter on TRGMUX input 63, and if the field = 1, it bypasses glitch filter on TRGMUX input 63. 0b - Enables 1b - Bypasses
12-11 —	Reserved
10 MUX_MODE_EN_ADC2_S9	Mux Mode Enable ADC2 Standard Channel 9 Controls the selection of GPIOs to drive ADC_2 standard channel 9. 0b - GPIO_132 1b - GPIO_46
9 MUX_MODE_EN_ADC2_S8	Mux Mode Enable ADC2 Standard Channel 8 Controls the selection of GPIOs to drive ADC_2 standard channel 8. 0b - GPIO_133 1b - GPIO_45

Table continues on the next page...

Table continued from the previous page...

Field	Function
8-7 —	Reserved
6 MUX_MODE_EN_ADC1_S23	Mux Mode Enable ADC_1 Standard Channel 23 Controls the selection of GPIOs to drive ADC_1 standard channel 23. 0b - GPIO_125 1b - GPIO_146
5 MUX_MODE_EN_ADC1_S22	Mux Mode Enable ADC_1 Standard Channel 22 Controls the selection of GPIOs to drive ADC_1 standard channel 22. 0b - GPIO_124 1b - GPIO_145
4 MUX_MODE_EN_ADC1_S15	Mux Mode Enable ADC_1 Standard Channel 15 Controls the selection of GPIOs to drive ADC_1 standard channel 15. 0b - GPIO_4 1b - GPIO_33
3 MUX_MODE_EN_ADC1_S14	Mux Mode Enable ADC_1 Standard Channel 14 Controls the selection of GPIOs to drive ADC_1 standard channel 14. 0b - GPIO_69 1b - GPIO_32
2 MUX_MODE_EN_ADC0_S9	Mux Mode Enable ADC_0 Standard Channel 9 Controls the selection of GPIOs to drive ADC_0 standard channel 9. 0b - GPIO_1 1b - GPIO_46
1 MUX_MODE_EN_ADC0_S8	Mux Mode Enable ADC_0 Standard Channel 8 Controls the selection of GPIOs to drive ADC_0 standard channel 8. 0b - GPIO_0 1b - GPIO_45
0 —	Reserved

38.2.52 Read Write GPR On Functional Reset 5 (DCMRWF5)

Offset

Register	Offset
DCMRWF5	610h

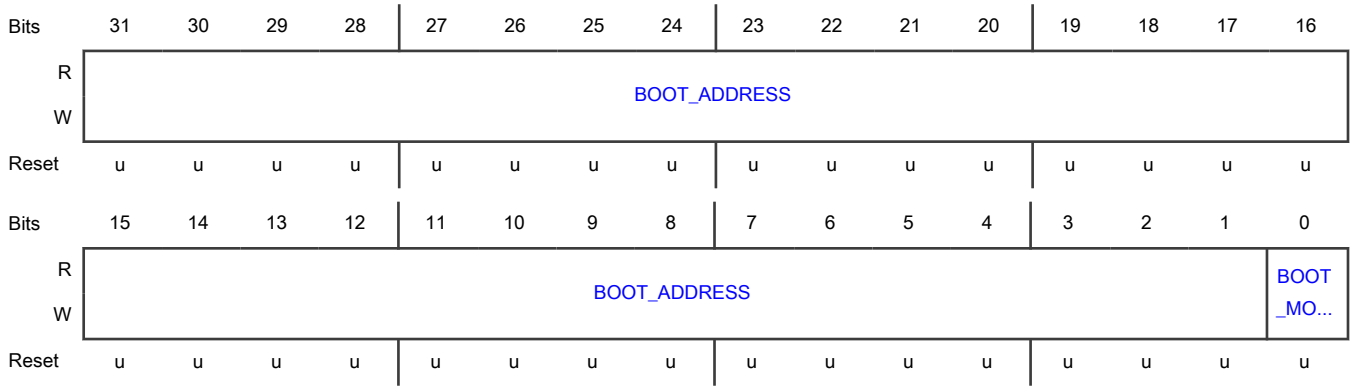
Function

Contains boot address and boot mode.

This register resets after functional reset 5.

The reset value of this register is undefined on reset and is loaded from the flash memory contents at the end of the reset sequence.

Diagram



Fields

Field	Function
31-1 BOOT_ADDRESS	Boot Address Specifies the Cortex-M7_0 base address of the vector table to be used after exiting Standby mode (only to be considered in Fast Standby mode).
0 BOOT_MODE	Boot Mode Selects the type of Boot mode after exiting Standby mode. 0b - Normal 1b - Fast Standby

38.2.53 Read-Only GPR On PMCPOR Reset 1 (DCMROPP1)

Offset

Register	Offset
DCMROPP1	700h

Function

Resets after PMCPOR reset 1 and captures the status of the functional reset sequence process when POR_WDG overflows.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	POR_WDG...	POR_WDG...	POR_WDG...	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	POR_WDG...	Reserved	Reserved	POR_WDG...	Reserved
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	POR_WDG...	Reserved	Reserved	POR_WDG...	POR_WDG...	Reserved	Reserved	Reserved	POR_WDG...	POR_WDG...	POR_WDG...	POR_WDG...	POR_WDG...	POR_WDG...	POR_WDG...
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 POR_WDG_STAT31	POR_WDG Status 31 Captures the status of the MC_RGM reset event (if occurred) while the chip is in Standby mode. **NOTE** This field is used only for standby sequence monitoring. 0b - Not detected 1b - Detected
30 POR_WDG_STAT30	POR_WDG Status 30 Captures the status of standby exit acknowledgement by MC_PCU when POR_WDG overflows. **NOTE** This field is used only for standby sequence monitoring. 0b - Not acknowledged

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Acknowledged
29 POR_WDG_STAT29	POR_WDG Status 29 Captures the status of the MC_ME standby entry request that MC_ME initiates when POR_WDG overflows. **NOTE** This field is used only for standby sequence monitoring. 0b - Active 1b - Inactive
28 —	Reserved
27 —	Reserved
26 —	Reserved
25 —	Reserved
24 —	Reserved
23 —	Reserved
22 —	Reserved
21 —	Reserved
20 POR_WDG_STAT20	POR_WDG Status 20 Specifies the status of the functional reset sequence process, DEST0, when POR_WDG overflows. 0b - Inactive 1b - Active
19 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
18 —	Reserved
17 POR_WDG_STAT17	POR_WDG Status 17 Specifies the status of the functional reset sequence process, FUNC10, when POR_WDG overflows. 0b - Inactive 1b - Active
16 —	Reserved
15 —	Reserved
14 POR_WDG_STAT14	POR_WDG Status 14 Specifies the status of the functional reset sequence process, FUNC9, when POR_WDG overflows. 0b - Inactive 1b - Active
13 —	Reserved
12 —	Reserved
11 POR_WDG_STAT11	POR_WDG Status 11 Specifies the status of the functional reset sequence process, FUNC8, when POR_WDG overflows. 0b - Inactive 1b - Active
10 POR_WDG_STAT10	POR_WDG Status 10 Specifies the status of the functional reset sequence process, FUNC7, when POR_WDG overflows. 0b - Inactive 1b - Active
9 —	Reserved
8 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
7 —	Reserved
6 POR_WDG_STAT6	POR_WDG Status 6 Specifies the status of the functional reset sequence process, FUNC6, when POR_WDG overflows. 0b - Inactive 1b - Active
5 POR_WDG_STAT5	POR_WDG Status 5 Specifies the status of the functional reset sequence process, FUNC5, when POR_WDG overflows. 0b - Inactive 1b - Active
4 POR_WDG_STAT4	POR_WDG Status 4 Specifies the status of the functional reset sequence process, FUNC4, when POR_WDG overflows. 0b - Inactive 1b - Active
3 POR_WDG_STAT3	POR_WDG Status 3 Specifies the status of the functional reset sequence process, FUNC3, when POR_WDG overflows. 0b - Inactive 1b - Active
2 POR_WDG_STAT2	POR_WDG Status 2 Specifies the status of the functional reset sequence process, FUNC2, when POR_WDG overflows. 0b - Inactive 1b - Active
1 POR_WDG_STAT1	POR_WDG Status 1 Specifies the status of the functional reset sequence process, FUNC1, when POR_WDG overflows. 0b - Inactive 1b - Active
0 POR_WDG_STAT0	POR_WDG Status 0 Specifies the status of the functional reset sequence process, FUNC0, when POR_WDG overflows. 0b - Inactive 1b - Active

38.2.54 Read-Only GPR On PMCPOR Reset 2 (DCMROPP2)

Offset

Register	Offset
DCMROPP2	704h

Function

Resets after PMCPOR reset 2 and captures the MC_RGM functional or external event status when POR_WDG overflows.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved	POR_WDG...	POR_WDG...	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	POR_WDG...	Reserved	Reserved	Reserved	POR_WDG...
W		W1C	W1C									W1C				W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	Reserved	Reserved	POR_WDG...	Reserved	Reserved	POR_WDG...	POR_WDG...	POR_WDG...	POR_WDG...	Reserved	POR_WDG...	POR_WDG...	Reserved	Reserved	POR_WDG...
W				W1C			W1C	W1C	W1C	W1C		W1C	W1C			W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30 POR_WDG_STAT62	POR_WDG Status 62 Specifies the value of MC_RGM.FES[DEBUG_FUNC] when POR_WDG overflows. 0b - 0 1b - 1
29 POR_WDG_STAT61	POR_WDG Status 61 Specifies the value of MC_RGM.FES[SW_FUNC] when POR_WDG overflows. 0b - 0 1b - 1
28	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
27 —	Reserved
26 —	Reserved
25 —	Reserved
24 —	Reserved
23 —	Reserved
22 —	Reserved
21 —	Reserved
20 POR_WDG_STAT52	POR_WDG Status 52 Specifies the value of MC_RGM.FES[HSE_BOOT_RST] when POR_WDG overflows. 0b - 0 1b - 1
19 —	Reserved
18 —	Reserved
17 —	Reserved
16 POR_WDG_STAT48	POR_WDG Status 48 Specifies the value of MC_RGM.FES[HSE_SWT_RST] when POR_WDG overflows. 0b - 0 1b - 1

Table continues on the next page...

Table continued from the previous page...

Field	Function
15 —	Reserved
14 —	Reserved
13 —	Reserved
12 POR_WDG_STAT44	POR_WDG Status 44 Specifies the value of MC_RGM.FES[PLL_AUX] when POR_WDG overflows. 0b - 0 1b - 1
11 —	Reserved
10 —	Reserved
9 POR_WDG_STAT41	POR_WDG Status 41 Specifies the value of MC_RGM.FES[JTAG_RST] when POR_WDG overflows. 0b - 0 1b - 1
8 POR_WDG_STAT40	POR_WDG Status 40 Specifies the value of MC_RGM.FES[Reserved] when POR_WDG overflows. 0b - 0 1b - 1
7 POR_WDG_STAT39	POR_WDG Status 39 Specifies the value of MC_RGM.FES[SWT1_RST] when POR_WDG overflows. 0b - 0 1b - 1
6 POR_WDG_STAT38	POR_WDG Status 38 Specifies the value of MC_RGM.FES[SWT0_RST] when POR_WDG overflows. 0b - 0 1b - 1

Table continues on the next page...

Table continued from the previous page...

Field	Function
5 —	Reserved
4 POR_WDG_STAT36	POR_WDG Status 36 Specifies the value of MC_RGM.FES[ST_DONE] when POR_WDG overflows. 0b - 0 1b - 1
3 POR_WDG_STAT35	POR_WDG Status 35 Specifies the value of MC_RGM.FES[FCCU_RST] when POR_WDG overflows. 0b - 0 1b - 1
2 —	Reserved
1 —	Reserved
0 POR_WDG_STAT32	POR_WDG Status 32 Specifies the value of MC_RGM.FES[F_EXR] when POR_WDG overflows. 0b - 0 1b - 1

38.2.55 Read-Only GPR On PMCPOR Reset 3 (DCMROPP3)

Offset

Register	Offset
DCMROPP3	708h

Function

Resets after PMCPOR reset 3 and captures the MC_RGM destructive event status when POR_WDG overflows.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved	POR_WDG...	POR_WDG...	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	POR_WDG...	POR_WDG...	POR_WDG...
W		W1C	W1C											W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	POR_WDG...	POR_WDG...	Reserved	POR_WDG...	Reserved	POR_WDG...	POR_WDG...	POR_WDG...	Reserved	POR_WDG...	Reserved	POR_WDG...	POR_WDG...	Reserved	Reserved	POR_WDG...
W	W1C	W1C		W1C		W1C	W1C	W1C		W1C		W1C	W1C			W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30 POR_WDG_STAT94	POR_WDG Status 94 Specifies the value of MC_RGM.DES[DEBUG_DEST] when POR_WDG overflows. 0b - 0 1b - 1
29 POR_WDG_STAT93	POR_WDG Status 93 Specifies the value of MC_RGM.DES[SW_DEST] when POR_WDG overflows. 0b - 0 1b - 1
28 —	Reserved
27 —	Reserved
26 —	Reserved
25 —	Reserved
24	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
23 —	Reserved
22 —	Reserved
21 —	Reserved
20 —	Reserved
19 —	Reserved
18 POR_WDG_STAT82	POR_WDG Status 82 Specifies the value of MC_RGM.DES[HSE_SNVS_RST] when POR_WDG overflows. 0b - 0 1b - 1
17 POR_WDG_STAT81	POR_WDG Status 81 Specifies the value of MC_RGM.DES[HSE_TMPR_RST] when POR_WDG overflows. 0b - 0 1b - 1
16 POR_WDG_STAT80	POR_WDG Status 80 Specifies the value of MC_RGM.DES[CM7_CORE_CLK_FAIL] when POR_WDG overflows. 0b - 0 1b - 1
15 POR_WDG_STAT79	POR_WDG Status 79 Specifies the value of MC_RGM.DES[SYS_DIV_FAIL] when POR_WDG overflows. 0b - 0 1b - 1
14 POR_WDG_STAT78	POR_WDG Status 78 Specifies the value of MC_RGM.DES[HSE_CLK_FAIL] when POR_WDG overflows. 0b - 0

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - 1
13 —	Reserved
12 POR_WDG_STAT76	POR_WDG Status 76 Specifies the value of MC_RGM.DES[AIPS_PLAT_CLK_FAIL] when POR_WDG overflows. 0b - 0 1b - 1
11 —	Reserved
10 POR_WDG_STAT74	POR_WDG Status 74 Specifies the value of MC_RGM.DES[CORE_CLK_FAIL] when POR_WDG overflows. 0b - 0 1b - 1
9 POR_WDG_STAT73	POR_WDG Status 73 Specifies the value of MC_RGM.DES[PLL_LOL] when POR_WDG overflows. 0b - 0 1b - 1
8 POR_WDG_STAT72	POR_WDG Status 72 Specifies the value of MC_RGM.DES[SWT2_RST] when POR_WDG overflows. 0b - 0 1b - 1
7 —	Reserved
6 POR_WDG_STAT70	POR_WDG Status 70 Specifies the value of MC_RGM.DES[MC_RGM_FRE] when POR_WDG overflows. 0b - 0 1b - 1
5 —	Reserved
4	POR_WDG Status 68

Table continues on the next page...

Table continued from the previous page...

Field	Function
POR_WDG_STAT68	Specifies the value of MC_RGM.DES[STCU_URF] when POR_WDG overflows. 0b - 0 1b - 1
3 POR_WDG_STAT67	POR_WDG Status 67 Specifies the value of MC_RGM.DES[FCCU_FTR] when POR_WDG overflows. 0b - 0 1b - 1
2 —	Reserved
1 —	Reserved
0 POR_WDG_STAT64	POR_WDG Status 64 Specifies the value of MC_RGM.DES[F_POR] when POR_WDG overflows. 0b - 0 1b - 1

38.2.56 Read-Only GPR On PMCPOR Reset 4 (DCMROPP4)

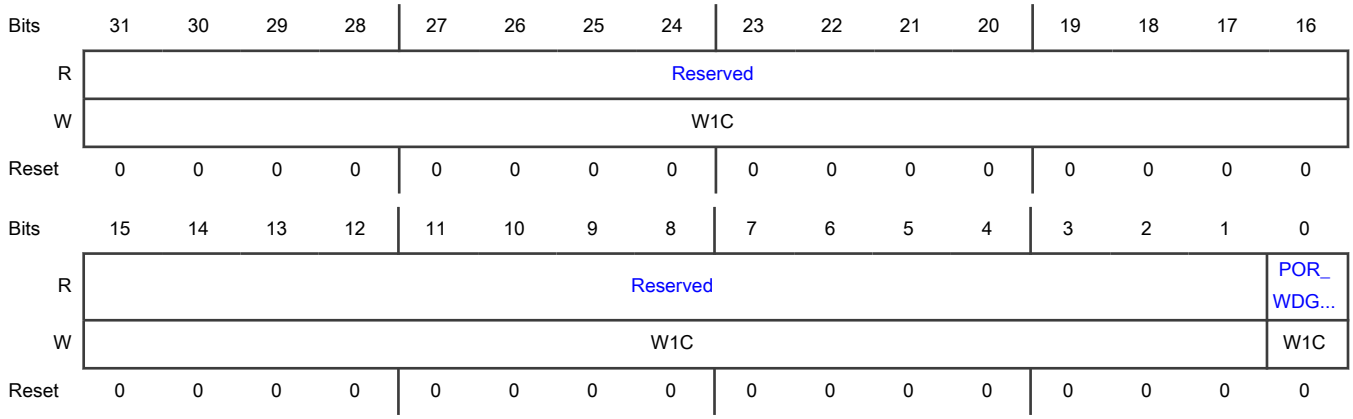
Offset

Register	Offset
DCMROPP4	70Ch

Function

Resets after PMCPOR reset 4 and captures the POR_WDG reset event if POR_WDG initiates a POR sequence.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 POR_WDG_STAT96	POR_WDG Status 96 Specifies the status of POR_WDG. If this field = 1, it indicates that a stuck scenario is detected and the chip POR event is raised. See POR_WDG_STAT[95:0] for the chip status when POR_WDG overflows. 0b - Inactive 1b - Active

Chapter 39
Device Configuration Module (DCM)

39.1 Overview

DCM controls:

- LC
- DCF client registers
- Debug authorization (Export Control mode)

The module also establishes a RoT for the chip by parsing the master root key and other security records.

39.1.1 Block diagram

This figure is a block representation of DCM.

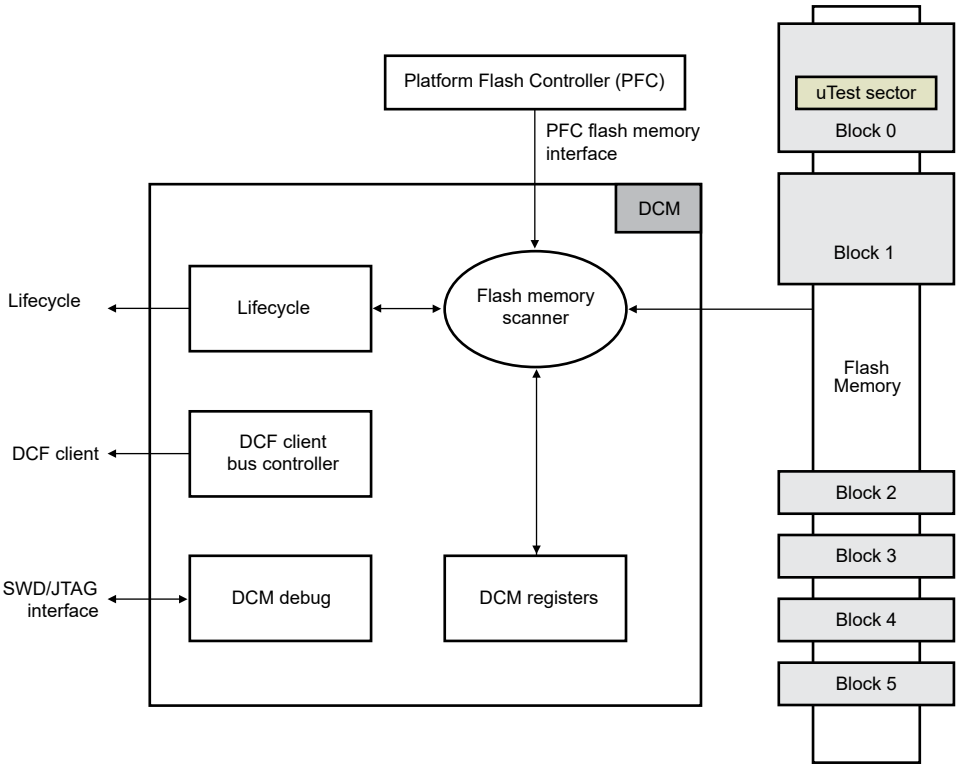


Figure 161. DCM block diagram

39.1.2 Features

- Scans flash memory and configures the system for following information:
  - LC detection
  - DCF records configuration using flash memory
- Allows debug features for flash memory content and the chip
- Provides debug enable control for HSE and application

- Provides a valid boot address detection
- Enables the DCF client to be writable via the IPS bus
- Parametrizes control to mask a set of DCF client chip select
- Supports temporary advancement of the LC by programming DCMLCC[DCMFLC]
- Manages debug features for flash memory content and the chip like debug enable control and debug authorization.

39.2 Functional description

DCM provides information about the current state and configuration of the system that you could use to:

- Configure the application software.
- Debug the system.

39.2.1 Modes of operation

DCM operates identically in all system modes of operation.

39.2.2 DCF mechanism

The DCF mechanism handles chip parameter settings via the OTP flash memory.

You can store a series of DCF records in UTEST flash memory, and each record is 64 bits in length. The chip processes these records during the system reset sequence before the CPU leaves reset.

39.2.3 DCF error recording

DCF errors are recorded in DCM for both types of clients—spread spectrum safe and normal. See DCF client error mechanism for details on spread spectrum safe clients.

39.2.4 DCF client error mechanism

The DCM consists of DCMMISC[DCMCERS] for detecting faulty DCF records. The DCMSRRn registers capture the details up to 16 faulty records. The DCMCERS bit can be cleared by writing 1 to it.

While scanning the flash, in case if a faulty record is encountered, the DCMMISC[DCMCERS] gets set indicating that there is atleast one faulty record. In such a case, the user can identify the details about the faulty record in the DCMSRRn registers and should update the flash memory with a new record, provided the record is not write-once.

For example, in the image below, if the faulty DCF record is DCF2, then the correct DCF record must also be DCF2. Even in case if the faulty record is updated, the DCM stores the faulty record via the DCMMISC[DCMCERS] and DCMSRRn registers.

This figure shows that a faulty DCF record is loaded between the original and final DCF record sets.

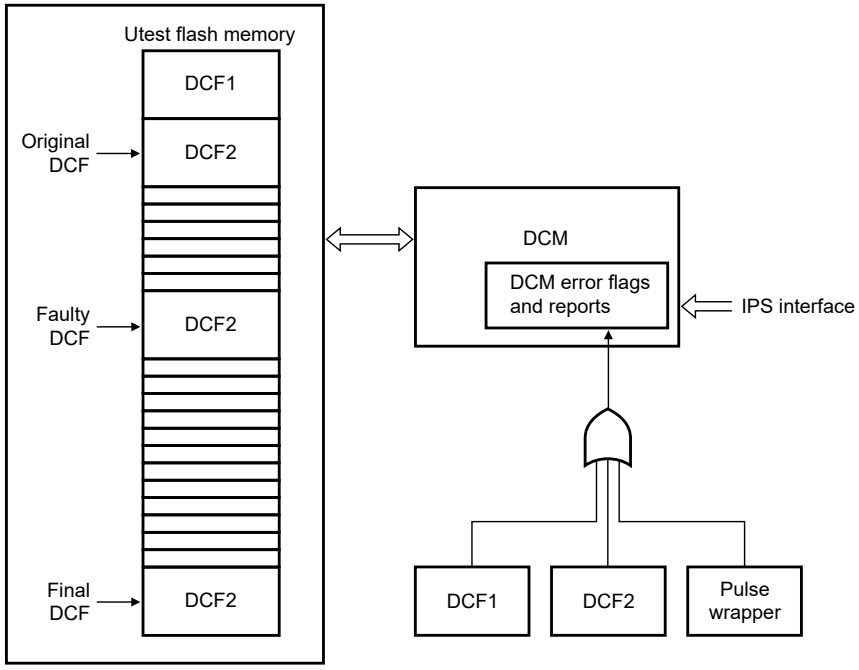


Figure 162. DCF error handling

39.2.5 DCM error detection mechanism

This figure shows the flow diagram of errors logged into DCM from DCF clients and Flash memory scanner. It also shows that DCM provides software control for lifecycle and export control mode.

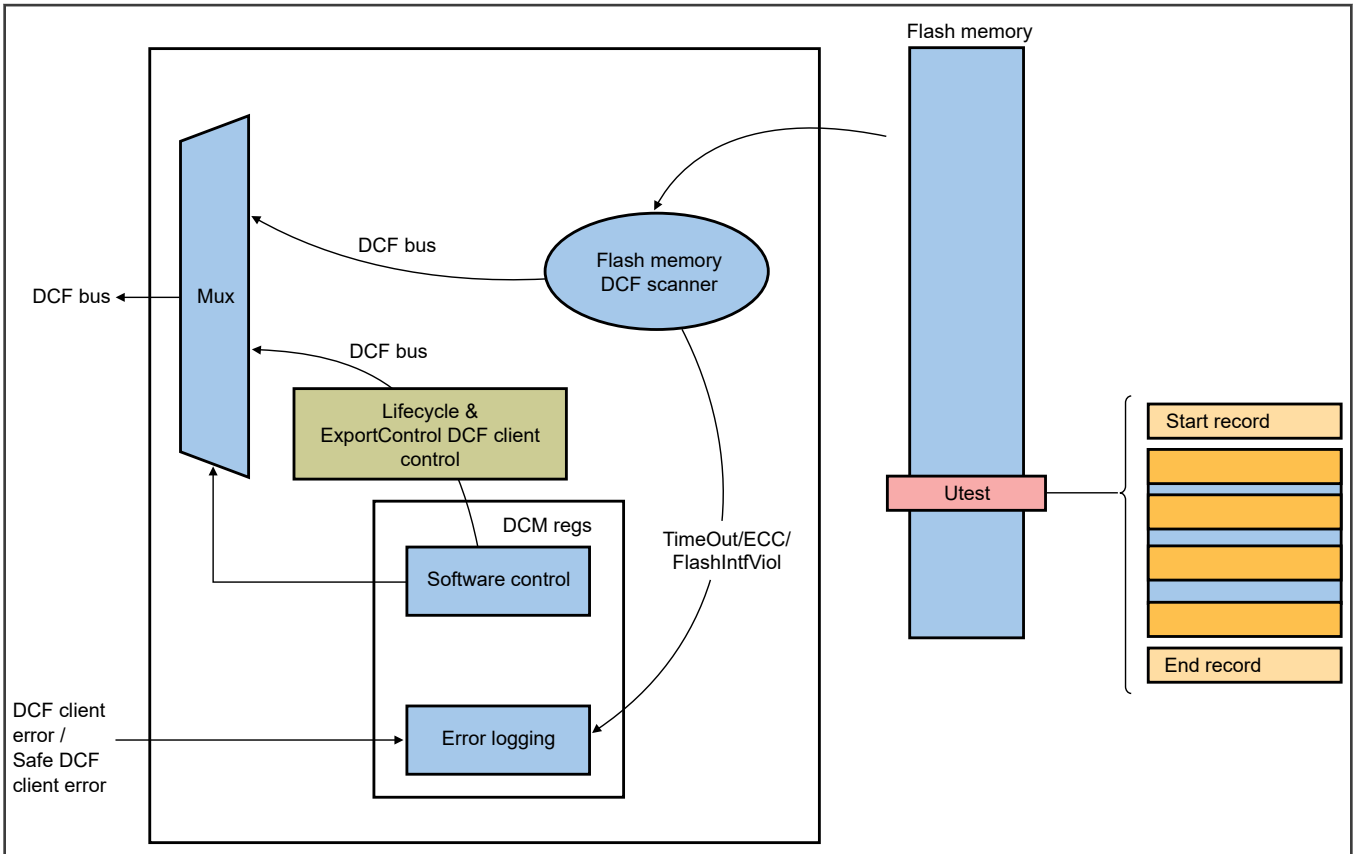


Figure 163. DCM error detection mechanism

39.2.6 LC

DCM determines the LC of the chip by reading the LC slots from the Utest flash memory. This read operation is performed during the reset phase, with normal timings. The operating monitors and an ECC check protect the operation. Additionally, a set of sanity checks executed over the LC read data guarantee the integrity of the final LC value.

At the end of the reset phase, the LC contains one of the following values:

- OEM production (OEM_PROD)
- In field (IN_FIELD)
- Pre-FA
- FA

The DCM LC progresses in the direction shown in this figure:

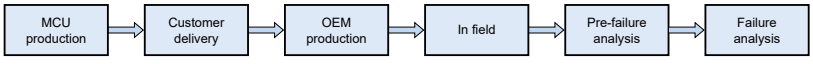


Figure 164. LC sequence

The LC is written into six slots, 128 bits each, and at fixed locations in the Utest flash memory block. Each LC slot is read in a single atomic operation and is organized in two types of fields:

- Valid (lower 64b of LC slot)
- Invalid (higher 64b of LC slot)

Depending on the possible combinations of data programmed into these fields, each LC slot indicates one of the four possible statuses as shown in Table 232. To know more about LC slots, see Table 233.

Table 232. LC slot status

LC slots		LC slot value
Valid field (64 bits)	Invalid field (64 bits)	
Erased	Erased	Erased
Marked	Erased	Active
Marked	Marked	Inactive
Other values		Illegal

In this case, "Marked" refers to a value that is configured based on the bit pattern 55AA_50AF_55AA_50AFh, and "Erased" is detected using the bit pattern FFFF_FFFF_FFFF_FFFFh.

Table 233. LC slots

LC slot 2 OEM_PROD	LC slot 3 IN_FIELD	LC slot 4 Pre-FA	LC slot 5 FA	Resulting LC
Active	Erased	Erased	Erased	OEM_PROD
Inactive	Active	Erased	Erased	IN_FIELD
Inactive	Inactive	Active	Erased	Pre-FA
Inactive	Inactive	Inactive	Active	FA

NOTE

When triggering DCM for rescanning, the software must ensure that the flash memory program and erase fields are not set. Otherwise, DCM does not configure the chip and ignores all data returned from the UTEST flash memory. This results in LC becoming IN_FIELD. All DCF clients indicate their default values in this case.

39.3 DCM register descriptions

39.3.1 DCM memory map

DCM base address: 402A_C000h

Offset	Register	Width (In bits)	Access	Reset value
0h	DCM Status (DCMSTAT)	32	R	0000_0371h
4h	LC and LC Control (DCMLCC)	32	R	See section
8h	LC Scan Status (DCMLCS)	32	RW	0000_0000h
1Ch	DCM Miscellaneous (DCMMISC)	32	RW	0000_0001h
20h	Debug Status and Configuration (DCMDEB)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
2Ch	DCF Error Count (DCMEC)	32	R	0000_0000h
30h	DCF Scan Report (DCMSRR1)	32	RW	0000_0000h
34h	DCF Scan Report (DCMSRR2)	32	RW	0000_0000h
38h	DCF Scan Report (DCMSRR3)	32	RW	0000_0000h
3Ch	DCF Scan Report (DCMSRR4)	32	RW	0000_0000h
40h	DCF Scan Report (DCMSRR5)	32	RW	0000_0000h
44h	DCF Scan Report (DCMSRR6)	32	RW	0000_0000h
48h	DCF Scan Report (DCMSRR7)	32	RW	0000_0000h
4Ch	DCF Scan Report (DCMSRR8)	32	RW	0000_0000h
50h	DCF Scan Report (DCMSRR9)	32	RW	0000_0000h
54h	DCF Scan Report (DCMSRR10)	32	RW	0000_0000h
58h	DCF Scan Report (DCMSRR11)	32	RW	0000_0000h
5Ch	DCF Scan Report (DCMSRR12)	32	RW	0000_0000h
60h	DCF Scan Report (DCMSRR13)	32	RW	0000_0000h
64h	DCF Scan Report (DCMSRR14)	32	RW	0000_0000h
68h	DCF Scan Report (DCMSRR15)	32	RW	0000_0000h
6Ch	DCF Scan Report (DCMSRR16)	32	RW	0000_0000h
80h	LC Scan Status 2 (DCMLCS_2)	32	RW	0000_0000h

39.3.2 DCM Status (DCMSTAT)

Offset

Register	Offset
DCMSTAT	0h

Function

Indicates the status of DCM at different stages.

NOTE

This register resets on functional reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0													0		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				0	DCMDBGPS	0	DCMUTS	0			DCMLCST	0		DCMERR	DCMDONE
W																
Reset	0	0	0	0	0	0	1	1	0	1	1	1	0	0	0	1

Fields

Field	Function
31-19 —	Reserved
18-16 —	Reserved
15-12 —	Reserved
11 —	Reserved
10 DCMDBGPS	Debug Password Scanning Status Indicates the DCM debug password scanning status. 0b - Completed with errors 1b - Completed successfully
9 —	Reserved
8 DCMUTS	DCM Utest DCF Scanning Status (valid only if DCMDONE bit is set) **NOTE** This bit always returns 0 in In Field LC. 0b - Completed with errors. 1b - Completed successfully.
7-5	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
4 DCMLCST	LC Scanning Status (valid only if DCMDONE bit is set) Indicates the DCM LC scanning status. **NOTE** This bit always returns 0 in INFIELD lifecycle. 0b - Completed with errors 1b - Completed successfully
3-2 —	Reserved
1 DCMERR	DCM Completion of Flash Scan with Error Status (valid only if DCMDONE bit is set) 0b - Completed with success. 1b - Completed with error.
0 DCMDONE	DCM Flash Scanning Status Indicates whether the DCM scanning is in progress or complete. 0b - Running 1b - Completed

39.3.3 LC and LC Control (DCMLCC)

Offset

Register	Offset
DCMLCC	4h

Function
Resets on the functional reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0						Reserved		0	DCMRLC			Reserved	DCMCLC		
W																
Reset	0	0	0	0	0	0	0	0	0	u[1]	u	u	0	u[1]	u	u

1. Post reset, the reset value of this register is 0000_0077h, and after scanning, it changes according to the programmed value.

Fields

Field	Function
31-10 —	Reserved
9-8 —	Reserved
7 —	Reserved
6-4 DCMRLC	Real LC Projects the real LC of the chip. The LC can move in this sequence: (010 : OEM_PROD) > (111 : IN_FIELD) > (001 : Pre-FA) > (000 : FA) 000b - FA 001b - Pre-FA 010b - OEM_PROD 100b - Reserved 101b - Reserved 111b - IN_FIELD
3 —	Reserved
2-0 DCMCLC	Current LC Projects the current LC of the chip. The LC can move in this sequence: (010 : OEM_PROD) > (111 : IN_FIELD) > (001 : Pre-FA) > (000 : FA)

Table continues on the next page...

Table continued from the previous page...

Field	Function
	000b - FA 001b - Pre-FA 010b - OEM_PROD 100b - Reserved 101b - Reserved 111b - IN_FIELD

39.3.4 LC Scan Status (DCMLCS)

Offset

Register	Offset
DCMLCS	8h

Function

Stores the status of LC scanning. By default, the status of each LC is "not yet scanned."

This register:

- Resets on destructive reset.
- Always returns 0 in a valid IN_FIELD LC (in LC without an error).

This register captures the errors related to LC scanning on each of these resets: POR, destructive, and functional. If an error is captured, its status in this register is cleared by writing 1 to the corresponding field or to any of the destructive or POR events.

All LC slot errors are captured and cleared independently.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		DCMLCFE5	DCMLCE5	DCMLCC5			DCMLCSS5	DCMLCFE4	DCMLCE4	DCMLCC4			DCMLCSS4	DCMLCFE3	DCMLCE3
W			W1C	W1C	W1C			W1C	W1C	W1C	W1C			W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DCMLCC3			DCMLCSS3	Reserved											
W	W1C			W1C	W1C											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29 DCMLCFE5	Pre-FA Flash Memory Error Check Indicates the status of the Pre-FA flash memory error check. 0b - No errors 1b - Errors exist
28 DCMLCE5	Pre-FA ECC Errors Indicates if Pre-FA is successful or has ECC errors. 0b - No errors 1b - Errors exist
27-25 DCMLCC5	Pre-FA Marking Status Indicates the Pre-FA marking status. These errors may cause this field to indicate the "not scanned yet" status: • If the reading completes too early and DCM has not yet scanned the LC. • If there is an error in the flash memory after completion of the reading. 000b - Not scanned yet 001b - Marked as active 010b - Marked as inactive 011b - Region is erased/virgin 101b - Marked as inactive by an unknown pattern 110b - Scanning timed out
24 DCMLCSS5	Pre-FA Scan Status Indicates the status of the Pre-FA scan. 0b - No errors 1b - Errors exist
23 DCMLCFE4	IN_FIELD Flash Memory Error Check Indicates the status of IN_FIELD flash memory error check. 0b - No errors 1b - Errors exist
22 DCMLCE4	IN_FIELD ECC Errors Indicates if IN_FIELD has ECC errors.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No errors 1b - Errors exist
21-19 DCMLCC4	IN_FIELD Marking Status Indicates the IN_FIELD marking status. These errors may cause this field to indicate the "not scanned yet" status: • If the reading completes too early and DCM has not yet scanned the LC. • If there is an error in the flash memory after completion of the reading. 000b - Not scanned yet 001b - Marked as active 010b - Marked as inactive 011b - Region is erased/virgin 101b - Marked as inactive by an unknown pattern 110b - Scanning timed out
18 DCMLCSS4	IN_FIELD Scan Status Indicates the status of the IN_FIELD scan. 0b - No errors 1b - Errors exist
17 DCMLCFE3	OEM_PROD Flash Memory Error Check Indicates the status of the OEM_PROD flash memory error check. 0b - No errors 1b - Errors exist
16 DCMLCE3	OEM_PROD ECC Errors Indicates if OEM_PROD has ECC errors. 0b - No errors 1b - Errors exist
15-13 DCMLCC3	OEM_PROD Marking Indicates the OEM_PROD marking status. These errors may cause this field to indicate the "not scanned yet" status: • If the reading completes too early and DCM has not yet scanned the LC. • If there is an error in the flash memory after completion of the reading. 000b - Not scanned yet

Table continues on the next page...

Table continued from the previous page...

Field	Function
	001b - Marked as active 010b - Marked as inactive 011b - Region is erased/virgin 101b - Marked as inactive by an unknown pattern 110b - Scanning timed out
12 DCMLCSS3	OEM_PROD Scan Status Indicates the status of OEM_PROD scan. 0b - No errors 1b - Errors exist
11-0 —	Reserved

39.3.5 DCM Miscellaneous (DCMMISC)

Offset

Register	Offset
DCMMISC	1Ch

Function

Resets on destructive reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	DCMCERS	0				0							
W				W1C												
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		0		DCMDBGE	DCMDBGT	0									
W					W1C	W1C										
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Fields

Field	Function
31 —	Reserved
30 —	Reserved
29 —	Reserved
28 DCMCERS	DCF Client Errors Records the status of errors from DCF clients. 0b - No errors on any of the DCF clients 1b - Atleast one safety DCF client has an error
27-24 —	Reserved
23-14 —	Reserved
13-12 —	Reserved
11 DCMDBGE	DCM ECC error on DBG sections This bit is set if there is any ECC error during scanning of CUST_PWD, or UID. 0b - No ECC error 1b - ECC error
10 DCMDBGT	DBG Section Timeout Error Indicates if there is a DCM UTEST flash memory timeout error in DBG sections. The value of this field is 1 in case a timeout error occurs when scanning CUST_PWD, or UID. 0b - No error 1b - Error exists
9-0 —	Reserved

39.3.6 Debug Status and Configuration (DCMDEB)

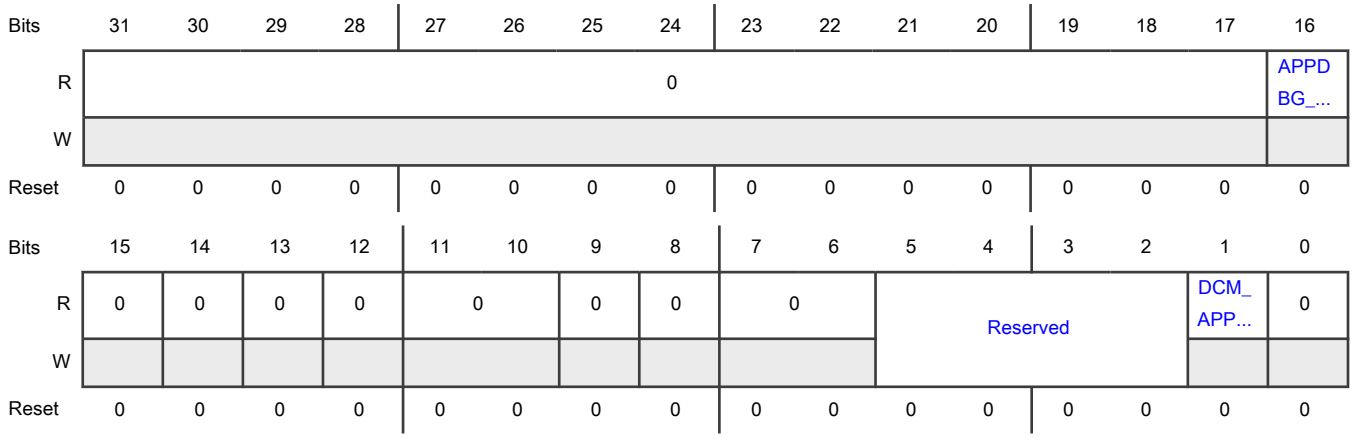
Offset

Register	Offset
DCMDEB	20h

Function

Resets on destructive reset.

Diagram



Fields

Field	Function
31-17 —	Reserved
16 APPDBG_STAT_SOC	Application Debug Status Indicates the application debug status of the chip. 0b - Disabled 1b - Enabled
15 —	Reserved
14 —	Reserved
13 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
12 —	Reserved
11-10 —	Reserved
9 —	Reserved
8 —	Reserved
7-6 —	Reserved
5-2 —	Reserved
1 DCM_APPDBG_STAT	DCM Authentication Engine Status for Application Core **NOTE** This bit will read 0 in non-export control mode.
0 —	Reserved

39.3.7 DCF Error Count (DCMEC)

Offset

Register	Offset
DCMEC	2Ch

Function

Indicates the number of faulty DCF records.

This register resets on destructive reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DCMECT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-0 DCMECT	DCF Error Count Indicates the number of faulty DCF records.

39.3.8 DCF Scan Report (DCMSRR1)

Offset

Register	Offset
DCMSRR1	30h

Function

Resets on destructive reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		DCMDCFT1	DCMESD1	DCMESF1	DCMDCFF1			0				DCMDCFE1			
W			W1C	W1C	W1C	W1C							W1C			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DCMDCFE1															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29 DCMDCFT1	Scanning Timeout On Flash Memory Indicates if scanning timeout exists on flash memory address. 0b - Does not exist 1b - Exists
28 DCMESD1	Chip Side Error Indicates if an error exists on chip side. These errors could be parity errors or the ones reported by the DCF client, such as write-once error. 0b - No errors 1b - Errors exist
27 DCMESF1	Flash Memory Error Indicates if a UTEST flash memory ECC error exists. 0b - No errors 1b - Errors exist
26-24 DCMDCFF1	DCF Record Location Indicates the DCF record location. 010b - Utest flash memory 101b - Others: Reserved
23-21 —	Reserved
20-0 DCMDCFE1	Flash Memory Address Indicates the flash memory address of the DCF client having an error.

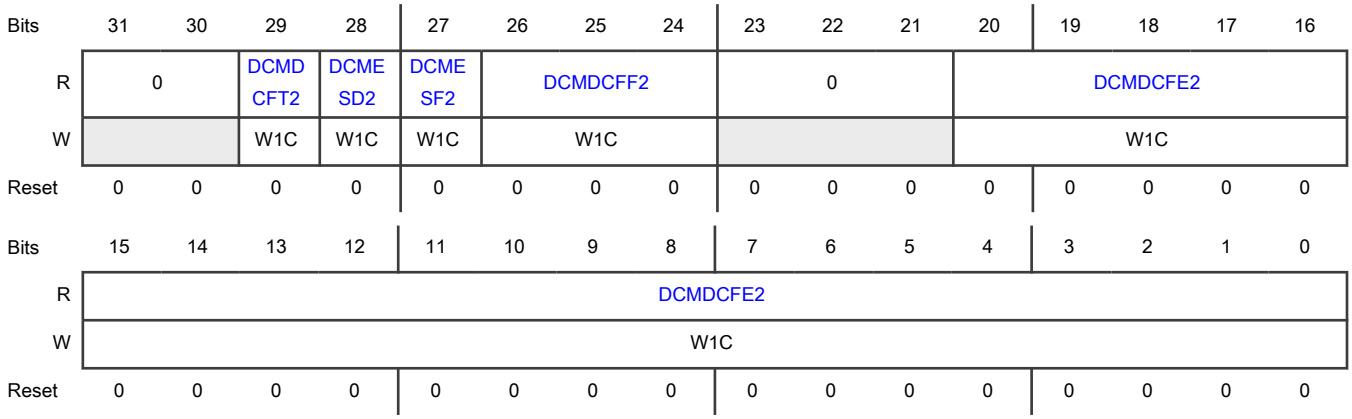
39.3.9 DCF Scan Report (DCMSRR2)

Offset

Register	Offset
DCMSRR2	34h

Function
Resets on destructive reset.

Diagram



Fields

Field	Function
31-30 —	Reserved
29 DCMDCFT2	Scanning Timeout On Flash Memory Indicates if scanning timeout exists on flash memory address. 0b - Does not exist 1b - Exists
28 DCMESD2	Chip Side Error Indicates if an error exists on chip side. These errors could be parity errors or the ones reported by the DCF client, such as write-once error. 0b - No errors 1b - Errors exist
27 DCMESF2	Flash Memory Error Indicates if a UTEST flash memory ECC error exists. 0b - No errors 1b - Errors exist
26-24 DCMDCFF2	DCF Record Location Indicates the DCF record location. 010b - Utest flash memory 101b - Others: Reserved
23-21 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
20-0 DCMDCFE2	Flash Memory Address Indicates the flash memory address of the DCF client having an error.

39.3.10 DCF Scan Report (DCMSRR3)

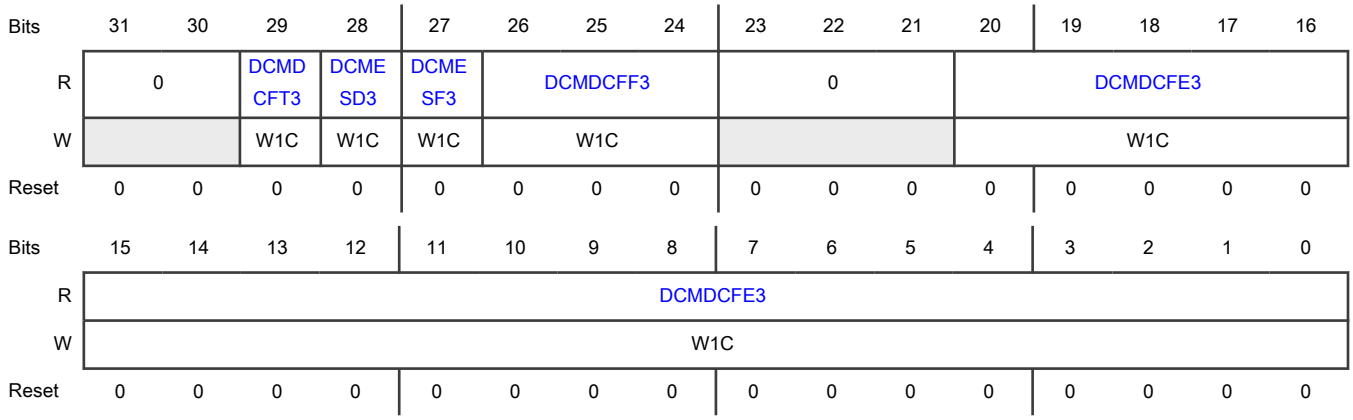
Offset

Register	Offset
DCMSRR3	38h

Function

Resets on destructive reset.

Diagram



Fields

Field	Function
31-30 —	Reserved
29 DCMDCFT3	Scanning Timeout On Flash Memory Indicates if scanning timeout exists on flash memory address. 0b - Does not exist 1b - Exists
28 DCMESD3	Chip Side Error

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Indicates if an error exists on chip side. These errors could be parity errors or the ones reported by the DCF client, such as write-once error. 0b - No errors 1b - Errors exist
27 DCMESF3	Flash Memory Error Indicates if a UTEST flash memory ECC error exists. 0b - No errors 1b - Errors exist
26-24 DCMDCFF3	DCF Record Location Indicates the DCF record location. 010b - Utest flash memory 101b - Others: Reserved
23-21 —	Reserved
20-0 DCMDCFE3	Flash Memory Address Indicates the flash memory address of the DCF client having an error.

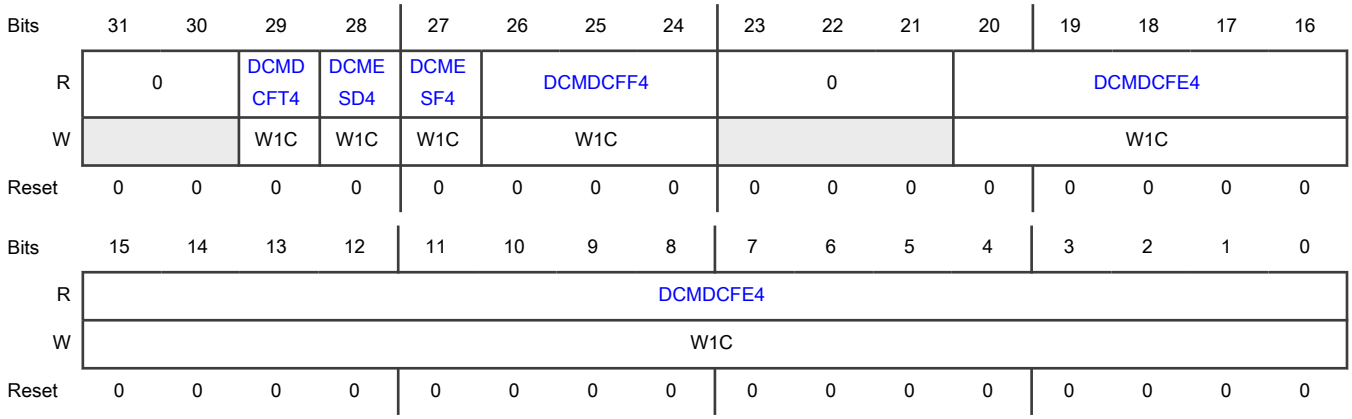
39.3.11 DCF Scan Report (DCMSRR4)

Offset

Register	Offset
DCMSRR4	3Ch

Function
Resets on destructive reset.

Diagram



Fields

Field	Function
31-30 —	Reserved
29 DCMDCFT4	Scanning Timeout On Flash Memory Indicates if scanning timeout exists on flash memory address. 0b - Does not exist 1b - Exists
28 DCMESD4	Chip Side Error Indicates if an error exists on chip side. These errors could be parity errors or the ones reported by the DCF client, such as write-once error. 0b - No errors 1b - Errors exist
27 DCMESF4	Flash Memory Error Indicates if a UTEST flash memory ECC error exists. 0b - No errors 1b - Errors exist
26-24 DCMDCFF4	DCF Record Location Indicates the DCF record location. 010b - Utest flash memory 101b - Others: Reserved
23-21 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
20-0 DCMDCFE4	Flash Memory Address Indicates the flash memory address of the DCF client having an error.

39.3.12 DCF Scan Report (DCMSRR5)

Offset

Register	Offset
DCMSRR5	40h

Function
Resets on destructive reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		DCMDCFT5	DCMESD5	DCMESF5	DCMDCFF5			0				DCMDCFE5			
W			W1C	W1C	W1C	W1C							W1C			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DCMDCFE5															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29 DCMDCFT5	Scanning Timeout On Flash Memory Indicates if scanning timeout exists on flash memory address. 0b - Does not exist 1b - Exists
28 DCMESD5	Chip Side Error

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Indicates if an error exists on chip side. These errors could be parity errors or the ones reported by the DCF client, such as write-once error. 0b - No errors 1b - Errors exist
27 DCMESF5	Flash Memory Error Indicates if a UTEST flash memory ECC error exists. 0b - No errors 1b - Errors exist
26-24 DCMDCFF5	DCF Record Location Indicates the DCF record location. 010b - Utest flash memory 101b - Others: Reserved
23-21 —	Reserved
20-0 DCMDCFE5	Flash Memory Address Indicates the flash memory address of the DCF client having an error.

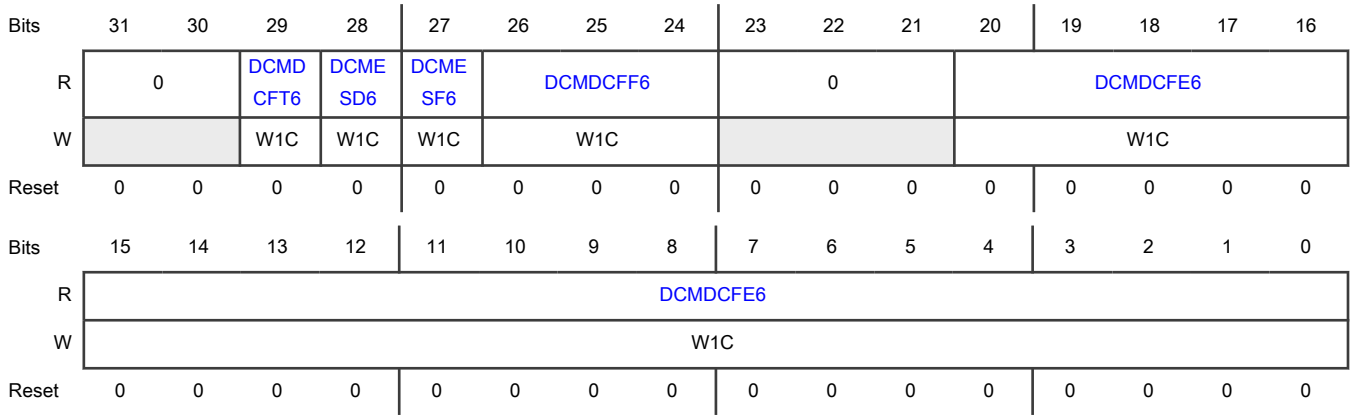
39.3.13 DCF Scan Report (DCMSRR6)

Offset

Register	Offset
DCMSRR6	44h

Function
Resets on destructive reset.

Diagram



Fields

Field	Function
31-30 —	Reserved
29 DCMDCFT6	Scanning Timeout On Flash Memory Indicates if scanning timeout exists on flash memory address. 0b - Does not exist 1b - Exists
28 DCMESD6	Chip Side Error Indicates if an error exists on chip side. These errors could be parity errors or the ones reported by the DCF client, such as write-once error. 0b - No errors 1b - Errors exist
27 DCMESF6	Flash Memory Error Indicates if a UTEST flash memory ECC error exists. 0b - No errors 1b - Errors exist
26-24 DCMDCFF6	DCF Record Location Indicates the DCF record location. 010b - Utest flash memory 101b - Others: Reserved
23-21 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
20-0 DCMDCFE6	Flash Memory Address Indicates the flash memory address of the DCF client having an error.

39.3.14 DCF Scan Report (DCMSRR7)

Offset

Register	Offset
DCMSRR7	48h

Function

Resets on destructive reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		DCMDCFT7	DCMESD7	DCMESF7	DCMDCFF7			0				DCMDCFE7			
W			W1C	W1C	W1C	W1C							W1C			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DCMDCFE7															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29 DCMDCFT7	Scanning Timeout On Flash Memory Indicates if scanning timeout exists on flash memory address. 0b - Does not exist 1b - Exists
28 DCMESD7	Chip Side Error

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Indicates if an error exists on chip side. These errors could be parity errors or the ones reported by the DCF client, such as write-once error. 0b - No errors 1b - Errors exist
27 DCMESF7	Flash Memory Error Indicates if a UTEST flash memory ECC error exists. 0b - No errors 1b - Errors exist
26-24 DCMDCFF7	DCF Record Location Indicates the DCF record location. 010b - Utest flash memory 101b - Others: Reserved
23-21 —	Reserved
20-0 DCMDCFE7	Flash Memory Address Indicates the flash memory address of the DCF client having an error.

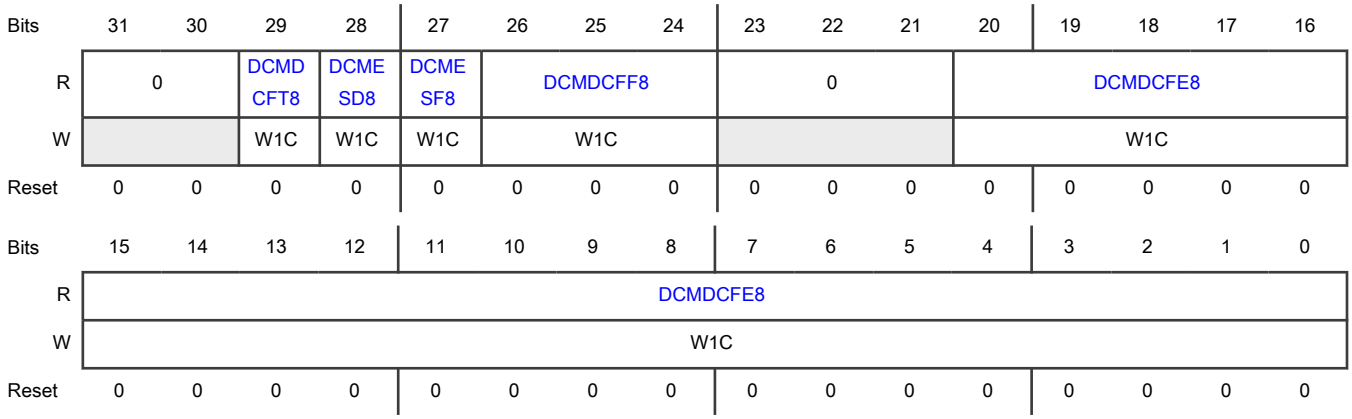
39.3.15 DCF Scan Report (DCMSRR8)

Offset

Register	Offset
DCMSRR8	4Ch

Function
Resets on destructive reset.

Diagram



Fields

Field	Function
31-30 —	Reserved
29 DCMDCFT8	Scanning Timeout On Flash Memory Indicates if scanning timeout exists on flash memory address. 0b - Does not exist 1b - Exists
28 DCMESD8	Chip Side Error Indicates if an error exists on chip side. These errors could be parity errors or the ones reported by the DCF client, such as write-once error. 0b - No errors 1b - Errors exist
27 DCMESF8	Flash Memory Error Indicates if a UTEST flash memory ECC error exists. 0b - No errors 1b - Errors exist
26-24 DCMDCFF8	DCF Record Location Indicates the DCF record location. 010b - Utest flash memory 101b - Others: Reserved
23-21 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
20-0 DCMDCFE8	Flash Memory Address Indicates the flash memory address of the DCF client having an error.

39.3.16 DCF Scan Report (DCMSRR9)

Offset

Register	Offset
DCMSRR9	50h

Function

Resets on destructive reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		DCMDCFT9	DCMESD9	DCMESF9	DCMDCFF9			0				DCMDCFE9			
W			W1C	W1C	W1C	W1C							W1C			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DCMDCFE9															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29 DCMDCFT9	Scanning Timeout On Flash Memory Indicates if scanning timeout exists on flash memory address. 0b - Does not exist 1b - Exists
28 DCMESD9	Chip Side Error

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Indicates if an error exists on chip side. These errors could be parity errors or the ones reported by the DCF client, such as write-once error. 0b - No errors 1b - Errors exist
27 DCMESF9	Flash Memory Error Indicates if a UTEST flash memory ECC error exists. 0b - No errors 1b - Errors exist
26-24 DCMDCFF9	DCF Record Location Indicates the DCF record location. 010b - Utest flash memory 101b - Others: Reserved
23-21 —	Reserved
20-0 DCMDCFE9	Flash Memory Address Indicates the flash memory address of the DCF client having an error.

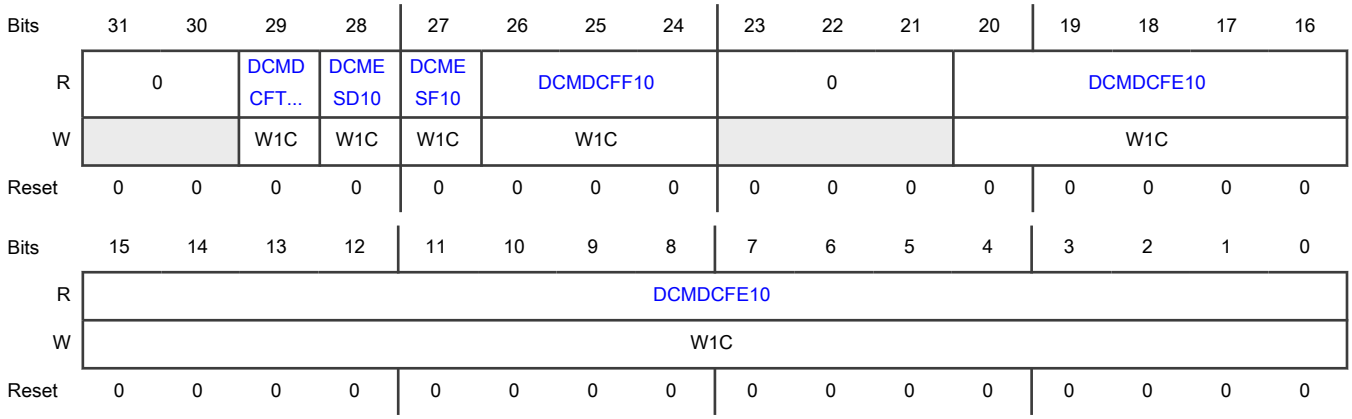
39.3.17 DCF Scan Report (DCMSRR10)

Offset

Register	Offset
DCMSRR10	54h

Function
Resets on destructive reset.

Diagram



Fields

Field	Function
31-30 —	Reserved
29 DCMDCFT10	Scanning Timeout On Flash Memory Indicates if scanning timeout exists on flash memory address. 0b - Does not exist 1b - Exists
28 DCMESD10	Chip Side Error Indicates if an error exists on chip side. These errors could be parity errors or the ones reported by the DCF client, such as write-once error. 0b - No errors 1b - Errors exist
27 DCMESF10	Flash Memory Error Indicates if a UTEST flash memory ECC error exists. 0b - No errors 1b - Errors exist
26-24 DCMDCFF10	DCF Record Location Indicates the DCF record location. 010b - Utest flash memory 101b - Others: Reserved
23-21 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
20-0 DCMDCFE10	Flash Memory Address Indicates the flash memory address of the DCF client having an error.

39.3.18 DCF Scan Report (DCMSRR11)

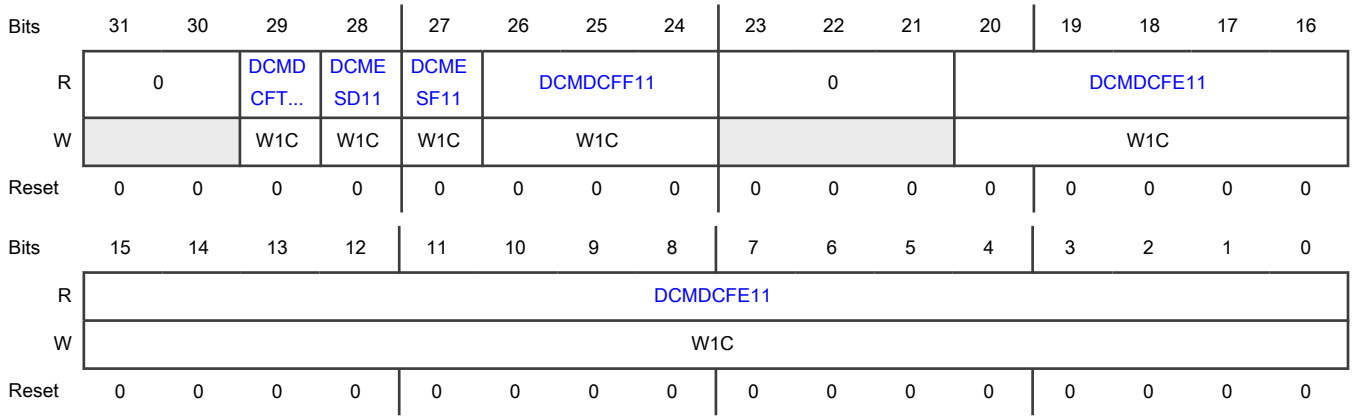
Offset

Register	Offset
DCMSRR11	58h

Function

Resets on destructive reset.

Diagram



Fields

Field	Function
31-30 —	Reserved
29 DCMDCFT11	Scanning Timeout On Flash Memory Indicates if scanning timeout exists on flash memory address. 0b - Does not exist 1b - Exists
28 DCMESD11	Chip Side Error

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Indicates if an error exists on chip side. These errors could be parity errors or the ones reported by the DCF client, such as write-once error. 0b - No errors 1b - Errors exist
27 DCMESF11	Flash Memory Error Indicates if a UTEST flash memory ECC error exists. 0b - No errors 1b - Errors exist
26-24 DCMDCFF11	DCF Record Location Indicates the DCF record location. 010b - Utest flash memory 101b - Others: Reserved
23-21 —	Reserved
20-0 DCMDCFE11	Flash Memory Address Indicates the flash memory address of the DCF client having an error.

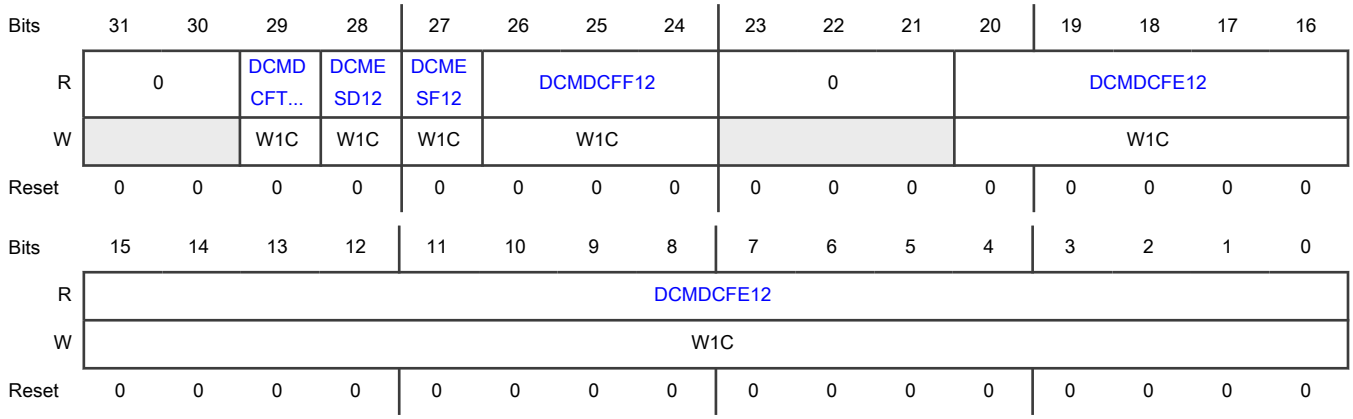
39.3.19 DCF Scan Report (DCMSRR12)

Offset

Register	Offset
DCMSRR12	5Ch

Function
Resets on destructive reset.

Diagram



Fields

Field	Function
31-30 —	Reserved
29 DCMDCFT12	Scanning Timeout On Flash Memory Indicates if scanning timeout exists on flash memory address. 0b - Does not exist 1b - Exists
28 DCMESD12	Chip Side Error Indicates if an error exists on chip side. These errors could be parity errors or the ones reported by the DCF client, such as write-once error. 0b - No errors 1b - Errors exist
27 DCMESF12	Flash Memory Error Indicates if a UTEST flash memory ECC error exists. 0b - No errors 1b - Errors exist
26-24 DCMDCFF12	DCF Record Location Indicates the DCF record location. 010b - Utest flash memory 101b - Others: Reserved
23-21 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
20-0 DCMDCFE12	Flash Memory Address Indicates the flash memory address of the DCF client having an error.

39.3.20 DCF Scan Report (DCMSRR13)

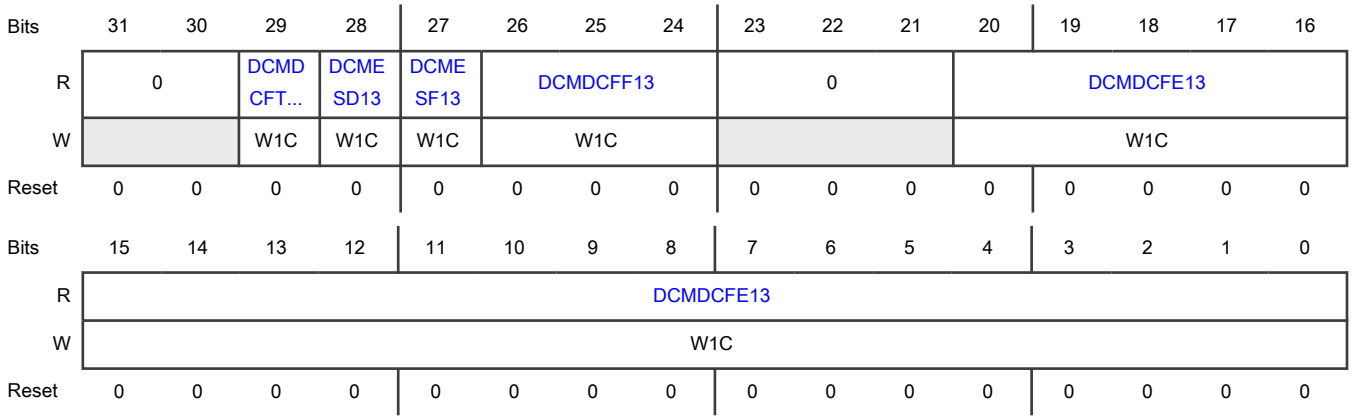
Offset

Register	Offset
DCMSRR13	60h

Function

Resets on destructive reset.

Diagram



Fields

Field	Function
31-30 —	Reserved
29 DCMDCFT13	Scanning Timeout On Flash Memory Indicates if scanning timeout exists on flash memory address. 0b - Does not exist 1b - Exists
28 DCMESD13	Chip Side Error

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Indicates if an error exists on chip side. These errors could be parity errors or the ones reported by the DCF client, such as write-once error. 0b - No errors 1b - Errors exist
27 DCMESF13	Flash Memory Error Indicates if a UTEST flash memory ECC error exists. 0b - No errors 1b - Errors exist
26-24 DCMDCFF13	DCF Record Location Indicates the DCF record location. 010b - Utest flash memory 101b - Others: Reserved
23-21 —	Reserved
20-0 DCMDCFE13	Flash Memory Address Indicates the flash memory address of the DCF client having an error.

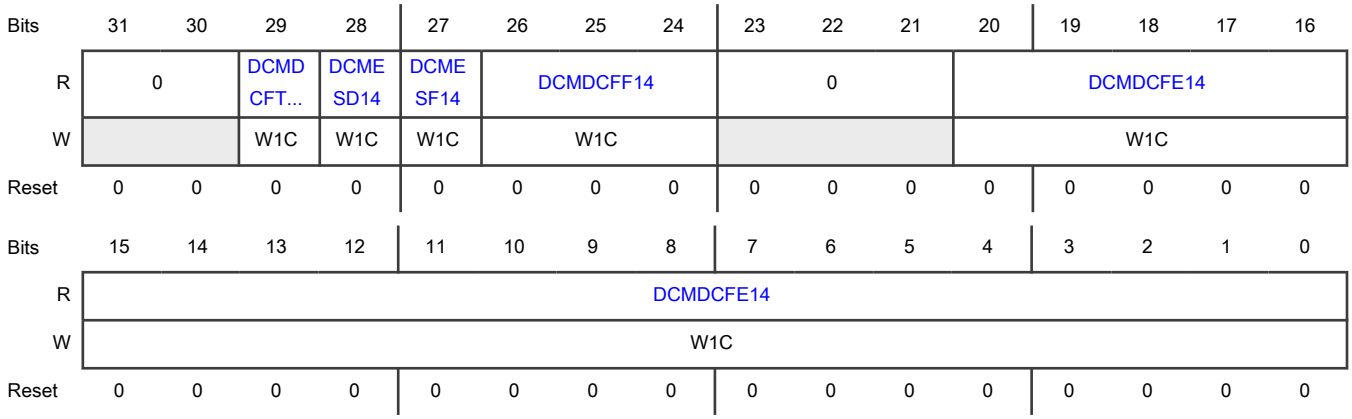
39.3.21 DCF Scan Report (DCMSRR14)

Offset

Register	Offset
DCMSRR14	64h

Function
Resets on destructive reset.

Diagram



Fields

Field	Function
31-30 —	Reserved
29 DCMDCFT14	Scanning Timeout On Flash Memory Indicates if scanning timeout exists on flash memory address. 0b - Does not exist 1b - Exists
28 DCMESD14	Chip Side Error Indicates if an error exists on chip side. These errors could be parity errors or the ones reported by the DCF client, such as write-once error. 0b - No errors 1b - Errors exist
27 DCMESF14	Flash Memory Error Indicates if a UTEST flash memory ECC error exists. 0b - No errors 1b - Errors exist
26-24 DCMDCFF14	DCF Record Location Indicates the DCF record location. 010b - Utest flash memory 101b - Others: Reserved
23-21 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
20-0 DCMDCFE14	Flash Memory Address Indicates the flash memory address of the DCF client having an error.

39.3.22 DCF Scan Report (DCMSRR15)

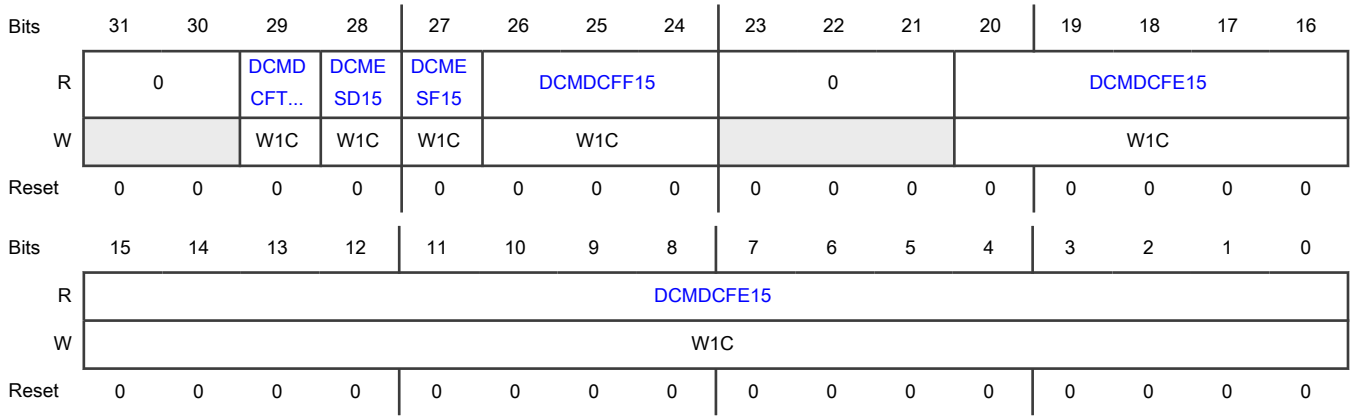
Offset

Register	Offset
DCMSRR15	68h

Function

Resets on destructive reset.

Diagram



Fields

Field	Function
31-30 —	Reserved
29 DCMDCFT15	Scanning Timeout On Flash Memory Indicates if scanning timeout exists on flash memory address. 0b - Does not exist 1b - Exists
28 DCMESD15	Chip Side Error

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Indicates if an error exists on chip side. These errors could be parity errors or the ones reported by the DCF client, such as write-once error. 0b - No errors 1b - Errors exist
27 DCMESF15	Flash Memory Error Indicates if a UTEST flash memory ECC error exists. 0b - No errors 1b - Errors exist
26-24 DCMDCFF15	DCF Record Location Indicates the DCF record location. 010b - Utest flash memory 101b - Others: Reserved
23-21 —	Reserved
20-0 DCMDCFE15	Flash Memory Address Indicates the flash memory address of the DCF client having an error.

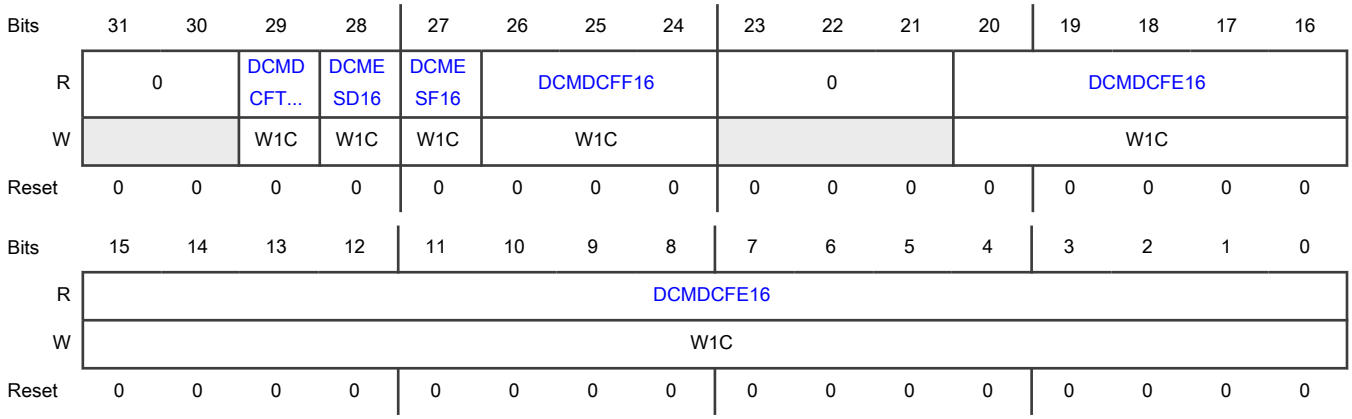
39.3.23 DCF Scan Report (DCMSRR16)

Offset

Register	Offset
DCMSRR16	6Ch

Function
Resets on destructive reset.

Diagram



Fields

Field	Function
31-30 —	Reserved
29 DCMDCFT16	Scanning Timeout On Flash Memory Indicates if scanning timeout exists on flash memory address. 0b - Does not exist 1b - Exists
28 DCMESD16	Chip Side Error Indicates if an error exists on chip side. These errors could be parity errors or the ones reported by the DCF client, such as write-once error. 0b - No errors 1b - Errors exist
27 DCMESF16	Flash Memory Error Indicates if a UTEST flash memory ECC error exists. 0b - No errors 1b - Errors exist
26-24 DCMDCFF16	DCF Record Location Indicates the DCF record location. 010b - Utest flash memory 101b - Others: Reserved
23-21 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
20-0 DCMDCFE16	Flash Memory Address Indicates the flash memory address of the DCF client having an error.

39.3.24 LC Scan Status 2 (DCMLCS_2)

Offset

Register	Offset
DCMLCS_2	80h

Function

Stores the status of LC scanning. By default, the status of each LC is "not yet scanned."

This register:

- Resets on destructive reset.
- Always returns 0 in a valid IN_FIELD LC (in LC without an error).

This register captures the errors related to LC scanning on each of these resets: POR, destructive, and functional. If an error is captured, its status in this register is cleared by writing 1 to the corresponding field or to any of the destructive or POR events.

All LC slot errors are captured and cleared independently.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0										DCMLCFE6	DCMLCE6	DCMLCC6			DCMLCSS6
W											W1C	W1C	W1C			W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-6 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
5 DCMLCFE6	Flash Memory Error Check Indicates the status of flash memory check. 0b - No errors 1b - Errors exist
4 DCMLCE6	FA ECC Errors Indicates if ECC errors exist in FA. 0b - No errors 1b - Errors exist
3-1 DCMLCC6	FA Marking Indicates the FA marking status. These errors may cause this field to indicate the "not scanned yet" status: • If the reading completes too early and DCM has not yet scanned the LC. • If there is an error in the flash memory after completion of the reading. 000b - Not scanned yet 001b - Marked as active 010b - Marked as inactive 011b - Region is erased/virgin 101b - Marked as inactive by an unknown pattern 110b - Scanning timed out
0 DCMLCSS6	FA Scan Status Indicates if errors exist in the FA scan. 0b - No errors 1b - Errors exist

39.4 Glossary

FA	Failure analysis
LC	Life cycle
Pre-FA	Pre-failure analysis
RoT	Root of trust
OTP	One Time Programmable
DCF	Device Configuration Format
PFC	Platform Flash Controller

Chapter 40
Messaging Unit (MU)

40.1 Chip-specific MU information

40.1.1 MU instances and configuration

This chip includes MUs for communication across the different cores. Each MU includes two interfaces, MUA and MUB. Two different processors control them for communication.

Table 234 indicates the MUs and their interfaces present in different parts of the S32K3 chip family. The table also summarizes the implementation of this module in each chip of the S32K3 product series.

NOTE

The HSE_B core controls the MUA interface of MU_0 and MU_1. Therefore, the application core cannot control the interface.

Table 234. MU instances

Instance	S32K388, S32K389	S32K328, S32K338, S32K358, S32K356	S32K348	S32K344	S32K324	S32K314	S32K342, S32K341	S32K322	S32K312	S32K311, S32K310	Use case
MU_0	Yes										Communication between HSE_B and application cores
MU_1	Yes										
MU_2	Yes		No	Yes	No		Yes		No		Communication between application cores
MU_3	Yes	No									
MU_4	Yes	No									

The base address of MU_1's MUB interface is different across the S32K3xx product series because AIPS_2 is unavailable in S32K312 and S32K311. Table 235 summarizes this difference.

Table 235. Base-address difference in MU_1's MUB interface across S32K3xx product series

	S32K388, S32K389, S32K328, S32K338, S32K348, S32K358, S32K356, S32K344, S32K324, S32K314, S32K342, S32K341, S32K322	S32K312, S32K311, S32K310
Base address	404E_C000h	4039_0000h

NOTE

- For S32K344/S32K324/S32K314/S32K312/S32K311/S32K310 reset value of MUA_VER and MUB_VER register is 0300_000Fh.
- For others, reset value of MUA_VER and MUB_VER register is 0309_000Fh.

40.2 Overview

MU enables two processors on a chip to communicate and coordinate by passing messages (for example, data, status, and control) through the MU interface. MU also allows one processor to signal the other processor using interrupts.

MU must synchronize the accesses from one side to the other because MU can manage messaging between processors using different clocks. MU accomplishes synchronization using two sets of matching registers: processor A-facing and processor B-facing.

40.2.1 Block diagram

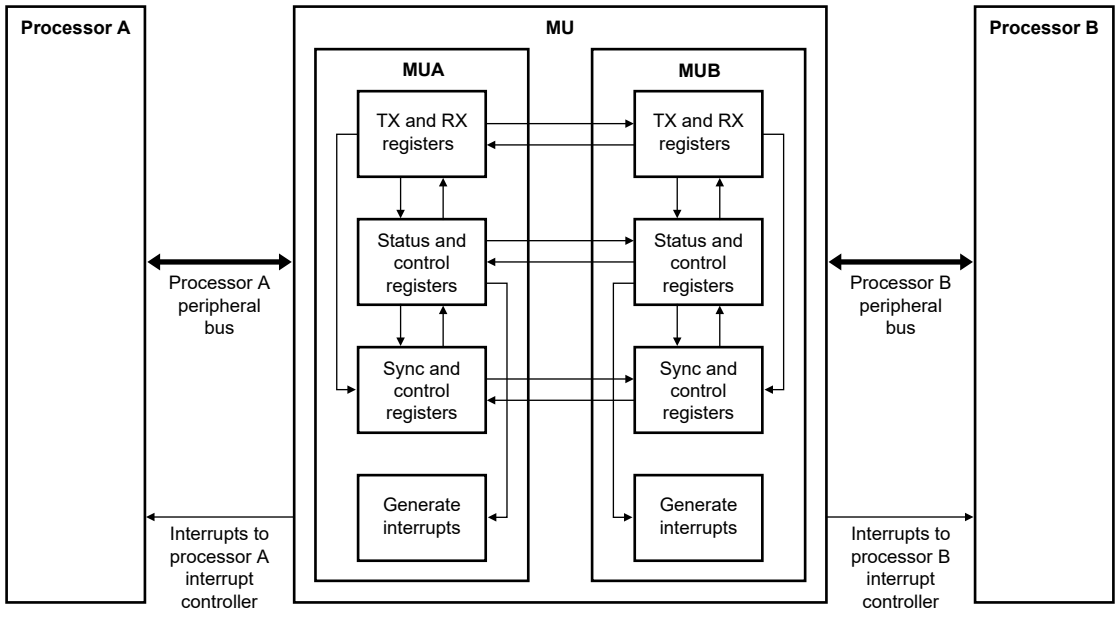


Figure 165. Block diagram

40.2.2 Features

- Memory-mapped registers (MU is connected as a peripheral under the peripheral bus on the processor A-side and processor B-side)
- Synchronized message transfers between cores
  - To send data or messages from one side to the other, MUA provides 4 transmit registers and 4 receive registers. MUB provides 4 transmit registers and 4 receive registers.
  - Transmit empty flags (TSR[TEn]) and receive full flags (RSR[RFn]) facilitate the transfer of data or messages between cores on both sides of MU.
  - A synchronization mechanism updates the transmit and the receive flags. There is an inherent latency between updating the flag on one side and reflecting its status on the other side. See Event update timing.
  - MU has a 3-bit flag data register, which you can use to send flag data between the two MU sides.
- Interprocessor interrupts
  - MU has 9 asynchronous interrupt sources on each side (processor A-side, processor B-side) for signaling the other processor. You can use the interrupts for notification of receive and transmit events and for general-purpose signaling between processors. There are 1 general-purpose interrupt requests available and 8 receive and transmit interrupt sources.
- Reset (Each processor can issue a reset to MU via CR[MUR], which is a self-clearing field).

40.3 Functional description

MU enables two cores (processor A and processor B) to communicate with each other:

- By sharing messages and data.
- By enabling one core to wake the other core by using interrupts.

The messaging, control, and status registers of the two cores are mapped to processor A memory and processor B memory as a regular peripheral. The peripheral data bus is 32 bits wide inside MU.

Messaging logic is used with shared memory. Various messaging methods can implement a messaging protocol. For example, a message could mean one of the following:

- A message of *n* words has been written starting at offset *x* in the memory.
- The previous data block that was sent has been read.

The ability to keep messaging logic independent of the shared memory is not restricted to a predefined hardware protocol. The software required to manage the messaging is short and straightforward.

Most of the messaging mechanisms are symmetric. They are duplicated and are available on both the processor A side and processor B side. The messaging mechanisms are:

- 4 32-bit transmit registers, which are each reflected in 4 read-only receive registers on the side of the other processor. These registers can transfer 32-bit word messages or the frame information of the messages written to the shared memory. For example, they can transfer the number of words, initial address, and message type code.
- Writing to a transmitter-side transmit register (TR*n*) clears the Transmitter Empty flag in the transmitter-side Status register (TS*n*[TE*n*]), and sets the Receiver Full flag in the receiver-side Status register (RS*n*[RF*n*]). Setting the flag on the receiver side can trigger an interrupt on the receiver side (a maskable receive interrupt).
- Reading a receiver-side receive register (RR*n*) clears the Receiver Full flag in the receiver-side Status register (RS*n*[RF*n*]), and sets the Transmitter Empty flag in the transmitter-side Status register (TS*n*[TE*n*]). Setting the Transmitter Empty flag can trigger an interrupt on the transmitter side (a maskable transmit interrupt).
- 1 general-purpose interrupt request flags are reflected in General-purpose Status (GSR) on the receiver side.
- 3-bit flag data is transmitted from Flag Control (FCR) to Flag Status (FSR) on the side receiving the flag data. SR[FUP] sets when the flag data is transmitted and clears when the receiving side acknowledges the flag data (the flag is updated).

Writing to a transmit register signals to the receiver side that data is ready for retrieval.

Do not write to the transmit register again without verifying that the data is retrieved. The transmitter side cannot determine the exact time that the receiver attempts to retrieve the data. Before attempting to write to the transmit register again, the transmitter side must wait for the Transmitter Empty interrupt or it must poll TSR[TE*n*].

Reading a receive register signals to the transmitter side that data can be written to that register.

Do not read the receive register again without verifying that the data is written. The receiver side cannot determine the exact time that the transmitter attempts to write the data. Before attempting to read the receive register again, the receiver side must wait for the Receiver Full interrupt or it must poll RSR[RF*n*].

40.3.1 Submodules

40.3.1.1 MUA side

MUA receives its register configuration via the processor A peripheral bus. It sends or receives messages to or from MUB. Processor A can receive messages by reading MUA registers, and MUA can send interrupts to processor A when interrupts are enabled.

NOTE

Processor B should not access MUA registers. Processor C, if exist, should also not access MUA registers. TRDC might be needed to prevent such illegal access of Processor B or C to MUA registers.

40.3.1.2 MUB side

MUB receives its register configuration via the processor B peripheral bus. It sends or receives messages to or from MUA. Processor B can receive messages by reading MUB registers, and MUB can send interrupts to processor B when interrupts are enabled.

NOTE

Processor A should not access MUB registers. Processor C, if exist, should also not access MUB registers. TRDC might be needed to prevent such illegal access of Processor A or C to MUB registers.

40.3.2 Event update timing

The messaging side of each processor has a hardware mechanism to send event update requests to the other processor. An event occurs when the status register of the receiving processor must reflect any information change. The event update latency is the delay between the event being ready at one processor and the resulting update at the status register of the other processor:

- The minimum event latency is (1 clock cycle of the sending side) + (2.5 clock cycles of the receiving side). This minimum case happens when no event is pending when the new event occurs.
- The maximum event latency is (6 clock cycles of the sending side) + (6.5 clock cycles of the receiving side). The maximum case happens when the event occurs just after a previous event is sent to the other side.

The event update latency varies depending on the time at which the subsequent event is triggered.

40.3.3 Clocking

Table 236. MU clocks

Clock name	Description
Bus clock MUA	Used only for bus accesses to MUA control and configuration registers
Bus clock MUB	Used only for bus accesses to MUB control and configuration registers

40.3.4 Resets

The following sections list the reset sources included in MU. Each reset performs a different function for the MU module compared to the function it performs for the system.

40.3.4.1 Asynchronous system reset

When the asynchronous system reset on either side of MU is asserted, SR[MURS] becomes 1 until the asynchronous system reset sequence on both the MUA and MUB sides ends. Verify that SR[MURS] becomes 0 before accessing MU.

The asynchronous system reset on one side of MU resets the other side of MU. The reset forces all control and status registers to return to their default values and clears all internal states. The following table shows the exceptions to this behavior.

Table 237. Exceptions to asynchronous system reset

MUA-side exceptions	MUB-side exceptions
MUB_CR[MURIE]	MUA_CR[MURIE]
MUB_SR[MURIP]	MUA_SR[MURIP]
MUB_SR[MURS]	MUA_SR[MURS]

40.3.4.2 MU software reset

Writing 1 to CR[MUR] causes most control and status registers to return to their default values and clears all internal states.

The instruction immediately following the assertion of CR[MUR] must not write to the MU registers. The reset sequence may overwrite such a write, with the register retaining the reset value. To know the end of the reset sequence for both processors, monitor the value of SR[MURS]. After the reset sequence on both processors has ended, a write to the MU registers can be attempted.

NOTE

The process of CR[MUR] becoming 1 is delicate because it asynchronously affects the registers on the other side. CR[MUR] becoming 1 may cause unpredictable behavior if, for example, the other processor is concurrently testing an MU register field (TSR[TE*n*] in the other processor). Before writing 1 to CR[MUR], verify that the other processor is not engaged in an MU signaling activity.

40.3.5 Interrupts

MU controls interrupt requests that one processor makes to the other processor. This section describes all the interrupts that the module generates.

MU can generate these interrupt sources individually to send to the processors:

- 4 receive interrupts (asserted when the Receive Full flags are set in Receive Status (RSR) and enabled in Receive Control (RCR)) for each receive register. These interrupts are Ored into a single interrupt for each MUA or MUB, please consult the interrupt map file attached to this document.
- 4 transmit interrupts (asserted when the Transmit Empty flags are set in Transmit Status (TSR) and enabled in Transmit Control (TCR)) for each transmit register. These interrupts are Ored into a single interrupt for each MUA or MUB, please consult the interrupt map file attached to this document.
- 1 general-purpose interrupt (asserted when the GIP flag is set in General-Purpose Interrupt Enable (GIER) and enabled in General-Purpose Interrupt Enable (GIER))

All interrupts are maskable in the processor control registers: TCR, RCR, GIER, and CR. MU does not assume any internal priority of these interrupts. Multiple interrupts (for example, receive 0 and receive 1, or any transmit or general-purpose interrupt) can be asserted simultaneously. The interrupt controller must resolve the priority of these interrupts at the chip level.

Triggering any enabled interrupt wakes the processor from below mode before servicing the interrupt

The software (as part of the interrupt handler) must clear the general-purpose interrupt pending flag (GSR[GIP0]) to deassert the request to the interrupt controller.

When a processor writes to the general-purpose interrupt flag (GCR[GIR0]), MU synchronizes the write event to the other processor to set the general-interrupt request pending flag (GSR[GIP0]). When GSR[GIP0] is set, if the general-purpose interrupt is enabled on the receiving processor side (GIER[GIE0] is 1), the transmitting-side general-purpose interrupt is issued to the receiving processor, which clears this interrupt by writing 1 to GSR[GIP0]. The interrupt is deasserted as soon as the write to GSR[GIP0] occurs. MU synchronizes the write event of GSR[GIP0] to the other processor. The synchronized signal clears the GIR0 flag.

Before writing 1 to GCR[GIR0], verify that this field is 0, which means that a general interrupt is not pending. Generally, MU ignores writing 1 to this field while the field is already 1, but in some cases it may issue a second interrupt.

40.4 External signals

This module has no external signals.

40.5 Initialization

MU does not require initialization.

40.6 Application information

MU facilitates messages between processors. For example, MU passes:

- Short messages. Transmit registers can pass short messages from 1 to 4 words in length for processor A and from 1 to 4 words for processor B. For example, for a four-word message, only one register must have its corresponding interrupt enabled on the receiving side. The first three words of the message are written to the registers with masked interrupts. The fourth word is written to the last register, triggering an interrupt on the receiving side.
- Frame information. Transmit registers can pass frame information for long messages written to the shared system memory. Such frame information normally includes a start address and a number of words. It can include a message type code.
- Event notices and requests. MU can signal events and requests that do not include data words between processors using general-purpose interrupts. For example, one such event is acknowledging that a long message is read from the shared system memory.
- Fixed-length data. Formatted data with a fixed length can be written to predetermined locations in the shared memory. A processor can use general-purpose interrupts to signal to the other processor that the data is ready.
- Announcements. A processor can use the 3 flags to announce its current program state or other billboard messages to the other processor.

40.6.1 Messaging protocols using interrupts

The example below describes a four-word messaging sequence sent between the processor A and processor B.

The transmitting processor writes to the transmit registers sequentially. When *n* = 0, 1, and 2, the interrupts are disabled, so no interrupt goes to the processor (although interrupt conditions occur). For *n* = 3, the interrupt is enabled, and MU generates the last receive interrupt request.

1. Write sequence:
    a. The transmitting processor writes the message information sequentially to its Transmit registers 0, 1, 2.
    b. When the write to Transmit register 3 occurs, RSR[RF3] is set after synchronization. It immediately triggers the receive full 3 interrupt on the receiving processor.
2. Read sequence:
    a. The receiving processor receives the receive full 3 interrupt and starts reading the message transferred from the receive registers.
    b. After the receiving processor reads Receive register 3, MU clears RSR[RF3].

Table 238 and Figure 166 describe the messaging model, using transmit and receive registers and the interrupt messaging protocol.

Table 238. Interrupt messaging protocol (generalized)

Step	Action	Description
1	Processor A writes data.	RR*n* on processor B's side reflects a data write to TR*n* by processor A.
2	Clear the transmitter empty flag and set the receiver full flag.	The data write to TR*n*: • Clears TSR[TE*n*] on the processor A side. • Sets RSR[RF*n*] on the processor B side.
3	Generate the receive interrupt request.	Setting RSR[RF*n*] generates a receive interrupt request to processor B.
4	Processor B reads the data.	After receiving the receive interrupt request, processor B performs a data read of RR*n*.

Table continues on the next page...

Table 238. Interrupt messaging protocol (generalized) (continued)

Step	Action	Description
5	Clear the receiver full flag and set the transmitter empty flag.	Reading the data from the RR*n* register: • Clears RSR[RF*n*] on the processor B side. • Sets TSR[TE*n*] on the processor A side.
6	Generate the transmit interrupt request.	Setting TSR[TE*n*] generates a transmit interrupt request to processor A.

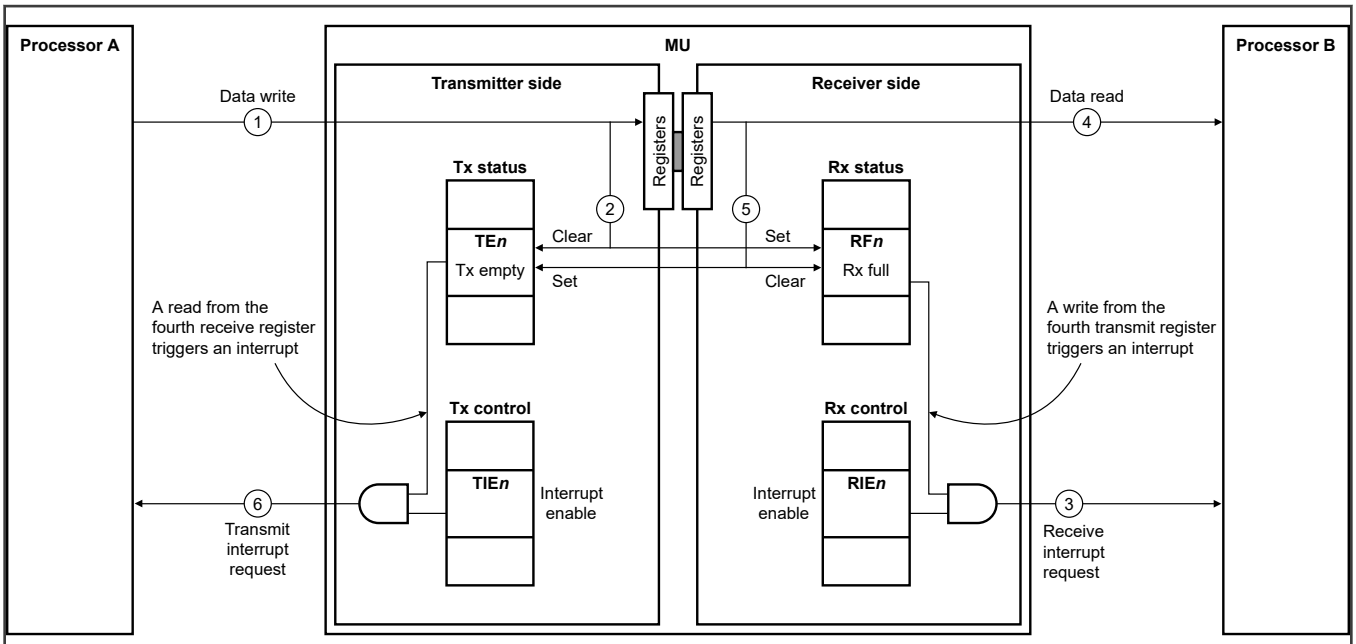


Figure 166. Messaging model using transmit and receive registers

You can use the messaging hardware to implement messaging protocols for an array of message types. MU provides full support for interrupt and polling management schemes.

40.6.2 Messaging protocols using event interrupts

MU can signal events and requests that do not include data words between processors using general-purpose interrupts.

Formatted data with a fixed length can be written to predetermined locations in the shared memory. A processor can use a general-purpose interrupt to signal to the other processor that the data is ready.

A processor can use the 3 flags to announce its current program state (or similar messages) to the other processor.

Table 239 and Figure 167 describe the event steps when the processor triggers a general-purpose interrupt.

Table 239. General-purpose interrupt messaging protocol (generalized)

Step	Action	Description
1	Processor A sets its associated general-purpose interrupt request flag.	Processor A sets GCR[GIR0].

Table continues on the next page...

Table 239. General-purpose interrupt messaging protocol (generalized) (continued)

Step	Action	Description
2	The general-purpose interrupt request pending status flag is set.	GSR[GIP0] is set.
3	MU generates the general-purpose interrupt request to processor B.	Setting GSR[GIP0] generates the general-purpose interrupt request to processor B. GIER[GIE0] must be set for processor B.
4	Processor B reads the status register.	Processor B reads GSR[GIP0].
5	Processor B services the interrupt.	—
6	Processor B sets GSR[GIP0] to clear the interrupt.	Processor B writes 1 to the corresponding GSR[GIP0] flag to clear the interrupt.
7	GCR[GIR0] is cleared.	Setting GSR[GIP0] clears GCR[GIR0] on the processor A side.

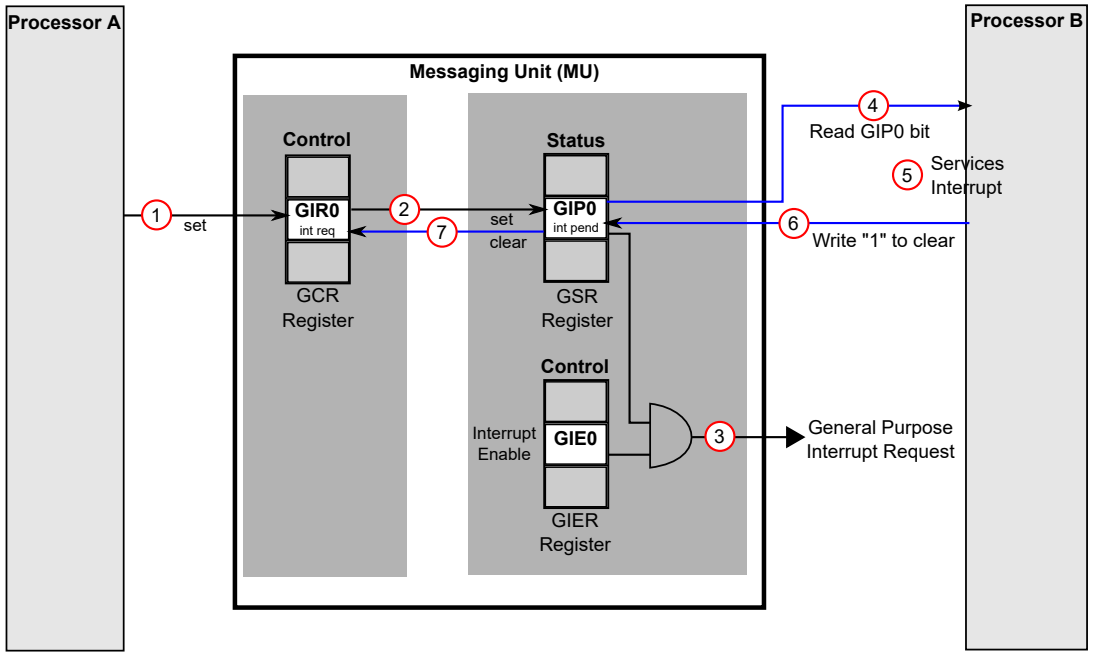


Figure 167. Messaging model using a general-purpose interrupt

40.6.3 Exclusive access to shared memory

MU can signal one processor about its current access to shared memory. This signaling prevents the other processor from overwriting data during the exclusive memory access period.

The following tables describe the signaling protocol that processor A uses to inform processor B about its current access (write) to shared memory:

- Table 240 shows processor A performing an exclusive access to shared memory.
- Table 241 shows processor B scanning for transaction information.

- Table 242 shows processor B accepting exclusive access from processor A.
- Table 243 shows processor B rejecting exclusive access from processor A.

According to the examples shown in the following tables, GCR[GIR0], RR*n*, and TR*n* are reserved to support exclusive access to the shared memory protocol.

Table 240. Processor A performs an exclusive access to shared memory

Step	Action	Description
1	Processor A sends the GIR0 request to processor B using the processor A control register.	Before processor A performs an exclusive access to the shared memory, it sends a GIR0 request to processor B.
2	Processor A sends an exclusive-access request using a transmit data register (TR*n*).	Processor A sends an exclusive-access request (command, location, and length of target access) to processor B using a selected transmit data register (TR0).
3	Processor A waits for a dedicated interrupt from processor B.	Processor A waits for a dedicated interrupt (as an acknowledgment) triggered by processor B before proceeding.
4	Processor A accesses the shared memory.	After receiving a dedicated interrupt from processor B, processor A proceeds.

Table 241. Processor B scans for transaction information

Step	Action	Description
1	Processor B receives an interrupt from a receive data register (RR*n*).	—
2	Processor B reads the receive data register (RR*n*).	—
3	Processor B scans the receive data register contents.	Processor B scans for transaction information (whether processor A has requested exclusive access).

Table 242. Processor B accepts exclusive access from processor A

Step	Action	Description
1	Processor B triggers a dedicated interrupt.	Processor B acknowledges the request from processor A by triggering a dedicated interrupt (ack) to processor A.
2	Processor B sends a code message to processor A.	With the acknowledge interrupt, processor B sends a code message to processor A through the selected transmit register (TR*n*). The message informs processor A that it can exclusively access the shared memory.

Table 243. Processor B rejects exclusive access from processor A

Step	Action	Description
1	Processor B ignores the request from processor A for exclusive access.	If processor B does not provide permission to processor A, processor B ignores the exclusive-access request.

40.6.4 Packet data transfers

The following table describes an example packet transfer sequence between processor B and processor A subsystems.

Table 244. Packet data transfer sequence

Step	Action	Description
1	Processor B requests DMA.	Processor B sends a DMA request to initiate the packet data transfer.
2	DMA data transfer	DMA acknowledges.
3		DMA starts transferring data from the specified processor B memory location to the specified shared memory.
4		DMA interrupts processor B to signal that the packet transfer has finished.
5	Processor B informs processor A that data is in shared memory.	Using a B-side transmit register, processor B sends a packet information message to processor A about the arrival of new packet data stored in shared memory. The message contains the command, location, and length of packet data.
6	Processor A receives an interrupt.	Processor A receives an interrupt (assuming its corresponding processor A MU-side receive interrupt is enabled). The pending processing task becomes active and processes packet data from memory.
7	Processor A reads data, then writes data.	Processor A reads or processes packet data from shared memory.
8		Processor A writes the result from packet processing to a separate buffer.
9	Processor A informs processor B that the transfer is finished.	After the processing of the packet data finishes, processor A informs processor B (using MU processor A-side transmit register, MUA_TR*n*).
10	Processor A sends an interrupt to processor B (a request for more data).	Processor B receives the next interrupt from processor A, in which processor A requests more packet data.

40.6.5 Freeing a processor from deadlock

During normal operation, one processor may determine that the other processor is not working or is deadlocked. Using Status (SR), the processor can use the methods in the following table to identify and correct the problem.

Table 245. Freeing a processor from deadlock

Processor mode	Technique	Description
—	Processor issues an interrupt.	The other processor can interrupt the processor by issuing one of the 9 (general purpose, receive, or transmit) interrupts.

40.7 Register definition

MU provides transmit and receive data registers (*x*TR0–*n*, *x*RR0–*n*) for communication between processor A and processor B. It also provides control registers (*x*CR) for operations such as interrupts and resets, and status registers (*x*SR) for checking the status of the other MU-side. Figure 168 shows the schematic for MU registers.

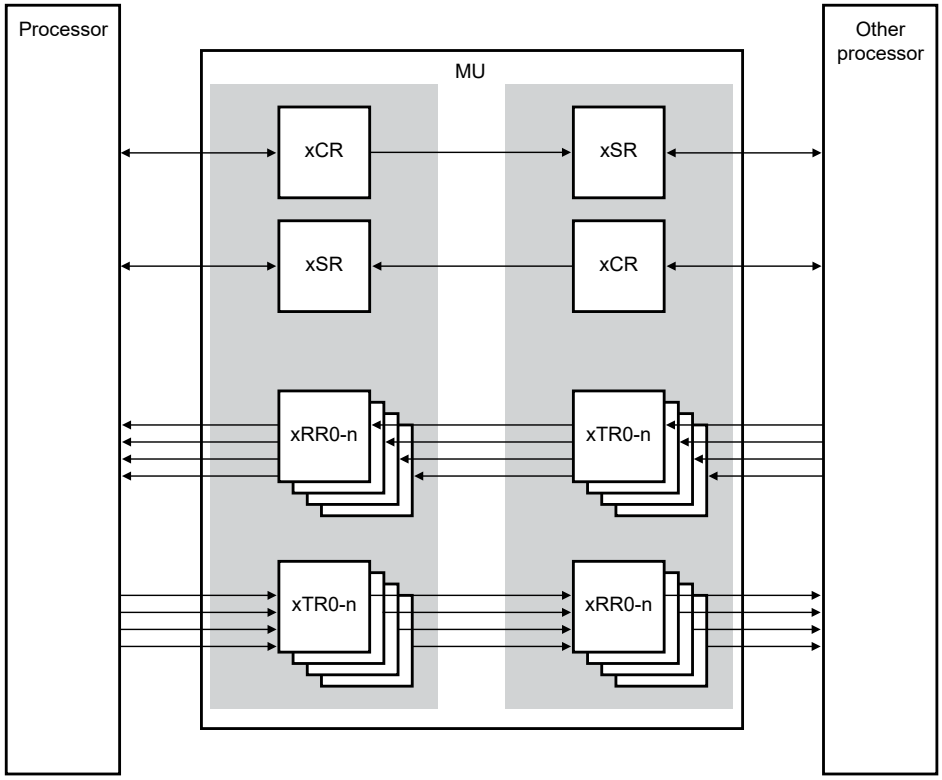


Figure 168. MU registers

40.7.1 MU register descriptions

This section contains the detailed register descriptions for MUA registers.

NOTE

A module transfer error to processor A or processor B is generated when:

- A read or write access is made to an invalid location.
- A write operation is performed on a read-only register on the processor A side or processor B side of MU.

40.7.1.1 MU memory map

MU_2.MUA base address: 400B_8000h

MU_3.MUA base address: 400C_4000h

MU_4.MUA base address: 400C_C000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID (VER)	32	R	0309_000Fh
4h	Parameter (PAR)	32	R	0301_0404h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
8h	Control (CR)	32	RW	0000_0000h
Ch	Status (SR)	32	RW	See section
100h	Flag Control (FCR)	32	RW	0000_0000h
104h	Flag Status (FSR)	32	R	0000_0000h
110h	General-Purpose Interrupt Enable (GIER)	32	RW	0000_0000h
114h	General-Purpose Control (GCR)	32	RW	0000_0000h
118h	General-purpose Status (GSR)	32	RW	0000_0000h
120h	Transmit Control (TCR)	32	RW	0000_0000h
124h	Transmit Status (TSR)	32	R	0000_000Fh
128h	Receive Control (RCR)	32	RW	0000_0000h
12Ch	Receive Status (RSR)	32	R	0000_0000h
200h - 20Ch	Transmit (TR0 - TR3)	32	RW	0000_0000h
280h - 28Ch	Receive (RR0 - RR3)	32	R	0000_0000h

40.7.1.2 Version ID (VER)

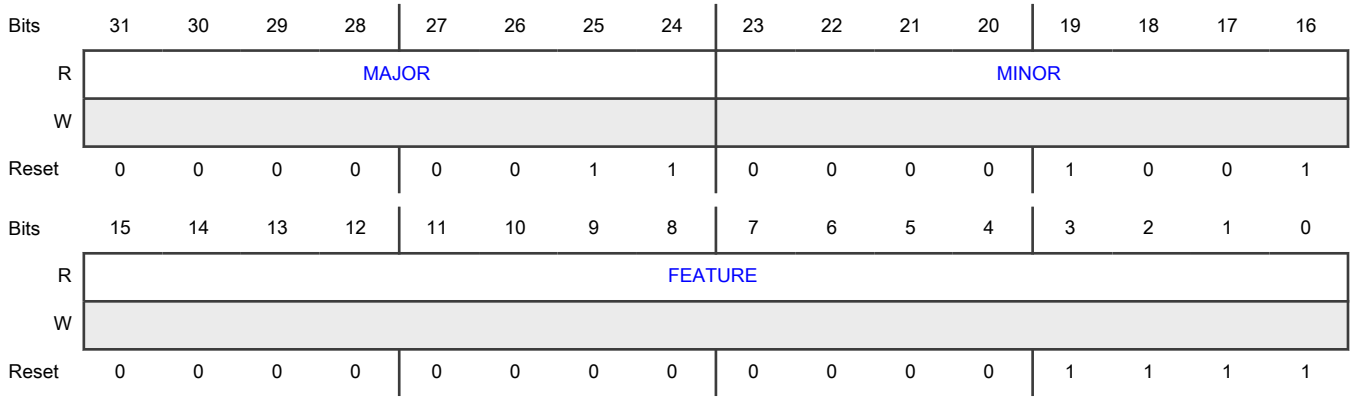
Offset

Register	Offset
VER	0h

Function

Determines the version ID and feature set number of MUA.

Diagram



Fields

Field	Function
31-24 MAJOR	Major Version Number
23-16 MINOR	Minor Version Number
15-0 FEATURE	Feature Set Number Indicates the feature set number. MU implements: • Standard features • Expanded number of TR*n*/RR*n* registers

40.7.1.3 Parameter (PAR)

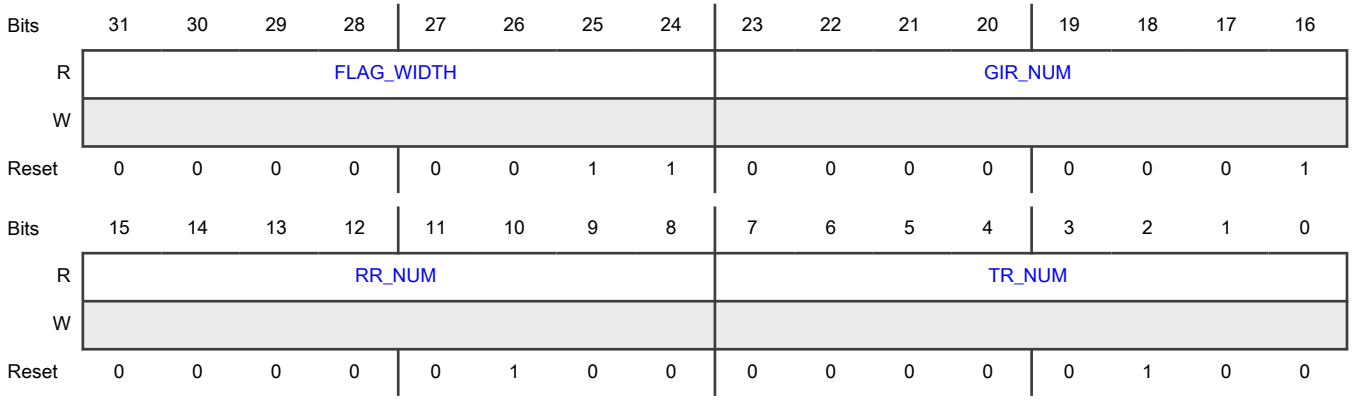
Offset

Register	Offset
PAR	4h

Function

Defines the number of flags, transmit registers, receive registers, and general-purpose interrupt requests available for MU.

Diagram



Fields

Field	Function
31-24	Flag Width

Table continues on the next page...

Table continued from the previous page...

Field	Function
FLAG_WIDTH	Specifies the number of flag bits (3) in the Flag Control (FCR) and Flag Status (FSR) registers.
23-16 GIR_NUM	General-Purpose Interrupt Request Number Specifies the number of general-purpose interrupt requests available (1).
15-8 RR_NUM	Receive Register Number Specifies the number of receive registers (4).
7-0 TR_NUM	Transmit Register Number Specifies the number of transmit registers (4).

40.7.1.4 Control (CR)

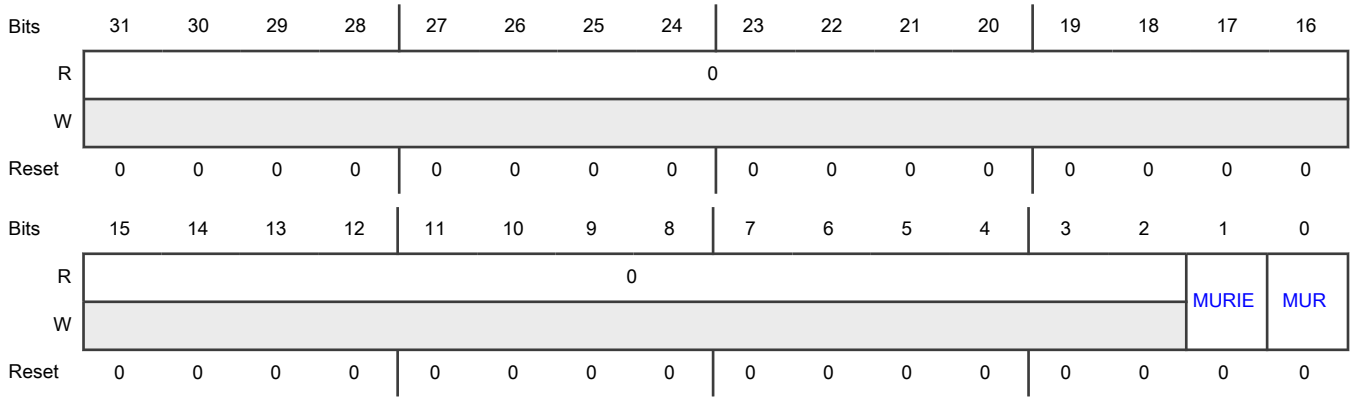
Offset

Register	Offset
CR	8h

Function

Controls MU reset and reset interrupt enable.

Diagram



Fields

Field	Function
31-2 —	Reserved
1	MUA Reset Interrupt Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
MURIE	Enables the processor A-side MU reset interrupt request due to MU reset issued by MUB. If the value of this field is 1, an MU reset interrupt request is issued to processor A when MUA_SR[MURIP] = 1. If the value of this field is 0, MU ignores the value of MURIP and issues no MU reset interrupt request. Only a system reset can reset this field. CR[MUR] cannot reset this field. 0b - Disable 1b - Enable
0 MUR	MU Reset Resets MU. Writing 1 to this field resets the MUA and MUB sides. All internal states are cleared. It forces all control and status registers to return to their default values (except MURIE in MUA/B_CR registers; MURIP and MURS in MUA/B_SR registers). Before writing 1 to this field, interrupt processor B because writing 1 to this field may affect the ongoing processor B program. After writing 1 to this field, monitor the value of MUA_SR[MURS] to know when the reset sequence on the processor B-side has ended. This field always reads 0, and it becomes 0 during the MU reset sequence. 0b - Idle 1b - Reset

40.7.1.5 Status (SR)

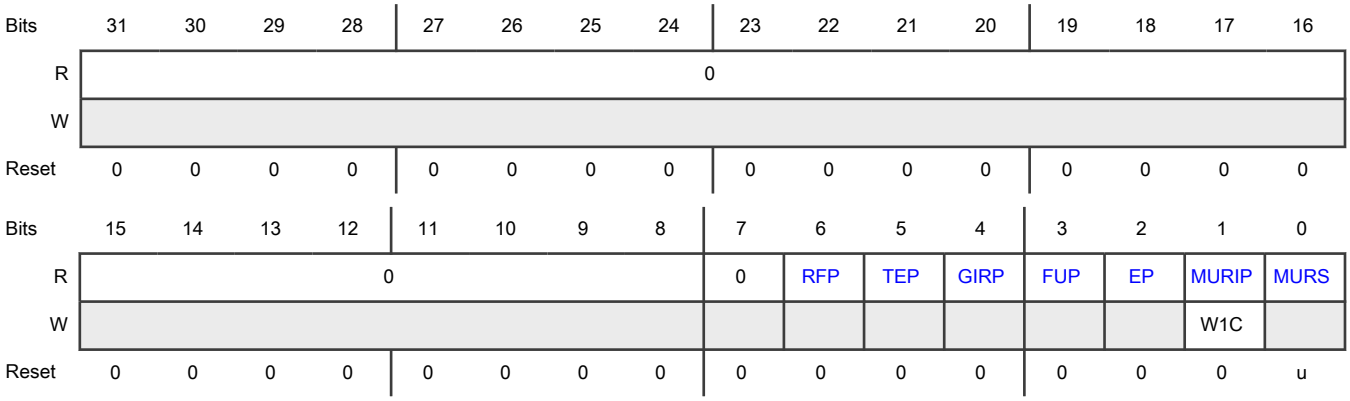
Offset

Register	Offset
SR	Ch

Function

Shows the status of MU resets and the status of pending events and requests.

Diagram



Fields

Field	Function
31-8 —	Reserved
7 —	Reserved
6 RFP	MUA Receive Full Pending Indicates whether a receive full message is pending. This field becomes 1 when MUB writes to a TR*n* register to send data to MUA. After this field becomes 1, MU checks RSR[RF*n*] to determine whether the data in the Receive register is ready for MUA to read it. This field becomes 0 when all MUA RR*n* registers are read, or when MU is reset. 0b - Not pending; MUB is not writing to a Transmit register 1b - Pending; MUB is writing to a Transmit register
5 TEP	MUA Transmit Empty Pending Indicates whether a transmit empty message is pending. This field becomes 1 when any TCR[TIE*n*] field is 1 and TSR[TE*n*] flag is set. After this field becomes 1, MU checks TSR[TE*n*] to determine whether the data in the Transmit (TR*n*) register is ready for MUA to write to it. This field becomes 0 when write operations to all MUA Transmit (TR*n*) registers where TCR[TIE*n*] = 1 (transfer interrupt enabled) are completed, or when MU is reset. 0b - No MUA transmit empty event pending 1b - Pending; any TCR[TIEn] field is 1 and TSR[TEn] flag is set
4 GIRP	MUA General-Purpose Interrupt Pending Indicates that MUB has sent a general-purpose interrupt request. This field becomes 1 when the MUB side sends a general-purpose interrupt request to the MUA side. GSR[GIP0] identifies which general-purpose interrupt request is received.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field becomes 0 when MUA_GSR[GIP0] is cleared, or when MU is reset. 0b - No request sent 1b - Request sent
3 FUP	MUA Flag Update Pending Indicates whether a flag update request is pending. MU generates this request when there is a change to the F*n*[2:0] bits of MUA_FCR. This field becomes 1 when the MUA side sends a flag update request to the MUB side. This field becomes 0 when MU acknowledges this flag update request internally (the flag is updated) from the MUB side, or during MU reset. No flag update changes are allowed when this field is 1. When FUP = 1, a write to the F*n*[2:0] bits of MUA_FCR does not generate a flag update event. The F*n*[2:0] bits do not change. If SR[EP] = 1 (event pending), writing to MUA_FCR does not immediately cause this field to become 1. 0b - No pending update flags (initiated by MUA) 1b - Pending update flags (initiated by MUA)
2 EP	MUA Side Event Pending Indicates a pending side event when the MUA side sends an event update request to the MUB side. An event is any hardware message that the Status register on the MUB side reflects. For example, an event occurs when Transmit register 0 is the target of a write operation. During normal operations, the update mechanism for this field operates automatically. MU clears this field automatically when the event update acknowledgment is received, or when MU resets. To ensure that events are posted to MUB, verify that this field is 0. If it is 1, wait and continue to poll this field until it becomes 0. 0b - Not pending 1b - Pending
1 MURIP	MU Reset Interrupt Pending Flag Indicates whether processor B has issued an MU reset. This flag is set after processor B initiates an MU reset by setting MUB_CR[MUR]. If CR[MURIE] = 1, the processor A MU reset interrupt request is issued when processor B writes 1 to MUB_CR[MUR]. Clearing this flag also clears the MU reset interrupt request. Only a system reset can reset this flag. MU reset cannot reset this flag. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Reset not issued

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Reset issued When writing 0b - No effect 1b - Clear the flag
0 MURS	MUA and MUB Reset State Indicates the reset state of MUA and MUB. This field becomes 1 during any system reset or MU reset from the MUA or MUB side. This field becomes 0 when the reset sequence on both MUA and MUB sides ends. After issuing any of the aforementioned reset events, verify that this field is 0 before starting any access. 0b - Out of reset 1b - In reset

40.7.1.6 Flag Control (FCR)

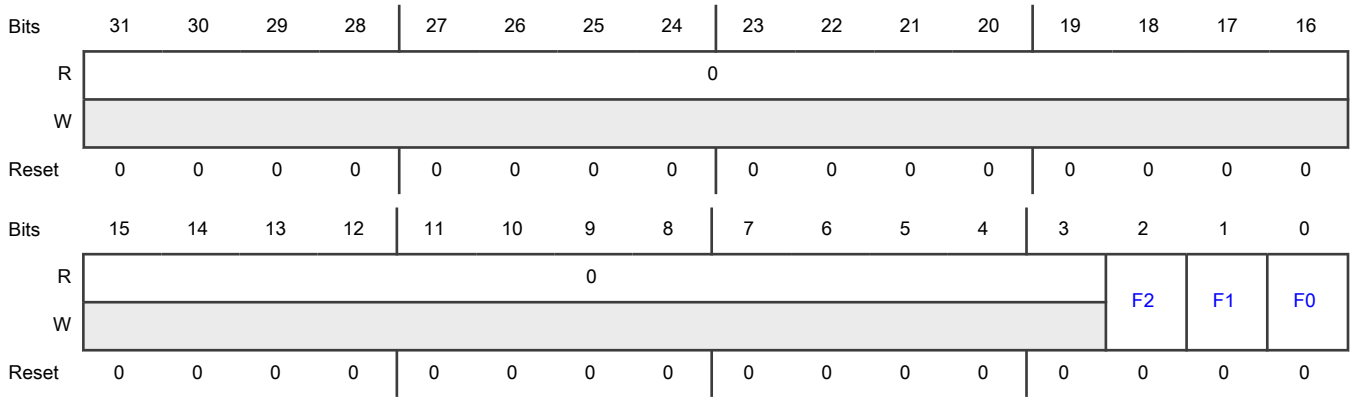
Offset

Register	Offset
FCR	100h

Function

Configures MUB_FSR[F*n*] flags.

Diagram



Fields

Field	Function
31-3 —	Reserved
2-0 Fn	MUA to MUB Flag Configures MUB_FSR[F*n*] flags, where *n* = 0 to 2. F*n* configures the corresponding MUB_FSR[F*n*] flag. F*n* becomes 0 when MU resets. 0b - Clear MUB_FSR[Fn] 1b - Set MUB_FSR[Fn]

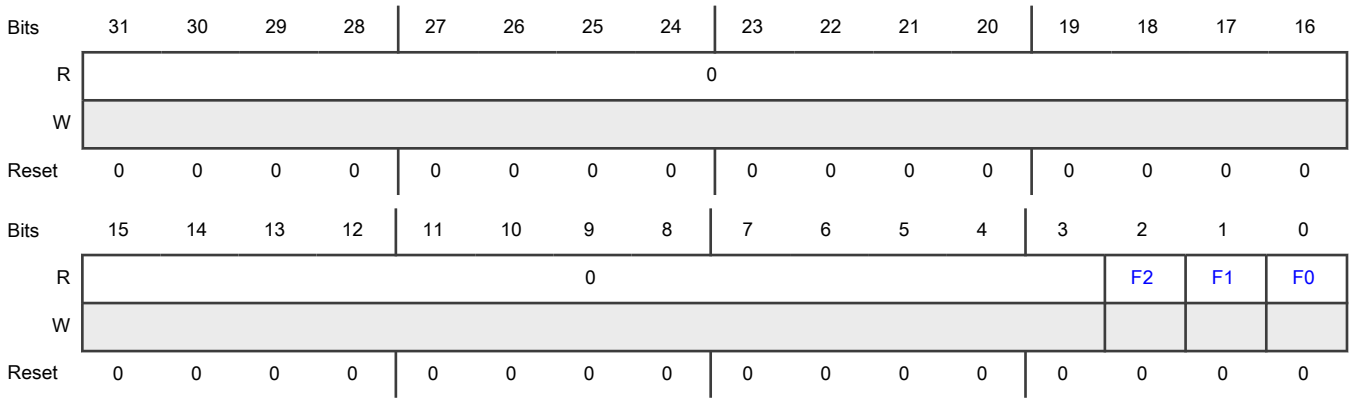
40.7.1.7 Flag Status (FSR)

Offset

Register	Offset
FSR	104h

Function
Contains flags configured by the values written to MUB_FCR[F*n*].

Diagram



Fields

Field	Function
31-3 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
2-0 Fn	MUB to MUA-Side Flag Contains flags configured by the values written to MUB_FCR[F*n*], where *n* = 0 to 2. F*n* is the MUA-side flag configured by the values written to MUB_FCR[F*n*]. When MUB_FCR[F*n*] is written to, the write event updates MUA_FSR[F*n*], after the event update latency. 0b - MUB_FCR[Fn] = 0 1b - MUB_FCR[Fn] = 1

40.7.1.8 General-Purpose Interrupt Enable (GIER)

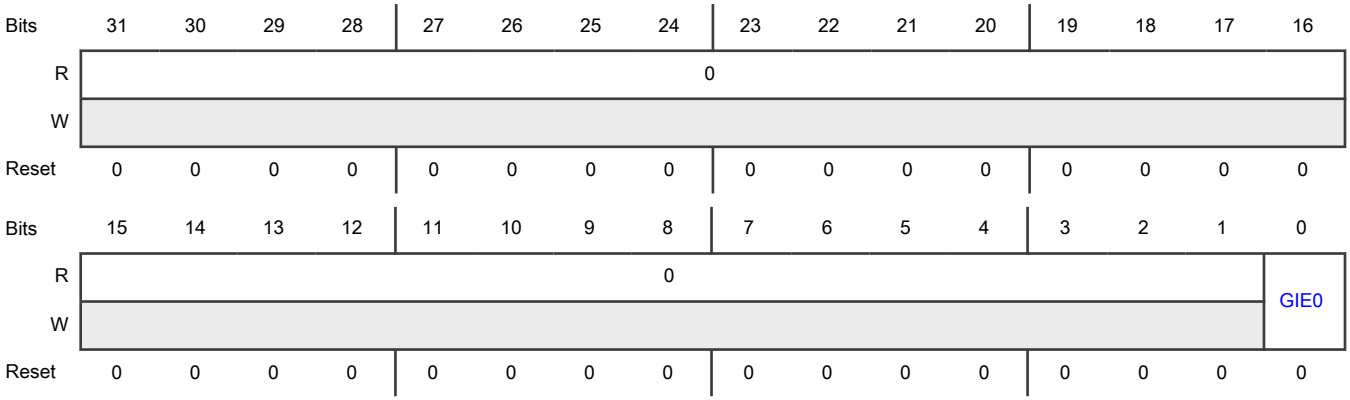
Offset

Register	Offset
GIER	110h

Function

Contains the MUA general-purpose interrupt enable fields.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 GIEn	MUA General-purpose Interrupt Enable Enables general-purpose interrupt. There is one general-purpose interrupt (*n* = 0).

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When GIE0 = 1, a general-purpose interrupt *n* request is issued to processor A when MUA GSR[GIP0] = 1. If GIE0 = 0, the general-purpose interrupt request pending does not trigger the general-purpose interrupt. GIE0 becomes 0 when MU resets. 0b - Disable 1b - Enable

40.7.1.9 General-Purpose Control (GCR)

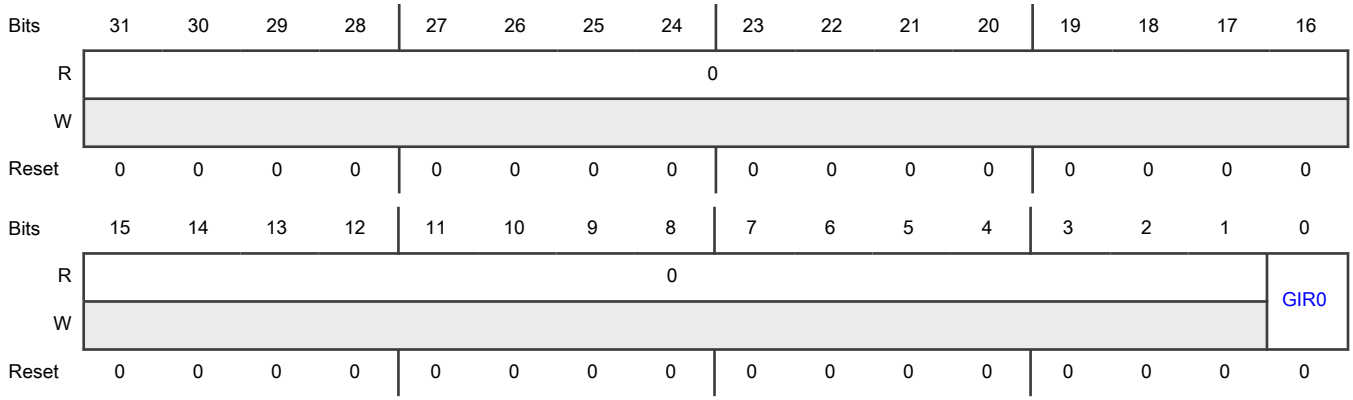
Offset

Register	Offset
GCR	114h

Function

Contains the MUA general-purpose interrupt request fields.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 GIRn	MUA General-Purpose Interrupt Request Specifies whether general-purpose interrupts are requested to MUB. There is one general-purpose interrupt (*n* = 0).

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Writing 1 to GIR0 sets MUB_GSR[GIP0]. If MUB_GIER[GIE0] = 1, a general-purpose interrupt request is triggered on processor B. This field becomes 0 when MUB_GSR[GIP0] is cleared. This clearing informs MUA that the interrupt is accepted (cleared by software). To ensure proper operations, verify that GIR0 is 0 (no pending interrupt) before writing 1 to it. This field becomes 0 when MU resets. 0b - Not requested 1b - Requested

40.7.1.10 General-purpose Status (GSR)

Offset

Register	Offset
GSR	118h

Function

Contains the status of the MUA general-purpose interrupt pending requests.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															GIP0
W																W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-1 —	Reserved
0	MUA General-Purpose Interrupt Request Pending

Table continues on the next page...

Table continued from the previous page...

Field	Function
GIPn	Indicates whether a general-purpose interrupt request is pending. There is one general-purpose interrupt (*n* = 0). GIP0 informs MUA that MUB_GCR[GIR0] changed from 0 to 1. If MUA_GIER[GIE0] = 1, a general-purpose interrupt request is issued to processor A. GIP0 is cleared when MU resets. After GIP0 is cleared, if MUA_GIER[GIE0] = 1, the general-purpose interrupt request is cleared on the MUA side. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not pending 1b - Pending When writing 0b - No effect 1b - Clear the flag

40.7.1.11 Transmit Control (TCR)

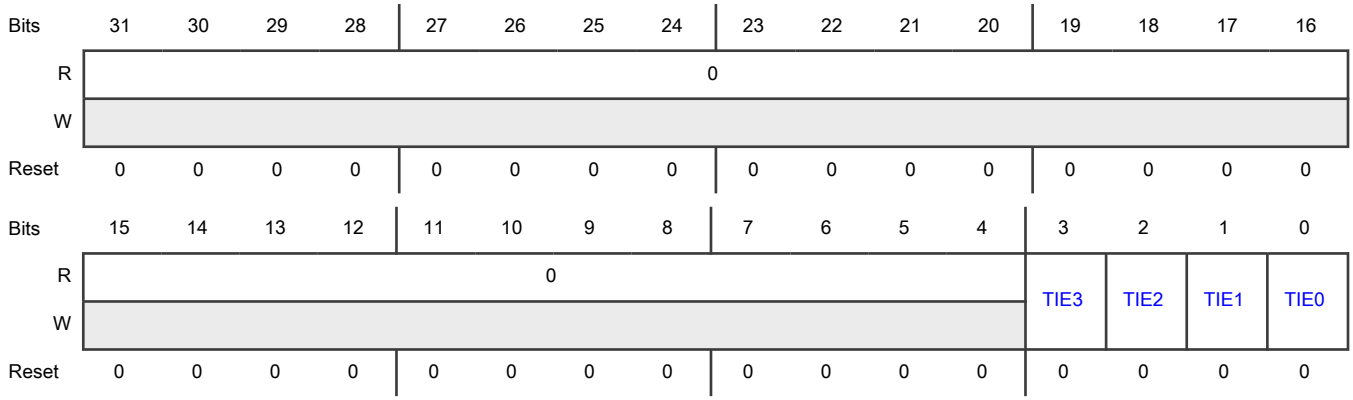
Offset

Register	Offset
TCR	120h

Function

Contains the MUA transmit interrupt enable fields.

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 TIEn	MUA Transmit Interrupt Enable Enables MUA transmit interrupt *n*, where *n* = 0 to 3. If this field is 1, an MUA transmit interrupt *n* request is issued when MUA_TSR[TE*n*] is set. If this field is 0, MU ignores the value of MUA_TSR[TE*n*], and no MUA transmit interrupt *n* request is issued. 0b - Disable 1b - Enable

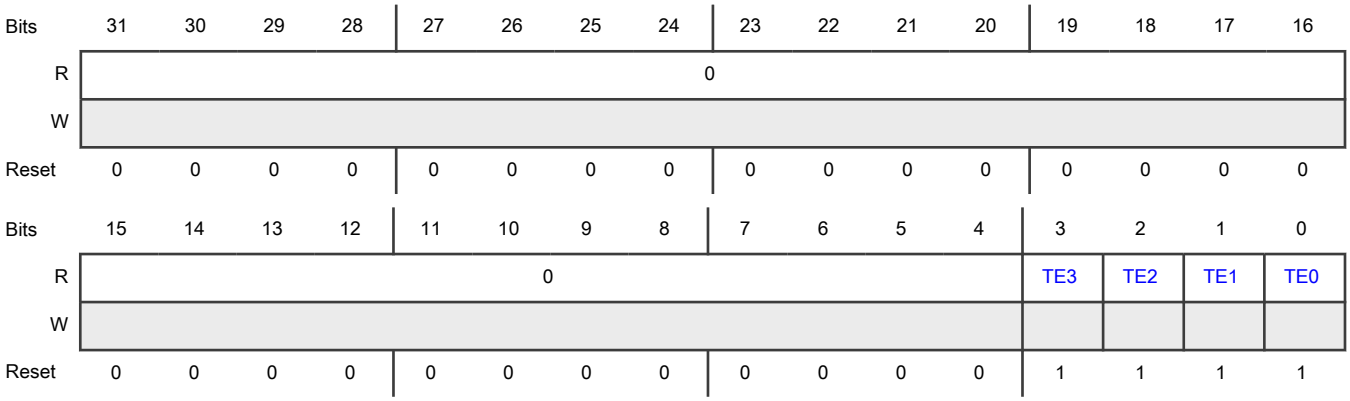
40.7.1.12 Transmit Status (TSR)

Offset

Register	Offset
TSR	124h

Function
Indicates whether the MUA transmit registers are empty.

Diagram



Fields

Field	Function
31-4 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
3-0 TEn	MUA Transmit Empty Indicates whether MUA Transmit (TR*n*) register is empty, where *n* = 0 to 3. This field becomes 1 after the MUB_RR*n* register is read on the MUB side. When TE*n* = 1, it indicates to the MUA side that the MUA_TR*n* register is ready to be written on the MUA side. If MUA_TCR[TIE*n*] = 1, a transmit *n* interrupt is issued on the MUA side. This field becomes 0 after the MUA_TR*n* register is written to on the MUA side. After this field becomes 0, if MUA_TCR[TIE*n*] = 1, the transmit *n* interrupt request is cleared on the MUA side. This field becomes 1 when MU resets. 0b - Not empty 1b - Empty

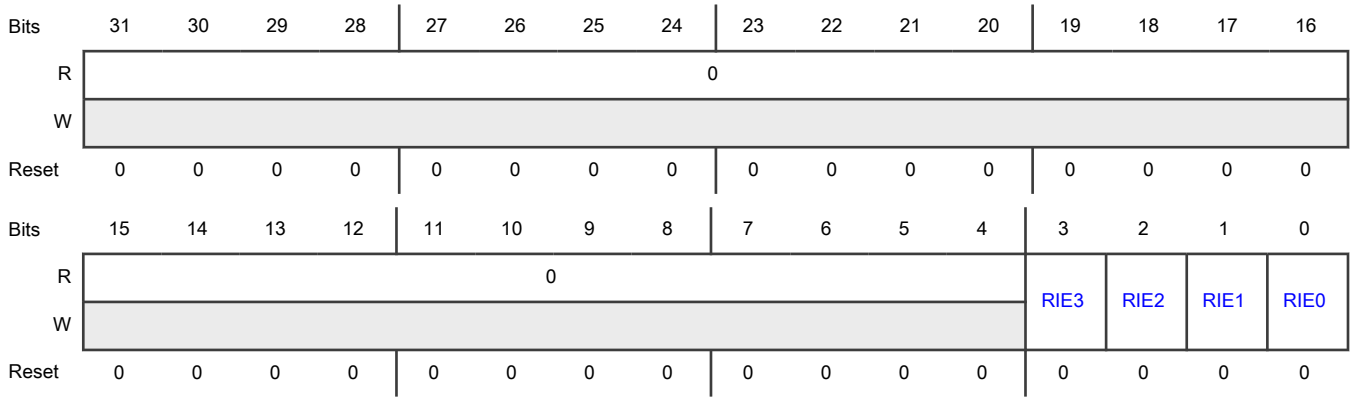
40.7.1.13 Receive Control (RCR)

Offset

Register	Offset
RCR	128h

Function
Contains the MUA receive interrupt enables.

Diagram



Fields

Field	Function
31-4 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
3-0 RIEn	MUA Receive Interrupt Enable Enables MUA receive interrupt *n*, where *n* = 0 to 3. If this field is 1, an MUA receive interrupt *n* request is issued when MUA_RSR[RF*n*] is set. If this field is 0, MU ignores the value of MUA_RSR[RF*n*], and no MUA receive interrupt request is issued. 0b - Disable 1b - Enable

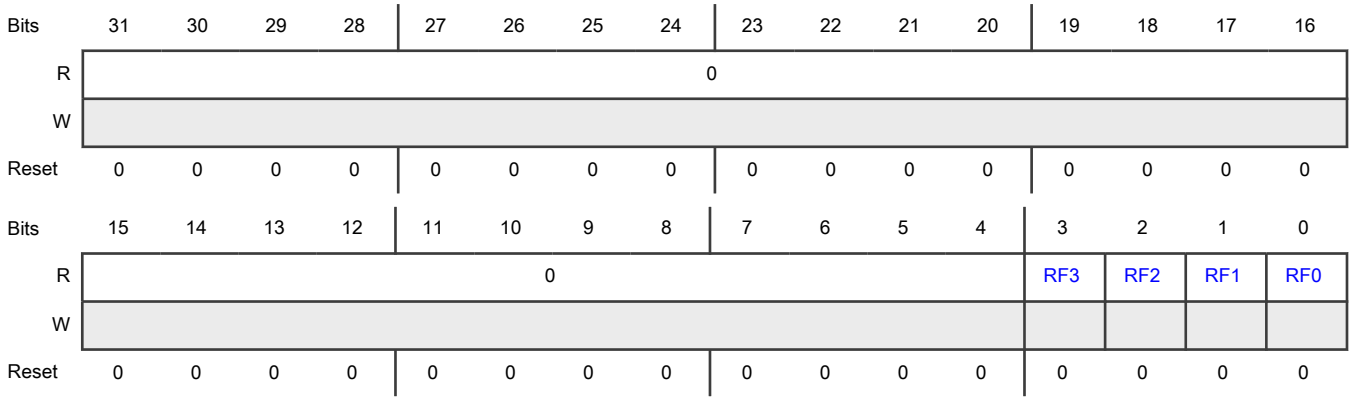
40.7.1.14 Receive Status (RSR)

Offset

Register	Offset
RSR	12Ch

Function
Indicates whether the MUA receive registers are full.

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 RFn	MUA Receive Register Full Indicates whether MUA Receive register (RR*n*) is full, where *n* = 0 to 3.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	RF*n* becomes 1 when the MUB_TR*n* register is written to on the MUB side. When RF*n* is 1, it indicates to the MUA side that new data in the MUA_RR*n* register is ready for MUA to read it. If MUA_RCR[RIE*n*] = 1, a receive *n* interrupt is issued on the MUA side. RF*n* becomes 0 when the MUA_RR*n* register is read, or when MU is reset. After RF*n* becomes 0, if MUA_RCR[RIE*n*] = 1, the receive *n* interrupt request is cleared on the MUA side. 0b - Not full 1b - Full

40.7.1.15 Transmit (TR0 - TR3)

Offset

Register	Offset
TR0	200h
TR1	204h
TR2	208h
TR3	20Ch

Function
Contains MUA transmit data.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W	TR_DATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W	TR_DATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0	MUA Transmit Data

Table continues on the next page...

Field	Function
TR_DATA	Contains MUA transmit data. MUB_RR*n* reflects the data written to this register. The TR*n* and RR*n* registers are not double-buffered. Writing to MUA_TR*n* overrides the data readable in the MUA_RR*n* register. A write to the Transmit register clears MUA_TSR[TE*n*] on the transmitter side, and sets MUB_RSR[RF*n*] on the receiver side. You can write to this register only when MUA_TSR[TE*n*] = 1. Reading this register returns all zeroes.

40.7.1.16 Receive (RR0 - RR3)

Offset

Register	Offset
RR0	280h
RR1	284h
RR2	288h
RR3	28Ch

Function

Contains MUA receive data.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RR_DATA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RR_DATA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 RR_DATA	MUA Receive Data Reflects the data written to MUB TR*n*.

Table continues on the next page...

Field	Function
	Reading this register clears MUA_RSR[RF*n*] on the receiver side, and sets MUB_TSR[TE*n*] on the transmitter side. You can read this register only when MUA_RSR[RF*n*] = 1. Reading it before MUA_RSR[RF*n*] becomes 1 may result in reading incorrect data. Poll MUA_RSR[RF*n*] to confirm it is set before reading RR*n*. Writing to this register generates an error response to MUA.

40.7.2 MU register descriptions

This section contains the detailed register descriptions for MUB registers.

NOTE

A module transfer error to processor A or processor B is generated when:

- A read or write access is made to an invalid location.
- A write operation is performed on a read-only register on the processor A side or processor B side of MU.

40.7.2.1 MU memory map

MU_0.MUB base address: 4038_C000h

MU_1.MUB base address: 404E_C000h

MU_2.MUB base address: 400B_C000h

MU_3.MUB base address: 400C_8000h

MU_4.MUB base address: 400D_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID (VER)	32	R	0309_000Fh
4h	Parameter (PAR)	32	R	See section
8h	Control (CR)	32	RW	0000_0000h
Ch	Status (SR)	32	RW	See section
10h	Core Control 0 (CCR0)	32	RW	0000_0000h
18h	Core Sticky Status 0 (CSSR0)	32	RW	See section
100h	Flag Control (FCR)	32	RW	0000_0000h
104h	Flag Status (FSR)	32	R	0000_0000h
110h	General-Purpose Interrupt Enable (GIER)	32	RW	0000_0000h
114h	General-Purpose Control (GCR)	32	RW	0000_0000h
118h	General-purpose Status (GSR)	32	RW	0000_0000h
120h	Transmit Control (TCR)	32	RW	0000_0000h
124h	Transmit Status (TSR)	32	R	0000_000Fh

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
128h	Receive Control (RCR)	32	RW	0000_0000h
12Ch	Receive Status (RSR)	32	R	0000_0000h
200h - 20Ch	Transmit (TR0 - TR3)	32	RW	0000_0000h
280h - 28Ch	Receive (RR0 - RR3)	32	R	0000_0000h

40.7.2.2 Version ID (VER)

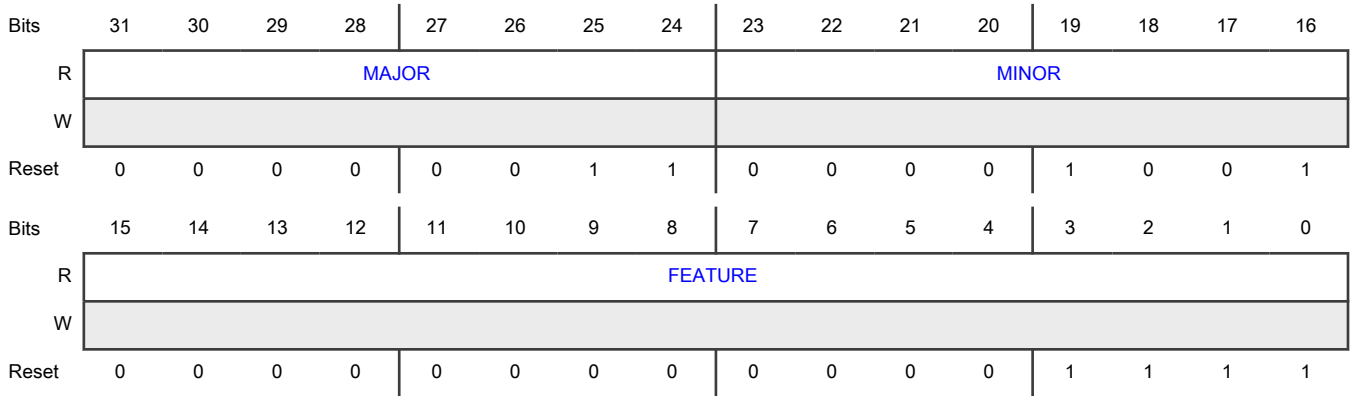
Offset

Register	Offset
VER	0h

Function

Determines the version ID and feature set number of MUB.

Diagram



Fields

Field	Function
31-24 MAJOR	Major Version Number
23-16 MINOR	Minor Version Number
15-0	Feature Set Number

Table continues on the next page...

Table continued from the previous page...

Field	Function
FEATURE	Indicates the feature set number. MU implements: • Standard features • Expanded number of TR*n*/RR*n* registers

40.7.2.3 Parameter (PAR)

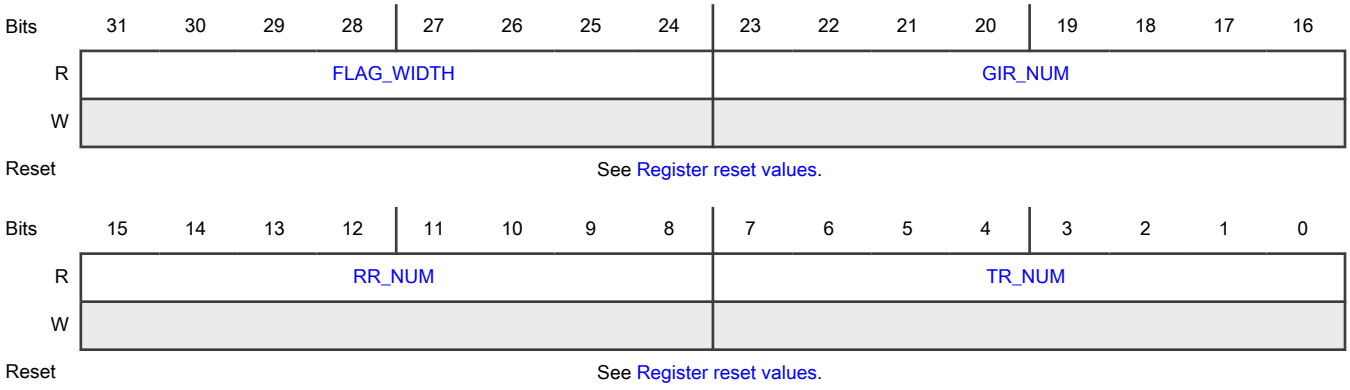
Offset

Register	Offset
PAR	4h

Function

Defines the number of flags, transmit registers, receive registers, and general-purpose interrupt requests available for MU.

Diagram



Register reset values

Register	Reset value
PAR	MU_0.MUB,MU_1.MUB: 2020_0404h MU_2.MUB–MU_4.MUB: 0301_0404h

Fields

Field	Function
31-24	Flag Width

Table continues on the next page...

Table continued from the previous page...

Field	Function
FLAG_WIDTH	Specifies the number of flag bits (32) in the Flag Control (FCR) and Flag Status (FSR) registers.
23-16 GIR_NUM	General-Purpose Interrupt Request Number Specifies the number of general-purpose interrupt requests available (32).
15-8 RR_NUM	Receive Register Number Specifies the number of receive registers (4).
7-0 TR_NUM	Transmit Register Number Specifies the number of transmit registers (4).

40.7.2.4 Control (CR)

Offset

Register	Offset
CR	8h

Function

Controls MU reset and reset interrupt enable.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0														MURIE	MUR
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-2 —	Reserved
1	MUB Reset Interrupt Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
MURIE	Enables the processor B-side MU reset interrupt request due to MU reset issued by MUA. If the value of this field is 1, an MU reset interrupt request is issued to processor B when MUB_SR[MURIP] = 1. If the value of this field is 0, MU ignores the value of MURIP and issues no MU reset interrupt request. Only a system reset can reset this field. CR[MUR] cannot reset this field. 0b - Disable 1b - Enable
0 MUR	MU Reset Resets MU. Writing 1 to this field resets the MUB and MUA sides. All internal states are cleared. It forces all control and status registers to return to their default values (except in MUB/A_CCR0 registers; MURIE in MUB/A_CR registers; MURIP and MURS in MUB/A_SR registers). Before writing 1 to this field, interrupt processor A because writing 1 to this field may affect the ongoing processor A program. After writing 1 to this field, monitor the value of MUB_SR[MURS] to know when the reset sequence on the processor A-side has ended. This field always reads 0, and it becomes 0 during the MU reset sequence. 0b - Idle 1b - Reset

40.7.2.5 Status (SR)

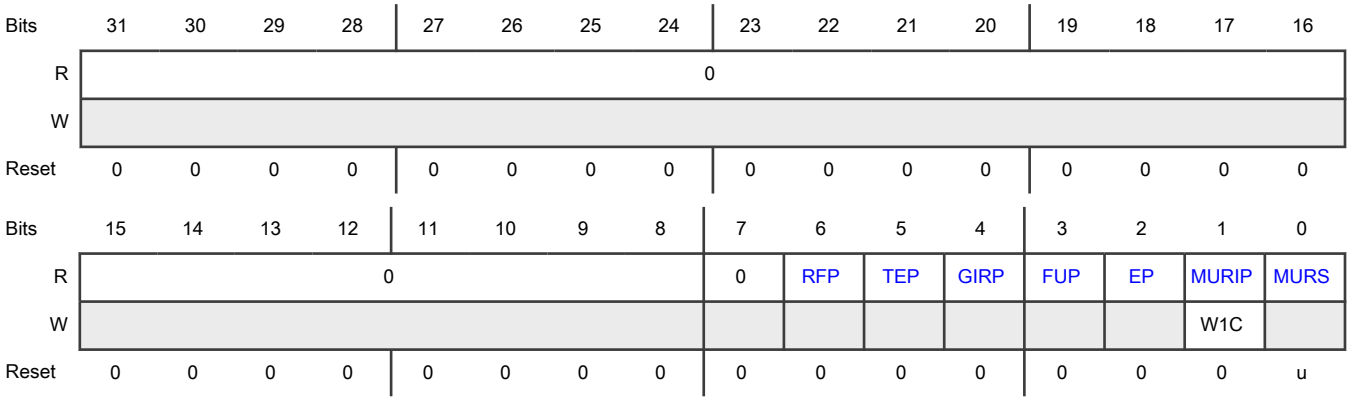
Offset

Register	Offset
SR	Ch

Function

Shows the status of MU resets and the status of pending events and requests.

Diagram



Fields

Field	Function
31-8 —	Reserved
7 —	Reserved
6 RFP	MUB Receive Full Pending Indicates whether a receive full message is pending. This field becomes 1 when MUA writes to a TR*n* register to send data to MUB. After this field becomes 1, MU checks RSR[RF*n*] to determine whether the data in the Receive register is ready for MUB to read it. This field becomes 0 when all MUB RR*n* registers are read, or when MU is reset. 0b - Not pending; MUA is not writing to a Transmit register 1b - Pending; MUA is writing to a Transmit register
5 TEP	MUB Transmit Empty Pending Indicates whether a transmit empty message is pending. This field becomes 1 when any TCR[TIE*n*] field is 1 and TSR[TE*n*] flag is set. After this field becomes 1, MU checks TSR[TE*n*] to determine whether the data in the Transmit (TR*n*) register is ready for MUB to write to it. This field becomes 0 when write operations to all MUB Transmit (TR*n*) registers where TCR[TIE*n*] = 1 (transfer interrupt enabled) are completed, or when MU is reset. 0b - No MUB transmit empty event pending 1b - Pending; any TCR[TIEn] field is 1 and TSR[TEn] flag is set
4 GIRP	MUB General-Purpose Interrupt Pending Indicates that MUA has sent a general-purpose interrupt request. This field becomes 1 when the MUA side sends a general-purpose interrupt request to the MUB side. GSR[GIP*n*] identifies which general-purpose interrupt request is received.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field becomes 0 when all MUB_GSR[GIP*n*] fields are cleared, or when MU is reset. 0b - No request sent 1b - Request sent
3 FUP	MUB Flag Update Pending Indicates whether a flag update request is pending. MU generates this request when there is a change to the F*n*[31:0] bits of MUB_FCR. This field becomes 1 when the MUB side sends a flag update request to the MUA side. This field becomes 0 when MU acknowledges this flag update request internally (the flag is updated) from the MUA side, or during MU reset. No flag update changes are allowed when this field is 1. When FUP = 1, a write to the F*n*[31:0] bits of MUB_FCR does not generate a flag update event. The F*n*[31:0] bits do not change. If SR[EP] = 1 (event pending), writing to MUB_FCR does not immediately cause this field to become 1. 0b - No pending update flags (initiated by MUB) 1b - Pending update flags (initiated by MUB)
2 EP	MUB Side Event Pending Indicates a pending side event when the MUB side sends an event update request to the MUA side. An event is any hardware message that the Status register on the MUA side reflects. For example, an event occurs when Transmit register 0 is the target of a write operation. During normal operations, the update mechanism for this field operates automatically. MU clears this field automatically when the event update acknowledgment is received, or when MU resets. To ensure that events are posted to MUA, verify that this field is 0. If it is 1, wait and continue to poll this field until it becomes 0. 0b - Not pending 1b - Pending
1 MURIP	MU Reset Interrupt Pending Flag Indicates whether processor A has issued an MU reset. This flag is set after processor A initiates an MU reset by setting MUB_CR[MUR]. If CR[MURIE] = 1, the processor B MU reset interrupt request is issued when processor A writes 1 to MUA_CR[MUR]. Clearing this flag also clears the MU reset interrupt request. Only a system reset can reset this flag. MU reset cannot reset this flag. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Reset not issued

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Reset issued When writing 0b - No effect 1b - Clear the flag
0 MURS	MUA and MUB Reset State Indicates the reset state of MUA and MUB. This field becomes 1 during any system reset or MU reset from the MUA or MUB side. This field becomes 0 when the reset sequence on both MUA and MUB sides ends. After issuing any of the aforementioned reset events, verify that this field is 0 before starting any access. 0b - Out of reset 1b - In reset

40.7.2.6 Core Control 0 (CCR0)

Offset

Register	Offset
CCR0	10h

Function

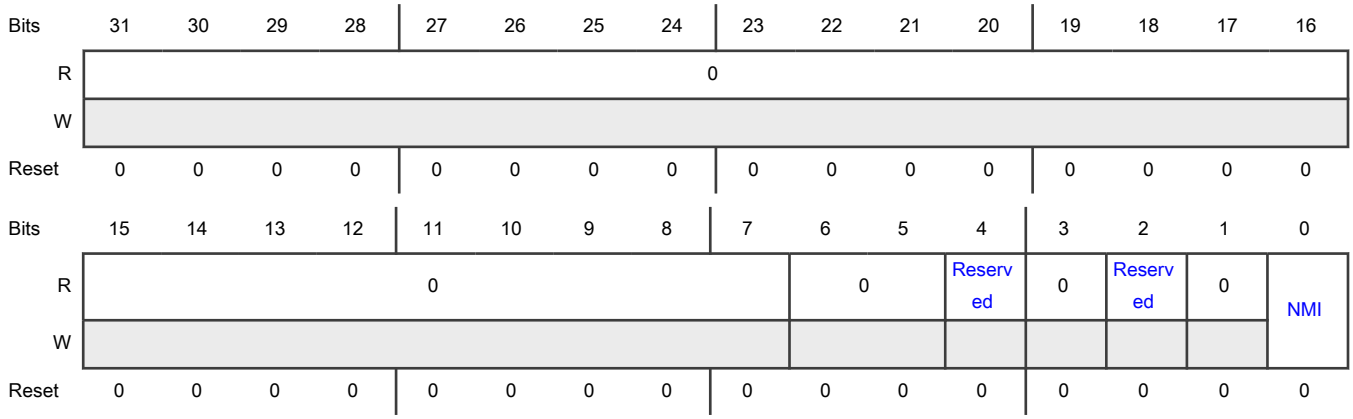
Allows MUB to control the processor on the MUA side.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
MU_0.MUB	CCR0	—
MU_1.MUB	CCR0	—
MU_2.MUB	—	CCR0
MU_3.MUB	—	CCR0
MU_4.MUB	—	CCR0

Diagram



Fields

Field	Function
31-7 —	Reserved
6-5 —	Reserved
4 —	Reserved
3 —	Reserved
2 —	Reserved
1 —	Reserved
0 NMI	MUA Nonmaskable Interrupt Request Indicates whether Processor B has issued a nonmaskable interrupt to Processor A. When this field becomes 1, it initiates a nonmaskable interrupt to processor A. This field becomes 0 after 1 is written to MUA_CSSR0[NMIC] to clear that field. After this field becomes 0, MUB can initiate another nonmaskable interrupt to MUA. This field is cleared when MU resets. 0b - Nonmaskable interrupt not issued 1b - Nonmaskable interrupt issued

40.7.2.7 Core Sticky Status 0 (CSSR0)

Offset

Register	Offset
CSSR0	18h

Function

Shows the status of interrupts pending (W1C).

The reset value is chip-specific.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
MU_0.MUB	CSSR0	—
MU_1.MUB	CSSR0	—
MU_2.MUB	—	CSSR0
MU_3.MUB	—	CSSR0
MU_4.MUB	—	CSSR0

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								0	0	0	0	0	0	0	NMIC
W																W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	u

Fields

Field	Function
31-8 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
7 —	Reserved
6 —	Reserved
5 —	Reserved
4 —	Reserved
3 —	Reserved
2 —	Reserved
1 —	Reserved
0 NMIC	Processor B Nonmaskable Interrupt Clear Clears the nonmaskable interrupt (NMI) request from the MUA side. The MUB-side NMI service routine uses this field. Writing 1 to this field clears MUA_CCR0[NMI], deasserting the NMI request and enabling MUA_CCR0[NMI] to trigger another NMI request. This field always reads as 0, so you cannot poll it. You can only use this field as part of the NMI service routine, in which you must write 1 to this field only once. This field is cleared when MU resets. 0b - Default 1b - Clear MUA_CCR0[NMI]

40.7.2.8 Flag Control (FCR)

Offset

Register	Offset
FCR	100h

Function
Configures MUA_FSR[F*n*] flags.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	F31	F30	F29	F28	F27	F26	F25	F24	F23	F22	F21	F20	F19	F18	F17	F16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	F15	F14	F13	F12	F11	F10	F9	F8	F7	F6	F5	F4	F3	F2	F1	F0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>31
F31</td><td>MUB to MUA Flag
Configures MUA_FSR[Fn] flags, where n = 0 to 31.
Fn configures the corresponding MUA_FSR[Fn] flag.
Fn becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FCR</td></tr>
<tr><td>MU_3.MUB</td><td>—</td><td>FCR</td></tr>
<tr><td>MU_4.MUB</td><td>—</td><td>FCR</td></tr>
</table>
0b - Clear MUA_FSR[Fn]
1b - Set MUA_FSR[Fn]</td></tr>
<tr><td>30
F30</td><td>MUB to MUA Flag
Configures MUA_FSR[Fn] flags, where n = 0 to 31.
Fn configures the corresponding MUA_FSR[Fn] flag.
Fn becomes 0 when MU resets.</td></tr>
</table>

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FCR</td></tr>
<tr><td>MU_3.MUB</td><td>—</td><td>FCR</td></tr>
<tr><td>MU_4.MUB</td><td>—</td><td>FCR</td></tr>
</table>
0b - Clear MUA_FSR[Fn]
1b - Set MUA_FSR[Fn]
</td></tr>
<tr><td>29
F29</td><td>
MUB to MUA Flag
Configures MUA_FSR[Fn] flags, where n = 0 to 31.
Fn configures the corresponding MUA_FSR[Fn] flag.
Fn becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FCR</td></tr>
<tr><td>MU_3.MUB</td><td>—</td><td>FCR</td></tr>
<tr><td>MU_4.MUB</td><td>—</td><td>FCR</td></tr>
</table>
0b - Clear MUA_FSR[Fn]
1b - Set MUA_FSR[Fn]
</td></tr>
<tr><td>28</td><td>MUB to MUA Flag</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>F28</td><td>Configures MUA_FSR[Fn] flags, where n = 0 to 31.
Fn configures the corresponding MUA_FSR[Fn] flag.
Fn becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>FCR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>FCR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>FCR</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>FCR</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>FCR</td></tr></table>0b - Clear MUA_FSR[Fn]
1b - Set MUA_FSR[Fn]</td></tr>
<tr><td>27
F27</td><td>MUB to MUA Flag
Configures MUA_FSR[Fn] flags, where n = 0 to 31.
Fn configures the corresponding MUA_FSR[Fn] flag.
Fn becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>FCR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>FCR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>FCR</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>FCR</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>FCR</td></tr></table></td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
</table>
0b - Clear MUA_FSR[Fn]
1b - Set MUA_FSR[Fn]
</td></tr>
<tr><td>26
F26</td><td>MUB to MUA Flag
Configures MUA_FSR[Fn] flags, where n = 0 to 31.
Fn configures the corresponding MUA_FSR[Fn] flag.
Fn becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FCR</td></tr>
<tr><td>MU_3.MUB</td><td>—</td><td>FCR</td></tr>
<tr><td>MU_4.MUB</td><td>—</td><td>FCR</td></tr>
</table>
0b - Clear MUA_FSR[Fn]
1b - Set MUA_FSR[Fn]
</td></tr>
<tr><td>25
F25</td><td>MUB to MUA Flag
Configures MUA_FSR[Fn] flags, where n = 0 to 31.
Fn configures the corresponding MUA_FSR[Fn] flag.
Fn becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FCR</td><td>—</td></tr>
</table>
</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_1.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FCR</td></tr>
<tr><td>MU_3.MUB</td><td>—</td><td>FCR</td></tr>
<tr><td>MU_4.MUB</td><td>—</td><td>FCR</td></tr>
</table>
0b - Clear MUA_FSR[Fn]
1b - Set MUA_FSR[Fn]
</td></tr>
<tr><td>24
F24</td><td>MUB to MUA Flag
Configures MUA_FSR[Fn] flags, where n = 0 to 31.
Fn configures the corresponding MUA_FSR[Fn] flag.
Fn becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FCR</td></tr>
<tr><td>MU_3.MUB</td><td>—</td><td>FCR</td></tr>
<tr><td>MU_4.MUB</td><td>—</td><td>FCR</td></tr>
</table>
0b - Clear MUA_FSR[Fn]
1b - Set MUA_FSR[Fn]
</td></tr>
<tr><td>23
F23</td><td>MUB to MUA Flag
Configures MUA_FSR[Fn] flags, where n = 0 to 31.
Fn configures the corresponding MUA_FSR[Fn] flag.
Fn becomes 0 when MU resets.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FCR</td></tr>
<tr><td>MU_3.MUB</td><td>—</td><td>FCR</td></tr>
<tr><td>MU_4.MUB</td><td>—</td><td>FCR</td></tr>
</table>
0b - Clear MUA_FSR[Fn]
1b - Set MUA_FSR[Fn]
</td></tr>
<tr><td>22
F22</td><td>
MUB to MUA Flag
Configures MUA_FSR[Fn] flags, where n = 0 to 31.
Fn configures the corresponding MUA_FSR[Fn] flag.
Fn becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FCR</td></tr>
<tr><td>MU_3.MUB</td><td>—</td><td>FCR</td></tr>
<tr><td>MU_4.MUB</td><td>—</td><td>FCR</td></tr>
</table>
0b - Clear MUA_FSR[Fn]
1b - Set MUA_FSR[Fn]
</td></tr>
<tr><td>21</td><td>MUB to MUA Flag</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>F21</td><td>Configures MUA_FSR[Fn] flags, where n = 0 to 31.
Fn configures the corresponding MUA_FSR[Fn] flag.
Fn becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>FCR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>FCR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>FCR</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>FCR</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>FCR</td></tr></table>0b - Clear MUA_FSR[Fn]
1b - Set MUA_FSR[Fn]</td></tr>
<tr><td>20
F20</td><td>MUB to MUA Flag
Configures MUA_FSR[Fn] flags, where n = 0 to 31.
Fn configures the corresponding MUA_FSR[Fn] flag.
Fn becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>FCR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>FCR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>FCR</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>FCR</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>FCR</td></tr></table></td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
</table>
0b - Clear MUA_FSR[Fn]
1b - Set MUA_FSR[Fn]
</td></tr>
<tr><td>19
F19</td><td>MUB to MUA Flag
Configures MUA_FSR[Fn] flags, where n = 0 to 31.
Fn configures the corresponding MUA_FSR[Fn] flag.
Fn becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FCR</td></tr>
<tr><td>MU_3.MUB</td><td>—</td><td>FCR</td></tr>
<tr><td>MU_4.MUB</td><td>—</td><td>FCR</td></tr>
</table>
0b - Clear MUA_FSR[Fn]
1b - Set MUA_FSR[Fn]
</td></tr>
<tr><td>18
F18</td><td>MUB to MUA Flag
Configures MUA_FSR[Fn] flags, where n = 0 to 31.
Fn configures the corresponding MUA_FSR[Fn] flag.
Fn becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FCR</td><td>—</td></tr>
</table>
</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_1.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FCR</td></tr>
<tr><td>MU_3.MUB</td><td>—</td><td>FCR</td></tr>
<tr><td>MU_4.MUB</td><td>—</td><td>FCR</td></tr>
</table>
0b - Clear MUA_FSR[Fn]
1b - Set MUA_FSR[Fn]
</td></tr>
<tr><td>17
F17</td><td>MUB to MUA Flag
Configures MUA_FSR[Fn] flags, where n = 0 to 31.
Fn configures the corresponding MUA_FSR[Fn] flag.
Fn becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FCR</td></tr>
<tr><td>MU_3.MUB</td><td>—</td><td>FCR</td></tr>
<tr><td>MU_4.MUB</td><td>—</td><td>FCR</td></tr>
</table>
0b - Clear MUA_FSR[Fn]
1b - Set MUA_FSR[Fn]
</td></tr>
<tr><td>16
F16</td><td>MUB to MUA Flag
Configures MUA_FSR[Fn] flags, where n = 0 to 31.
Fn configures the corresponding MUA_FSR[Fn] flag.
Fn becomes 0 when MU resets.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FCR</td></tr>
<tr><td>MU_3.MUB</td><td>—</td><td>FCR</td></tr>
<tr><td>MU_4.MUB</td><td>—</td><td>FCR</td></tr>
</table>
0b - Clear MUA_FSR[Fn]
1b - Set MUA_FSR[Fn]
</td></tr>
<tr><td>15
F15</td><td>
MUB to MUA Flag
Configures MUA_FSR[Fn] flags, where n = 0 to 31.
Fn configures the corresponding MUA_FSR[Fn] flag.
Fn becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FCR</td></tr>
<tr><td>MU_3.MUB</td><td>—</td><td>FCR</td></tr>
<tr><td>MU_4.MUB</td><td>—</td><td>FCR</td></tr>
</table>
0b - Clear MUA_FSR[Fn]
1b - Set MUA_FSR[Fn]
</td></tr>
<tr><td>14</td><td>MUB to MUA Flag</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>F14</td><td>Configures MUA_FSR[Fn] flags, where n = 0 to 31.
Fn configures the corresponding MUA_FSR[Fn] flag.
Fn becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>FCR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>FCR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>FCR</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>FCR</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>FCR</td></tr></table>0b - Clear MUA_FSR[Fn]
1b - Set MUA_FSR[Fn]</td></tr>
<tr><td>13
F13</td><td>MUB to MUA Flag
Configures MUA_FSR[Fn] flags, where n = 0 to 31.
Fn configures the corresponding MUA_FSR[Fn] flag.
Fn becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>FCR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>FCR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>FCR</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>FCR</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>FCR</td></tr></table></td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
</table>
0b - Clear MUA_FSR[Fn]
1b - Set MUA_FSR[Fn]
</td></tr>
<tr><td>12
F12</td><td>MUB to MUA Flag
Configures MUA_FSR[Fn] flags, where n = 0 to 31.
Fn configures the corresponding MUA_FSR[Fn] flag.
Fn becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FCR</td></tr>
<tr><td>MU_3.MUB</td><td>—</td><td>FCR</td></tr>
<tr><td>MU_4.MUB</td><td>—</td><td>FCR</td></tr>
</table>
0b - Clear MUA_FSR[Fn]
1b - Set MUA_FSR[Fn]
</td></tr>
<tr><td>11
F11</td><td>MUB to MUA Flag
Configures MUA_FSR[Fn] flags, where n = 0 to 31.
Fn configures the corresponding MUA_FSR[Fn] flag.
Fn becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FCR</td><td>—</td></tr>
</table>
</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_1.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FCR</td></tr>
<tr><td>MU_3.MUB</td><td>—</td><td>FCR</td></tr>
<tr><td>MU_4.MUB</td><td>—</td><td>FCR</td></tr>
</table>
0b - Clear MUA_FSR[Fn]
1b - Set MUA_FSR[Fn]
</td></tr>
<tr><td>10
F10</td><td>MUB to MUA Flag
Configures MUA_FSR[Fn] flags, where n = 0 to 31.
Fn configures the corresponding MUA_FSR[Fn] flag.
Fn becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FCR</td></tr>
<tr><td>MU_3.MUB</td><td>—</td><td>FCR</td></tr>
<tr><td>MU_4.MUB</td><td>—</td><td>FCR</td></tr>
</table>
0b - Clear MUA_FSR[Fn]
1b - Set MUA_FSR[Fn]
</td></tr>
<tr><td>9
F9</td><td>MUB to MUA Flag
Configures MUA_FSR[Fn] flags, where n = 0 to 31.
Fn configures the corresponding MUA_FSR[Fn] flag.
Fn becomes 0 when MU resets.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FCR</td></tr>
<tr><td>MU_3.MUB</td><td>—</td><td>FCR</td></tr>
<tr><td>MU_4.MUB</td><td>—</td><td>FCR</td></tr>
</table>
0b - Clear MUA_FSR[Fn]
1b - Set MUA_FSR[Fn]
</td></tr>
<tr><td>8
F8</td><td>
MUB to MUA Flag
Configures MUA_FSR[Fn] flags, where n = 0 to 31.
Fn configures the corresponding MUA_FSR[Fn] flag.
Fn becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FCR</td></tr>
<tr><td>MU_3.MUB</td><td>—</td><td>FCR</td></tr>
<tr><td>MU_4.MUB</td><td>—</td><td>FCR</td></tr>
</table>
0b - Clear MUA_FSR[Fn]
1b - Set MUA_FSR[Fn]
</td></tr>
<tr><td>7</td><td>MUB to MUA Flag</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>F7</td><td>Configures MUA_FSR[Fn] flags, where n = 0 to 31.
Fn configures the corresponding MUA_FSR[Fn] flag.
Fn becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>FCR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>FCR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>FCR</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>FCR</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>FCR</td></tr></table>0b - Clear MUA_FSR[Fn]
1b - Set MUA_FSR[Fn]</td></tr>
<tr><td>6
F6</td><td>MUB to MUA Flag
Configures MUA_FSR[Fn] flags, where n = 0 to 31.
Fn configures the corresponding MUA_FSR[Fn] flag.
Fn becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>FCR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>FCR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>FCR</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>FCR</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>FCR</td></tr></table></td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
</table>
0b - Clear MUA_FSR[Fn]
1b - Set MUA_FSR[Fn]
</td></tr>
<tr><td>5
F5</td><td>MUB to MUA Flag
Configures MUA_FSR[Fn] flags, where n = 0 to 31.
Fn configures the corresponding MUA_FSR[Fn] flag.
Fn becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FCR</td></tr>
<tr><td>MU_3.MUB</td><td>—</td><td>FCR</td></tr>
<tr><td>MU_4.MUB</td><td>—</td><td>FCR</td></tr>
</table>
0b - Clear MUA_FSR[Fn]
1b - Set MUA_FSR[Fn]
</td></tr>
<tr><td>4
F4</td><td>MUB to MUA Flag
Configures MUA_FSR[Fn] flags, where n = 0 to 31.
Fn configures the corresponding MUA_FSR[Fn] flag.
Fn becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FCR</td><td>—</td></tr>
</table>
</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_1.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FCR</td></tr>
<tr><td>MU_3.MUB</td><td>—</td><td>FCR</td></tr>
<tr><td>MU_4.MUB</td><td>—</td><td>FCR</td></tr>
</table>
0b - Clear MUA_FSR[Fn]
1b - Set MUA_FSR[Fn]
</td></tr>
<tr><td>3
F3</td><td>MUB to MUA Flag
Configures MUA_FSR[Fn] flags, where n = 0 to 31.
Fn configures the corresponding MUA_FSR[Fn] flag.
Fn becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FCR</td></tr>
<tr><td>MU_3.MUB</td><td>—</td><td>FCR</td></tr>
<tr><td>MU_4.MUB</td><td>—</td><td>FCR</td></tr>
</table>
0b - Clear MUA_FSR[Fn]
1b - Set MUA_FSR[Fn]
</td></tr>
<tr><td>2
F2</td><td>MUB to MUA Flag
Configures MUA_FSR[Fn] flags, where n = 0 to 31.
Fn configures the corresponding MUA_FSR[Fn] flag.
Fn becomes 0 when MU resets.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Clear MUA_FSR[Fn] 1b - Set MUA_FSR[Fn]
1 F1	MUB to MUA Flag Configures MUA_FSR[F*n*] flags, where *n* = 0 to 31. F*n* configures the corresponding MUA_FSR[F*n*] flag. F*n* becomes 0 when MU resets. 0b - Clear MUA_FSR[Fn] 1b - Set MUA_FSR[Fn]
0 F0	MUB to MUA Flag Configures MUA_FSR[F*n*] flags, where *n* = 0 to 31. F*n* configures the corresponding MUA_FSR[F*n*] flag. F*n* becomes 0 when MU resets. 0b - Clear MUA_FSR[Fn] 1b - Set MUA_FSR[Fn]

40.7.2.9 Flag Status (FSR)

Offset

Register	Offset
FSR	104h

Function

Contains flags configured by the values written to MUA_FCR[F*n*].

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	F31	F30	F29	F28	F27	F26	F25	F24	F23	F22	F21	F20	F19	F18	F17	F16
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	F15	F14	F13	F12	F11	F10	F9	F8	F7	F6	F5	F4	F3	F2	F1	F0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>31
F31</td><td>MUA to MUB-Side Flag
Contains flags configured by the values written to MUA_FCR[Fn], where n = 0 to 31.
Fn is the MUB-side flag configured by the values written to MUA_FCR[Fn].
When MUA_FCR[Fn] is written to, the write event updates MUB_FSR[Fn], after the event update latency.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>FSR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>FSR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>FSR</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>FSR</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>FSR</td></tr></table>
0b - MUA_FCR[Fn] = 0
1b - MUA_FCR[Fn] = 1</td></tr>
<tr><td>30
F30</td><td>MUA to MUB-Side Flag
Contains flags configured by the values written to MUA_FCR[Fn], where n = 0 to 31.
Fn is the MUB-side flag configured by the values written to MUA_FCR[Fn].
When MUA_FCR[Fn] is written to, the write event updates MUB_FSR[Fn], after the event update latency.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>FSR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>FSR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>FSR</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>FSR</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>FSR</td></tr></table></td></tr>
</table>

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
</table>
0b - MUA_FCR[Fn] = 0
1b - MUA_FCR[Fn] = 1
</td></tr>
<tr><td>29
F29</td><td>MUA to MUB-Side Flag
Contains flags configured by the values written to MUA_FCR[Fn], where n = 0 to 31.
Fn is the MUB-side flag configured by the values written to MUA_FCR[Fn].
When MUA_FCR[Fn] is written to, the write event updates MUB_FSR[Fn], after the event update latency.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FSR</td></tr>
<tr><td>MU_3.MUB</td><td>—</td><td>FSR</td></tr>
<tr><td>MU_4.MUB</td><td>—</td><td>FSR</td></tr>
</table>
0b - MUA_FCR[Fn] = 0
1b - MUA_FCR[Fn] = 1</td></tr>
<tr><td>28
F28</td><td>MUA to MUB-Side Flag
Contains flags configured by the values written to MUA_FCR[Fn], where n = 0 to 31.
Fn is the MUB-side flag configured by the values written to MUA_FCR[Fn].
When MUA_FCR[Fn] is written to, the write event updates MUB_FSR[Fn], after the event update latency.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FSR</td><td>—</td></tr>
</table></td></tr>
</table>

Table continued from the previous page...

Field	Function
	<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_1.MUB</td><td>FSR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>FSR</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>FSR</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>FSR</td></tr></table> 0b - MUA_FCR[Fn] = 0 1b - MUA_FCR[Fn] = 1
27 F27	MUA to MUB-Side Flag Contains flags configured by the values written to MUA_FCR[F*n*], where *n* = 0 to 31. F*n* is the MUB-side flag configured by the values written to MUA_FCR[F*n*]. When MUA_FCR[F*n*] is written to, the write event updates MUB_FSR[F*n*], after the event update latency. **NOTE** This field is not supported in every instance. The following table includes only supported registers. <table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>FSR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>FSR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>FSR</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>FSR</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>FSR</td></tr></table> 0b - MUA_FCR[Fn] = 0 1b - MUA_FCR[Fn] = 1
26 F26	MUA to MUB-Side Flag Contains flags configured by the values written to MUA_FCR[F*n*], where *n* = 0 to 31. F*n* is the MUB-side flag configured by the values written to MUA_FCR[F*n*]. When MUA_FCR[F*n*] is written to, the write event updates MUB_FSR[F*n*], after the event update latency.

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FSR</td></tr>
<tr><td>MU_3.MUB</td><td>—</td><td>FSR</td></tr>
<tr><td>MU_4.MUB</td><td>—</td><td>FSR</td></tr>
</table>
0b - MUA_FCR[Fn] = 0
1b - MUA_FCR[Fn] = 1
</td></tr>
<tr><td>25
F25</td><td>
MUA to MUB-Side Flag
Contains flags configured by the values written to MUA_FCR[Fn], where n = 0 to 31.
Fn is the MUB-side flag configured by the values written to MUA_FCR[Fn].
When MUA_FCR[Fn] is written to, the write event updates MUB_FSR[Fn], after the event update latency.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FSR</td></tr>
<tr><td>MU_3.MUB</td><td>—</td><td>FSR</td></tr>
<tr><td>MU_4.MUB</td><td>—</td><td>FSR</td></tr>
</table>
0b - MUA_FCR[Fn] = 0
1b - MUA_FCR[Fn] = 1
</td></tr>
<tr><td>24</td><td>MUA to MUB-Side Flag</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>F24</td><td>Contains flags configured by the values written to MUA_FCR[Fn], where n = 0 to 31.
Fn is the MUB-side flag configured by the values written to MUA_FCR[Fn].
When MUA_FCR[Fn] is written to, the write event updates MUB_FSR[Fn], after the event update latency.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>FSR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>FSR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>FSR</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>FSR</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>FSR</td></tr></table>
0b - MUA_FCR[Fn] = 0
1b - MUA_FCR[Fn] = 1</td></tr>
<tr><td>23
F23</td><td>MUA to MUB-Side Flag
Contains flags configured by the values written to MUA_FCR[Fn], where n = 0 to 31.
Fn is the MUB-side flag configured by the values written to MUA_FCR[Fn].
When MUA_FCR[Fn] is written to, the write event updates MUB_FSR[Fn], after the event update latency.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>FSR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>FSR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>FSR</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>FSR</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>FSR</td></tr></table></td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
</table>
0b - MUA_FCR[Fn] = 0
1b - MUA_FCR[Fn] = 1
</td></tr>
<tr><td>22
F22</td><td>MUA to MUB-Side Flag
Contains flags configured by the values written to MUA_FCR[Fn], where n = 0 to 31.
Fn is the MUB-side flag configured by the values written to MUA_FCR[Fn].
When MUA_FCR[Fn] is written to, the write event updates MUB_FSR[Fn], after the event update latency.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FSR</td></tr>
<tr><td>MU_3.MUB</td><td>—</td><td>FSR</td></tr>
<tr><td>MU_4.MUB</td><td>—</td><td>FSR</td></tr>
</table>
0b - MUA_FCR[Fn] = 0
1b - MUA_FCR[Fn] = 1
</td></tr>
<tr><td>21
F21</td><td>MUA to MUB-Side Flag
Contains flags configured by the values written to MUA_FCR[Fn], where n = 0 to 31.
Fn is the MUB-side flag configured by the values written to MUA_FCR[Fn].
When MUA_FCR[Fn] is written to, the write event updates MUB_FSR[Fn], after the event update latency.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FSR</td><td>—</td></tr>
</table>
</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_1.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FSR</td></tr>
<tr><td>MU_3.MUB</td><td>—</td><td>FSR</td></tr>
<tr><td>MU_4.MUB</td><td>—</td><td>FSR</td></tr>
</table>
0b - MUA_FCR[Fn] = 0
1b - MUA_FCR[Fn] = 1
</td></tr>
<tr><td>20
F20</td><td>MUA to MUB-Side Flag
Contains flags configured by the values written to MUA_FCR[Fn], where n = 0 to 31.
Fn is the MUB-side flag configured by the values written to MUA_FCR[Fn].
When MUA_FCR[Fn] is written to, the write event updates MUB_FSR[Fn], after the event update latency.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FSR</td></tr>
<tr><td>MU_3.MUB</td><td>—</td><td>FSR</td></tr>
<tr><td>MU_4.MUB</td><td>—</td><td>FSR</td></tr>
</table>
0b - MUA_FCR[Fn] = 0
1b - MUA_FCR[Fn] = 1
</td></tr>
<tr><td>19
F19</td><td>MUA to MUB-Side Flag
Contains flags configured by the values written to MUA_FCR[Fn], where n = 0 to 31.
Fn is the MUB-side flag configured by the values written to MUA_FCR[Fn].
When MUA_FCR[Fn] is written to, the write event updates MUB_FSR[Fn], after the event update latency.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FSR</td></tr>
<tr><td>MU_3.MUB</td><td>—</td><td>FSR</td></tr>
<tr><td>MU_4.MUB</td><td>—</td><td>FSR</td></tr>
</table>
0b - MUA_FCR[Fn] = 0
1b - MUA_FCR[Fn] = 1
</td></tr>
<tr><td>18
F18</td><td>
MUA to MUB-Side Flag
Contains flags configured by the values written to MUA_FCR[Fn], where n = 0 to 31.
Fn is the MUB-side flag configured by the values written to MUA_FCR[Fn].
When MUA_FCR[Fn] is written to, the write event updates MUB_FSR[Fn], after the event update latency.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FSR</td></tr>
<tr><td>MU_3.MUB</td><td>—</td><td>FSR</td></tr>
<tr><td>MU_4.MUB</td><td>—</td><td>FSR</td></tr>
</table>
0b - MUA_FCR[Fn] = 0
1b - MUA_FCR[Fn] = 1
</td></tr>
<tr><td>17</td><td>MUA to MUB-Side Flag</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>F17</td><td>Contains flags configured by the values written to MUA_FCR[Fn], where n = 0 to 31.
Fn is the MUB-side flag configured by the values written to MUA_FCR[Fn].
When MUA_FCR[Fn] is written to, the write event updates MUB_FSR[Fn], after the event update latency.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>FSR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>FSR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>FSR</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>FSR</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>FSR</td></tr></table>
0b - MUA_FCR[Fn] = 0
1b - MUA_FCR[Fn] = 1</td></tr>
<tr><td>16
F16</td><td>MUA to MUB-Side Flag
Contains flags configured by the values written to MUA_FCR[Fn], where n = 0 to 31.
Fn is the MUB-side flag configured by the values written to MUA_FCR[Fn].
When MUA_FCR[Fn] is written to, the write event updates MUB_FSR[Fn], after the event update latency.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>FSR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>FSR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>FSR</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>FSR</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>FSR</td></tr></table></td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
</table>
0b - MUA_FCR[Fn] = 0
1b - MUA_FCR[Fn] = 1
</td></tr>
<tr><td>15
F15</td><td>MUA to MUB-Side Flag
Contains flags configured by the values written to MUA_FCR[Fn], where n = 0 to 31.
Fn is the MUB-side flag configured by the values written to MUA_FCR[Fn].
When MUA_FCR[Fn] is written to, the write event updates MUB_FSR[Fn], after the event update latency.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FSR</td></tr>
<tr><td>MU_3.MUB</td><td>—</td><td>FSR</td></tr>
<tr><td>MU_4.MUB</td><td>—</td><td>FSR</td></tr>
</table>
0b - MUA_FCR[Fn] = 0
1b - MUA_FCR[Fn] = 1</td></tr>
<tr><td>14
F14</td><td>MUA to MUB-Side Flag
Contains flags configured by the values written to MUA_FCR[Fn], where n = 0 to 31.
Fn is the MUB-side flag configured by the values written to MUA_FCR[Fn].
When MUA_FCR[Fn] is written to, the write event updates MUB_FSR[Fn], after the event update latency.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FSR</td><td>—</td></tr>
</table></td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_1.MUB</td><td>FSR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>FSR</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>FSR</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>FSR</td></tr></table> 0b - MUA_FCR[Fn] = 0 1b - MUA_FCR[Fn] = 1
13 F13	MUA to MUB-Side Flag Contains flags configured by the values written to MUA_FCR[F*n*], where *n* = 0 to 31. F*n* is the MUB-side flag configured by the values written to MUA_FCR[F*n*]. When MUA_FCR[F*n*] is written to, the write event updates MUB_FSR[F*n*], after the event update latency. **NOTE** This field is not supported in every instance. The following table includes only supported registers. <table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>FSR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>FSR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>FSR</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>FSR</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>FSR</td></tr></table> 0b - MUA_FCR[Fn] = 0 1b - MUA_FCR[Fn] = 1
12 F12	MUA to MUB-Side Flag Contains flags configured by the values written to MUA_FCR[F*n*], where *n* = 0 to 31. F*n* is the MUB-side flag configured by the values written to MUA_FCR[F*n*]. When MUA_FCR[F*n*] is written to, the write event updates MUB_FSR[F*n*], after the event update latency.

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FSR</td></tr>
<tr><td>MU_3.MUB</td><td>—</td><td>FSR</td></tr>
<tr><td>MU_4.MUB</td><td>—</td><td>FSR</td></tr>
</table>
0b - MUA_FCR[Fn] = 0
1b - MUA_FCR[Fn] = 1
</td></tr>
<tr><td>11
F11</td><td>
MUA to MUB-Side Flag
Contains flags configured by the values written to MUA_FCR[Fn], where n = 0 to 31.
Fn is the MUB-side flag configured by the values written to MUA_FCR[Fn].
When MUA_FCR[Fn] is written to, the write event updates MUB_FSR[Fn], after the event update latency.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FSR</td></tr>
<tr><td>MU_3.MUB</td><td>—</td><td>FSR</td></tr>
<tr><td>MU_4.MUB</td><td>—</td><td>FSR</td></tr>
</table>
0b - MUA_FCR[Fn] = 0
1b - MUA_FCR[Fn] = 1
</td></tr>
<tr><td>10</td><td>MUA to MUB-Side Flag</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>F10</td><td>Contains flags configured by the values written to MUA_FCR[Fn], where n = 0 to 31.
Fn is the MUB-side flag configured by the values written to MUA_FCR[Fn].
When MUA_FCR[Fn] is written to, the write event updates MUB_FSR[Fn], after the event update latency.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>FSR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>FSR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>FSR</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>FSR</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>FSR</td></tr></table>0b - MUA_FCR[Fn] = 0
1b - MUA_FCR[Fn] = 1</td></tr>
<tr><td>9
F9</td><td>MUA to MUB-Side Flag
Contains flags configured by the values written to MUA_FCR[Fn], where n = 0 to 31.
Fn is the MUB-side flag configured by the values written to MUA_FCR[Fn].
When MUA_FCR[Fn] is written to, the write event updates MUB_FSR[Fn], after the event update latency.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>FSR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>FSR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>FSR</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>FSR</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>FSR</td></tr></table></td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
</table>
0b - MUA_FCR[Fn] = 0
1b - MUA_FCR[Fn] = 1
</td></tr>
<tr><td>8
F8</td><td>MUA to MUB-Side Flag
Contains flags configured by the values written to MUA_FCR[Fn], where n = 0 to 31.
Fn is the MUB-side flag configured by the values written to MUA_FCR[Fn].
When MUA_FCR[Fn] is written to, the write event updates MUB_FSR[Fn], after the event update latency.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FSR</td></tr>
<tr><td>MU_3.MUB</td><td>—</td><td>FSR</td></tr>
<tr><td>MU_4.MUB</td><td>—</td><td>FSR</td></tr>
</table>
0b - MUA_FCR[Fn] = 0
1b - MUA_FCR[Fn] = 1
</td></tr>
<tr><td>7
F7</td><td>MUA to MUB-Side Flag
Contains flags configured by the values written to MUA_FCR[Fn], where n = 0 to 31.
Fn is the MUB-side flag configured by the values written to MUA_FCR[Fn].
When MUA_FCR[Fn] is written to, the write event updates MUB_FSR[Fn], after the event update latency.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FSR</td><td>—</td></tr>
</table>
</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function

Instance	Field supported in	Field not supported in
MU_1.MUB	FSR	—
MU_2.MUB	—	FSR
MU_3.MUB	—	FSR
MU_4.MUB	—	FSR

0b - MUA_FCR[Fn] = 0

1b - MUA_FCR[Fn] = 1

6

F6

MUA to MUB-Side Flag

Contains flags configured by the values written to MUA_FCR[F*n*], where *n* = 0 to 31.

F*n* is the MUB-side flag configured by the values written to MUA_FCR[F*n*].

When MUA_FCR[F*n*] is written to, the write event updates MUB_FSR[F*n*], after the event update latency.

NOTE

This field is not supported in every instance. The following table includes only supported registers.

Instance	Field supported in	Field not supported in
MU_0.MUB	FSR	—
MU_1.MUB	FSR	—
MU_2.MUB	—	FSR
MU_3.MUB	—	FSR
MU_4.MUB	—	FSR

0b - MUA_FCR[Fn] = 0

1b - MUA_FCR[Fn] = 1

5

F5

MUA to MUB-Side Flag

Contains flags configured by the values written to MUA_FCR[F*n*], where *n* = 0 to 31.

F*n* is the MUB-side flag configured by the values written to MUA_FCR[F*n*].

When MUA_FCR[F*n*] is written to, the write event updates MUB_FSR[F*n*], after the event update latency.

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FSR</td></tr>
<tr><td>MU_3.MUB</td><td>—</td><td>FSR</td></tr>
<tr><td>MU_4.MUB</td><td>—</td><td>FSR</td></tr>
</table>
0b - MUA_FCR[Fn] = 0
1b - MUA_FCR[Fn] = 1
</td></tr>
<tr><td>4
F4</td><td>
MUA to MUB-Side Flag
Contains flags configured by the values written to MUA_FCR[Fn], where n = 0 to 31.
Fn is the MUB-side flag configured by the values written to MUA_FCR[Fn].
When MUA_FCR[Fn] is written to, the write event updates MUB_FSR[Fn], after the event update latency.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>FSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>FSR</td></tr>
<tr><td>MU_3.MUB</td><td>—</td><td>FSR</td></tr>
<tr><td>MU_4.MUB</td><td>—</td><td>FSR</td></tr>
</table>
0b - MUA_FCR[Fn] = 0
1b - MUA_FCR[Fn] = 1
</td></tr>
<tr><td>3</td><td>MUA to MUB-Side Flag</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>F3</td><td>Contains flags configured by the values written to MUA_FCR[Fn], where n = 0 to 31.
Fn is the MUB-side flag configured by the values written to MUA_FCR[Fn].
When MUA_FCR[Fn] is written to, the write event updates MUB_FSR[Fn], after the event update latency.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>FSR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>FSR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>FSR</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>FSR</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>FSR</td></tr></table>0b - MUA_FCR[Fn] = 0
1b - MUA_FCR[Fn] = 1</td></tr>
<tr><td>2
F2</td><td>MUA to MUB-Side Flag
Contains flags configured by the values written to MUA_FCR[Fn], where n = 0 to 31.
Fn is the MUB-side flag configured by the values written to MUA_FCR[Fn].
When MUA_FCR[Fn] is written to, the write event updates MUB_FSR[Fn], after the event update latency.
0b - MUA_FCR[Fn] = 0
1b - MUA_FCR[Fn] = 1</td></tr>
<tr><td>1
F1</td><td>MUA to MUB-Side Flag
Contains flags configured by the values written to MUA_FCR[Fn], where n = 0 to 31.
Fn is the MUB-side flag configured by the values written to MUA_FCR[Fn].
When MUA_FCR[Fn] is written to, the write event updates MUB_FSR[Fn], after the event update latency.
0b - MUA_FCR[Fn] = 0
1b - MUA_FCR[Fn] = 1</td></tr>
<tr><td>0
F0</td><td>MUA to MUB-Side Flag
Contains flags configured by the values written to MUA_FCR[Fn], where n = 0 to 31.
Fn is the MUB-side flag configured by the values written to MUA_FCR[Fn].</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When MUA_FCR[F*n*] is written to, the write event updates MUB_FSR[F*n*], after the event update latency. 0b - MUA_FCR[Fn] = 0 1b - MUA_FCR[Fn] = 1

40.7.2.10 General-Purpose Interrupt Enable (GIER)

Offset

Register	Offset
GIER	110h

Function

Contains the MUB general-purpose interrupt enable fields.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	GIE31	GIE30	GIE29	GIE28	GIE27	GIE26	GIE25	GIE24	GIE23	GIE22	GIE21	GIE20	GIE19	GIE18	GIE17	GIE16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	GIE15	GIE14	GIE13	GIE12	GIE11	GIE10	GIE9	GIE8	GIE7	GIE6	GIE5	GIE4	GIE3	GIE2	GIE1	GIE0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 GIE31	MUB General-purpose Interrupt Enable Enables general-purpose interrupt. There are 32 general-purpose interrupts (*n* = 0 to 31). When GIE*n* = 1, a general-purpose interrupt *n* request is issued to processor B when MUB GSR[GIP*n*] = 1. If GIE*n* = 0, the general-purpose interrupt request pending does not trigger the general-purpose interrupt. GIE*n* becomes 0 when MU resets. **NOTE** This field is not supported in every instance. The following table includes only supported registers.

Table continued from the previous page...

Field	Function

Instance	Field supported in	Field not supported in
MU_0.MUB	GIER	—
MU_1.MUB	GIER	—
MU_2.MUB	—	GIER
MU_3.MUB	—	GIER
MU_4.MUB	—	GIER

0b - Disable

1b - Enable

30

GIE30

MUB General-purpose Interrupt Enable

Enables general-purpose interrupt. There are 32 general-purpose interrupts (*n* = 0 to 31).

When GIE*n* = 1, a general-purpose interrupt *n* request is issued to processor B when MUB GSR[GIP*n*] = 1.

If GIE*n* = 0, the general-purpose interrupt request pending does not trigger the general-purpose interrupt.

GIE*n* becomes 0 when MU resets.

NOTE

This field is not supported in every instance. The following table includes only supported registers.

Instance	Field supported in	Field not supported in
MU_0.MUB	GIER	—
MU_1.MUB	GIER	—
MU_2.MUB	—	GIER
MU_3.MUB	—	GIER
MU_4.MUB	—	GIER

0b - Disable

1b - Enable

29

GIE29

MUB General-purpose Interrupt Enable

Enables general-purpose interrupt. There are 32 general-purpose interrupts (*n* = 0 to 31).

When GIE*n* = 1, a general-purpose interrupt *n* request is issued to processor B when MUB GSR[GIP*n*] = 1.

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>If GIEn = 0, the general-purpose interrupt request pending does not trigger the general-purpose interrupt.
GIEn becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>GIER</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>GIER</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>GIER</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>GIER</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>GIER</td></tr></table>0b - Disable
1b - Enable</td></tr>
<tr><td>28
GIE28</td><td>MUB General-purpose Interrupt Enable
Enables general-purpose interrupt. There are 32 general-purpose interrupts (n = 0 to 31).
When GIEn = 1, a general-purpose interrupt n request is issued to processor B when MUB GSR[GIPn] = 1.
If GIEn = 0, the general-purpose interrupt request pending does not trigger the general-purpose interrupt.
GIEn becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>GIER</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>GIER</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>GIER</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>GIER</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>GIER</td></tr></table></td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
</table>
0b - Disable
1b - Enable</td></tr>
<tr><td>27
GIE27</td><td>MUB General-purpose Interrupt Enable
Enables general-purpose interrupt. There are 32 general-purpose interrupts (n = 0 to 31).
When GIEn = 1, a general-purpose interrupt n request is issued to processor B when MUB GSR[GIPn] = 1.
If GIEn = 0, the general-purpose interrupt request pending does not trigger the general-purpose interrupt.
GIEn becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GIER</td></tr>
<tr><td>MU_3.MUB</td><td>—</td><td>GIER</td></tr>
<tr><td>MU_4.MUB</td><td>—</td><td>GIER</td></tr>
</table>
0b - Disable
1b - Enable</td></tr>
<tr><td>26
GIE26</td><td>MUB General-purpose Interrupt Enable
Enables general-purpose interrupt. There are 32 general-purpose interrupts (n = 0 to 31).
When GIEn = 1, a general-purpose interrupt n request is issued to processor B when MUB GSR[GIPn] = 1.
If GIEn = 0, the general-purpose interrupt request pending does not trigger the general-purpose interrupt.
GIEn becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function

Instance	Field supported in	Field not supported in
MU_0.MUB	GIER	—
MU_1.MUB	GIER	—
MU_2.MUB	—	GIER
MU_3.MUB	—	GIER
MU_4.MUB	—	GIER

0b - Disable

1b - Enable

25

GIE25

MUB General-purpose Interrupt Enable

Enables general-purpose interrupt. There are 32 general-purpose interrupts (*n* = 0 to 31).

When GIE*n* = 1, a general-purpose interrupt *n* request is issued to processor B when MUB GSR[GIP*n*] = 1.

If GIE*n* = 0, the general-purpose interrupt request pending does not trigger the general-purpose interrupt.

GIE*n* becomes 0 when MU resets.

NOTE

This field is not supported in every instance. The following table includes only supported registers.

Instance	Field supported in	Field not supported in
MU_0.MUB	GIER	—
MU_1.MUB	GIER	—
MU_2.MUB	—	GIER
MU_3.MUB	—	GIER
MU_4.MUB	—	GIER

0b - Disable

1b - Enable

24

GIE24

MUB General-purpose Interrupt Enable

Enables general-purpose interrupt. There are 32 general-purpose interrupts (*n* = 0 to 31).

When GIE*n* = 1, a general-purpose interrupt *n* request is issued to processor B when MUB GSR[GIP*n*] = 1.

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>If GIEn = 0, the general-purpose interrupt request pending does not trigger the general-purpose interrupt.
GIEn becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>GIER</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>GIER</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>GIER</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>GIER</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>GIER</td></tr></table>0b - Disable
1b - Enable</td></tr>
<tr><td>23
GIE23</td><td>MUB General-purpose Interrupt Enable
Enables general-purpose interrupt. There are 32 general-purpose interrupts (n = 0 to 31).
When GIEn = 1, a general-purpose interrupt n request is issued to processor B when MUB GSR[GIPn] = 1.
If GIEn = 0, the general-purpose interrupt request pending does not trigger the general-purpose interrupt.
GIEn becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>GIER</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>GIER</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>GIER</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>GIER</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>GIER</td></tr></table></td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
</table>
0b - Disable
1b - Enable</td></tr>
<tr><td>22
GIE22</td><td>MUB General-purpose Interrupt Enable
Enables general-purpose interrupt. There are 32 general-purpose interrupts (n = 0 to 31).
When GIEn = 1, a general-purpose interrupt n request is issued to processor B when MUB GSR[GIPn] = 1.
If GIEn = 0, the general-purpose interrupt request pending does not trigger the general-purpose interrupt.
GIEn becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GIER</td></tr>
<tr><td>MU_3.MUB</td><td>—</td><td>GIER</td></tr>
<tr><td>MU_4.MUB</td><td>—</td><td>GIER</td></tr>
</table>
0b - Disable
1b - Enable</td></tr>
<tr><td>21
GIE21</td><td>MUB General-purpose Interrupt Enable
Enables general-purpose interrupt. There are 32 general-purpose interrupts (n = 0 to 31).
When GIEn = 1, a general-purpose interrupt n request is issued to processor B when MUB GSR[GIPn] = 1.
If GIEn = 0, the general-purpose interrupt request pending does not trigger the general-purpose interrupt.
GIEn becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GIER</td></tr>
<tr><td>MU_3.MUB</td><td>—</td><td>GIER</td></tr>
<tr><td>MU_4.MUB</td><td>—</td><td>GIER</td></tr>
</table>
0b - Disable
1b - Enable
</td></tr>
<tr><td>20
GIE20</td><td>MUB General-purpose Interrupt Enable
Enables general-purpose interrupt. There are 32 general-purpose interrupts (n = 0 to 31).
When GIEn = 1, a general-purpose interrupt n request is issued to processor B when MUB GSR[GIPn] = 1.
If GIEn = 0, the general-purpose interrupt request pending does not trigger the general-purpose interrupt.
GIEn becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GIER</td></tr>
<tr><td>MU_3.MUB</td><td>—</td><td>GIER</td></tr>
<tr><td>MU_4.MUB</td><td>—</td><td>GIER</td></tr>
</table>
0b - Disable
1b - Enable
</td></tr>
<tr><td>19
GIE19</td><td>MUB General-purpose Interrupt Enable
Enables general-purpose interrupt. There are 32 general-purpose interrupts (n = 0 to 31).
When GIEn = 1, a general-purpose interrupt n request is issued to processor B when MUB GSR[GIPn] = 1.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>If GIEn = 0, the general-purpose interrupt request pending does not trigger the general-purpose interrupt.
GIEn becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>GIER</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>GIER</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>GIER</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>GIER</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>GIER</td></tr></table>0b - Disable
1b - Enable</td></tr>
<tr><td>18
GIE18</td><td>MUB General-purpose Interrupt Enable
Enables general-purpose interrupt. There are 32 general-purpose interrupts (n = 0 to 31).
When GIEn = 1, a general-purpose interrupt n request is issued to processor B when MUB GSR[GIPn] = 1.
If GIEn = 0, the general-purpose interrupt request pending does not trigger the general-purpose interrupt.
GIEn becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>GIER</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>GIER</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>GIER</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>GIER</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>GIER</td></tr></table></td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
</table>
0b - Disable
1b - Enable
</td></tr>
<tr><td>17
GIE17</td><td>MUB General-purpose Interrupt Enable
Enables general-purpose interrupt. There are 32 general-purpose interrupts (n = 0 to 31).
When GIEn = 1, a general-purpose interrupt n request is issued to processor B when MUB GSR[GIPn] = 1.
If GIEn = 0, the general-purpose interrupt request pending does not trigger the general-purpose interrupt.
GIEn becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GIER</td></tr>
<tr><td>MU_3.MUB</td><td>—</td><td>GIER</td></tr>
<tr><td>MU_4.MUB</td><td>—</td><td>GIER</td></tr>
</table>
0b - Disable
1b - Enable
</td></tr>
<tr><td>16
GIE16</td><td>MUB General-purpose Interrupt Enable
Enables general-purpose interrupt. There are 32 general-purpose interrupts (n = 0 to 31).
When GIEn = 1, a general-purpose interrupt n request is issued to processor B when MUB GSR[GIPn] = 1.
If GIEn = 0, the general-purpose interrupt request pending does not trigger the general-purpose interrupt.
GIEn becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GIER</td></tr>
<tr><td>MU_3.MUB</td><td>—</td><td>GIER</td></tr>
<tr><td>MU_4.MUB</td><td>—</td><td>GIER</td></tr>
</table>
0b - Disable
1b - Enable
</td></tr>
<tr><td>15
GIE15</td><td>MUB General-purpose Interrupt Enable
Enables general-purpose interrupt. There are 32 general-purpose interrupts (n = 0 to 31).
When GIEn = 1, a general-purpose interrupt n request is issued to processor B when MUB GSR[GIPn] = 1.
If GIEn = 0, the general-purpose interrupt request pending does not trigger the general-purpose interrupt.
GIEn becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GIER</td></tr>
<tr><td>MU_3.MUB</td><td>—</td><td>GIER</td></tr>
<tr><td>MU_4.MUB</td><td>—</td><td>GIER</td></tr>
</table>
0b - Disable
1b - Enable
</td></tr>
<tr><td>14
GIE14</td><td>MUB General-purpose Interrupt Enable
Enables general-purpose interrupt. There are 32 general-purpose interrupts (n = 0 to 31).
When GIEn = 1, a general-purpose interrupt n request is issued to processor B when MUB GSR[GIPn] = 1.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>If GIEn = 0, the general-purpose interrupt request pending does not trigger the general-purpose interrupt.
GIEn becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>GIER</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>GIER</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>GIER</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>GIER</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>GIER</td></tr></table>0b - Disable
1b - Enable</td></tr>
<tr><td>13
GIE13</td><td>MUB General-purpose Interrupt Enable
Enables general-purpose interrupt. There are 32 general-purpose interrupts (n = 0 to 31).
When GIEn = 1, a general-purpose interrupt n request is issued to processor B when MUB GSR[GIPn] = 1.
If GIEn = 0, the general-purpose interrupt request pending does not trigger the general-purpose interrupt.
GIEn becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>GIER</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>GIER</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>GIER</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>GIER</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>GIER</td></tr></table></td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
</table>
0b - Disable
1b - Enable
</td></tr>
<tr><td>12
GIE12</td><td>MUB General-purpose Interrupt Enable
Enables general-purpose interrupt. There are 32 general-purpose interrupts (n = 0 to 31).
When GIEn = 1, a general-purpose interrupt n request is issued to processor B when MUB GSR[GIPn] = 1.
If GIEn = 0, the general-purpose interrupt request pending does not trigger the general-purpose interrupt.
GIEn becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GIER</td></tr>
<tr><td>MU_3.MUB</td><td>—</td><td>GIER</td></tr>
<tr><td>MU_4.MUB</td><td>—</td><td>GIER</td></tr>
</table>
0b - Disable
1b - Enable</td></tr>
<tr><td>11
GIE11</td><td>MUB General-purpose Interrupt Enable
Enables general-purpose interrupt. There are 32 general-purpose interrupts (n = 0 to 31).
When GIEn = 1, a general-purpose interrupt n request is issued to processor B when MUB GSR[GIPn] = 1.
If GIEn = 0, the general-purpose interrupt request pending does not trigger the general-purpose interrupt.
GIEn becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function

Instance	Field supported in	Field not supported in
MU_0.MUB	GIER	—
MU_1.MUB	GIER	—
MU_2.MUB	—	GIER
MU_3.MUB	—	GIER
MU_4.MUB	—	GIER

0b - Disable

1b - Enable

10

GIE10

MUB General-purpose Interrupt Enable

Enables general-purpose interrupt. There are 32 general-purpose interrupts (*n* = 0 to 31).

When GIE*n* = 1, a general-purpose interrupt *n* request is issued to processor B when MUB GSR[GIP*n*] = 1.

If GIE*n* = 0, the general-purpose interrupt request pending does not trigger the general-purpose interrupt.

GIE*n* becomes 0 when MU resets.

NOTE

This field is not supported in every instance. The following table includes only supported registers.

Instance	Field supported in	Field not supported in
MU_0.MUB	GIER	—
MU_1.MUB	GIER	—
MU_2.MUB	—	GIER
MU_3.MUB	—	GIER
MU_4.MUB	—	GIER

0b - Disable

1b - Enable

9

GIE9

MUB General-purpose Interrupt Enable

Enables general-purpose interrupt. There are 32 general-purpose interrupts (*n* = 0 to 31).

When GIE*n* = 1, a general-purpose interrupt *n* request is issued to processor B when MUB GSR[GIP*n*] = 1.

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>If GIEn = 0, the general-purpose interrupt request pending does not trigger the general-purpose interrupt.
GIEn becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>GIER</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>GIER</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>GIER</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>GIER</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>GIER</td></tr></table>0b - Disable
1b - Enable</td></tr>
<tr><td>8
GIE8</td><td>MUB General-purpose Interrupt Enable
Enables general-purpose interrupt. There are 32 general-purpose interrupts (n = 0 to 31).
When GIEn = 1, a general-purpose interrupt n request is issued to processor B when MUB GSR[GIPn] = 1.
If GIEn = 0, the general-purpose interrupt request pending does not trigger the general-purpose interrupt.
GIEn becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>GIER</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>GIER</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>GIER</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>GIER</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>GIER</td></tr></table></td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
</table>
0b - Disable
1b - Enable</td></tr>
<tr><td>7
GIE7</td><td>MUB General-purpose Interrupt Enable
Enables general-purpose interrupt. There are 32 general-purpose interrupts (n = 0 to 31).
When GIEn = 1, a general-purpose interrupt n request is issued to processor B when MUB GSR[GIPn] = 1.
If GIEn = 0, the general-purpose interrupt request pending does not trigger the general-purpose interrupt.
GIEn becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GIER</td></tr>
<tr><td>MU_3.MUB</td><td>—</td><td>GIER</td></tr>
<tr><td>MU_4.MUB</td><td>—</td><td>GIER</td></tr>
</table>
0b - Disable
1b - Enable</td></tr>
<tr><td>6
GIE6</td><td>MUB General-purpose Interrupt Enable
Enables general-purpose interrupt. There are 32 general-purpose interrupts (n = 0 to 31).
When GIEn = 1, a general-purpose interrupt n request is issued to processor B when MUB GSR[GIPn] = 1.
If GIEn = 0, the general-purpose interrupt request pending does not trigger the general-purpose interrupt.
GIEn becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>GIER</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>GIER</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>GIER</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>GIER</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>GIER</td></tr></table> 0b - Disable 1b - Enable
5 GIE5	MUB General-purpose Interrupt Enable Enables general-purpose interrupt. There are 32 general-purpose interrupts (*n* = 0 to 31). When GIE*n* = 1, a general-purpose interrupt *n* request is issued to processor B when MUB GSR[GIP*n*] = 1. If GIE*n* = 0, the general-purpose interrupt request pending does not trigger the general-purpose interrupt. GIE*n* becomes 0 when MU resets. **NOTE** This field is not supported in every instance. The following table includes only supported registers. <table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>GIER</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>GIER</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>GIER</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>GIER</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>GIER</td></tr></table> 0b - Disable 1b - Enable
4 GIE4	MUB General-purpose Interrupt Enable Enables general-purpose interrupt. There are 32 general-purpose interrupts (*n* = 0 to 31). When GIE*n* = 1, a general-purpose interrupt *n* request is issued to processor B when MUB GSR[GIP*n*] = 1.

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>If GIEn = 0, the general-purpose interrupt request pending does not trigger the general-purpose interrupt.
GIEn becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>GIER</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>GIER</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>GIER</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>GIER</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>GIER</td></tr></table>0b - Disable
1b - Enable</td></tr>
<tr><td>3
GIE3</td><td>MUB General-purpose Interrupt Enable
Enables general-purpose interrupt. There are 32 general-purpose interrupts (n = 0 to 31).
When GIEn = 1, a general-purpose interrupt n request is issued to processor B when MUB GSR[GIPn] = 1.
If GIEn = 0, the general-purpose interrupt request pending does not trigger the general-purpose interrupt.
GIEn becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>GIER</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>GIER</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>GIER</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>GIER</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>GIER</td></tr></table></td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
</table>
0b - Disable
1b - Enable
</td></tr>
<tr><td>2
GIE2</td><td>MUB General-purpose Interrupt Enable
Enables general-purpose interrupt. There are 32 general-purpose interrupts (n = 0 to 31).
When GIEn = 1, a general-purpose interrupt n request is issued to processor B when MUB GSR[GIPn] = 1.
If GIEn = 0, the general-purpose interrupt request pending does not trigger the general-purpose interrupt.
GIEn becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GIER</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GIER</td></tr>
<tr><td>MU_3.MUB</td><td>—</td><td>GIER</td></tr>
<tr><td>MU_4.MUB</td><td>—</td><td>GIER</td></tr>
</table>
0b - Disable
1b - Enable
</td></tr>
<tr><td>1
GIE1</td><td>MUB General-purpose Interrupt Enable
Enables general-purpose interrupt. There are 32 general-purpose interrupts (n = 0 to 31).
When GIEn = 1, a general-purpose interrupt n request is issued to processor B when MUB GSR[GIPn] = 1.
If GIEn = 0, the general-purpose interrupt request pending does not trigger the general-purpose interrupt.
GIEn becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>GIER</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>GIER</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>GIER</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>GIER</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>GIER</td></tr></table> 0b - Disable 1b - Enable
0 GIE0	MUB General-purpose Interrupt Enable Enables general-purpose interrupt. There are 32 general-purpose interrupts (*n* = 0 to 31). When GIE*n* = 1, a general-purpose interrupt *n* request is issued to processor B when MUB GSR[GIP*n*] = 1. If GIE*n* = 0, the general-purpose interrupt request pending does not trigger the general-purpose interrupt. GIE*n* becomes 0 when MU resets. 0b - Disable 1b - Enable

40.7.2.11 General-Purpose Control (GCR)

Offset

Register	Offset
GCR	114h

Function

Contains the MUB general-purpose interrupt request fields.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	GIR31	GIR30	GIR29	GIR28	GIR27	GIR26	GIR25	GIR24	GIR23	GIR22	GIR21	GIR20	GIR19	GIR18	GIR17	GIR16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	GIR15	GIR14	GIR13	GIR12	GIR11	GIR10	GIR9	GIR8	GIR7	GIR6	GIR5	GIR4	GIR3	GIR2	GIR1	GIR0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>31
GIR31</td><td>MUB General-Purpose Interrupt Request
Specifies whether general-purpose interrupts are requested to MUA. There are 32 general-purpose interrupts (n = 0 to 31).
Writing 1 to GIRn sets MUA_GSR[GIPn]. If MUA_GIER[GIEn] = 1, a general-purpose interrupt request is triggered on processor A.
This field becomes 0 when MUA_GSR[GIPn] is cleared. This clearing informs MUB that the interrupt is accepted (cleared by software).
To ensure proper operations, verify that GIRn is 0 (no pending interrupt) before writing 1 to it.
This field becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GCR</td></tr>
<tr><td>MU_3.MUB</td><td>—</td><td>GCR</td></tr>
<tr><td>MU_4.MUB</td><td>—</td><td>GCR</td></tr>
</table>
0b - Not requested
1b - Requested</td></tr>
<tr><td>30
GIR30</td><td>MUB General-Purpose Interrupt Request</td></tr>
</table>

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>Specifies whether general-purpose interrupts are requested to MUA. There are 32 general-purpose interrupts (n = 0 to 31).
Writing 1 to GIRn sets MUA_GSR[GIPn]. If MUA_GIER[GIEn] = 1, a general-purpose interrupt request is triggered on processor A.
This field becomes 0 when MUA_GSR[GIPn] is cleared. This clearing informs MUB that the interrupt is accepted (cleared by software).
To ensure proper operations, verify that GIRn is 0 (no pending interrupt) before writing 1 to it.
This field becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GCR</td></tr>
<tr><td>MU_3.MUB</td><td>—</td><td>GCR</td></tr>
<tr><td>MU_4.MUB</td><td>—</td><td>GCR</td></tr>
</table>
0b - Not requested
1b - Requested</td></tr>
<tr><td>29
GIR29</td><td>MUB General-Purpose Interrupt Request
Specifies whether general-purpose interrupts are requested to MUA. There are 32 general-purpose interrupts (n = 0 to 31).
Writing 1 to GIRn sets MUA_GSR[GIPn]. If MUA_GIER[GIEn] = 1, a general-purpose interrupt request is triggered on processor A.
This field becomes 0 when MUA_GSR[GIPn] is cleared. This clearing informs MUB that the interrupt is accepted (cleared by software).
To ensure proper operations, verify that GIRn is 0 (no pending interrupt) before writing 1 to it.
This field becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GCR</td></tr>
<tr><td>MU_3.MUB</td><td>—</td><td>GCR</td></tr>
<tr><td>MU_4.MUB</td><td>—</td><td>GCR</td></tr>
</table>
0b - Not requested
1b - Requested
</td></tr>
<tr><td>28
GIR28</td><td>MUB General-Purpose Interrupt Request
Specifies whether general-purpose interrupts are requested to MUA. There are 32 general-purpose interrupts (n = 0 to 31).
Writing 1 to GIRn sets MUA_GSR[GIPn]. If MUA_GIER[GIEn] = 1, a general-purpose interrupt request is triggered on processor A.
This field becomes 0 when MUA_GSR[GIPn] is cleared. This clearing informs MUB that the interrupt is accepted (cleared by software).
To ensure proper operations, verify that GIRn is 0 (no pending interrupt) before writing 1 to it.
This field becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GCR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GCR</td></tr>
<tr><td>MU_3.MUB</td><td>—</td><td>GCR</td></tr>
<tr><td>MU_4.MUB</td><td>—</td><td>GCR</td></tr>
</table>
0b - Not requested
1b - Requested
</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>27
GIR27</td><td>MUB General-Purpose Interrupt Request
Specifies whether general-purpose interrupts are requested to MUA. There are 32 general-purpose interrupts (n = 0 to 31).
Writing 1 to GIRn sets MUA_GSR[GIPn]. If MUA_GIER[GIEn] = 1, a general-purpose interrupt request is triggered on processor A.
This field becomes 0 when MUA_GSR[GIPn] is cleared. This clearing informs MUB that the interrupt is accepted (cleared by software).
To ensure proper operations, verify that GIRn is 0 (no pending interrupt) before writing 1 to it.
This field becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>GCR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>GCR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>GCR</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>GCR</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>GCR</td></tr></table>0b - Not requested
1b - Requested</td></tr>
<tr><td>26
GIR26</td><td>MUB General-Purpose Interrupt Request
Specifies whether general-purpose interrupts are requested to MUA. There are 32 general-purpose interrupts (n = 0 to 31).
Writing 1 to GIRn sets MUA_GSR[GIPn]. If MUA_GIER[GIEn] = 1, a general-purpose interrupt request is triggered on processor A.
This field becomes 0 when MUA_GSR[GIPn] is cleared. This clearing informs MUB that the interrupt is accepted (cleared by software).
To ensure proper operations, verify that GIRn is 0 (no pending interrupt) before writing 1 to it.
This field becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function

Instance	Field supported in	Field not supported in
MU_0.MUB	GCR	—
MU_1.MUB	GCR	—
MU_2.MUB	—	GCR
MU_3.MUB	—	GCR
MU_4.MUB	—	GCR

0b - Not requested

1b - Requested

25

GIR25

MUB General-Purpose Interrupt Request

Specifies whether general-purpose interrupts are requested to MUA. There are 32 general-purpose interrupts (*n* = 0 to 31).

Writing 1 to GIR*n* sets MUA_GSR[GIP*n*]. If MUA_GIER[GIE*n*] = 1, a general-purpose interrupt request is triggered on processor A.

This field becomes 0 when MUA_GSR[GIP*n*] is cleared. This clearing informs MUB that the interrupt is accepted (cleared by software).

To ensure proper operations, verify that GIR*n* is 0 (no pending interrupt) before writing 1 to it.

This field becomes 0 when MU resets.

NOTE

This field is not supported in every instance. The following table includes only supported registers.

Instance	Field supported in	Field not supported in
MU_0.MUB	GCR	—
MU_1.MUB	GCR	—
MU_2.MUB	—	GCR
MU_3.MUB	—	GCR
MU_4.MUB	—	GCR

0b - Not requested

1b - Requested

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>24
GIR24</td><td>MUB General-Purpose Interrupt Request
Specifies whether general-purpose interrupts are requested to MUA. There are 32 general-purpose interrupts (n = 0 to 31).
Writing 1 to GIRn sets MUA_GSR[GIPn]. If MUA_GIER[GIEn] = 1, a general-purpose interrupt request is triggered on processor A.
This field becomes 0 when MUA_GSR[GIPn] is cleared. This clearing informs MUB that the interrupt is accepted (cleared by software).
To ensure proper operations, verify that GIRn is 0 (no pending interrupt) before writing 1 to it.
This field becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>GCR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>GCR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>GCR</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>GCR</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>GCR</td></tr></table>0b - Not requested
1b - Requested</td></tr>
<tr><td>23
GIR23</td><td>MUB General-Purpose Interrupt Request
Specifies whether general-purpose interrupts are requested to MUA. There are 32 general-purpose interrupts (n = 0 to 31).
Writing 1 to GIRn sets MUA_GSR[GIPn]. If MUA_GIER[GIEn] = 1, a general-purpose interrupt request is triggered on processor A.
This field becomes 0 when MUA_GSR[GIPn] is cleared. This clearing informs MUB that the interrupt is accepted (cleared by software).
To ensure proper operations, verify that GIRn is 0 (no pending interrupt) before writing 1 to it.
This field becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function

Instance	Field supported in	Field not supported in
MU_0.MUB	GCR	—
MU_1.MUB	GCR	—
MU_2.MUB	—	GCR
MU_3.MUB	—	GCR
MU_4.MUB	—	GCR

0b - Not requested

1b - Requested

22

GIR22

MUB General-Purpose Interrupt Request

Specifies whether general-purpose interrupts are requested to MUA. There are 32 general-purpose interrupts (*n* = 0 to 31).

Writing 1 to GIR*n* sets MUA_GSR[GIP*n*]. If MUA_GIER[GIE*n*] = 1, a general-purpose interrupt request is triggered on processor A.

This field becomes 0 when MUA_GSR[GIP*n*] is cleared. This clearing informs MUB that the interrupt is accepted (cleared by software).

To ensure proper operations, verify that GIR*n* is 0 (no pending interrupt) before writing 1 to it.

This field becomes 0 when MU resets.

NOTE

This field is not supported in every instance. The following table includes only supported registers.

Instance	Field supported in	Field not supported in
MU_0.MUB	GCR	—
MU_1.MUB	GCR	—
MU_2.MUB	—	GCR
MU_3.MUB	—	GCR
MU_4.MUB	—	GCR

0b - Not requested

1b - Requested

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>21
GIR21</td><td>MUB General-Purpose Interrupt Request
Specifies whether general-purpose interrupts are requested to MUA. There are 32 general-purpose interrupts (n = 0 to 31).
Writing 1 to GIRn sets MUA_GSR[GIPn]. If MUA_GIER[GIEn] = 1, a general-purpose interrupt request is triggered on processor A.
This field becomes 0 when MUA_GSR[GIPn] is cleared. This clearing informs MUB that the interrupt is accepted (cleared by software).
To ensure proper operations, verify that GIRn is 0 (no pending interrupt) before writing 1 to it.
This field becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>GCR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>GCR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>GCR</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>GCR</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>GCR</td></tr></table>0b - Not requested
1b - Requested</td></tr>
<tr><td>20
GIR20</td><td>MUB General-Purpose Interrupt Request
Specifies whether general-purpose interrupts are requested to MUA. There are 32 general-purpose interrupts (n = 0 to 31).
Writing 1 to GIRn sets MUA_GSR[GIPn]. If MUA_GIER[GIEn] = 1, a general-purpose interrupt request is triggered on processor A.
This field becomes 0 when MUA_GSR[GIPn] is cleared. This clearing informs MUB that the interrupt is accepted (cleared by software).
To ensure proper operations, verify that GIRn is 0 (no pending interrupt) before writing 1 to it.
This field becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function

Instance	Field supported in	Field not supported in
MU_0.MUB	GCR	—
MU_1.MUB	GCR	—
MU_2.MUB	—	GCR
MU_3.MUB	—	GCR
MU_4.MUB	—	GCR

0b - Not requested

1b - Requested

19

GIR19

MUB General-Purpose Interrupt Request

Specifies whether general-purpose interrupts are requested to MUA. There are 32 general-purpose interrupts (*n* = 0 to 31).

Writing 1 to GIR*n* sets MUA_GSR[GIP*n*]. If MUA_GIER[GIE*n*] = 1, a general-purpose interrupt request is triggered on processor A.

This field becomes 0 when MUA_GSR[GIP*n*] is cleared. This clearing informs MUB that the interrupt is accepted (cleared by software).

To ensure proper operations, verify that GIR*n* is 0 (no pending interrupt) before writing 1 to it.

This field becomes 0 when MU resets.

NOTE

This field is not supported in every instance. The following table includes only supported registers.

Instance	Field supported in	Field not supported in
MU_0.MUB	GCR	—
MU_1.MUB	GCR	—
MU_2.MUB	—	GCR
MU_3.MUB	—	GCR
MU_4.MUB	—	GCR

0b - Not requested

1b - Requested

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>18
GIR18</td><td>MUB General-Purpose Interrupt Request
Specifies whether general-purpose interrupts are requested to MUA. There are 32 general-purpose interrupts (n = 0 to 31).
Writing 1 to GIRn sets MUA_GSR[GIPn]. If MUA_GIER[GIEn] = 1, a general-purpose interrupt request is triggered on processor A.
This field becomes 0 when MUA_GSR[GIPn] is cleared. This clearing informs MUB that the interrupt is accepted (cleared by software).
To ensure proper operations, verify that GIRn is 0 (no pending interrupt) before writing 1 to it.
This field becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>GCR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>GCR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>GCR</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>GCR</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>GCR</td></tr></table>0b - Not requested
1b - Requested</td></tr>
<tr><td>17
GIR17</td><td>MUB General-Purpose Interrupt Request
Specifies whether general-purpose interrupts are requested to MUA. There are 32 general-purpose interrupts (n = 0 to 31).
Writing 1 to GIRn sets MUA_GSR[GIPn]. If MUA_GIER[GIEn] = 1, a general-purpose interrupt request is triggered on processor A.
This field becomes 0 when MUA_GSR[GIPn] is cleared. This clearing informs MUB that the interrupt is accepted (cleared by software).
To ensure proper operations, verify that GIRn is 0 (no pending interrupt) before writing 1 to it.
This field becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function

Instance	Field supported in	Field not supported in
MU_0.MUB	GCR	—
MU_1.MUB	GCR	—
MU_2.MUB	—	GCR
MU_3.MUB	—	GCR
MU_4.MUB	—	GCR

0b - Not requested

1b - Requested

16

GIR16

MUB General-Purpose Interrupt Request

Specifies whether general-purpose interrupts are requested to MUA. There are 32 general-purpose interrupts (*n* = 0 to 31).

Writing 1 to GIR*n* sets MUA_GSR[GIP*n*]. If MUA_GIER[GIE*n*] = 1, a general-purpose interrupt request is triggered on processor A.

This field becomes 0 when MUA_GSR[GIP*n*] is cleared. This clearing informs MUB that the interrupt is accepted (cleared by software).

To ensure proper operations, verify that GIR*n* is 0 (no pending interrupt) before writing 1 to it.

This field becomes 0 when MU resets.

NOTE

This field is not supported in every instance. The following table includes only supported registers.

Instance	Field supported in	Field not supported in
MU_0.MUB	GCR	—
MU_1.MUB	GCR	—
MU_2.MUB	—	GCR
MU_3.MUB	—	GCR
MU_4.MUB	—	GCR

0b - Not requested

1b - Requested

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>15
GIR15</td><td>MUB General-Purpose Interrupt Request
Specifies whether general-purpose interrupts are requested to MUA. There are 32 general-purpose interrupts (n = 0 to 31).
Writing 1 to GIRn sets MUA_GSR[GIPn]. If MUA_GIER[GIEn] = 1, a general-purpose interrupt request is triggered on processor A.
This field becomes 0 when MUA_GSR[GIPn] is cleared. This clearing informs MUB that the interrupt is accepted (cleared by software).
To ensure proper operations, verify that GIRn is 0 (no pending interrupt) before writing 1 to it.
This field becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>GCR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>GCR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>GCR</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>GCR</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>GCR</td></tr></table>0b - Not requested
1b - Requested</td></tr>
<tr><td>14
GIR14</td><td>MUB General-Purpose Interrupt Request
Specifies whether general-purpose interrupts are requested to MUA. There are 32 general-purpose interrupts (n = 0 to 31).
Writing 1 to GIRn sets MUA_GSR[GIPn]. If MUA_GIER[GIEn] = 1, a general-purpose interrupt request is triggered on processor A.
This field becomes 0 when MUA_GSR[GIPn] is cleared. This clearing informs MUB that the interrupt is accepted (cleared by software).
To ensure proper operations, verify that GIRn is 0 (no pending interrupt) before writing 1 to it.
This field becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function

Instance	Field supported in	Field not supported in
MU_0.MUB	GCR	—
MU_1.MUB	GCR	—
MU_2.MUB	—	GCR
MU_3.MUB	—	GCR
MU_4.MUB	—	GCR

0b - Not requested

1b - Requested

13

GIR13

MUB General-Purpose Interrupt Request

Specifies whether general-purpose interrupts are requested to MUA. There are 32 general-purpose interrupts (*n* = 0 to 31).

Writing 1 to GIR*n* sets MUA_GSR[GIP*n*]. If MUA_GIER[GIE*n*] = 1, a general-purpose interrupt request is triggered on processor A.

This field becomes 0 when MUA_GSR[GIP*n*] is cleared. This clearing informs MUB that the interrupt is accepted (cleared by software).

To ensure proper operations, verify that GIR*n* is 0 (no pending interrupt) before writing 1 to it.

This field becomes 0 when MU resets.

NOTE

This field is not supported in every instance. The following table includes only supported registers.

Instance	Field supported in	Field not supported in
MU_0.MUB	GCR	—
MU_1.MUB	GCR	—
MU_2.MUB	—	GCR
MU_3.MUB	—	GCR
MU_4.MUB	—	GCR

0b - Not requested

1b - Requested

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>12
GIR12</td><td>MUB General-Purpose Interrupt Request
Specifies whether general-purpose interrupts are requested to MUA. There are 32 general-purpose interrupts (n = 0 to 31).
Writing 1 to GIRn sets MUA_GSR[GIPn]. If MUA_GIER[GIEn] = 1, a general-purpose interrupt request is triggered on processor A.
This field becomes 0 when MUA_GSR[GIPn] is cleared. This clearing informs MUB that the interrupt is accepted (cleared by software).
To ensure proper operations, verify that GIRn is 0 (no pending interrupt) before writing 1 to it.
This field becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>GCR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>GCR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>GCR</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>GCR</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>GCR</td></tr></table>0b - Not requested
1b - Requested</td></tr>
<tr><td>11
GIR11</td><td>MUB General-Purpose Interrupt Request
Specifies whether general-purpose interrupts are requested to MUA. There are 32 general-purpose interrupts (n = 0 to 31).
Writing 1 to GIRn sets MUA_GSR[GIPn]. If MUA_GIER[GIEn] = 1, a general-purpose interrupt request is triggered on processor A.
This field becomes 0 when MUA_GSR[GIPn] is cleared. This clearing informs MUB that the interrupt is accepted (cleared by software).
To ensure proper operations, verify that GIRn is 0 (no pending interrupt) before writing 1 to it.
This field becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function

Instance	Field supported in	Field not supported in
MU_0.MUB	GCR	—
MU_1.MUB	GCR	—
MU_2.MUB	—	GCR
MU_3.MUB	—	GCR
MU_4.MUB	—	GCR

0b - Not requested

1b - Requested

10

GIR10

MUB General-Purpose Interrupt Request

Specifies whether general-purpose interrupts are requested to MUA. There are 32 general-purpose interrupts (*n* = 0 to 31).

Writing 1 to GIR*n* sets MUA_GSR[GIP*n*]. If MUA_GIER[GIE*n*] = 1, a general-purpose interrupt request is triggered on processor A.

This field becomes 0 when MUA_GSR[GIP*n*] is cleared. This clearing informs MUB that the interrupt is accepted (cleared by software).

To ensure proper operations, verify that GIR*n* is 0 (no pending interrupt) before writing 1 to it.

This field becomes 0 when MU resets.

NOTE

This field is not supported in every instance. The following table includes only supported registers.

Instance	Field supported in	Field not supported in
MU_0.MUB	GCR	—
MU_1.MUB	GCR	—
MU_2.MUB	—	GCR
MU_3.MUB	—	GCR
MU_4.MUB	—	GCR

0b - Not requested

1b - Requested

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>9
GIR9</td><td>MUB General-Purpose Interrupt Request
Specifies whether general-purpose interrupts are requested to MUA. There are 32 general-purpose interrupts (n = 0 to 31).
Writing 1 to GIRn sets MUA_GSR[GIPn]. If MUA_GIER[GIEn] = 1, a general-purpose interrupt request is triggered on processor A.
This field becomes 0 when MUA_GSR[GIPn] is cleared. This clearing informs MUB that the interrupt is accepted (cleared by software).
To ensure proper operations, verify that GIRn is 0 (no pending interrupt) before writing 1 to it.
This field becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>GCR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>GCR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>GCR</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>GCR</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>GCR</td></tr></table>0b - Not requested
1b - Requested</td></tr>
<tr><td>8
GIR8</td><td>MUB General-Purpose Interrupt Request
Specifies whether general-purpose interrupts are requested to MUA. There are 32 general-purpose interrupts (n = 0 to 31).
Writing 1 to GIRn sets MUA_GSR[GIPn]. If MUA_GIER[GIEn] = 1, a general-purpose interrupt request is triggered on processor A.
This field becomes 0 when MUA_GSR[GIPn] is cleared. This clearing informs MUB that the interrupt is accepted (cleared by software).
To ensure proper operations, verify that GIRn is 0 (no pending interrupt) before writing 1 to it.
This field becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function

Instance	Field supported in	Field not supported in
MU_0.MUB	GCR	—
MU_1.MUB	GCR	—
MU_2.MUB	—	GCR
MU_3.MUB	—	GCR
MU_4.MUB	—	GCR

0b - Not requested

1b - Requested

7

GIR7

MUB General-Purpose Interrupt Request

Specifies whether general-purpose interrupts are requested to MUA. There are 32 general-purpose interrupts (*n* = 0 to 31).

Writing 1 to GIR*n* sets MUA_GSR[GIP*n*]. If MUA_GIER[GIE*n*] = 1, a general-purpose interrupt request is triggered on processor A.

This field becomes 0 when MUA_GSR[GIP*n*] is cleared. This clearing informs MUB that the interrupt is accepted (cleared by software).

To ensure proper operations, verify that GIR*n* is 0 (no pending interrupt) before writing 1 to it.

This field becomes 0 when MU resets.

NOTE

This field is not supported in every instance. The following table includes only supported registers.

Instance	Field supported in	Field not supported in
MU_0.MUB	GCR	—
MU_1.MUB	GCR	—
MU_2.MUB	—	GCR
MU_3.MUB	—	GCR
MU_4.MUB	—	GCR

0b - Not requested

1b - Requested

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>6
GIR6</td><td>MUB General-Purpose Interrupt Request
Specifies whether general-purpose interrupts are requested to MUA. There are 32 general-purpose interrupts (n = 0 to 31).
Writing 1 to GIRn sets MUA_GSR[GIPn]. If MUA_GIER[GIEn] = 1, a general-purpose interrupt request is triggered on processor A.
This field becomes 0 when MUA_GSR[GIPn] is cleared. This clearing informs MUB that the interrupt is accepted (cleared by software).
To ensure proper operations, verify that GIRn is 0 (no pending interrupt) before writing 1 to it.
This field becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>GCR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>GCR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>GCR</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>GCR</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>GCR</td></tr></table>0b - Not requested
1b - Requested</td></tr>
<tr><td>5
GIR5</td><td>MUB General-Purpose Interrupt Request
Specifies whether general-purpose interrupts are requested to MUA. There are 32 general-purpose interrupts (n = 0 to 31).
Writing 1 to GIRn sets MUA_GSR[GIPn]. If MUA_GIER[GIEn] = 1, a general-purpose interrupt request is triggered on processor A.
This field becomes 0 when MUA_GSR[GIPn] is cleared. This clearing informs MUB that the interrupt is accepted (cleared by software).
To ensure proper operations, verify that GIRn is 0 (no pending interrupt) before writing 1 to it.
This field becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function

Instance	Field supported in	Field not supported in
MU_0.MUB	GCR	—
MU_1.MUB	GCR	—
MU_2.MUB	—	GCR
MU_3.MUB	—	GCR
MU_4.MUB	—	GCR

0b - Not requested

1b - Requested

4

GIR4

MUB General-Purpose Interrupt Request

Specifies whether general-purpose interrupts are requested to MUA. There are 32 general-purpose interrupts (*n* = 0 to 31).

Writing 1 to GIR*n* sets MUA_GSR[GIP*n*]. If MUA_GIER[GIE*n*] = 1, a general-purpose interrupt request is triggered on processor A.

This field becomes 0 when MUA_GSR[GIP*n*] is cleared. This clearing informs MUB that the interrupt is accepted (cleared by software).

To ensure proper operations, verify that GIR*n* is 0 (no pending interrupt) before writing 1 to it.

This field becomes 0 when MU resets.

NOTE

This field is not supported in every instance. The following table includes only supported registers.

Instance	Field supported in	Field not supported in
MU_0.MUB	GCR	—
MU_1.MUB	GCR	—
MU_2.MUB	—	GCR
MU_3.MUB	—	GCR
MU_4.MUB	—	GCR

0b - Not requested

1b - Requested

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>3
GIR3</td><td>MUB General-Purpose Interrupt Request
Specifies whether general-purpose interrupts are requested to MUA. There are 32 general-purpose interrupts (n = 0 to 31).
Writing 1 to GIRn sets MUA_GSR[GIPn]. If MUA_GIER[GIEn] = 1, a general-purpose interrupt request is triggered on processor A.
This field becomes 0 when MUA_GSR[GIPn] is cleared. This clearing informs MUB that the interrupt is accepted (cleared by software).
To ensure proper operations, verify that GIRn is 0 (no pending interrupt) before writing 1 to it.
This field becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>GCR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>GCR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>GCR</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>GCR</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>GCR</td></tr></table>0b - Not requested
1b - Requested</td></tr>
<tr><td>2
GIR2</td><td>MUB General-Purpose Interrupt Request
Specifies whether general-purpose interrupts are requested to MUA. There are 32 general-purpose interrupts (n = 0 to 31).
Writing 1 to GIRn sets MUA_GSR[GIPn]. If MUA_GIER[GIEn] = 1, a general-purpose interrupt request is triggered on processor A.
This field becomes 0 when MUA_GSR[GIPn] is cleared. This clearing informs MUB that the interrupt is accepted (cleared by software).
To ensure proper operations, verify that GIRn is 0 (no pending interrupt) before writing 1 to it.
This field becomes 0 when MU resets.
NOTE
This field is not supported in every instance. The following table includes only supported registers.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function

Instance	Field supported in	Field not supported in
MU_0.MUB	GCR	—
MU_1.MUB	GCR	—
MU_2.MUB	—	GCR
MU_3.MUB	—	GCR
MU_4.MUB	—	GCR

0b - Not requested

1b - Requested

1

GIR1

MUB General-Purpose Interrupt Request

Specifies whether general-purpose interrupts are requested to MUA. There are 32 general-purpose interrupts (*n* = 0 to 31).

Writing 1 to GIR*n* sets MUA_GSR[GIP*n*]. If MUA_GIER[GIE*n*] = 1, a general-purpose interrupt request is triggered on processor A.

This field becomes 0 when MUA_GSR[GIP*n*] is cleared. This clearing informs MUB that the interrupt is accepted (cleared by software).

To ensure proper operations, verify that GIR*n* is 0 (no pending interrupt) before writing 1 to it.

This field becomes 0 when MU resets.

NOTE

This field is not supported in every instance. The following table includes only supported registers.

Instance	Field supported in	Field not supported in
MU_0.MUB	GCR	—
MU_1.MUB	GCR	—
MU_2.MUB	—	GCR
MU_3.MUB	—	GCR
MU_4.MUB	—	GCR

0b - Not requested

1b - Requested

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 GIR0	MUB General-Purpose Interrupt Request Specifies whether general-purpose interrupts are requested to MUA. There are 32 general-purpose interrupts (*n* = 0 to 31). Writing 1 to GIR*n* sets MUA_GSR[GIP*n*]. If MUA_GIER[GIE*n*] = 1, a general-purpose interrupt request is triggered on processor A. This field becomes 0 when MUA_GSR[GIP*n*] is cleared. This clearing informs MUB that the interrupt is accepted (cleared by software). To ensure proper operations, verify that GIR*n* is 0 (no pending interrupt) before writing 1 to it. This field becomes 0 when MU resets. 0b - Not requested 1b - Requested

40.7.2.12 General-purpose Status (GSR)

Offset

Register	Offset
GSR	118h

Function

Contains the status of the MUB general-purpose interrupt pending requests.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	GIP31	GIP30	GIP29	GIP28	GIP27	GIP26	GIP25	GIP24	GIP23	GIP22	GIP21	GIP20	GIP19	GIP18	GIP17	GIP16
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	GIP15	GIP14	GIP13	GIP12	GIP11	GIP10	GIP9	GIP8	GIP7	GIP6	GIP5	GIP4	GIP3	GIP2	GIP1	GIP0
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31	MUB General-Purpose Interrupt Request Pending

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>GIP31</td><td>Indicates whether a general-purpose interrupt request is pending. There are 32 general-purpose interrupts (n = 0 to 31).
GIPn informs MUB that MUA_GCR[GIRn] changed from 0 to 1. If MUB_GIER[GIEn] = 1, a general-purpose interrupt request is issued to processor B.
GIPn is cleared when MU resets.
After GIPn is cleared, if MUB_GIER[GIEn] = 1, the general-purpose interrupt request is cleared on the MUB side.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>GSR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>GSR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>GSR</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>GSR</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>GSR</td></tr></table>NOTE
This field behaves differently for register reads and writes.
When reading
0b - Not pending
1b - Pending
When writing
0b - No effect
1b - Clear the flag</td></tr>
<tr><td>30
GIP30</td><td>MUB General-Purpose Interrupt Request Pending
Indicates whether a general-purpose interrupt request is pending. There are 32 general-purpose interrupts (n = 0 to 31).
GIPn informs MUB that MUA_GCR[GIRn] changed from 0 to 1. If MUB_GIER[GIEn] = 1, a general-purpose interrupt request is issued to processor B.
GIPn is cleared when MU resets.
After GIPn is cleared, if MUB_GIER[GIEn] = 1, the general-purpose interrupt request is cleared on the MUB side.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field is not supported in every instance. The following table includes only supported registers.

Instance	Field supported in	Field not supported in
MU_0.MUB	GSR	—
MU_1.MUB	GSR	—
MU_2.MUB	—	GSR
MU_3.MUB	—	GSR
MU_4.MUB	—	GSR

NOTE
This field behaves differently for register reads and writes.

When reading

0b - Not pending

1b - Pending

When writing

0b - No effect

1b - Clear the flag

29

GIP29

MUB General-Purpose Interrupt Request Pending

Indicates whether a general-purpose interrupt request is pending. There are 32 general-purpose interrupts (*n* = 0 to 31).

GIP*n* informs MUB that MUA_GCR[GIR*n*] changed from 0 to 1. If MUB_GIER[GIE*n*] = 1, a general-purpose interrupt request is issued to processor B.

GIP*n* is cleared when MU resets.

After GIP*n* is cleared, if MUB_GIER[GIE*n*] = 1, the general-purpose interrupt request is cleared on the MUB side.

NOTE
This field is not supported in every instance. The following table includes only supported registers.

Instance	Field supported in	Field not supported in
MU_0.MUB	GSR	—

Table continues on the next page...

Table continued from the previous page...

Field	Function
	<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_1.MUB</td><td>GSR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>GSR</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>GSR</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>GSR</td></tr></table> **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not pending 1b - Pending When writing 0b - No effect 1b - Clear the flag
28 GIP28	MUB General-Purpose Interrupt Request Pending Indicates whether a general-purpose interrupt request is pending. There are 32 general-purpose interrupts (*n* = 0 to 31). GIP*n* informs MUB that MUA_GCR[GIR*n*] changed from 0 to 1. If MUB_GIER[GIE*n*] = 1, a general-purpose interrupt request is issued to processor B. GIP*n* is cleared when MU resets. After GIP*n* is cleared, if MUB_GIER[GIE*n*] = 1, the general-purpose interrupt request is cleared on the MUB side. **NOTE** This field is not supported in every instance. The following table includes only supported registers. <table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>GSR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>GSR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>GSR</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>GSR</td></tr></table>

Table continues on the next page...

Table continued from the previous page...

Field	Function

Instance	Field supported in	Field not supported in
MU_4.MUB	—	GSR

NOTE

This field behaves differently for register reads and writes.

When reading

0b - Not pending

1b - Pending

When writing

0b - No effect

1b - Clear the flag

27

GIP27

MUB General-Purpose Interrupt Request Pending

Indicates whether a general-purpose interrupt request is pending. There are 32 general-purpose interrupts (*n* = 0 to 31).

GIP*n* informs MUB that MUA_GCR[GIR*n*] changed from 0 to 1. If MUB_GIER[GIE*n*] = 1, a general-purpose interrupt request is issued to processor B.

GIP*n* is cleared when MU resets.

After GIP*n* is cleared, if MUB_GIER[GIE*n*] = 1, the general-purpose interrupt request is cleared on the MUB side.

NOTE

This field is not supported in every instance. The following table includes only supported registers.

Instance	Field supported in	Field not supported in
MU_0.MUB	GSR	—
MU_1.MUB	GSR	—
MU_2.MUB	—	GSR
MU_3.MUB	—	GSR
MU_4.MUB	—	GSR

NOTE

This field behaves differently for register reads and writes.

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>When reading
0b - Not pending
1b - Pending
When writing
0b - No effect
1b - Clear the flag</td></tr>
<tr><td>26
GIP26</td><td>MUB General-Purpose Interrupt Request Pending
Indicates whether a general-purpose interrupt request is pending. There are 32 general-purpose interrupts (n = 0 to 31).
GIPn informs MUB that MUA_GCR[GIRn] changed from 0 to 1. If MUB_GIER[GIEn] = 1, a general-purpose interrupt request is issued to processor B.
GIPn is cleared when MU resets.
After GIPn is cleared, if MUB_GIER[GIEn] = 1, the general-purpose interrupt request is cleared on the MUB side.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>GSR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>GSR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>GSR</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>GSR</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>GSR</td></tr></table>NOTE
This field behaves differently for register reads and writes.
When reading
0b - Not pending
1b - Pending
When writing
0b - No effect
1b - Clear the flag</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>25
GIP25</td><td>MUB General-Purpose Interrupt Request Pending
Indicates whether a general-purpose interrupt request is pending. There are 32 general-purpose interrupts (n = 0 to 31).
GIPn informs MUB that MUA_GCR[GIRn] changed from 0 to 1. If MUB_GIER[GIEn] = 1, a general-purpose interrupt request is issued to processor B.
GIPn is cleared when MU resets.
After GIPn is cleared, if MUB_GIER[GIEn] = 1, the general-purpose interrupt request is cleared on the MUB side.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>GSR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>GSR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>GSR</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>GSR</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>GSR</td></tr></table>NOTE
This field behaves differently for register reads and writes.
When reading
0b - Not pending
1b - Pending
When writing
0b - No effect
1b - Clear the flag</td></tr>
<tr><td>24
GIP24</td><td>MUB General-Purpose Interrupt Request Pending
Indicates whether a general-purpose interrupt request is pending. There are 32 general-purpose interrupts (n = 0 to 31).
GIPn informs MUB that MUA_GCR[GIRn] changed from 0 to 1. If MUB_GIER[GIEn] = 1, a general-purpose interrupt request is issued to processor B.
GIPn is cleared when MU resets.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>After GIPn is cleared, if MUB_GIER[GIEn] = 1, the general-purpose interrupt request is cleared on the MUB side.

NOTE
This field is not supported in every instance. The following table includes only supported registers.

<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GSR</td></tr>
<tr><td>MU_3.MUB</td><td>—</td><td>GSR</td></tr>
<tr><td>MU_4.MUB</td><td>—</td><td>GSR</td></tr>
</table>

NOTE
This field behaves differently for register reads and writes.

When reading
0b - Not pending
1b - Pending
When writing
0b - No effect
1b - Clear the flag</td></tr>
<tr><td>23
GIP23</td><td>MUB General-Purpose Interrupt Request Pending
Indicates whether a general-purpose interrupt request is pending. There are 32 general-purpose interrupts (n = 0 to 31).
GIPn informs MUB that MUA_GCR[GIRn] changed from 0 to 1. If MUB_GIER[GIEn] = 1, a general-purpose interrupt request is issued to processor B.
GIPn is cleared when MU resets.
After GIPn is cleared, if MUB_GIER[GIEn] = 1, the general-purpose interrupt request is cleared on the MUB side.

NOTE
This field is not supported in every instance. The following table includes only supported registers.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>GSR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>GSR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>GSR</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>GSR</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>GSR</td></tr></table> **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not pending 1b - Pending When writing 0b - No effect 1b - Clear the flag
22 GIP22	MUB General-Purpose Interrupt Request Pending Indicates whether a general-purpose interrupt request is pending. There are 32 general-purpose interrupts (*n* = 0 to 31). GIP*n* informs MUB that MUA_GCR[GIR*n*] changed from 0 to 1. If MUB_GIER[GIE*n*] = 1, a general-purpose interrupt request is issued to processor B. GIP*n* is cleared when MU resets. After GIP*n* is cleared, if MUB_GIER[GIE*n*] = 1, the general-purpose interrupt request is cleared on the MUB side. **NOTE** This field is not supported in every instance. The following table includes only supported registers. <table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>GSR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>GSR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>GSR</td></tr></table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_3.MUB</td><td>—</td><td>GSR</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>GSR</td></tr></table> **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not pending 1b - Pending When writing 0b - No effect 1b - Clear the flag
21 GIP21	MUB General-Purpose Interrupt Request Pending Indicates whether a general-purpose interrupt request is pending. There are 32 general-purpose interrupts (*n* = 0 to 31). GIP*n* informs MUB that MUA_GCR[GIR*n*] changed from 0 to 1. If MUB_GIER[GIE*n*] = 1, a general-purpose interrupt request is issued to processor B. GIP*n* is cleared when MU resets. After GIP*n* is cleared, if MUB_GIER[GIE*n*] = 1, the general-purpose interrupt request is cleared on the MUB side. **NOTE** This field is not supported in every instance. The following table includes only supported registers. <table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>GSR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>GSR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>GSR</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>GSR</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>GSR</td></tr></table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>NOTE
This field behaves differently for register reads and writes.
When reading
0b - Not pending
1b - Pending
When writing
0b - No effect
1b - Clear the flag</td></tr>
<tr><td>20
GIP20</td><td>MUB General-Purpose Interrupt Request Pending
Indicates whether a general-purpose interrupt request is pending. There are 32 general-purpose interrupts (n = 0 to 31).
GIPn informs MUB that MUA_GCR[GIRn] changed from 0 to 1. If MUB_GIER[GIEn] = 1, a general-purpose interrupt request is issued to processor B.
GIPn is cleared when MU resets.
After GIPn is cleared, if MUB_GIER[GIEn] = 1, the general-purpose interrupt request is cleared on the MUB side.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>GSR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>GSR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>GSR</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>GSR</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>GSR</td></tr></table>NOTE
This field behaves differently for register reads and writes.
When reading
0b - Not pending
1b - Pending
When writing</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>0b - No effect
1b - Clear the flag</td></tr>
<tr><td>19
GIP19</td><td>MUB General-Purpose Interrupt Request Pending
Indicates whether a general-purpose interrupt request is pending. There are 32 general-purpose interrupts (n = 0 to 31).
GIPn informs MUB that MUA_GCR[GIRn] changed from 0 to 1. If MUB_GIER[GIEn] = 1, a general-purpose interrupt request is issued to processor B.
GIPn is cleared when MU resets.
After GIPn is cleared, if MUB_GIER[GIEn] = 1, the general-purpose interrupt request is cleared on the MUB side.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>GSR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>GSR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>GSR</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>GSR</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>GSR</td></tr></table>NOTE
This field behaves differently for register reads and writes.
When reading
0b - Not pending
1b - Pending
When writing
0b - No effect
1b - Clear the flag</td></tr>
<tr><td>18
GIP18</td><td>MUB General-Purpose Interrupt Request Pending
Indicates whether a general-purpose interrupt request is pending. There are 32 general-purpose interrupts (n = 0 to 31).</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>GIPn informs MUB that MUA_GCR[GIRn] changed from 0 to 1. If MUB_GIER[GIEn] = 1, a general-purpose interrupt request is issued to processor B.

GIPn is cleared when MU resets.

After GIPn is cleared, if MUB_GIER[GIEn] = 1, the general-purpose interrupt request is cleared on the MUB side.

NOTE
This field is not supported in every instance. The following table includes only supported registers.

<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GSR</td></tr>
<tr><td>MU_3.MUB</td><td>—</td><td>GSR</td></tr>
<tr><td>MU_4.MUB</td><td>—</td><td>GSR</td></tr>
</table>
NOTE
This field behaves differently for register reads and writes.

When reading
0b - Not pending
1b - Pending
When writing
0b - No effect
1b - Clear the flag</td></tr>
<tr><td>17
GIP17</td><td>MUB General-Purpose Interrupt Request Pending

Indicates whether a general-purpose interrupt request is pending. There are 32 general-purpose interrupts (n = 0 to 31).

GIPn informs MUB that MUA_GCR[GIRn] changed from 0 to 1. If MUB_GIER[GIEn] = 1, a general-purpose interrupt request is issued to processor B.

GIPn is cleared when MU resets.

After GIPn is cleared, if MUB_GIER[GIEn] = 1, the general-purpose interrupt request is cleared on the MUB side.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GSR</td></tr>
<tr><td>MU_3.MUB</td><td>—</td><td>GSR</td></tr>
<tr><td>MU_4.MUB</td><td>—</td><td>GSR</td></tr>
</table>
NOTE
This field behaves differently for register reads and writes.
When reading
0b - Not pending
1b - Pending
When writing
0b - No effect
1b - Clear the flag
</td></tr>
<tr><td>16
GIP16</td><td>
MUB General-Purpose Interrupt Request Pending
Indicates whether a general-purpose interrupt request is pending. There are 32 general-purpose interrupts (n = 0 to 31).
GIPn informs MUB that MUA_GCR[GIRn] changed from 0 to 1. If MUB_GIER[GIEn] = 1, a general-purpose interrupt request is issued to processor B.
GIPn is cleared when MU resets.
After GIPn is cleared, if MUB_GIER[GIEn] = 1, the general-purpose interrupt request is cleared on the MUB side.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GSR</td><td>—</td></tr>
</table>
</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_1.MUB</td><td>GSR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>GSR</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>GSR</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>GSR</td></tr></table> **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not pending 1b - Pending When writing 0b - No effect 1b - Clear the flag
15 GIP15	MUB General-Purpose Interrupt Request Pending Indicates whether a general-purpose interrupt request is pending. There are 32 general-purpose interrupts (*n* = 0 to 31). GIP*n* informs MUB that MUA_GCR[GIR*n*] changed from 0 to 1. If MUB_GIER[GIE*n*] = 1, a general-purpose interrupt request is issued to processor B. GIP*n* is cleared when MU resets. After GIP*n* is cleared, if MUB_GIER[GIE*n*] = 1, the general-purpose interrupt request is cleared on the MUB side. **NOTE** This field is not supported in every instance. The following table includes only supported registers. <table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>GSR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>GSR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>GSR</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>GSR</td></tr></table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_4.MUB</td><td>—</td><td>GSR</td></tr></table> **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not pending 1b - Pending When writing 0b - No effect 1b - Clear the flag
14 GIP14	MUB General-Purpose Interrupt Request Pending Indicates whether a general-purpose interrupt request is pending. There are 32 general-purpose interrupts (*n* = 0 to 31). GIP*n* informs MUB that MUA_GCR[GIR*n*] changed from 0 to 1. If MUB_GIER[GIE*n*] = 1, a general-purpose interrupt request is issued to processor B. GIP*n* is cleared when MU resets. After GIP*n* is cleared, if MUB_GIER[GIE*n*] = 1, the general-purpose interrupt request is cleared on the MUB side. **NOTE** This field is not supported in every instance. The following table includes only supported registers. <table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>GSR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>GSR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>GSR</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>GSR</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>GSR</td></tr></table> **NOTE** This field behaves differently for register reads and writes.

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>When reading
0b - Not pending
1b - Pending
When writing
0b - No effect
1b - Clear the flag</td></tr>
<tr><td>13
GIP13</td><td>MUB General-Purpose Interrupt Request Pending
Indicates whether a general-purpose interrupt request is pending. There are 32 general-purpose interrupts (n = 0 to 31).
GIPn informs MUB that MUA_GCR[GIRn] changed from 0 to 1. If MUB_GIER[GIEn] = 1, a general-purpose interrupt request is issued to processor B.
GIPn is cleared when MU resets.
After GIPn is cleared, if MUB_GIER[GIEn] = 1, the general-purpose interrupt request is cleared on the MUB side.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>GSR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>GSR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>GSR</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>GSR</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>GSR</td></tr></table>NOTE
This field behaves differently for register reads and writes.
When reading
0b - Not pending
1b - Pending
When writing
0b - No effect
1b - Clear the flag</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>12
GIP12</td><td>MUB General-Purpose Interrupt Request Pending
Indicates whether a general-purpose interrupt request is pending. There are 32 general-purpose interrupts (n = 0 to 31).
GIPn informs MUB that MUA_GCR[GIRn] changed from 0 to 1. If MUB_GIER[GIEn] = 1, a general-purpose interrupt request is issued to processor B.
GIPn is cleared when MU resets.
After GIPn is cleared, if MUB_GIER[GIEn] = 1, the general-purpose interrupt request is cleared on the MUB side.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>GSR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>GSR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>GSR</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>GSR</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>GSR</td></tr></table>NOTE
This field behaves differently for register reads and writes.
When reading
0b - Not pending
1b - Pending
When writing
0b - No effect
1b - Clear the flag</td></tr>
<tr><td>11
GIP11</td><td>MUB General-Purpose Interrupt Request Pending
Indicates whether a general-purpose interrupt request is pending. There are 32 general-purpose interrupts (n = 0 to 31).
GIPn informs MUB that MUA_GCR[GIRn] changed from 0 to 1. If MUB_GIER[GIEn] = 1, a general-purpose interrupt request is issued to processor B.
GIPn is cleared when MU resets.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>After GIPn is cleared, if MUB_GIER[GIEn] = 1, the general-purpose interrupt request is cleared on the MUB side.

NOTE
This field is not supported in every instance. The following table includes only supported registers.

<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GSR</td></tr>
<tr><td>MU_3.MUB</td><td>—</td><td>GSR</td></tr>
<tr><td>MU_4.MUB</td><td>—</td><td>GSR</td></tr>
</table>

NOTE
This field behaves differently for register reads and writes.

When reading
0b - Not pending
1b - Pending
When writing
0b - No effect
1b - Clear the flag</td></tr>
<tr><td>10
GIP10</td><td>MUB General-Purpose Interrupt Request Pending
Indicates whether a general-purpose interrupt request is pending. There are 32 general-purpose interrupts (n = 0 to 31).
GIPn informs MUB that MUA_GCR[GIRn] changed from 0 to 1. If MUB_GIER[GIEn] = 1, a general-purpose interrupt request is issued to processor B.
GIPn is cleared when MU resets.
After GIPn is cleared, if MUB_GIER[GIEn] = 1, the general-purpose interrupt request is cleared on the MUB side.

NOTE
This field is not supported in every instance. The following table includes only supported registers.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GSR</td></tr>
<tr><td>MU_3.MUB</td><td>—</td><td>GSR</td></tr>
<tr><td>MU_4.MUB</td><td>—</td><td>GSR</td></tr>
</table>
NOTE
This field behaves differently for register reads and writes.

When reading
0b - Not pending
1b - Pending
When writing
0b - No effect
1b - Clear the flag</td></tr>
<tr><td>9
GIP9</td><td>MUB General-Purpose Interrupt Request Pending

Indicates whether a general-purpose interrupt request is pending. There are 32 general-purpose interrupts (n = 0 to 31).

GIPn informs MUB that MUA_GCR[GIRn] changed from 0 to 1. If MUB_GIER[GIEn] = 1, a general-purpose interrupt request is issued to processor B.

GIPn is cleared when MU resets.

After GIPn is cleared, if MUB_GIER[GIEn] = 1, the general-purpose interrupt request is cleared on the MUB side.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GSR</td></tr>
</table></td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_3.MUB</td><td>—</td><td>GSR</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>GSR</td></tr></table> **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not pending 1b - Pending When writing 0b - No effect 1b - Clear the flag
8 GIP8	MUB General-Purpose Interrupt Request Pending Indicates whether a general-purpose interrupt request is pending. There are 32 general-purpose interrupts (*n* = 0 to 31). GIP*n* informs MUB that MUA_GCR[GIR*n*] changed from 0 to 1. If MUB_GIER[GIE*n*] = 1, a general-purpose interrupt request is issued to processor B. GIP*n* is cleared when MU resets. After GIP*n* is cleared, if MUB_GIER[GIE*n*] = 1, the general-purpose interrupt request is cleared on the MUB side. **NOTE** This field is not supported in every instance. The following table includes only supported registers. <table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>GSR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>GSR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>GSR</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>GSR</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>GSR</td></tr></table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>NOTE
This field behaves differently for register reads and writes.
When reading
0b - Not pending
1b - Pending
When writing
0b - No effect
1b - Clear the flag</td></tr>
<tr><td>7
GIP7</td><td>MUB General-Purpose Interrupt Request Pending
Indicates whether a general-purpose interrupt request is pending. There are 32 general-purpose interrupts (n = 0 to 31).
GIPn informs MUB that MUA_GCR[GIRn] changed from 0 to 1. If MUB_GIER[GIEn] = 1, a general-purpose interrupt request is issued to processor B.
GIPn is cleared when MU resets.
After GIPn is cleared, if MUB_GIER[GIEn] = 1, the general-purpose interrupt request is cleared on the MUB side.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GSR</td></tr>
<tr><td>MU_3.MUB</td><td>—</td><td>GSR</td></tr>
<tr><td>MU_4.MUB</td><td>—</td><td>GSR</td></tr>
</table>
NOTE
This field behaves differently for register reads and writes.
When reading
0b - Not pending
1b - Pending
When writing</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>0b - No effect
1b - Clear the flag</td></tr>
<tr><td>6
GIP6</td><td>MUB General-Purpose Interrupt Request Pending
Indicates whether a general-purpose interrupt request is pending. There are 32 general-purpose interrupts (n = 0 to 31).
GIPn informs MUB that MUA_GCR[GIRn] changed from 0 to 1. If MUB_GIER[GIEn] = 1, a general-purpose interrupt request is issued to processor B.
GIPn is cleared when MU resets.
After GIPn is cleared, if MUB_GIER[GIEn] = 1, the general-purpose interrupt request is cleared on the MUB side.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>GSR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>GSR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>GSR</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>GSR</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>GSR</td></tr></table>NOTE
This field behaves differently for register reads and writes.
When reading
0b - Not pending
1b - Pending
When writing
0b - No effect
1b - Clear the flag</td></tr>
<tr><td>5
GIP5</td><td>MUB General-Purpose Interrupt Request Pending
Indicates whether a general-purpose interrupt request is pending. There are 32 general-purpose interrupts (n = 0 to 31).</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>GIPn informs MUB that MUA_GCR[GIRn] changed from 0 to 1. If MUB_GIER[GIEn] = 1, a general-purpose interrupt request is issued to processor B.
GIPn is cleared when MU resets.
After GIPn is cleared, if MUB_GIER[GIEn] = 1, the general-purpose interrupt request is cleared on the MUB side.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>GSR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>GSR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>GSR</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>GSR</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>GSR</td></tr></table>NOTE
This field behaves differently for register reads and writes.
When reading
0b - Not pending
1b - Pending
When writing
0b - No effect
1b - Clear the flag</td></tr>
<tr><td>4
GIP4</td><td>MUB General-Purpose Interrupt Request Pending
Indicates whether a general-purpose interrupt request is pending. There are 32 general-purpose interrupts (n = 0 to 31).
GIPn informs MUB that MUA_GCR[GIRn] changed from 0 to 1. If MUB_GIER[GIEn] = 1, a general-purpose interrupt request is issued to processor B.
GIPn is cleared when MU resets.
After GIPn is cleared, if MUB_GIER[GIEn] = 1, the general-purpose interrupt request is cleared on the MUB side.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field is not supported in every instance. The following table includes only supported registers.

Instance	Field supported in	Field not supported in
MU_0.MUB	GSR	—
MU_1.MUB	GSR	—
MU_2.MUB	—	GSR
MU_3.MUB	—	GSR
MU_4.MUB	—	GSR

NOTE
This field behaves differently for register reads and writes.

When reading

0b - Not pending

1b - Pending

When writing

0b - No effect

1b - Clear the flag

3

GIP3

MUB General-Purpose Interrupt Request Pending

Indicates whether a general-purpose interrupt request is pending. There are 32 general-purpose interrupts (*n* = 0 to 31).

GIP*n* informs MUB that MUA_GCR[GIR*n*] changed from 0 to 1. If MUB_GIER[GIE*n*] = 1, a general-purpose interrupt request is issued to processor B.

GIP*n* is cleared when MU resets.

After GIP*n* is cleared, if MUB_GIER[GIE*n*] = 1, the general-purpose interrupt request is cleared on the MUB side.

NOTE
This field is not supported in every instance. The following table includes only supported registers.

Instance	Field supported in	Field not supported in
MU_0.MUB	GSR	—

Table continues on the next page...

Table continued from the previous page...

Field	Function
	<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_1.MUB</td><td>GSR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>GSR</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>GSR</td></tr><tr><td>MU_4.MUB</td><td>—</td><td>GSR</td></tr></table> **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not pending 1b - Pending When writing 0b - No effect 1b - Clear the flag
2 GIP2	MUB General-Purpose Interrupt Request Pending Indicates whether a general-purpose interrupt request is pending. There are 32 general-purpose interrupts (*n* = 0 to 31). GIP*n* informs MUB that MUA_GCR[GIR*n*] changed from 0 to 1. If MUB_GIER[GIE*n*] = 1, a general-purpose interrupt request is issued to processor B. GIP*n* is cleared when MU resets. After GIP*n* is cleared, if MUB_GIER[GIE*n*] = 1, the general-purpose interrupt request is cleared on the MUB side. **NOTE** This field is not supported in every instance. The following table includes only supported registers. <table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MU_0.MUB</td><td>GSR</td><td>—</td></tr><tr><td>MU_1.MUB</td><td>GSR</td><td>—</td></tr><tr><td>MU_2.MUB</td><td>—</td><td>GSR</td></tr><tr><td>MU_3.MUB</td><td>—</td><td>GSR</td></tr></table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_4.MUB</td><td>—</td><td>GSR</td></tr>
</table>
NOTE
This field behaves differently for register reads and writes.
When reading
0b - Not pending
1b - Pending
When writing
0b - No effect
1b - Clear the flag
</td></tr>
<tr><td>1
GIP1</td><td>
MUB General-Purpose Interrupt Request Pending
Indicates whether a general-purpose interrupt request is pending. There are 32 general-purpose interrupts (n = 0 to 31).
GIPn informs MUB that MUA_GCR[GIRn] changed from 0 to 1. If MUB_GIER[GIEn] = 1, a general-purpose interrupt request is issued to processor B.
GIPn is cleared when MU resets.
After GIPn is cleared, if MUB_GIER[GIEn] = 1, the general-purpose interrupt request is cleared on the MUB side.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>MU_0.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_1.MUB</td><td>GSR</td><td>—</td></tr>
<tr><td>MU_2.MUB</td><td>—</td><td>GSR</td></tr>
<tr><td>MU_3.MUB</td><td>—</td><td>GSR</td></tr>
<tr><td>MU_4.MUB</td><td>—</td><td>GSR</td></tr>
</table>
NOTE
This field behaves differently for register reads and writes.
</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When reading 0b - Not pending 1b - Pending When writing 0b - No effect 1b - Clear the flag
0 GIP0	MUB General-Purpose Interrupt Request Pending Indicates whether a general-purpose interrupt request is pending. There are 32 general-purpose interrupts (*n* = 0 to 31). GIP*n* informs MUB that MUA_GCR[GIR*n*] changed from 0 to 1. If MUB_GIER[GIE*n*] = 1, a general-purpose interrupt request is issued to processor B. GIP*n* is cleared when MU resets. After GIP*n* is cleared, if MUB_GIER[GIE*n*] = 1, the general-purpose interrupt request is cleared on the MUB side. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not pending 1b - Pending When writing 0b - No effect 1b - Clear the flag

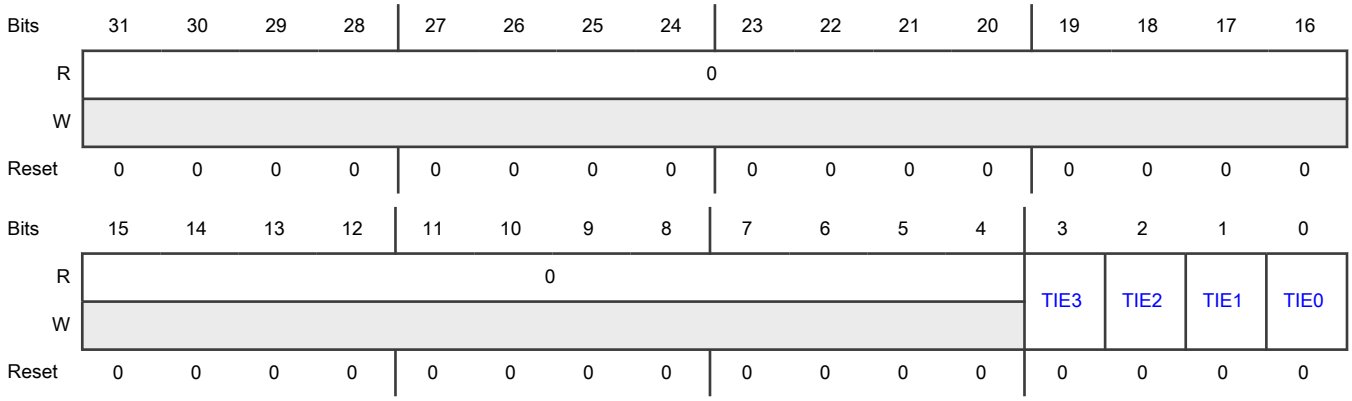
40.7.2.13 Transmit Control (TCR)

Offset

Register	Offset
TCR	120h

Function
Contains the MUB transmit interrupt enable fields.

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 TIEn	MUB Transmit Interrupt Enable Enables MUB transmit interrupt *n*, where *n* = 0 to 3. If this field is 1, an MUB transmit interrupt *n* request is issued when MUB_TSR[TE*n*] is set. If this field is 0, MU ignores the value of MUB_TSR[TE*n*], and no MUB transmit interrupt *n* request is issued. 0b - Disable 1b - Enable

40.7.2.14 Transmit Status (TSR)

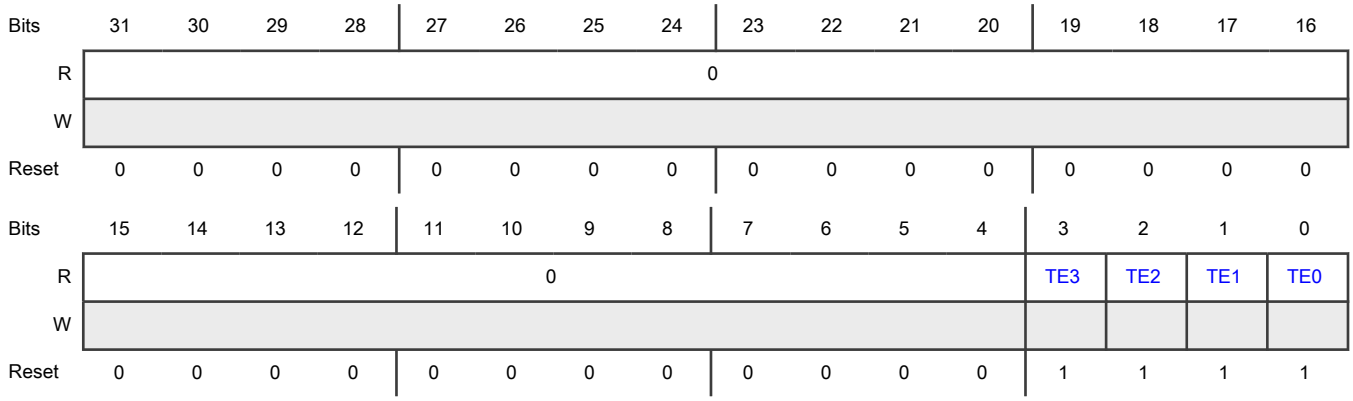
Offset

Register	Offset
TSR	124h

Function

Indicates whether the MUB transmit registers are empty.

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 TEn	MUB Transmit Empty Indicates whether MUB Transmit (TR*n*) register is empty, where *n* = 0 to 3. This field becomes 1 after the MUA_RR*n* register is read on the MUA side. When TE*n* = 1, it indicates to the MUB side that the MUB_TR*n* register is ready to be written on the MUB side. If MUB_TCR[TIE*n*] = 1, a transmit *n* interrupt is issued on the MUB side. This field becomes 0 after the MUB_TR*n* register is written to on the MUB side. After this field becomes 0, if MUB_TCR[TIE*n*] = 1, the transmit *n* interrupt request is cleared on the MUB side. This field becomes 1 when MU resets. 0b - Not empty 1b - Empty

40.7.2.15 Receive Control (RCR)

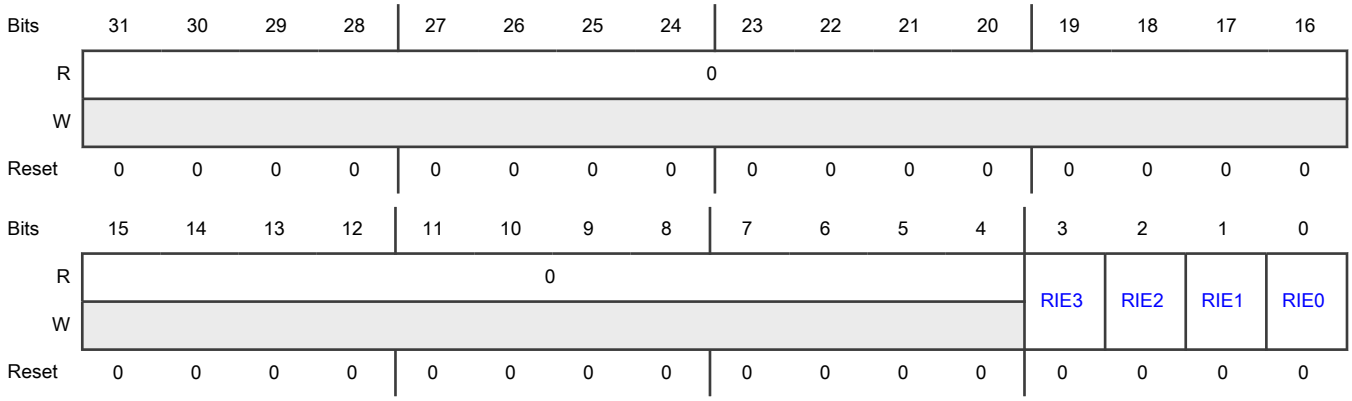
Offset

Register	Offset
RCR	128h

Function

Contains the MUB receive interrupt enables.

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 RIEn	MUB Receive Interrupt Enable Enables MUB receive interrupt *n*, where *n* = 0 to 3. If this field is 1, an MUB receive interrupt *n* request is issued when MUB_RSR[RF*n*] is set. If this field is 0, MU ignores the value of MUB_RSR[RF*n*], and no MUB receive interrupt request is issued. 0b - Disable 1b - Enable

40.7.2.16 Receive Status (RSR)

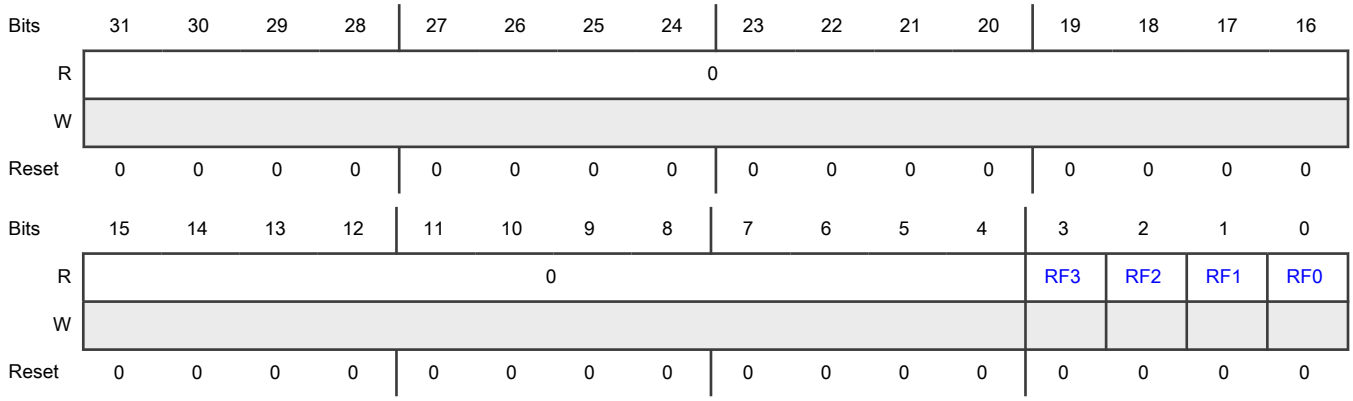
Offset

Register	Offset
RSR	12Ch

Function

Indicates whether the MUB receive registers are full.

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 RFn	MUB Receive Register Full Indicates whether MUB Receive register (RR*n*) is full, where *n* = 0 to 3. RF*n* becomes 1 when the MUA_TR*n* register is written to on the MUA side. When RF*n* is 1, it indicates to the MUB side that new data in the MUB_RR*n* register is ready for MUB to read it. If MUB_RCR[RIE*n*] = 1, a receive *n* interrupt is issued on the MUB side. RF*n* becomes 0 when the MUB_RR*n* register is read, or when MU is reset. After RF*n* becomes 0, if MUB_RCR[RIE*n*] = 1, the receive *n* interrupt request is cleared on the MUB side. 0b - Not full 1b - Full

40.7.2.17 Transmit (TR0 - TR3)

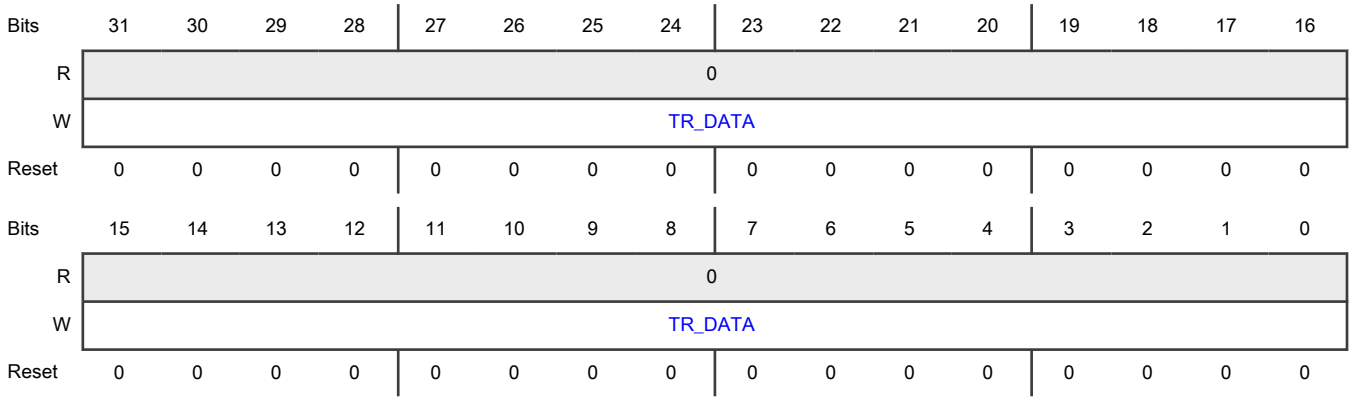
Offset

Register	Offset
TR0	200h
TR1	204h
TR2	208h
TR3	20Ch

Function

Contains MUB transmit data.

Diagram



Fields

Field	Function
31-0 TR_DATA	MUB Transmit Data Contains MUB transmit data. MUA_RR*n* reflects the data written to this register. The TR*n* and RR*n* registers are not double-buffered. Writing to MUB_TR*n* overrides the data readable in the MUA_RR*n* register. A write to the Transmit register clears MUB_TSR[TE*n*] on the transmitter side, and sets MUA_RSR[RF*n*] on the receiver side. You can write to this register only when MUB_TSR[TE*n*] = 1. Reading this register returns all zeroes.

40.7.2.18 Receive (RR0 - RR3)

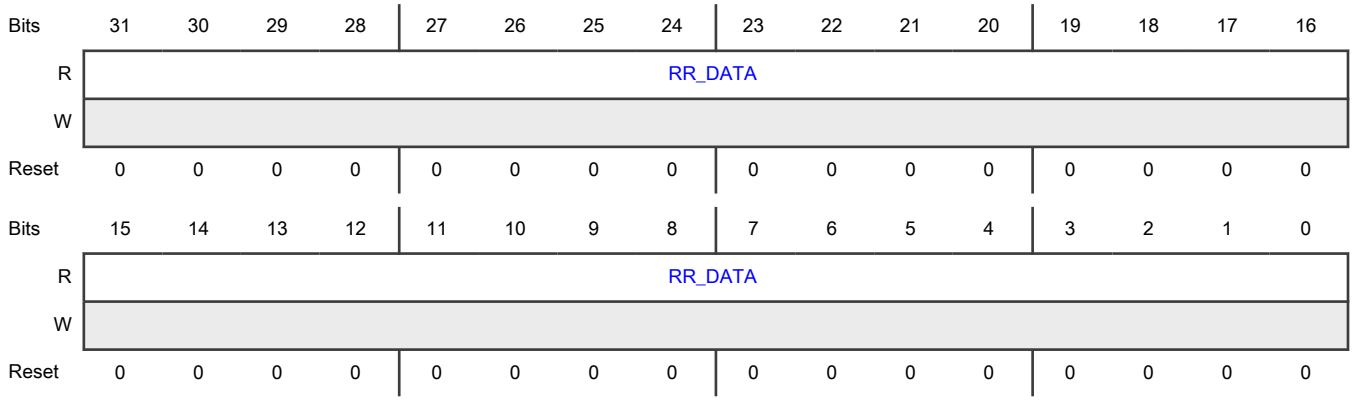
Offset

Register	Offset
RR0	280h
RR1	284h
RR2	288h
RR3	28Ch

Function

Contains MUB receive data.

Diagram



Fields

Field	Function
31-0 RR_DATA	MUB Receive Data Reflects the data written to MUA TR*n*. Reading this register clears MUB_RSR[RF*n*] on the receiver side, and sets MUA_TSR[TE*n*] on the transmitter side. You can read this register only when MUB_RSR[RF*n*] = 1. Reading it before MUB_RSR[RF*n*] becomes 1 may result in reading incorrect data. Poll MUB_RSR[RF*n*] to confirm it is set before reading RR*n*. Writing to this register generates an error response to MUB.

40.8 Glossary

EP	Event Pending
GIR	General-purpose Interrupt Request
GIP	General-purpose Interrupt Pending
MUR	Messaging Unit Reset
RF	Receiver Full
RFP	Receive Full Pending
TE	Transmitter Empty
TEP	Transmit Empty Pending
MUA	Messaging Unit A
MUB	Messaging Unit B

Chapter 41
Power Management

41.1 Introduction

The power management system generates, monitors, and controls power supplies and related resets. This chapter describes the system's interaction with other peripherals.

The power management system includes the modules listed below.

Table 246. Power management system modules and their functions

Part	Function
MC_ME	Initiates entry into Low-Power mode
MC_PCU	Controls entry into and exit from Low-Power mode
PMC	Regulates and monitors power supplies of the Run and the Standby domains
MC_RGM	Ensures a clean state and correct Run domain functionality by controlling the reset sequence
FIRC, WKPU, and other chip peripherals	Controls Standby mode wake-up and operations in Run and Standby modes

41.1.1 Power management system for S32K344, S32K324, S32K314, S32K341, S32K342, and S32K322

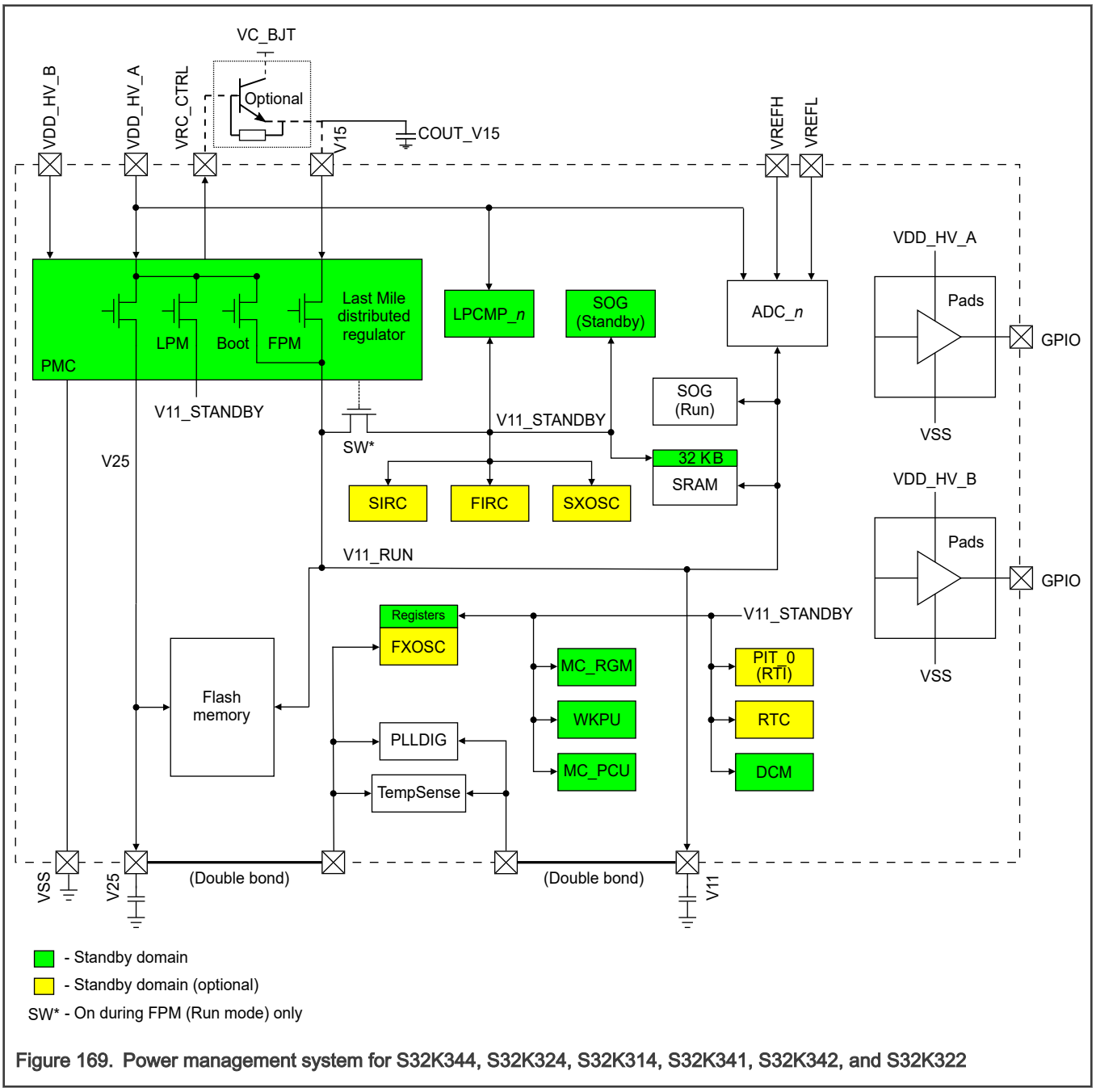


Figure 169. Power management system for S32K344, S32K324, S32K314, S32K341, S32K342, and S32K322

41.1.2 Power management system for S32K312 and S32K311, and S32K310

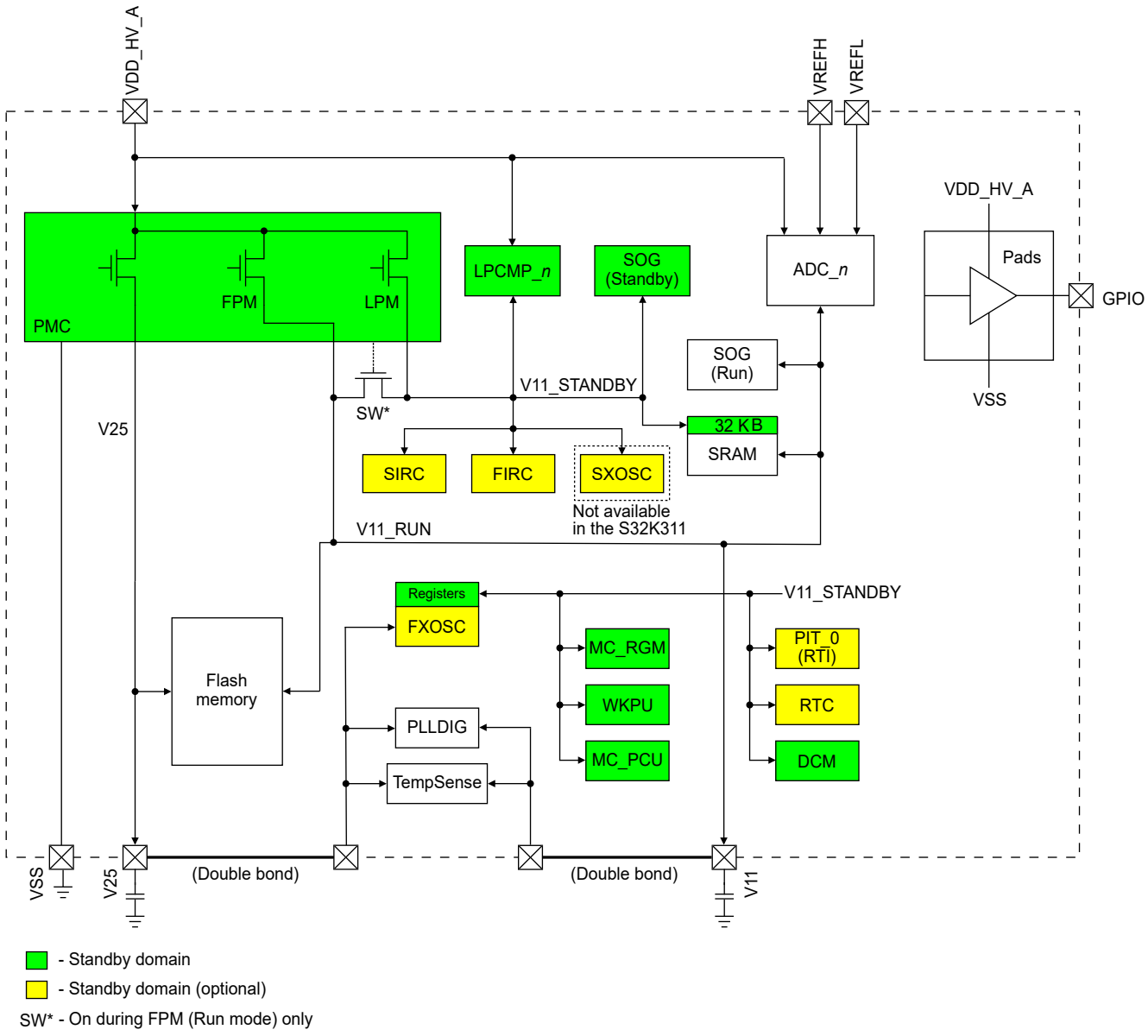
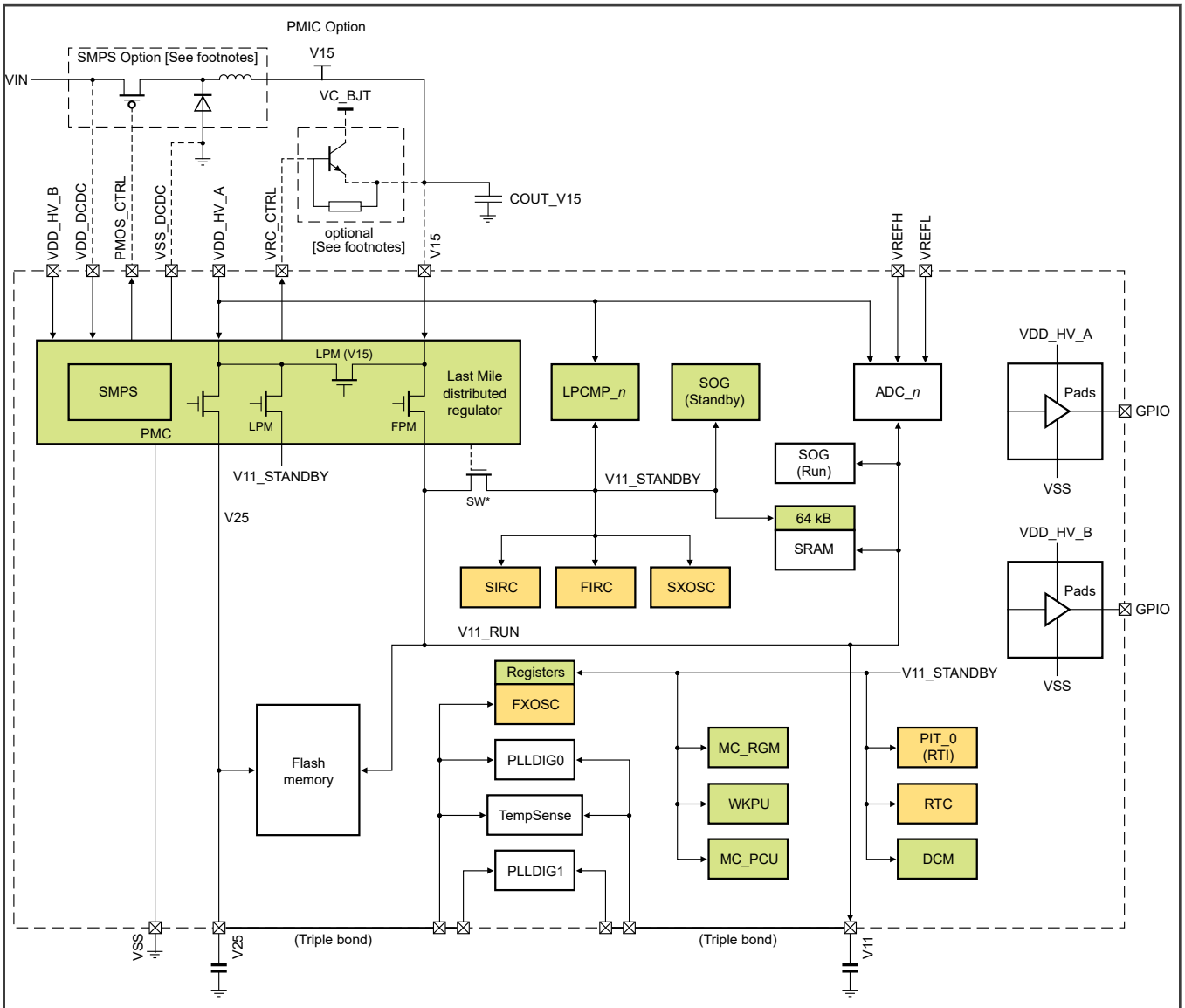


Figure 170. Power management system for S32K312 and S32K311, and S32K310

41.1.3 Power management system for S32K358/S32K356/S32K348/S32K338/S32K328



SW* - On during FPM (Run mode) only

The V15 input voltage can be supplied directly by an SBC or Voltage regulator. The S32K3x8 devices provide two options to derive the V15 regulator from the 5V or 3.3V supply.

Option 1: A DC/DC converter (SMPS) regulator can be implemented by connecting the external components (PMOS, inductor, diode) shown in the figure above, at pins VDD_DCDC, PMOS_CTRL, and VSS_DCDC. In this case, VRC_CTRL must not be connected. VDD_DCDC must be connected to the VDD_HV_A or VDD_HV_B, supply voltage. In this option COUT_V15 is specified in SMPS regulator electrical specifications.

Option 2: A linear regulator can be implemented by connecting a bipolar NPN transistor from the VDD_HV_A or VDD_HV_B, supply voltage shown as VC_BJT in the figure above. The VRC_CTRL pin must be connected to the base of the transistor. In this case, VDD_DCDC, PMOS_CTRL must not be connected. VSS_DCDC must be connected to MCU GND. In this option, COUT_V15 is specified in Recommended decoupling capacitors.

Note: In the EP package, you must connect the exposed pad to ground (VSS).

Figure 171. Power management system for S32K358/S32K356/S32K348/S32K338/S32K328

41.1.4 Power management system for S32K388/S32K389

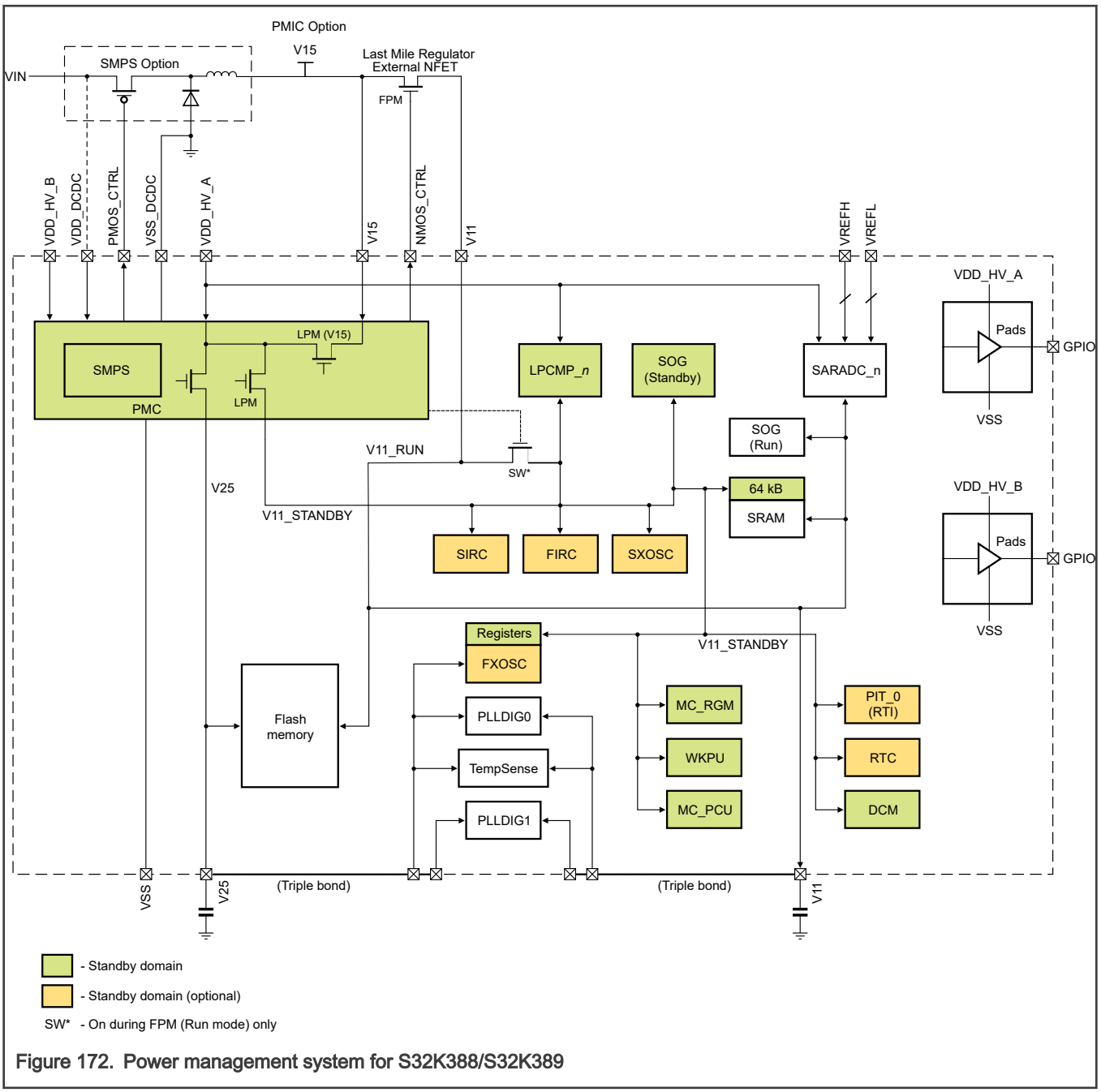


Figure 172. Power management system for S32K388/S32K389

41.1.5 Features

- For S32K310, S32K311, S32K312, PMC uses VDD_HV_A to generate 1.1 V (nominal) supply for the core logic (V11_RUN).
- For other chips, PMC uses a 1.5 V supply to generate 1.1 V (nominal) supply for the core logic (V11_RUN).
- S32K358/S32K356/S32K348/S32K338/S32K328, S32K342/S32K322/S32K341, S32K344/S32K324/S32K314 includes a linear regulator to use an optional external ballast (BJT) and VRC_CTRL output for 1.5 V generation.
- Supports a low-power regulator (LPM) supplying core logic during Standby mode (V11_STANDBY)
- Includes 1.1 V FPM regulator options for driving the logic in RUN mode (V11_RUN) from a 1.5V supply:

  - Boot LDO for S32K344/S32K324/S32K314, S32K342/S32K322/S32K341
  - Internal Last-mile (FPM) for S32K344/S32K324/S32K314, S32K342/S32K322/S32K341/S32K358/S32K356/S32K348/S32K338/S32K328
  - External Last-Mile (FPM) for S32K388/S32K389
- Supports power switches to isolate voltage islands and configure the chip in Standby mode
- S32K358/S32K356/S32K348/S32K338/S32K328/S32K388/S32K389 supports a DC-DC buck converter stage, with a dedicated pin to control an external Power MOSFET, in the range from 1.4V to 1.5V.
- S32K388/S32K389 controls an external NMOS transistor, and uses V15 generated from SMPS as input to generate V11 for a load current up to 1.5 A (disabled in Standby mode).
- Includes a linear regulator that generates a 2.5 V supply (V25) from VDD_HV_A
- Supports voltage monitors ensuring transitions to a safe state (POR) when a supply is out of a valid range
- Controls power-mode transitions through an interaction with the digital interface
- Offers a padkeeping feature that retains PAD state during standby mode till software boots up
- Provides separate ADC reference supplies (VREFH and VREFL)

41.1.6 Operational power modes

This chip has two power modes:

- Run mode (FPM): Main operation mode with full chip performance and a higher current consumption as compared to Standby mode.
- Standby mode (LPM): Low-performance mode of the chip in which the Run domain is turned off. The cores and most peripherals are off in this mode.

The boot regulator manages the chip during the booting process except S32K3x8, S32K312, S32K311 and S32K310.

The last-mile regulator is the full-performance regulator, which you enable for running applications. The upcoming sections discuss the sequence to enable or disable the regulator.

The LPM regulator manages the chip in Standby mode.

41.1.7 External 1.5 V source

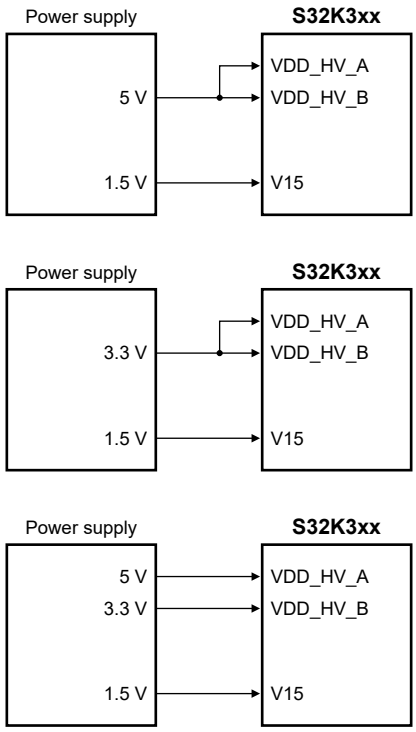


Figure 173. Core supply from external 1.5 V source (except for S32K388, S32K389, S32K312, S32K311, S32K310)

41.1.8 External 1.5 V source (applicable for S32K388/S32K389)

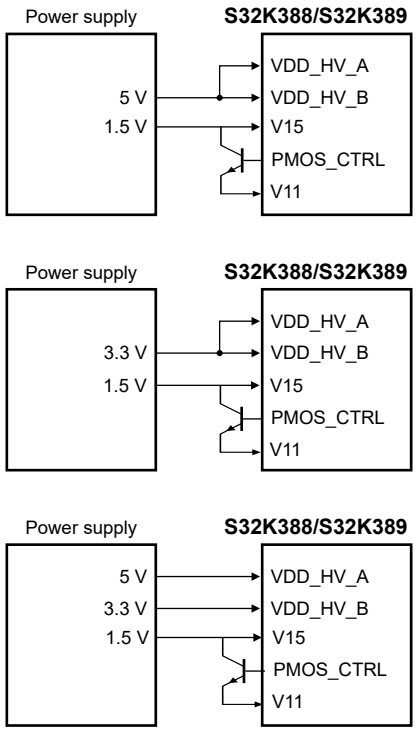


Figure 174. Core supply from external 1.5 V source, with external last-mile regulator (applicable for S32K388/S32K389)

41.1.9 Using a BJT for 1.5 V generation (not applicable for S32K388, S32K389, S32K312, S32K311, and S32K310)

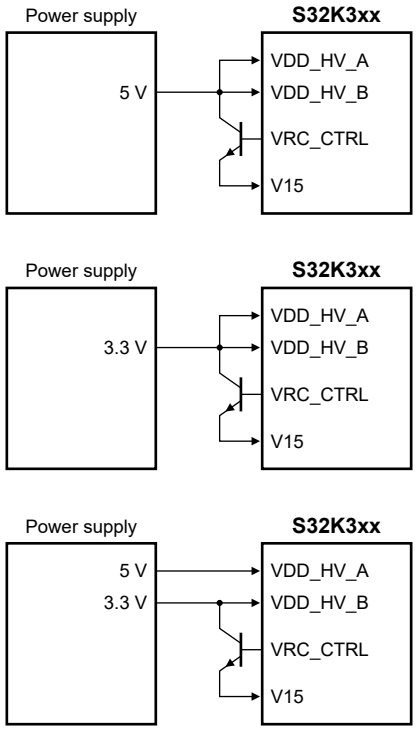


Figure 175. Using a BJT for 1.5 V generation

41.2 Power-up sequence

Figure 176 shows the reset sequence for a power-up or Standby mode event. This sequence starts from the POR phase. For the Run domain logic, Standby mode exit operation is same as power-up.

See the "Reset Overview" chapter for more information about the chip reset sequence.

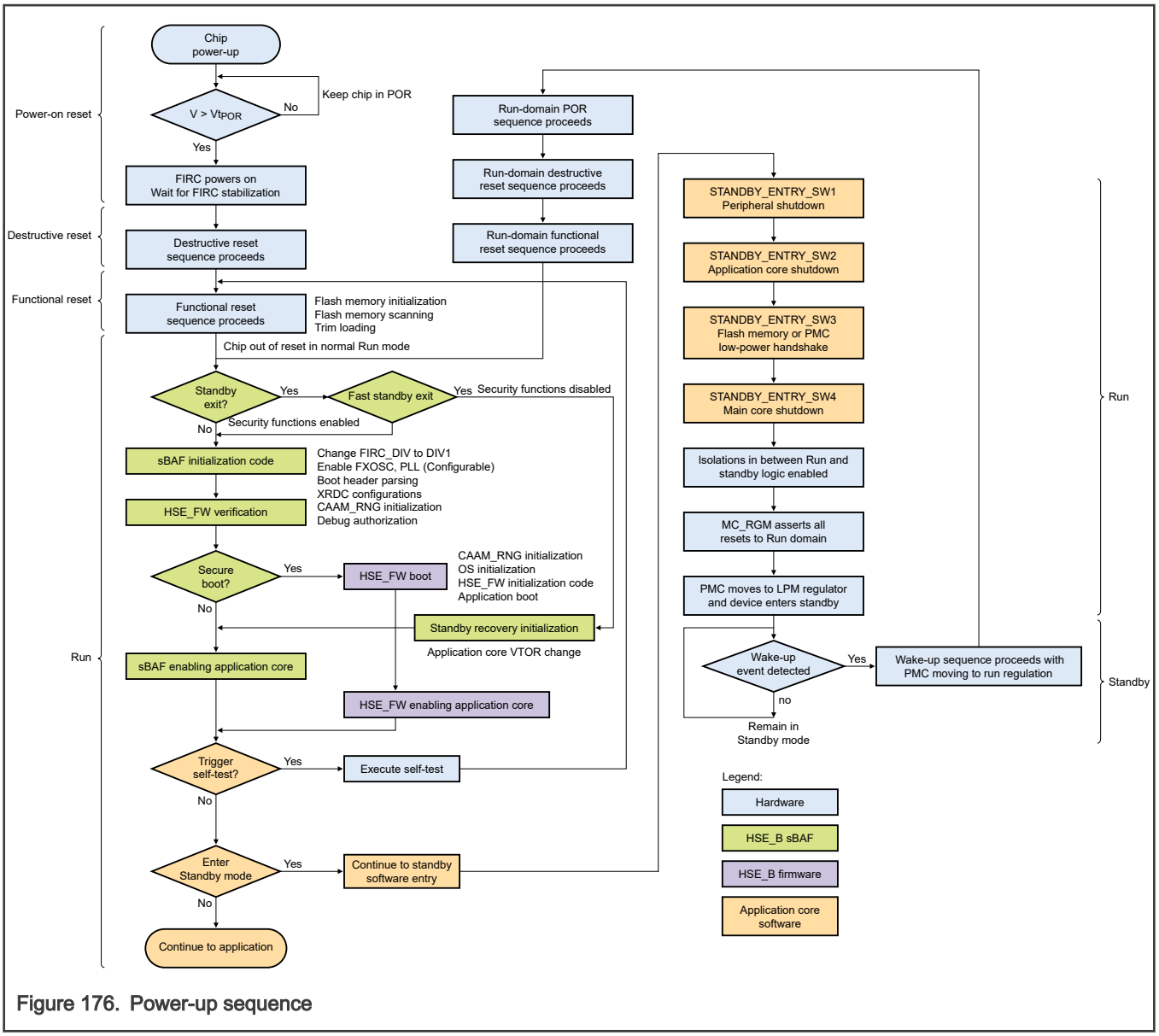


Figure 176. Power-up sequence

41.3 PMC last-mile regulator auto-enable feature (applicable for S32K344, S32K324, S32K314, S32K341, S32K342, and S32K322)

PMC includes an automatic last-mile auto-enable feature. After starting on boot regulator, this feature allows automatic switch over to last-mile regulator if 1.5 V is present during:

- Chip startup
- Standby mode recovery

You can control the PMC last-mile regulator by appropriately configuring:

- PMC.CONFIG[LMAUTOEN] field
- PMC.CONFIG[LM_EN] field

  For more information, see the descriptions of these fields in PMC.CONFIG register.

NOTE

You must write 1 to PMC.CONFIG[LM_EN] field before transtioning to faster clock frequencies irrespective of the setting of PMC.CONFIG[LMAUTOEN] field. This is because of the reduced clock speed when using the boot regulator.

41.3.1 Last-mile regulator with 1.5 V from an external source (applicable for S32K344, S32K324, S32K314, S32K341, S32K342, and S32K322)

Table 247. Last-mile regulator with 1.5 V from an external source

Operating condition	Last-mile regulator operation
After POR	Boots on boot regulator and then automatically switches to the last-mile regulator
After destructive reset	Remains on last-mile regulator **NOTE** If the destructive reset source is low voltage reset on 1.1 V then switchback happens from last-mile to boot regulator for boot.
After functional reset	Remains on last-mile regulator
After PMC.LVSC[LVD15S] field becomes 1	Switches, automatically, to the boot regulator to configure clocks for slow speed

41.3.2 Last-mile regulator using a BJT (applicable for S32K344, S32K324, S32K314, S32K341, S32K342, and S32K322)

Table 248. Last-mile regulator using a BJT

Operating condition	Last-mile regulator operation
After POR	Boots on boot regulator; switches to the last-mile regulator post reset when the Cortex-M7 core configures the software on FIRC
After destructive reset	Switches to boot regulator to check reset propagation delay
After functional reset	Remains on last-mile regulator
After PMC.LVSC[LVD15S] field becomes 1	Switches, automatically, to boot regulator to configure clocks for slow speed

41.4 PMC last-mile regulator auto-enable feature (applicable for S32K358, S32K356, S32K348, S32K338 and S32K328)

PMC includes an automatic last-mile auto-enable feature.

You can control the PMC last-mile regulator by appropriately configuring:

- PMC.CONFIG[LM_EN] field

  For more information, see the descriptions of these fields in PMC.CONFIG register.

41.4.1 Last-mile regulator with 1.5 V from an external source (applicable for S32K358, S32K356, S32K348, S32K338 and S32K328)

Table 249. Last-mile regulator with 1.5 V from an external source

Operating condition	Last-mile regulator operation
After POR	Waits for 1.5 V to start last-mile regulator
After destructive reset	Remains on last-mile regulator
After functional reset	Remains on last-mile regulator

41.4.2 Last-mile regulator using a BJT (applicable for S32K358, S32K356, S32K348, S32K338 and S32K328)

Table 250. Last-mile regulator using a BJT

Operating condition	Last-mile regulator operation
After POR	Wait for 1.5 V, if it is at sustained level, PMC starts the last-mile regulator
After destructive reset	Remains on last-mile regulator
After functional reset	Remains on last-mile regulator

41.5 Standby mode entry sequence

The Standby mode entry sequence includes three phases of operation:

- Standby mode entry configuration phase or software Standby mode entry sequence
- Standby mode entry handshake phase or hardware Standby mode entry sequence
- Standby mode entry or PMC Standby mode entry

NOTE

The Standby mode entry sequence described in this section is the only supported sequence. Contact NXP support if you require an alternate Standby mode entry sequence.

NOTE

In S32K388/S32K389, after triggering Standby Entry from Application, if Reset is asserted before Standby entry then reset can be issued to system, which will be visible at RESET_B pad.

NOTE

When "PMIC Handshake with MCU" is not used, to be able to exit from STANDBY mode, PMIC_PGOOD_HNDSHK_BYP must be set to 1.

41.5.1 Software Standby mode entry sequence

The Standby mode entry sequence includes chip configuration as described in the following sections.

Before entering the software Standby mode sequence, the system clock source must be changed to FIRC at 48 MHz because PLLDIG is not available in Standby mode. In Standby mode, all clock sources can be optionally disabled (including FIRC, which results in a no-clock, low-power consumption mode). You could also use FXOSC, if enabled, when the 2.5 V supply is available by appropriate configuration of PMC.CONFIG[LPM25EN].

The software Standby mode entry sequence consists of four steps:

1. SW1: Module shutdown process

2. SW2: Application core shutdown process
3. SW3: Flash memory low-power handshake and PMC last-mile regulator control[9]
4. SW4: Main core shutdown process

These processes are described in detail in the sections that follow. See Figure 176 that shows the relationship between these four steps in a flow diagram.

41.5.1.1 SW1: Module shutdown process

I/O and module configuration for Standby mode discusses the procedure to configure I/O and the chip modules for Standby mode.

The entry sequence for this mode includes module clock disabling steps (see the "Clocking" chapter for module clock turn on and turn off processes). You must use MC_ME.PRTN*n*_COFB*m*_CLKEN[REQ*p*] fields to enable or disable module clocks.

41.5.1.1.1 Disabling modules

Disable modules by configuring the appropriate fields in their registers for Standby mode operation. See specific module chapters for more information.

The Standby mode entry sequence includes the module clock disabling step, with which you can disable the modules that you do not need for Standby mode operation.

The sequence of disabling modules is shown in I/O and module configuration for Standby mode.

NOTE

While enabling or disabling the modules, you must verify that the module is disabled when you read MC_ME.PRTNn_COFBm_STAT register and the module disable field, if applicable. In case of a discrepancy, you must perform proper diagnostic steps.

You must clear the I/O controls for the pads that you do not require in Standby mode (OBE, IBE, and so on). This avoids any unwanted pad keeping settings. See Pad keeping for more information on the chip pad keeping process.

For any standby wake source, if an interrupt occurs it must be disabled before entering Standby mode and only the wake up event of that source must be enabled. This is to avoid any SW conflict in the interrupt handling for multi application core cases.

The MC_ME.PRTN*n*_COFB*m*_STAT register indicates the status of peripheral clock enable or disable. It may take up to three clock cycles for MC_ME.PRTN*n*_COFB*m*_STAT register to update after MC_ME.PRTN*n*_COFB*m*_CLKEN register is updated.

Once modules are disabled by following above steps:

- Switch to FIRC as the system clock.
- Disable FXOSC and wait for clock status.
- Configure Standby Entry.

[9] Not applicable for S32K388/S32K389.

41.5.1.1.2 I/O and module configuration for Standby mode

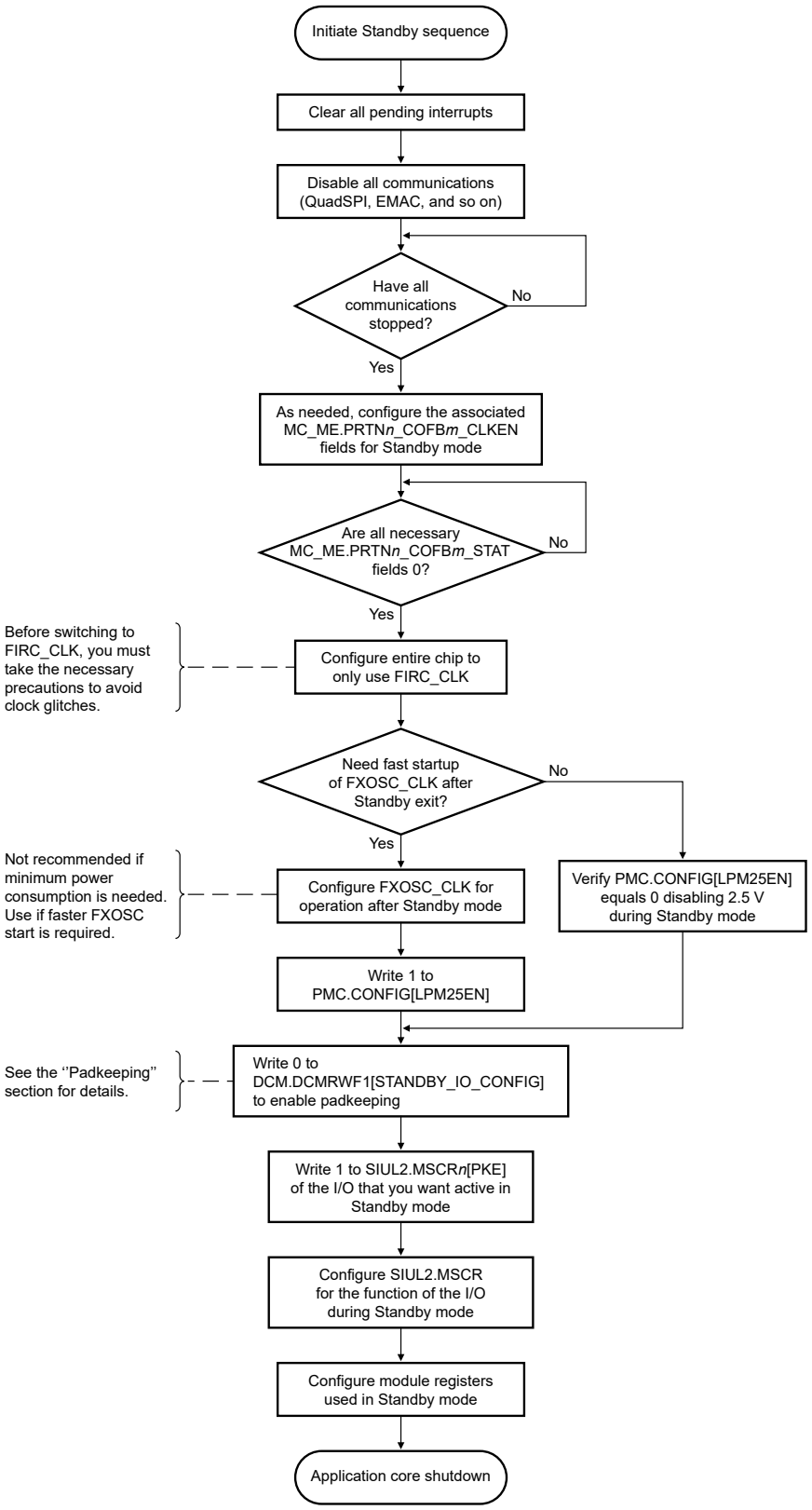


Figure 177. I/O and module configuration for Standby mode

41.5.1.2 SW2: Application core shutdown process

The application cores(s) execute a WFI (as opposed to the main core running the Standby mode entry sequence). See the section "Application core shutdown" in the MC_ME chapter for more information.

The main core configuration (programming valid core ID and enabling standby entry process) and wake-up source configuration must also be set in SW2, so that SW4 contains only the main core WFI execution.

41.5.1.3 SW3: Flash memory low-power handshake and PMC last-mile regulator

In this process, you execute a flash memory low-power handshake and disable the PMC last-mile regulator by executing the procedures indicated in Figure 178 and Figure 179.

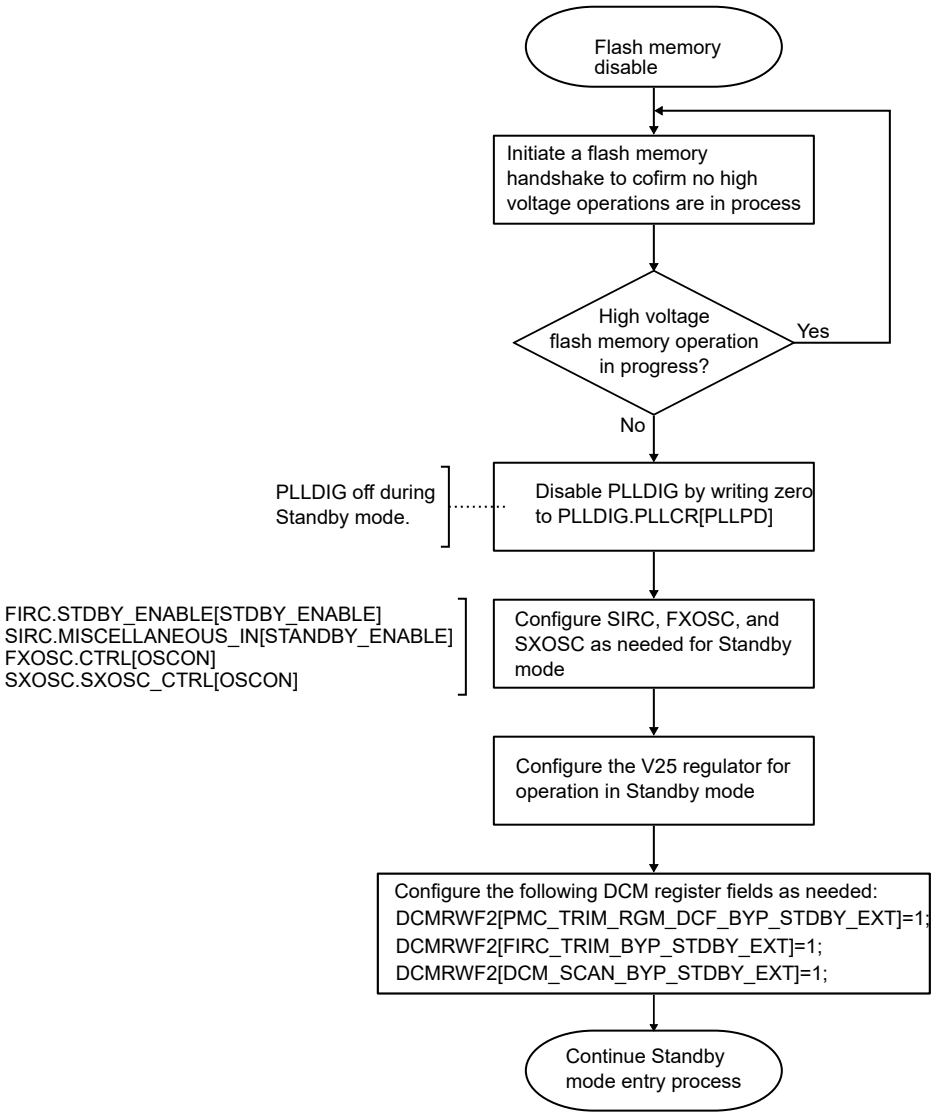


Figure 178. Flash memory Standby mode configuration

NOTE

Disable the last-mile regulator according to the Last-mile regulator disable sequence (not applicable for S32K3x8, S32K312, S32K311 and S32K310).

41.5.1.3.1 Last-mile regulator disable sequence (not applicable for S32K3x8, S32K312, S32K311 and S32K310)

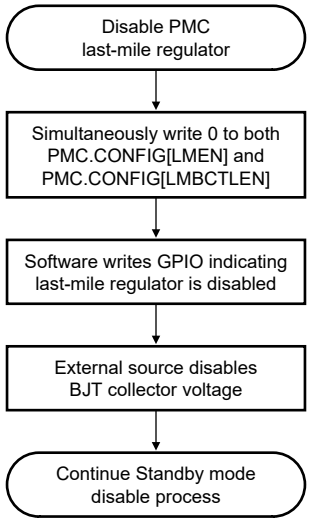


Figure 179. Last-mile regulator disable sequence

When exiting Standby mode, the 1.1 V capacitor is charging. You can charge the 1.5 V capacitor during or after the 1.1 V capacitor is charged. If you charge the 1.5 V capacitor sequentially after the 1.1 V capacitor, you will need additional time to complete the overall charging process.

Leaving PMC.CTRL[LMBCTLEN] = 1 saves time when exiting Standby mode and does not draw any additional current during this mode.

41.5.1.4 SW4: Main core shutdown process

For information on this process, see these in the "Mode Entry Module (MC_ME)" chapter:

- Figure "Standby entry sequence along with main core shutdown"
- Section "Main core shutdown and Standby mode entry"

You must configure WKPU before disabling interrupts to avoid missing any events as shown in the Standby entry sequence along with main core shutdown flowchart in the MC_ME chapter. You must program WKPU, disable interrupts, and configure MC_ME's Standby mode entry in SW1 (see SW1: Module shutdown process for more information). In SW4, you must perform only the main-core WFI execution.

41.5.2 Hardware Standby mode entry sequence

The hardware Standby mode entry sequence consists of handshaking between MC_PCU and MC_ME occurring after Software Standby mode entry sequence completes. The FSM in MC_PCU performs these steps automatically and does not require your intervention (see Power sequence FSM in the MC_PCU chapter for more information).

During the MC_ME and PCU phase, MC_ME and PCU:

- Enable FIRC.
- Deassert isolation.
- Deassert reset to PD1.

41.5.3 PMC Standby mode entry

The PMC Standby mode entry sequence starts after Hardware Standby mode entry sequence completes, and consists of these phases:

1. Standby mode entry acknowledgment and initiation of an internal low-power process on receiving Standby mode entry request.
2. Disabling of the boot regulator within the PMC low-power process. This point is only applicable for S32K344/S32K324/S32K314 and S32K342/S32K322/S32K341.
3. Disabling of the V25 regulator and the FPM LVR monitors.
4. Disabling of the V25 regulator (oscillator and flash memory supply) and the LPM monitor, unless it is enabled during Standby mode (see PMC.CONFIG[LPM25EN] field in the PMC chapter for more information).
5. Disabling of the VDD_HV_B LPM monitors except S32K312, S32K311 and S32K310.

During PMC phase, after receiving an LPM request, PMC:

- Deasserts FPM ready signal.
- Starts the courtesy timer.
- When the courtesy timer expires, PMC:
  - Disables the V25 regulator.
  - Disables the HP reference blocks.
  - Disables the external NMOS.
  - Opens the PD0 switch.
  - Enables a core LPM request.
  - If selected, enables a 2.5 V LPM request.
  - Disengages FPM monitors.
  - Disables the VDD_HV_B LPM monitor, if present and deselected.
- Waits for LPM signal deassertion.

41.6 Chip status at the end of Standby mode entry sequence

After PMC Standby mode entry completes, the chip completes Standby mode entry as follows:

1. Configures Standby domain peripherals according to SW1 (see SW1: Module shutdown process).
2. Enables pad keeping on pins as described in SW1 (see SW1: Module shutdown process). See Pad keeping for other details.
3. Powers down all memory types except Standby RAM (in SRAM0).
   - The V25 regulator can remain on during Standby mode. However, for maximum power savings, it must remain off in this mode.
4. Isolates Standby and Run domains from each other:
   - Standby domain is functional.
   - Run domain is held in reset.
5. Configures the system clock (FIRC_CLK or PLL_PHI*n*_CLK, depending on their configuration) and other clock sources according to SW3 (see Figure 178 for the procedure details).

   The cores are off and in the Standby domain.
6. Turns off the boot[10] and FPM regulators.

7. Waits for a wake-up event to initiate recovery from Standby mode.

41.7 Chip operation in Standby mode

This chip supports the following functionalities in Standby mode:

- STANDBY_RAM content retention during Standby mode.
- Wake-up from up to 60 digital inputs (for details, see the signals WKUP[*n*] functions of WKPU module in the IOMUX file attached to this document). The section "Wakeup Unit Configurations" in the "Wakeup Unit (WKPU)" chapter shows mapping of the wake-up sources.
- Wake-up from up to 16 analog inputs through the Trigger mode functionality (see the signals CMP*n*_IN*m* functions of CMP*n* modules in the IOMUX file attached to this document).
- Wake-up from on-chip timers:
  - RTI (function of PIT[0])
  - SWT0
  - RTC
- Ability to configure the chip clocking modules to optionally enable or disable in Standby mode (FIRC, SXOSC, FXOSC, and SIRC).

41.8 Standby mode exit

This chip supports Standby mode exit from a wake-up, functional reset, or destructive reset event. The sources that cause chip Standby mode exit are:

- MC_RGM functional reset event
- MC_RGM destructive reset event
- WKPU wake-up events, WKPU[0]–WKPU[63]. See the WKPU chapter for more information.

After Standby mode exit, the following events occur (for more information, see "Power sequence FSM" in the MC_PCU chapter for MC_PCU FSM transitions during entry into and exit from Standby mode):

1. A wake-up event arrives.
2. FIRC starts powering up (if disabled in Standby mode).
3. PMC starts the transition process to FPM (for example, enables the last-mile regulator and provides V11_RUN supply to the chip).
4. MC_PCU removes the isolation between Run and Standby domains.
5. Run domain reset deasserts (asserts on Standby mode entry) and the chip undergoes a functional reset exit sequence for this domain.
6. The chip enters Run mode of operation.

NOTE

A reset event during standby results in pad controls to get reset. Thereby, resulting in unpredictable toggling at GPIO.

[10] Not applicable for S32K388, S32K389, S32K312, S32K311, and S32K310.

41.8.1 Faster Standby mode exit

The chip supports an optional configuration for faster recovery from Standby mode on the expense of a higher capacitor recharging current. See the CONFIG[FASTREC] field description in the PMC chapter for more information on faster PMC recovery from Standby mode.

This chip supports an optional feature that bypasses:

- FIRC trimming
- PMC trimming
- DCM scanning

To use the bypass operation, write 1 to DCM.DCMRWF2[5], DCM.DCMRWF2[4], and DCM.DCMRWF2[3] respectively before Standby mode entry. This results in a considerable reduction in Standby mode exit duration. The trim values are retained across Standby mode and bypassing these values does not cause any impact. Even if the FIRC trimming is bypassed, the FIRC must be at 48 MHz before entering Standby mode.

Configuration to achieve the tMODE_STDBYEXIT_FAST as specified in Datasheet

DCMRWF2[PMC_TRIM_RGM_DCF__BYP_STDBY_EXT] = 1

DCMRWF2[FIRC_TRIM_BYP_STDBY_EXT] = 1

DCMRWF2[DCM_SCAN_BYP_STDBY_EXT] = 1

41.8.2 Last-mile regulator enable sequence (not applicable for S32K388, S32K389, S32K312, S32K311 and S32K310)

After Standby mode exit, if the Last Mile Regulator Auto Enable function is not enabled, you must re-enable the PMC last-mile regulator before transitioning to faster clock frequencies. Figure 180 shows the last-mile regulator enable sequence part of the Standby mode exit process.

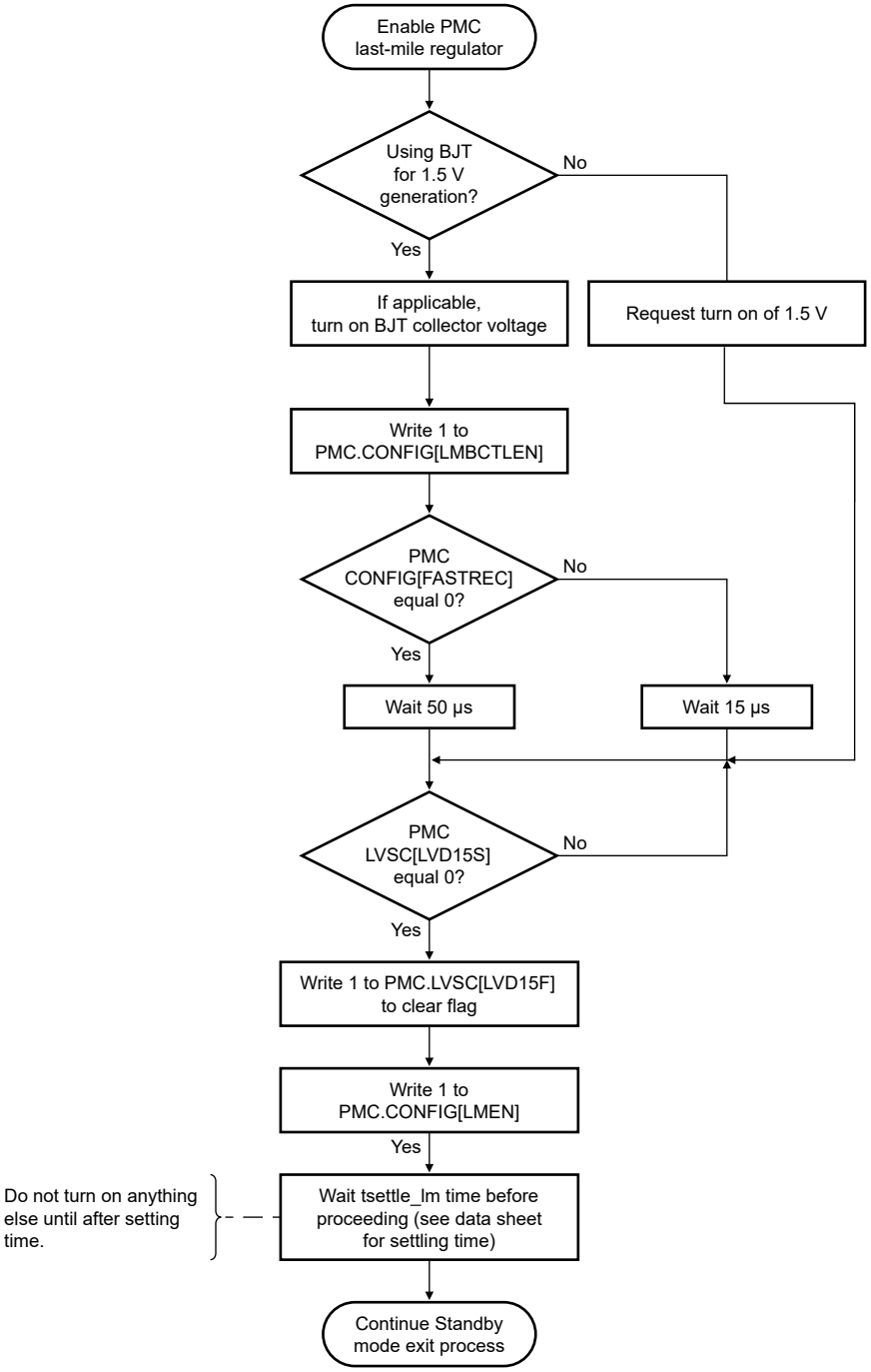


Figure 180. Last-mile regulator enable sequence

41.9 PMIC Handshake with MCU (applicable for S32K328, S32K338, S32K348, S32K358, S32K356, S32K388, and S32K389)

For interfacing with PMIC where handshake is required, MCU will follow a sequence to come out of standby. This handshake will make sure that power supply to the MCU is in proper condition.

3 GPIOs will be used to perform this handshake as shown in below figure.

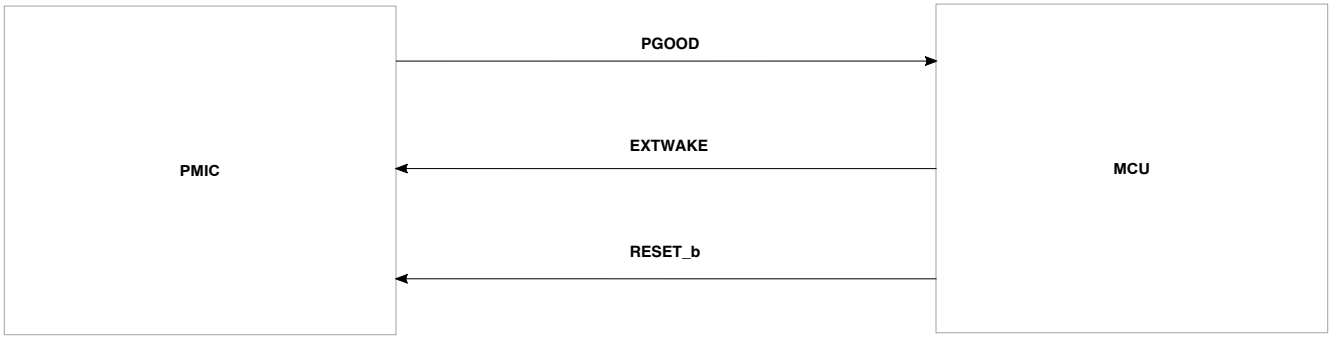


Figure 181. PMIC Handshake with MCU

MCU will assert EXTWAKE signal on receiving a wakeup request from any of the available wakeup sources. If the EXTWAKE signal is enabled via the MSCR configuration (see IOMIX file attached to this document), this wakeup will be triggered to the external PMIC and the PMIC will use it to come out of standby mode. MCU sends the reset_b signal to PMIC to reset the PMIC.

NOTE

After EXTWAKE assertion, POR WDOG is triggered. If PGOOD assertion does not arrive within the configured time, POR WDOG will issue a reset to the MCU.

PMIC will raise PGOOD signal towards MCU for indicating the MCU that it is now able to provide the supplies in proper conditions. If the PGOOD input is enabled via the corresponding IMCR configuration (see IOMIX file attached to this document) and the appropriate PGOOD polarity and enable settings are selected in the corresponding DCMRWF configuration registers (see PMIC_PGOOD_HNDSHK_BYP and PGOOD_POLARITY in Chapter 38, Device Configuration Module General-Purpose Registers (DCM_GPR)), the MCU will proceed to exit from STANDBY mode once the PGOOD input is asserted.

41.10 SMPS

SMPS is regarded as enhancement for the BJT regulator. The essential advantage of SMPS is that the power dissipation is much less compared to the linear regulators. SMPS generates the intermediate power rail of 1.5 V, which is then converted to 1.1 V by the last-mile regulator.

41.11 Chip power domain partitioning

The Standby domain includes the modules listed below. For more information, see "Peripheral reset status" in the "Reset Overview" chapter.

NOTE

Modules within the Standby domain do not participate in the LBIST operation.

41.11.1 Modules available in Standby domain

Table 251. Modules available in Standby domain

PD0 contents	S32K344/ S32K324/S32K314	S32K342/ S32K322/S32K341	S32K312	S32K311	S32K358/ S32K356/ S32K348/ S32K338/ S32K328/ S32K388/S32K389
External Pin wakeups	Minimum 60	Minimum 60	Minimum 60	33 on 100HDQFP & 12 on 48LQFP	Minimum 60
RTC_API	Yes	Yes	Yes	Yes	Yes
LPCMP_0	Yes	Yes	Yes	Yes	Yes
LPCMP_1	Yes	Yes	Yes	No	Yes
LPCMP_2	Yes	No	No	No	Yes
SWT_0	Yes	Yes	Yes	Yes	Yes
WKPU	Yes	Yes	Yes	Yes	Yes
PMC	Yes	Yes	Yes	Yes	Yes
MC_RGM	Yes	Yes	Yes	Yes	Yes
SXOSC(only used for RTC)	Yes	Yes	Yes	No	Yes
FXOSC	Yes	Yes	Yes	Yes	Yes
FIRC	Yes	Yes	Yes	Yes	Yes
SIRC	Yes	Yes	Yes	Yes	Yes
CLK OUT	Yes	Yes	Yes	Yes	Yes
SRAM	32K	32K	32K	32K	64K
PIT_RTI_0	Yes	Yes	Yes	Yes	Yes
DCM and DCF records	Yes	Yes	Yes	Yes	Yes
DCM Flash Interface	Yes	Yes	Yes	Yes	Yes

SWT0 (Cortex-M7_0) resides in the Standby domain and supports a configurable hardware-based timer operation during Standby mode depending on the configuration of SWTs.

All clock sources (except PLLDIG) are available in Standby mode. SIRC is present in the Standby domain for low-power wake-up, but can be enabled.

NOTE

This chip does not support Stop and Wait modes. The only low-power mode it supports is Standby.

41.12 Pad keeping

Pad-keeping allows you to retain the state of the pads while the MCU is in the Standby mode.

41.12.1 Pad keeping on Run domain pins

The process of entering Standby mode ensures that the chip maintains the state of the pads until it wakes-up and re-configures them by software. Software Standby mode entry sequence specifies that you must write 0 to DCM.DCMRWF1[STANDBY_IO_CONFIG] before configuring I/O pad keeping. You could consider DCM.DCMRWF1[STANDBY_IO_CONFIG] field as a global enable for all pad keeping purposes during Standby mode. If you are unable to write to this field as described, pad keeping works as explained in this chapter, during Standby mode.

After the chip exits Standby mode, you must first re-configure your pads, and then, write 1 to DCM.DCMRWF1[STANDBY_IO_CONFIG] field (see "Chip specific register descriptions" in the DCM chapter). Writing 1 to this field disables pad keeping.

41.12.2 Pad keeping on Standby domain pins

See Table 252 for the Standby domain pins (all the rest are Run domain pins).

The Standby domain pads can have pad keeping enabled or disabled based on pad availability in Standby mode. Writing to SIUL2.MSCR*n*[PKE] field configures Standby mode pad keeping on a selected I/O.

Ensure that DCM.DCMRWF1[STANDBY_IO_CONFIG] = 0 before Standby mode entry for pad keeping on Standby domain pads. Write 0 to the field in case its value is not 0 already.

If a pad is not required during Standby mode, the corresponding MSCR*n*[PKE] field in SIUL2 must be 0.

41.12.3 Pad keeping configuration procedure

This procedure specifies how to enable an I/O for pad keeping:

1. Configure SIUL2.MSCR*n* register to control the pad state prior to Standby mode entry (for example, writing 0 to SIUL2.MSCR*n*[OBE], MSCR*n*[IBE], and MSCR[PUE] fields tristates the corresponding I/O).
2. Configure SIUL2.MSCR*n*[PKE] field as needed for an I/O pad keeping state during Standby mode.
3. Write 0 to DCM.DCMRWF1[STANDBY_IO_CONFIG].

   The application core executes WFI, and Standby mode sequence starts.
4. Write 1 to DCM.DCMRWF1[STANDBY_IO_CONFIG] field on Standby mode exit to disable pad keeping.

If you write 1 to DCM.DCMRWF1[STANDBY_IO_CONFIG] field before entering Standby mode, the isolation removal hardware removes pad keeping on Standby mode exit. This (writing 1 to DCM.DCMRWF1[STANDBY_IO_CONFIG] field before Standby mode entry) is useful in case of low-power debug because enabling pad keeping does not allow low-power debug protocol to work properly (because the TDO pad is padkept). For low-power debug, you must write 1 to DCM.DCMRWF1[STANDBY_IO_CONFIG] field prior to Standby mode entry.

In case of Standby mode exit by reset, the pad keeping of the reset pin is removed when the chip is reset on Standby mode wake up. See the "GPIO padkeeping enable" signal in Figure 183. The signal corresponds to wake-up via reset event case.

41.12.4 Pad keeping waveforms

41.12.4.1 Pad keeping when the chip wakes up from Standby mode via an interrupt wake-up event

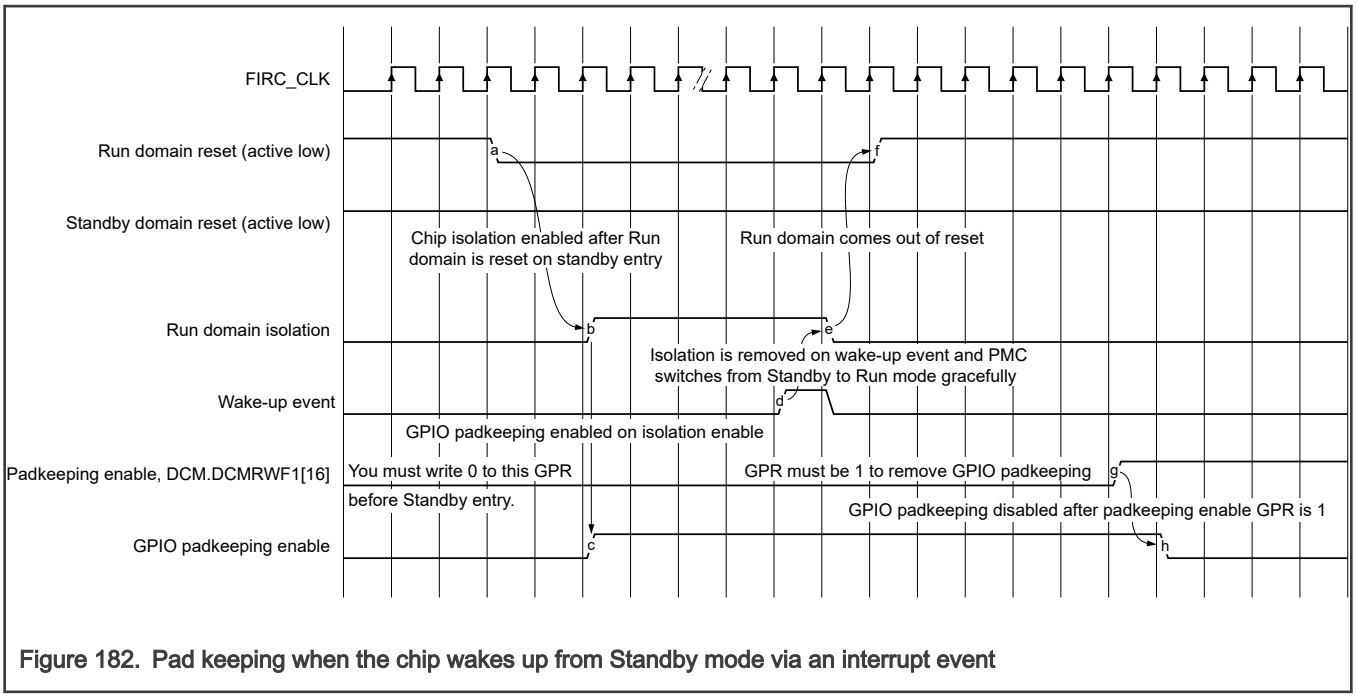


Figure 182. Pad keeping when the chip wakes up from Standby mode via an interrupt event

41.12.4.2 Pad keeping when the chip wakes up from Standby mode via reset

If the chip exits Standby mode via reset, the reset pad keeping is removed when the chip resets after Standby-mode wake-up. See the figure's "GPIO padkeeping enable" waveform corresponding to wake-up via a reset event.

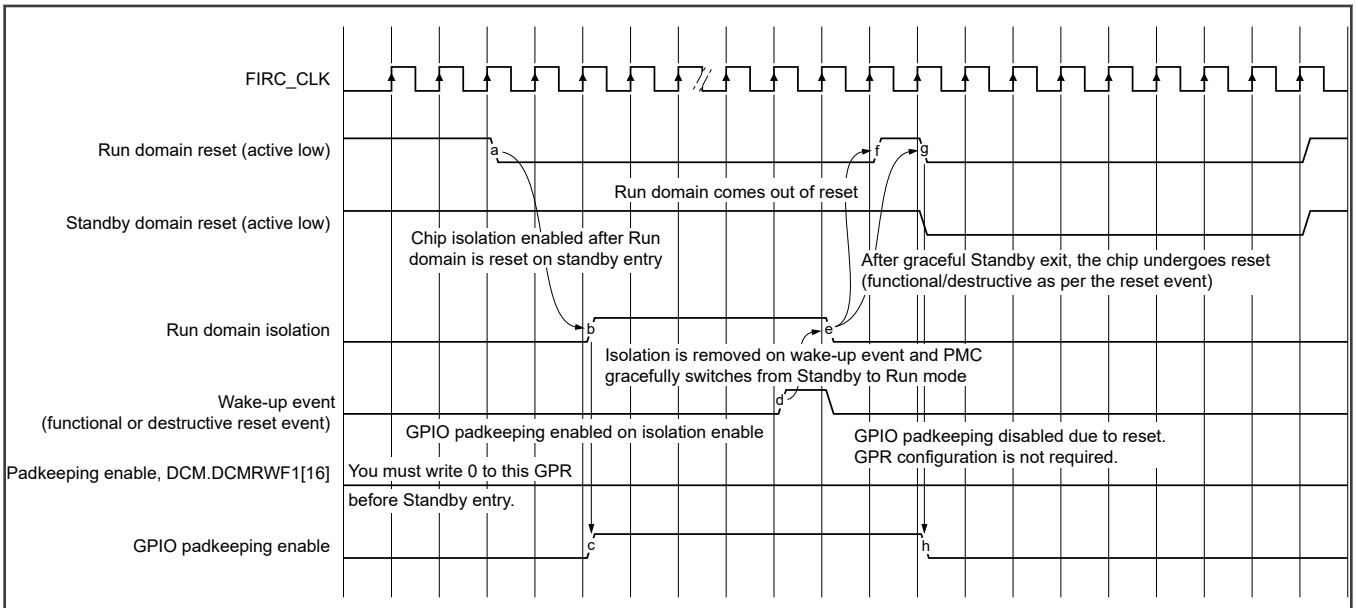


Figure 183. Pad keeping when the chip wakes up from Standby mode via reset

41.12.5 SIUL2's fields for actively-driven pins in Standby mode

If you use a WKPU pin as a wake-up input, perform these operations on SIUL2's fields:

1. Write 1 to SIUL2.MSCR*x*[IBE].

2. Program SIUL2.MSCR*x*[PUE] and SIUL2.MSCR*x*[PUS] according to the use case.
3. Write 0 to SIUL2.MSCR*x*[PKE].

For the GPIO pins that are driven to high impedance during Standby mode:

1. Write 0 to DCM.DCMRWF1[STANDBY_IO_CONFIG] field.
2. Write 0 to SIUL2.MSCR*x*[SSS] field (GPIO mode).
3. Write 0 to SIUL2.MSCR*x*[IBE] and SIUL2.MSCR*x*[OBE] fields.

Some of the pins are, by default, actively driven in Standby mode. If these pins retain a static value throughout the mode, program the corresponding SIUL2.MSCR*x*[PKE] bits individually.

This table shows the list of pins available in Standby mode along with the functions they perform.

Table 252. Active pins in Standby mode

Pin[1]	Function
GPIO_4	CMP0_OUT
GPIO_5	RESET_b
GPIO_9	CMP2_OUT
GPIO_11	CMP0_RRT
GPIO_12	CLKOUT_STANDBY; CMP1_OUT
GPIO_69	CMP2_RRT
GPIO_110	CMP0_RRT
GPIO_131	CMP0_OUT
GPIO_138	CLKOUT_STANDBY
GPIO_143	CMP1_RRT
GPIO_158	CMP1_OUT
GPIO_159	CMP1_RRT
GPIO_174	CMP0_OUT
GPIO_175	CMP0_RRT
GPIO_196	CMP2_OUT
GPIO_197	CMP2_RRT

1. See the IOMUX sheet attached to this document for details on pins configuration in different chips.

41.13 Glossary

Application core Core apart from the main core

Boot regulator The on chip 1.1 V regulator that is active during startup and entry and exit from Standby mode

FSM Finite state machine

FPM Full-power mode (Run mode)—uses the last-mile regulator and 1.5 V source to generate the 1.1 V core logic supply during a full-power operation (V11_RUN)

LPM Low-power mode (Standby mode)—uses the low-power regulator to generate the 1.1 V core logic supply during low-power operation (V11_STANDBY)

LVR	Low voltage reset
Main core	Core initiating the chip Standby mode request (for example, the core corresponding to the "core index" in MC_ME's MAIN_COREID register)
Pad keeping	Maintains I/O pad configuration during Standby mode, if enabled
V11_STANDBY	Core logic and clock sources, low-voltage supply to Standby domain
V11_RUN	Low-voltage supply to Run domain
V15	High-current input for core or logic supply from either an external BJT or from direct 1.5 V external supply
V25	Flash memory, FXOSC, and PLLDIG high-voltage supply
VDDA_ADC	ADC supply voltage
VREFH	ADC high-voltage reference supply
VREFL	ADC low-voltage reference supply
VSS	Core logic ground supply
VDD_HV_A	Main I/O voltage supply (5 V or 3.3 V)
VDD_HV_B	Other I/O domain voltage supply (5 V or 3.3 V)
VRC_CTRL	PMC voltage regulator control output using the BJT option to generate a 1.5 V supply
WFI	Wait for interrupt software instruction

Chapter 42
Power Management Controller (PMC for S32K34x, S32K322, S32K324, and S32K314)

42.1 Overview

PMC is the power management controller for the S32K3 family of microcontrollers. It provides multiple power options to allow you to optimize power consumption for the level of functionality needed. It includes internal voltage regulators, POR, and the integrated low/high voltage detect system with reset (brown-out) capability. The voltage regulator requires a 3.3 V or 5 V input to generate all the required secondary supplies.

42.1.1 Block diagram

The following figure shows the block diagram for this module.

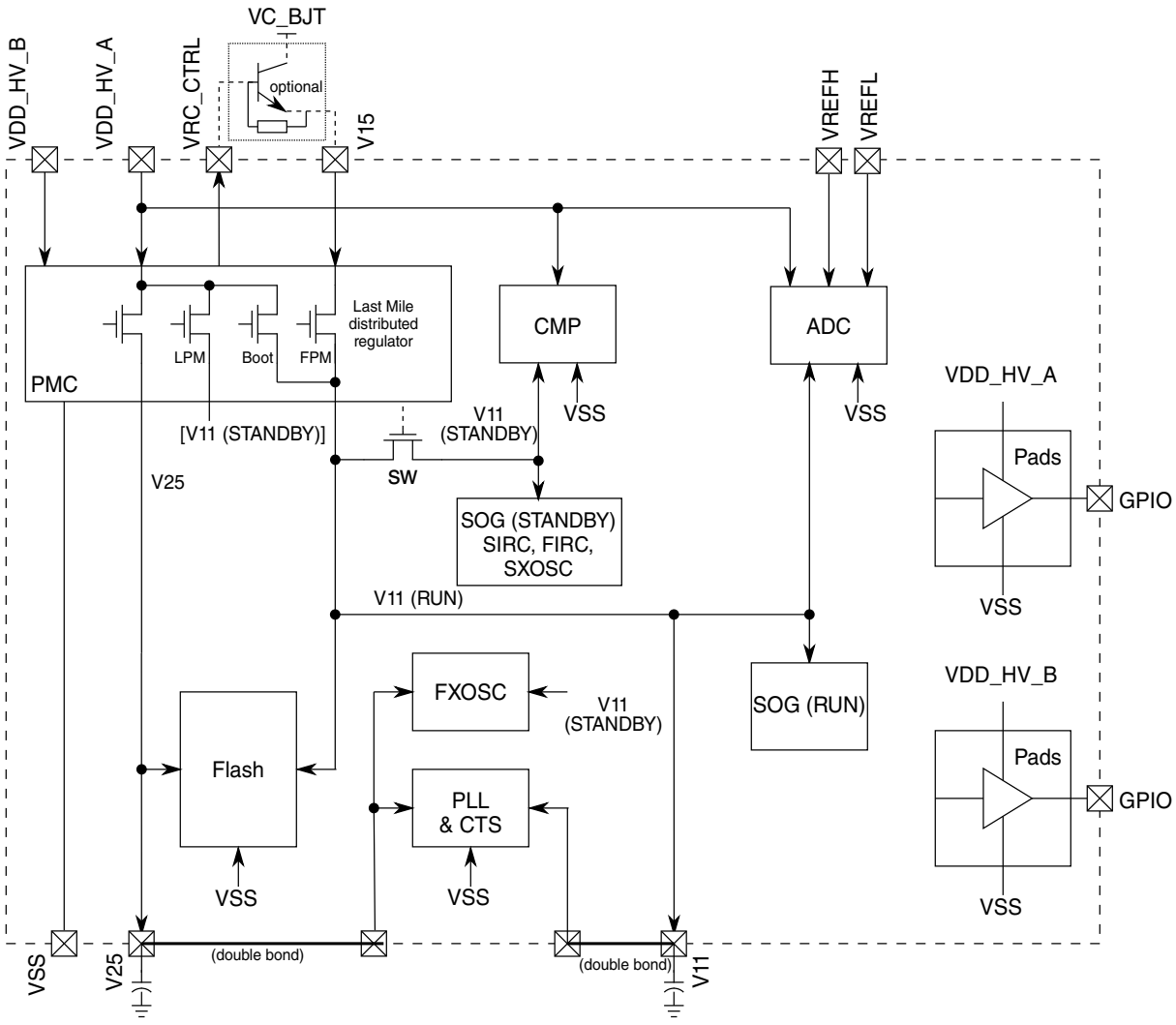


Figure 184. PMC block diagram

42.1.2 Features

PMC includes the following features:

- Combination of internal and external voltage regulator options, offering RUN and Standby modes
- Active POR providing brown-out detect
- LVR for all system-relevant power domains
- LVD and HVD as indication for software

42.2 Functional description

The following sections describe functional details of the PMC.

42.2.1 Modes of operation

PMC provides two basic modes of operation for the voltage regulators and monitors:

- FPM, which is used on chip-level in RUN modes: For high-current consumption
- LPM, which is used on chip-level for Standby modes: For low-current consumption

42.2.2 Reset

The POR and all LVRs are combined into one single MCU POR.

After an MCU POR event, it can be determined which power domain caused it, by reading in the PMC_LVSC register, the POR flag, and LVR flags.

After an initial power ramp up of the MCU in the PMC_LVSC register, the POR flag and the LVR flags are all set to 1. The Go/Nogo flags have an arbitrary value.

NOTE

After an initial power ramp up, all flags in the LVSC register must be cleared (by writing 0xFFFFFFFF to the LVSC register).

Because the flags are sticky bits, it is required to clear them before usage. So, in case of an unexpected MCU POR, the source of the problem can be tracked and debugged by reading the flags in the LVSC register.

42.2.3 Interrupts

PMC includes two interrupt sources:

- HVD interrupt: It combines all HVD monitors into one interrupt source. Interrupt enable is the HVDIE field in the CONFIG register. See the PMC Configuration Register (CONFIG) and Low Voltage Status and Control Register (LVSC) registers for details.
- LVD interrupt: It combines the LVD15 and LVD5A monitors into one interrupt source. Interrupt enable is the LVDIE field in the CONFIG register. See the CONFIG and LVSC registers for details.

42.3 Signals

This table describes the PMC module signals.

Table 253. Signal Description

Signal	I/O	Description
VDD_HV_A	Supply input	This is the primary high-voltage supply input to PMC. VDD_HV_A is used for the PMC internal precision references. After the VDD_HV_A

Table continues on the next page...

Table 253. Signal Description (continued)

Signal	I/O	Description
		domain is powered up, it must be kept powered at all times of operation (FPM and LPM).
VDD_HV_B[1]	Supply input	This is the secondary high-voltage supply input supervised by the PMC. After the VDD_HV_B domain is powered up, it must be kept powered at all times of operation (FPM and LPM).
V25	Supply output	V25 power supply domain is driven by a fully integrated low-dropout linear voltage regulator. It supplies the Flash memory and (via a double bond) the clock modules.
V15[1]	Supply input	This is the high-current input for core/logic supply that can be fed by an external BJT or another source.
VRC_CTRL	Output	VRC_CTRL connects to the base of external BJT, if this option is used to generate V15.
V11	Supply output	V11 is the core/logic supply. It is driven by a fully integrated low-dropout linear voltage regulator.
VSS	Ground	VSS must be grounded. All VSS Pins need to be externally connected to the same ground node.

1. VDD_HV_B and V15 not present in S32K312 and S32K311.

42.4 PMC register descriptions

42.4.1 PMC memory map

This section includes the PMC module memory map and detailed descriptions for all the registers.

PMC_S32K344 base address: 402E_8000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Low Voltage Status and Control Register (LVSC)	32	RW	See section
4h	PMC Configuration Register (CONFIG)	32	RW	See section
Ch	Version ID register (VERID)	32	R	0200_0001h

42.4.2 Low Voltage Status and Control Register (LVSC)

Offset

Register	Offset
LVSC	0h

Function

This register contains status and control bits to support the low-voltage reset and low- or high-voltage detect function. When the PMC is in LPM, the low- or high-voltage detect systems are disabled.

NOTE

For all flags that are not affected by reset (POR flag, all LVR flags, all GNG flags), in case a reset occurs at the same time while trying to clear the flags (by writing 1), the flag value is not defined appropriately. In this case, you need to clear the flag again after exit from reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PORF	0					GNG11OS...	GNG25OS...	LVR11LPF	LVR11F	LVR25LPF	LVR25F	LVRBLPF	LVRBF	LVRALPF	LVRAF
W	W1C						W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	u	0	0	0	0	0	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		LVD15S	LVD5AS	HVD11S	HVD25S	HVDBS	HVDAS	0		LVD15F	LVD5AF	HVD11F	HVD25F	HVDBF	HVDAF
W											W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 PORF	POR flag Indicates that a power-on reset event has occurred. Writing 1 to this field clears it, and other reset sources have no effect. 0b - No power-on reset event has occurred 1b - Power-on reset event has occurred
30-26 —	Reserved
25 GNG11OSCF	Go/NoGo detect flag on Osc part of V11 domain Indicates that the Go/NoGo sensor has detected a low voltage on the V11 domain in FPM. This applies to only that part of the domain which supplies the 1.1V clocking modules (for example, PLL). Writing 1 to the field clears it, and other reset sources have no effect. 0b - No event has occurred 1b - NoGo event has occurred
24 GNG25OSCF	GO/NoGo detect flag on Osc part of V25 domain Indicates that the Go/NoGo sensor has detected a low voltage on the V25 domain in FPM. This applies to only that part of the domain which supplies the 2.5V clocking modules (for example, XOSC and IRC). Writing 1 to the field clears it, and other reset sources have no effect. 0b - No event has occurred 1b - NoGo event has occurred

Table continues on the next page...

Table continued from the previous page...

Field	Function
23 LVR11LPF	LVR11LP flag on V11 domain Indicates that a low-voltage reset event has occurred on the 1.1V V11 power domain (FPM or LPM). Writing 1 to the field clears it, and other reset sources have no effect. 0b - No low-voltage reset event has occurred 1b - Low-voltage reset event has occurred
22 LVR11F	LVR11 flag on V11 domain in FPM Indicates that a low-voltage reset event has occurred on the 1.1V V11 power domain in the FPM. Writing 1 to this field clears it, and other reset sources have no effect. 0b - No low-voltage reset event has occurred 1b - Low-voltage reset event has occurred
21 LVR25LPF	LVR25LP flag on V25 domain Indicates that a low-voltage reset event has occurred on the V25 power domain (FPM or LPM). Writing 1 to this field clears it, and other reset sources have no effect. 0b - No low-voltage reset event has occurred 1b - Low-voltage reset event has occurred
20 LVR25F	LVR25 flag on V25 domain in FPM Indicates that a low-voltage reset event has occurred on the V25 power domain in FPM. Writing 1 to this field clears it, and other reset sources have no effect. 0b - No low-voltage reset event has occurred 1b - Low-voltage reset event has occurred
19 LVRBLPF	LVRBLP flag on VDD_HV_B domain Indicates that a low-voltage reset event has occurred on the VDD_HV_B power domain (FPM or LPM). Writing 1 to this field clears it, and other reset sources have no effect. 0b - No low-voltage reset event has occurred 1b - Low-voltage reset event has occurred
18 LVRBF	LVRB flag on VDD_HV_B domain in FPM Indicates that a low-voltage reset event has occurred on the VDD_HV_B power domain in FPM. Writing 1 to this field clears it, and other reset sources have no effect. 0b - No low-voltage reset event has occurred 1b - Low-voltage reset event has occurred
17 LVRALPF	LVRALP flag on VDD_HV_A domain Indicates that a low-voltage reset event has occurred on the VDD_HV_A power domain (FPM or LPM). Writing 1 to this field clears it, and other reset sources have no effect.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No low-voltage reset event has occurred 1b - Low-voltage reset event has occurred
16 LVRAF	LVRA flag on VDD_HV_A domain in FPM Indicates that a low-voltage reset event has occurred on the VDD_HV_A power domain in FPM. Writing 1 to this field clears it, and other reset sources have no effect. 0b - No low-voltage reset event has occurred 1b - Low-voltage reset event has occurred
15-14 —	Reserved
13 LVD15S	LVD15 status on V15 domain in FPM Shows the status of the 1.5V low-voltage detect, LVD15, on the V15 power domain. This monitor indicates when the V15 voltage level generated from external is on target. This feature is available only in FPM and disabled in LPM. After a reset or wakeup from LPM, the software should clear the LVD15F flag and check the status bit LVD15S to determine voltage level on V15 supply. 0b - Voltage on V15 is above low-voltage detect threshold or LPM. 1b - Voltage on V15 is below low-voltage detect threshold and FPM.
12 LVD5AS	LVD5A status on VDD_HV_A domain in FPM Shows the status of the 5V low-voltage detect, LVD5A, on the VDD_HV_A power domain. This monitor indicates if the voltage is below a certain threshold, which is set slightly below 4.5V (see Datasheet for exact value). The feature is only available in FPM and disabled in LPM. After a reset or wakeup from LPM, the software should clear the LVD5AF flag and check the status bit LVD5AS to determine voltage level on VDD_HV_A supply. 0b - Voltage on VDD_HV_A is above low-voltage detect threshold 1b - Voltage on VDD_HV_A is below low-voltage detect threshold
11 HVD11S	HVD11 status on V11 domain in FPM Shows the status of the high-voltage detect, HVD11, on the V11 power domain. This feature is only available in FPM and disabled in LPM. 0b - Voltage on V11 is below high-voltage detect threshold or LPM. 1b - Voltage on V11 is above high-voltage detect threshold and FPM.
10 HVD25S	HVD25 status on V25 domain in FPM Shows the status of the high-voltage detect, HVD25, on the V25 power domain. The feature is only available in FPM and disabled in LPM. 0b - Voltage on V25 is below high-voltage detect threshold or LPM. 1b - Voltage on V25 is above high-voltage detect threshold and FPM.

Table continues on the next page...

Table continued from the previous page...

Field	Function
9 HVDBS	HVDB status on VDD_HV_B domain in FPM Shows the status of the high-voltage detect, HVDB, on the VDD_HV_B power domain. The feature is only available in FPM and disabled in LPM. 0b - Voltage on VDD_HV_B is below high-voltage detect threshold or LPM. 1b - Voltage on VDD_HV_B is above high-voltage detect threshold and FPM.
8 HVDAS	HVDA status on VDD_HV_A domain in FPM Shows the status of the high-voltage detect HVDA on the VDD_HV_A power domain. The feature is only available in FPM and disabled in LPM. 0b - Voltage on VDD_HV_A is below high-voltage detect threshold or LPM. 1b - Voltage on VDD_HV_A is above high-voltage detect threshold and FPM.
7-6 —	Reserved
5 LVD15F	LVD15 flag on V15 domain in FPM PMC writes 1 to the LVD15F field when the LVD15S status field changes. To clear LVD15F, write 1 to it. If enabled, LVD15F causes an interrupt request. 0b - LVD15S has not changed. 1b - LVD15S has changed.
4 LVD5AF	LVD5A flag on VDD_HV_A domain in FPM PMC writes 1 to this field when LVD5AS status field changes. To clear LVD5AF, write 1 to it. If enabled, LVD5AF causes an interrupt request. 0b - LVD5AS has not changed. 1b - LVD5AS has changed.
3 HVD11F	HVD11 flag on V11 domain in FPM PMC writes 1 to this field when the HVD11S status field changes. To clear HVD11F, write 1 to it. If enabled, HVD11F causes an interrupt request. 0b - HVD11S has not changed. 1b - HVD11S has changed.
2 HVD25F	HVD25 flag on V25 domain in FPM PMC writes 1 to the HVD25F field when HVD25S status field changes. To clear HVD25F, write 1 to it. If enabled, HVD25F causes an interrupt request. 0b - HVD25S has not changed. 1b - HVD25S has changed.
1	HVDB flag on VDD_HV_B domain in FPM

Table continues on the next page...

Table continued from the previous page...

Field	Function
HVDBF	PMC writes 1 to the HVDBF field when HVDBS status field changes. To clear HVDBF, write 1 to it. If enabled, HVDBF causes an interrupt request. 0b - HVDBS has not changed. 1b - HVDBS has changed.
0 HVDAF	HVDA flag on VDD_HV_A domain in FPM PMC writes 1 to the HVDAF field when HVDAS status field changes. To clear HVDAF, write 1 to it. If enabled, HVDAF causes an interrupt request. 0b - HVDAS has not changed. 1b - HVDAS has changed.

42.4.3 PMC Configuration Register (CONFIG)

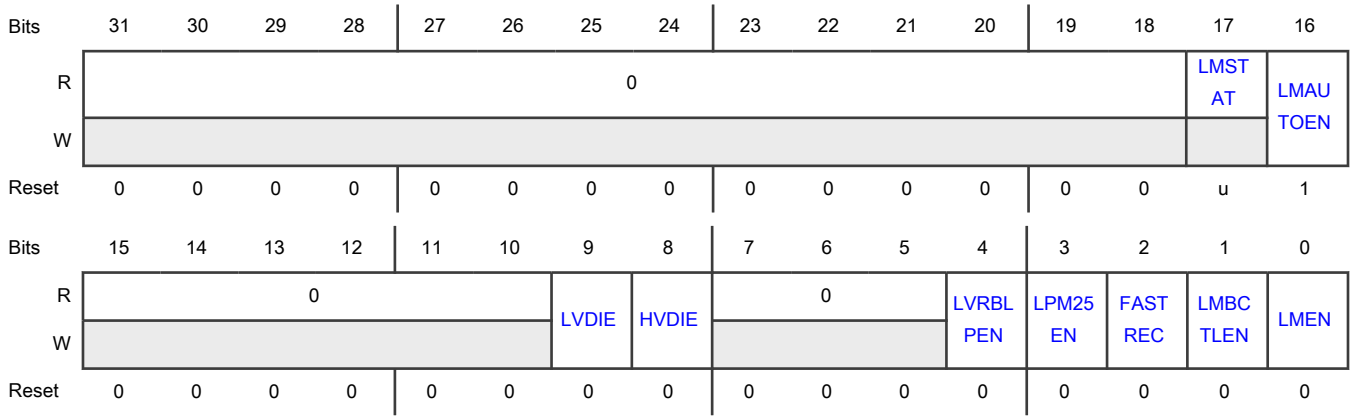
Offset

Register	Offset
CONFIG	4h

Function

If the Last Mile Regulator option is not available on your device (Version ID register (VERID) register: LMFEAT=0), then the bits LMEN, LMBCTLEN, LMAUTOEN and LMSTAT fields can not be written and read always 0.

Diagram



Fields

Field	Function
31-18	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
17 LMSTAT	Last Mile regulator status bit This bits reflects the current status of the Last Mile regulator. This information is required as when auto turn over feature is enabled (LMAUTOEN=1) the PMC will switch automatically between Boot regulator and Last Mile regulator depending on the V15 status (LVD15S). 0b - Last Mile Regulator is off 1b - Last Mile Regulator is on
16 LMAUTOEN	Last Mile regulator auto turn over bit Enables to turn over automatically from Boot Regulator Mode to Last Mile regulator mode and vice versa depending on the V15 voltage status (LVD15S). As long as LMEN=0 software must make sure that the system clock is on FIRC clock or slower. To use higher clock speed software must set LMEN=1. 0b - Auto turnover disabled 1b - Auto turnover enabled
15-10 —	Reserved
9 LVDIE	Low voltage detect interrupt enable Enables hardware interrupt requests if any of the following flags is set: LVD5AF, LVD15F. LVD interrupt must be disabled before going into LPM. 0b - LVD hardware interrupt is disabled (use polling). 1b - Request an LVD hardware interrupt when LVDA5F=1 or LVD15F=1.
8 HVDIE	High voltage detect interrupt enable Enables hardware interrupt requests if any of the following flags is set: HVDAF, HVDBF, HVD25F, HVD11F. 0b - HVD hardware interrupt is disabled (use polling). 1b - Request an HVD hardware interrupt when HVDAF=1, HVDBF=1, HVD25F=1, or HVD11F=1.
7-5 —	Reserved
4 LVRBLPEN	LVRBLP enable bit during LPM Controls whether the low-voltage reset detection (LVRBLP) on the VDD_HV_B power domain is active or inactive in LPM 0b - Low-voltage reset detection is disabled in LPM. 1b - Low-voltage reset detection is enabled in LPM.
3	V25 domain enable bit during LPM

Table continues on the next page...

Table continued from the previous page...

Field	Function
LPM25EN	Controls whether the V25 regulator and low-voltage reset detection (LVR25LP) are active or inactive in LPM 0b - V25 regulator and LVR25LP are disabled in LPM. 1b - V25 regulator and LVR25LP are enabled in LPM.
2 FASTREC	Fast recovery from LPM enable bit Controls the recovery time from LPM to FPM. At recovery from LPM, all the tank capacitors from the secondary supplies have to be recharged. This causes a high-current demand, which might not be met by the supply driving the VDD_HV_A primary domain. When selecting the fast recovery time, the current for recharging is approximately three times higher than that for FASTREC=0. The application must determine from the drive capability of the external VDD_HV_A regulator, the size of tank caps on the secondary supply pins and the selected recovery time if this is sufficient to start up from LPM in time. 0b - Normal recovery time from LPM 1b - Fast recovery time from LPM
1 LMBCTLEN	Last Mile regulator base control enable bit This field must be set to 1 if external BJT between VDD_HV_A and V15 is used on the PCB. The base of this BJT must be connected to the VRC_CTRL pin and is controlled by the PMC to regulate a voltage of 1.5V on V15 pin. After setting LMBCTLEN=1 the software has to wait for 15us (FASTREC=1) respectively 50us (FASTREC=0) before polling LVD15S=0 and then setting LMEN=1. This respects the softstart time of the V15 regulator. If LMAUTOEN=1 then LMBCTLEN can be left enabled when going into LPM, the hardware will turn the regulator off and back on automatically after recovery from LPM. 0b - External BCTL regulator for V15 disabled 1b - External BCTL regulator for V15 enabled
0 LMEN	Last Mile regulator enable bit Enables the Last Mile regulator, which regulates an external 1.5V voltage on V15 down to the core and logic supply (V11 power domain), which is typically 1.1V. Setting LMEN=1 hands over the V11 voltage generation from the Boot regulator to the Last Mile regulator. The software must ensure that before enabling the Last Mile regulator, the voltage on V15 is sufficiently high as indicated by the LVD15S status field (LVD15S=0). To use external BJT between VDD_HV_A and V15, the LMBCTLEN field must be set before the LMEN field, and the software must wait until 1.5V is up (LVD15S=0). If LMAUTOEN=0 then to disable the Last Mile regulator, LMEN and LMBCTLEN must be cleared simultaneously (single register write). The software must disable (LMEN=0) the Last Mile regulator before going into LPM. After setting LMEN=1, software must wait a minimum time of 1.5us before changing clock rate. 0b - Last Mile regulator is disabled. 1b - Last Mile regulator is enabled.

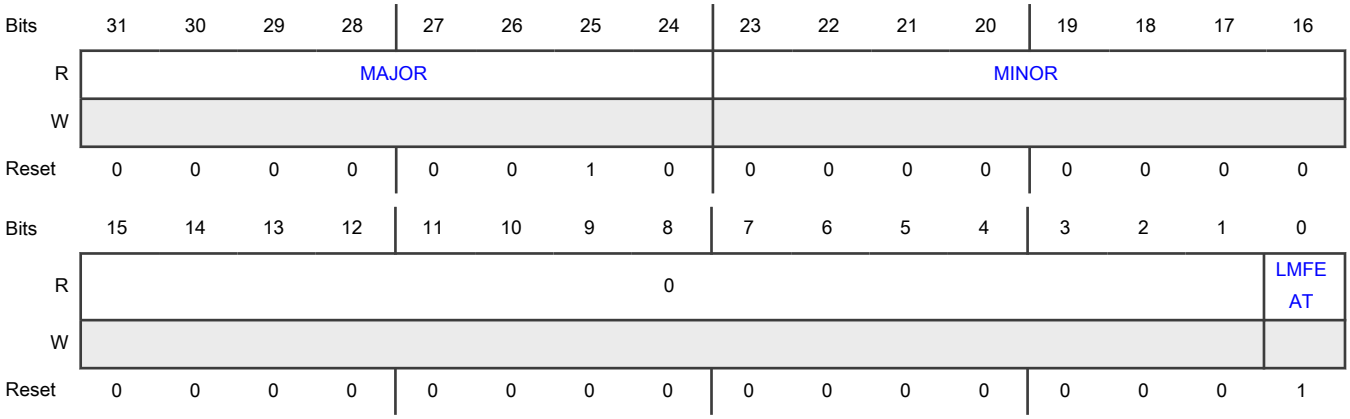
42.4.4 Version ID register (VERID)

Offset

Register	Offset
VERID	Ch

Function
This register returns the major and minor version numbers of hardware implementation.

Diagram



Fields

Field	Function
31-24 MAJOR	Major version number Returns the version number for the specification
23-16 MINOR	Minor version number Returns the version number for the hardware implementation
15-1 —	Reserved
0 LMFEAT	Last Mile Regulator Feature This read-only field shows if the Last Mile regulator feature is available. 0b - No Last Mile regulator 1b - Last Mile regulator (1.5V to 1.1V) is available

42.5 Glossary

FPM	Full Performance mode
HVD	High voltage detect

IRC	Internal RC oscillator
LM	Last mile regulator
LPM	Low Performance mode
LVD	Low voltage detect
LVR	Low voltage reset
NVM	Nonvolatile memory
POR	Power on reset
XOSC	External crystal oscillator

Chapter 43
Power Management Controller (PMC for S32K310, S32K311 and S32K312)

43.1 Overview

PMC provides multiple power options to allow you to optimize power consumption for the appropriate level of functionality. It includes:

- Internal voltage regulators
- POR
- Integrated low and high voltage detection system with reset (brownout) capability

The voltage regulator requires a 3.3 V or 5 V input to generate all the required secondary supplies.

43.1.1 Block diagram

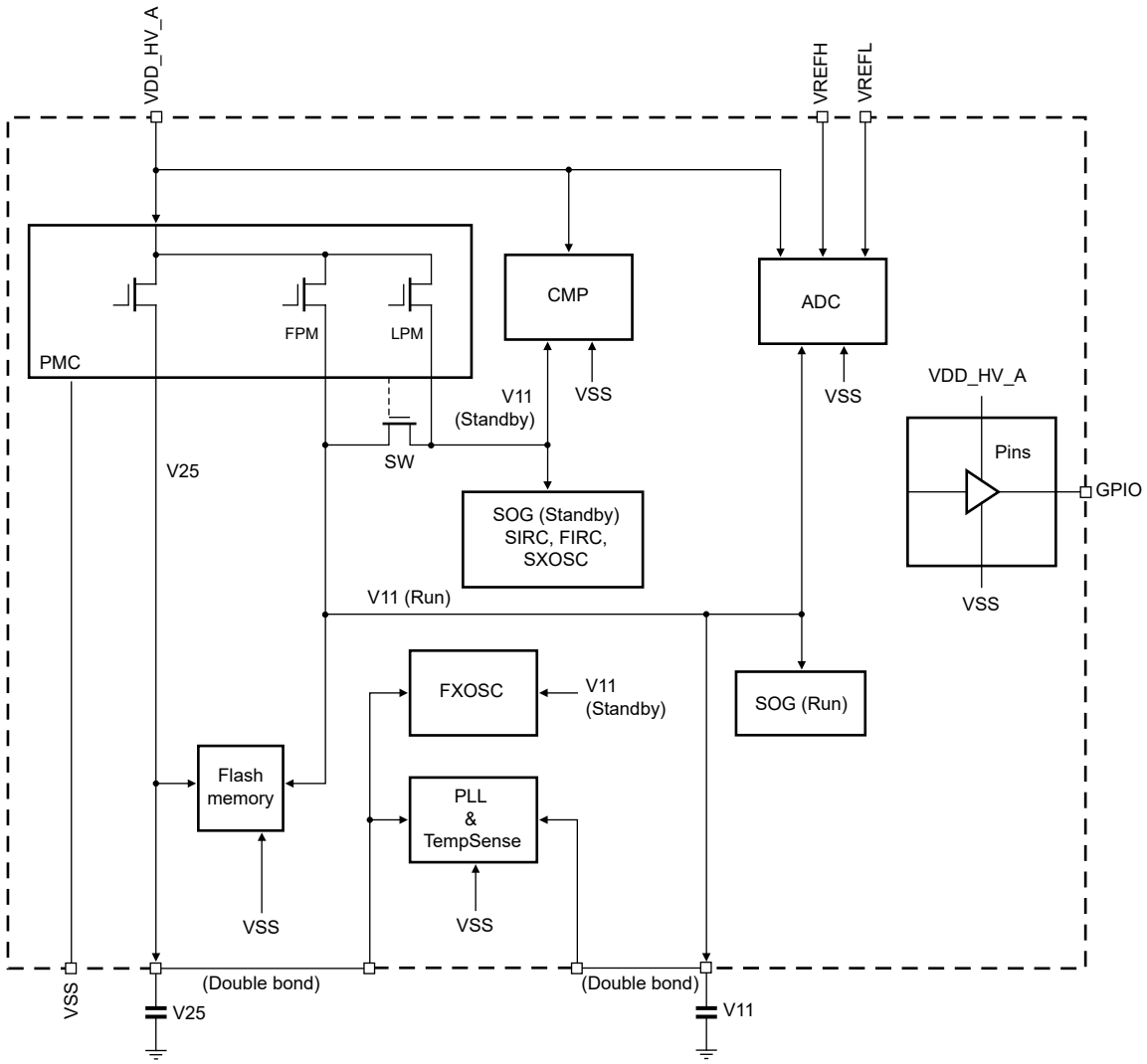


Figure 185. Block diagram

43.1.2 Features

- A combination of internal and external voltage regulator options, offering Run and Standby modes
- Active POR, providing brownout detect
- LVR for all system-relevant power domains
- Software-readable Low Voltage Status and Control register that contains flags that indicate low- or high-voltage conditions

43.2 Functional description

43.2.1 Modes of operation

PMC provides two basic modes of operation for voltage regulators and monitors:

- FPM supports chip run modes that have high current consumption.

- LPM supports chip standby modes that have low current consumption.

43.2.2 Reset

The POR and all LVRs combine into one single chip POR.

After a chip POR event, you can determine which power domain caused it by reading the PORF and LVR*x* flags in Low Voltage Status And Control (LVSC).

After an initial power ramp-up of the chip in LVSC, PMC sets the POR and LVR*x* flags. The go/no go flags have an arbitrary value.

NOTE

After an initial power ramp-up, you must clear all flags in LVSC by writing FFFF_FFFFh to that register.

Because the flags are sticky bits, you must clear them before using them. That way, if an unexpected chip POR occurs you can track and debug the source of the problem by reading the flags in LVSC.

43.2.3 Interrupts

PMC includes two interrupt sources:

- HVD interrupt: Combines all HVD monitors into one interrupt source. CONFIG[HVDIE] enables this interrupt. See PMC Configuration (CONFIG) and Low Voltage Status And Control (LVSC) for details.
- LVD interrupt: The LVD5A monitor is the only interrupt source. CONFIG[LVDIE] enables this interrupt. See PMC Configuration (CONFIG) and Low Voltage Status And Control (LVSC) for details.

43.3 Signals

Table 254. Signals

Signal	Type	Description
VDD_HV_A	Supply input	The primary high-voltage supply input to PMC. VDD_HV_A is used for the PMC internal precision references. After the VDD_HV_A power domain is powered up, it must always be kept powered for both FPM and LPM.
V25	Supply output	A fully integrated low-dropout linear voltage regulator drives the V25 power supply domain. V25 supplies the flash memory and (via a double bond) the clock modules.
V11	Supply output	V11 is the core and logic supply. A fully integrated low-dropout linear voltage regulator drives V11.
VSS	Ground	VSS must be grounded. You must connect all VSS pins externally to the same ground node.

43.4 PMC register descriptions

43.4.1 PMC memory map

PMC_S32K312 base address: 402E_8000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Low Voltage Status And Control (LVSC)	32	RW	See section
4h	PMC Configuration (CONFIG)	32	RW	0000_0000h
Ch	Version ID (VERID)	32	R	0300_0000h

43.4.2 Low Voltage Status And Control (LVSC)

Offset

Register	Offset
LVSC	0h

Function

Contains status and control fields that support the low-voltage reset and low- or high-voltage detect functions. When PMC is in LPM, the low- or high-voltage detect systems are disabled.

NOTE

For all flags that are not affected by reset (POR flag, all LVR flags, all GNG flags), if a reset occurs while trying to clear the flags (by writing 1), the flag value is not defined appropriately. In this case, you must clear the flag again after exiting from reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PORF	0					GNG11OS...	GNG25OS...	LVR11LPF	LVR11F	LVR25LPF	LVR25F	0		LVRALPF	LVRAF
W	W1C						W1C	W1C	W1C	W1C	W1C	W1C			W1C	W1C
Reset	u	0	0	0	0	0	u	u	u	u	u	u	0	0	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0			LVD5AS	HVD11S	HVD25S	0	HVDAS	0			LVD5AF	HVD11F	HVD25F	0	HVDAF
W												W1C	W1C	W1C		W1C
Reset	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 PORF	POR Flag Indicates that a power-on reset event has occurred. Other reset sources have no effect.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field behaves differently for register reads and writes. When reading 0b - Event did not occur 1b - Event occurred When writing 0b - No effect 1b - Clear the flag
30-26 —	Reserved
25 GNG11OSCF	Go/No Go Detect Flag On OSC Part Of V11 Power Domain Indicates that the go/no go sensor has detected a low voltage in the V11 power domain in FPM. This applies only to the part of the power domain that supplies the 1.1 V clocking modules (for example, PLL). Other reset sources have no effect. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Event did not occur 1b - Event occurred When writing 0b - No effect 1b - Clear the flag
24 GNG25OSCF	Go/No Go Detect Flag On OSC Part of V25 Power Domain Indicates that the go/no go sensor has detected a low voltage in the V25 power domain in FPM. This applies only to the part of the power domain that supplies the 2.5 V clocking modules (for example, XOSC and IRC). Other reset sources have no effect. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Event did not occur 1b - Event occurred When writing 0b - No effect

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Clear the flag
23 LVR11LPF	LVR11LP Flag On V11 Power Domain Indicates that a low-voltage reset event has occurred in the 1.1 V V11 power domain (FPM or LPM).Other reset sources have no effect. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Event did not occur 1b - Event occurred When writing 0b - No effect 1b - Clear the flag
22 LVR11F	LVR11 Flag On V11 Power Domain In FPM Indicates that a low-voltage reset event has occurred in the 1.1 V V11 power domain in FPM. Other reset sources have no effect. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Event did not occur 1b - Event occurred When writing 0b - No effect 1b - Clear the flag
21 LVR25LPF	LVR25LP Flag On V25 Power Domain Indicates that a low-voltage reset event has occurred in the V25 power domain (FPM or LPM). Other reset sources have no effect. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Event did not occur 1b - Event occurred When writing

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No effect 1b - Clear the flag
20 LVR25F	LVR25 Flag On V25 Power Domain In FPM Indicates that a low-voltage reset event has occurred in the V25 power domain in FPM. Other reset sources have no effect. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Event did not occur 1b - Event occurred When writing 0b - No effect 1b - Clear the flag
19-18 —	Reserved
17 LVRALPF	LVRALP Flag On VDD_HV_A Power Domain Indicates that a low-voltage reset event has occurred in the VDD_HV_A power domain (FPM or LPM). Other reset sources have no effect. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Event did not occur 1b - Event occurred When writing 0b - No effect 1b - Clear the flag
16 LVRAF	LVRA Flag On VDD_HV_A Power Domain In FPM Indicates that a low-voltage reset event has occurred in the VDD_HV_A power domain in FPM. Other reset sources have no effect. **NOTE** This field behaves differently for register reads and writes. When reading

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Event did not occur 1b - Event occurred When writing 0b - No effect 1b - Clear the flag
15-13 —	Reserved
12 LVD5AS	LVD5A Status On VDD_HV_A Power Domain In FPM Indicates whether the voltage on VDD_HV_A is above or below the low-voltage detect threshold. This monitor reflects the status of the 5 V low-voltage detect, LVD5A, and indicates if the voltage is below a certain threshold, which is set slightly below 4.5 V (see the chip data sheet for the exact value). The feature is only available in FPM and is disabled in LPM. After a reset or wake-up from LPM, you must clear the LVD5AF flag and check LVD5AS to determine the voltage level on the VDD_HV_A supply. 0b - Above 1b - Below
11 HVD11S	HVD11 Status On V11 Power Domain In FPM Indicates whether the voltage on V11 is above or below the high-voltage detect threshold. This field reflects the status of the high-voltage detect, HVD11, on the V11 power domain. This feature is only available in FPM and is disabled in LPM. 0b - Voltage is below threshold or chip is in LPM 1b - Voltage is above threshold and chip is in FPM
10 HVD25S	HVD25 Status On V25 Power Domain In FPM Indicates whether the voltage on V25 is above or below the high-voltage detect threshold. This field reflects the status of the high-voltage detect, HVD25, on the V25 power domain. The feature is only available in FPM and is disabled in LPM. 0b - Voltage is below threshold or chip is in LPM 1b - Voltage is above threshold and chip is in FPM
9 —	Reserved
8 HVDAS	HVDA Status On VDD_HV_A Power Domain In FPM Indicates whether the voltage on VDD_HV_A is above or below the high-voltage detect threshold. This field reflects the status of the high-voltage detect, HVDA, on the V25 power domain. The feature is only available in FPM and is disabled in LPM. 0b - Voltage is below threshold or chip is in LPM 1b - Voltage is above threshold and chip is in FPM

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-5 —	Reserved
4 LVD5AF	LVD5A Flag On VDD_HV_A Power Domain In FPM Indicates whether LVD5AS has changed. When LVD5AS changes, PMC changes LVD5AF to 1. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Did not change 1b - Changed When writing 0b - No effect 1b - Clear the flag
3 HVD11F	HVD11 Flag On V11 Power Domain In FPM Indicates whether HVD11S has changed. When HVD11S changes, PMC changes HVD11F to 1. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Did not change 1b - Changed When writing 0b - No effect 1b - Clear the flag
2 HVD25F	HVD25 Flag On V25 Power Domain In FPM Indicates whether HVD25S has changed. When HVD25S changes, PMC changes HVD25F to 1. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Did not change 1b - Changed When writing 0b - No effect 1b - Clear the flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 —	Reserved
0 HVDAF	HVDA Flag On VDD_HV_A Power Domain In FPM Indicates whether HVDAS has changed. When HVDAS changes, PMC changes HVDAF to 1. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Did not change 1b - Changed When writing 0b - No effect 1b - Clear the flag

43.4.3 PMC Configuration (CONFIG)

Offset

Register	Offset
CONFIG	4h

Function
Configures PMC.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0						LVDIE	HVDIE	0				LPM25EN	FASTREC	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-10 —	Reserved
9 LVDIE	Low Voltage Detect Interrupt Enable Enables hardware interrupt requests if LVSC[LVD5AF] = 1. You must disable the LVD interrupt before entering LPM. 0b - LVD hardware interrupt is disabled (use polling) 1b - Request an LVD hardware interrupt when LVDA5F = 1
8 HVDIE	High Voltage Detect Interrupt Enable Enables hardware interrupt requests if any of the following flags in Low Voltage Status And Control (LVSC) are set: • HVDAF • HVDBF • HVD25F • HVD11F 0b - HVD hardware interrupt is disabled (use polling) 1b - Request an HVD hardware interrupt when HVDAF=1, HVDBF=1, HVD25F=1, or HVD11F=1
7-4 —	Reserved
3 LPM25EN	V25 Power Domain Enable During LPM Controls whether the V25 regulator and low-voltage reset detection (LVR25LP) are enabled or disabled in LPM 0b - Disabled 1b - Enabled
2 FASTREC	Fast Recovery From LPM Enable Controls the recovery time from LPM to FPM. At recovery from LPM, all the tank capacitors from the secondary supplies must be recharged. This causes a high-current demand, which the supply driving the VDD_HV_A primary power domain might not meet. When you select the fast recovery time, the current for recharging is approximately three times higher than that for normal recovery time. You must determine whether this current is sufficient to start up from LPM in time, using these criteria: • Drive capability of the external VDD_HV_A regulator • Size of the tank caps on the secondary supply pin • Selected recovery time

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Normal 1b - Fast
1-0 —	Reserved

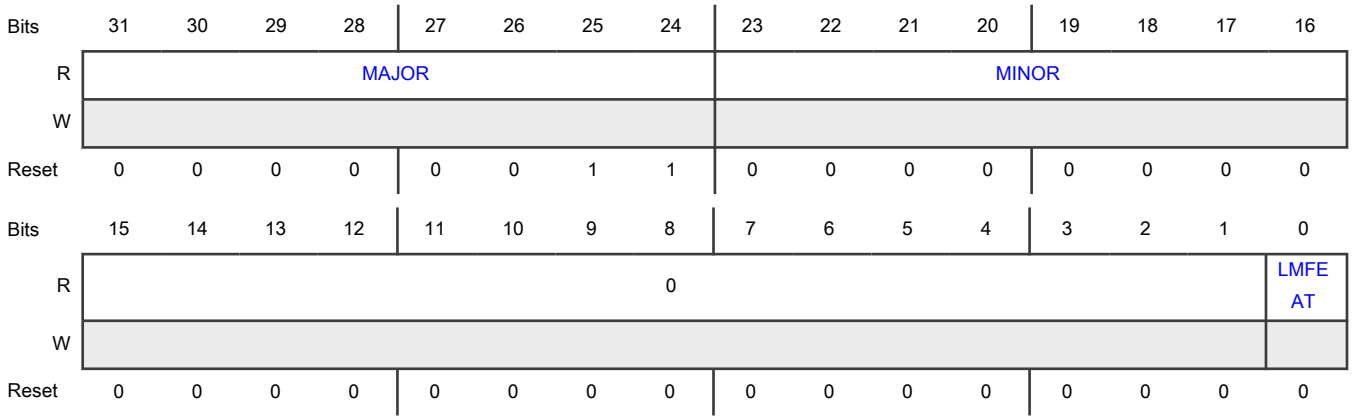
43.4.4 Version ID (VERID)

Offset

Register	Offset
VERID	Ch

Function
Records the specific PMC version in the chip.

Diagram



Fields

Field	Function
31-24 MAJOR	Major Version Number Indicates the major version number of the PMC design.
23-16 MINOR	Minor Version Number Indicates the minor version number of the PMC design.
15-1	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
0 LMFEAT	Last-Mile Regulator Feature Indicates whether the last-mile regulator (1.5 V to 1.1 V) is available. 0b - Not available 1b - Available

43.5 Glossary

FPM	Full Performance mode
HVD	High voltage detect
LPM	Low Power mode
LVD	Low voltage detect
LVR	Low voltage reset
POR	Power-on reset

Chapter 44
Power Management Controller (PMC for S32K358, S32K356, S32K328, S32K338, and S32K348)

44.1 Overview

PMC is the power management controller for the S32K3 family of microcontrollers. It provides multiple power options to allow you to optimize power consumption for the level of functionality needed. It includes internal voltage regulators, POR, and the integrated low/high voltage detect system with reset (brown-out) capability. The voltage regulator requires a 3.3 V or 5 V input to generate all the required secondary supplies.

44.1.1 Block diagram

The following figure shows the block diagram for this module.

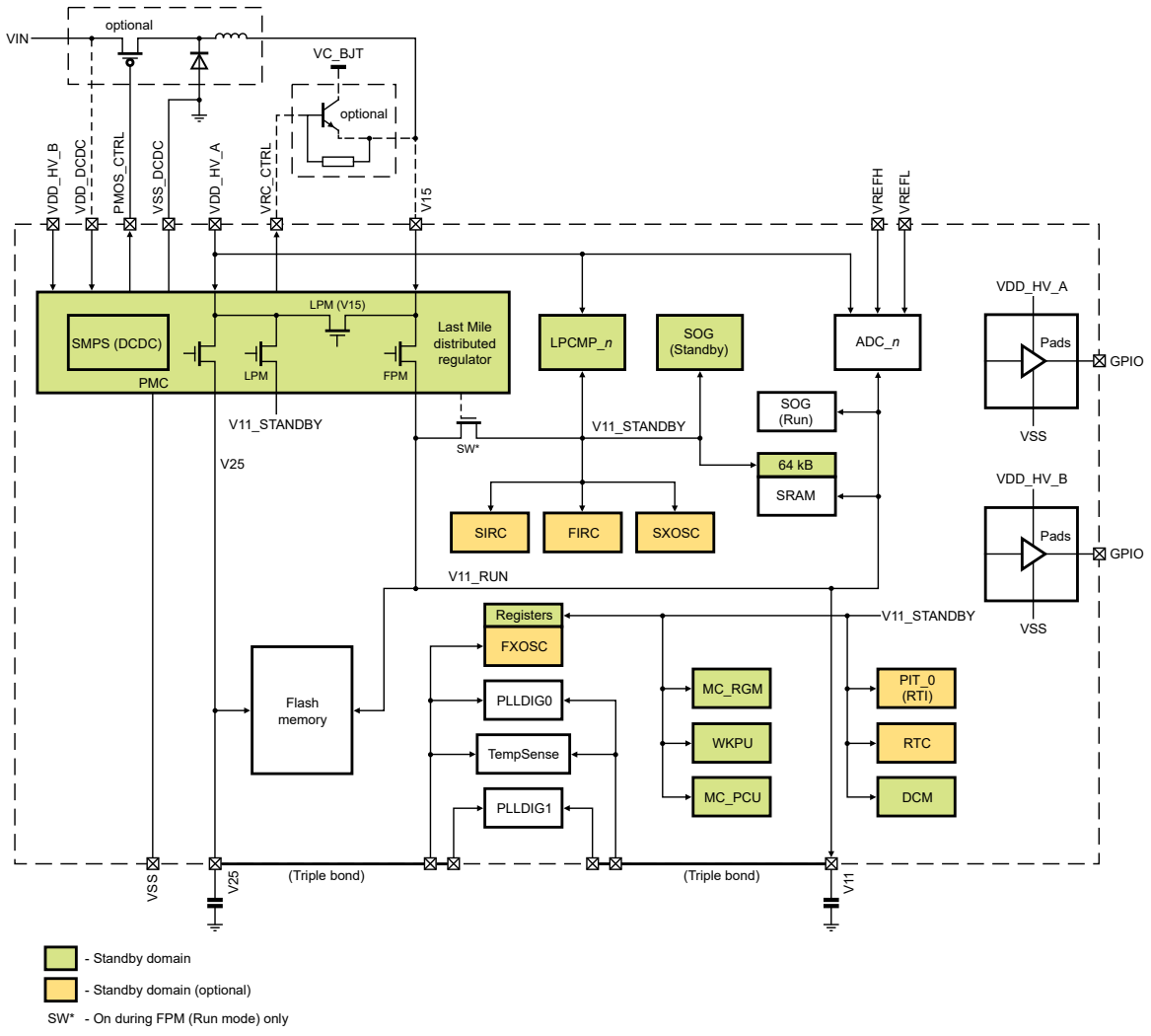


Figure 186. PMC block diagram

44.1.2 Features

PMC includes the following features:

- Combination of internal and external voltage regulator options, offering RUN and Standby modes
- Active POR providing brown-out detect
- LVR for all system-relevant power domains
- LVD and HVD as indication for software

44.2 Functional description

The following sections describe functional details of the PMC.

44.2.1 Modes of operation

PMC provides two basic modes of operation for the voltage regulators and monitors:

- FPM, which is used on chip-level in RUN modes: For high-current consumption
- LPM, which is used on chip-level for Standby modes: For low-current consumption

44.2.2 Reset

The POR and all LVRs are combined into one single MCU POR.

After an MCU POR event, it can be determined which power domain caused it, by reading in the PMC_LVSC register, the POR flag, and LVR flags.

After an initial power ramp up of the MCU in the PMC_LVSC register, the POR flag and the LVR flags are all set to 1. The Go/Nogo flags have an arbitrary value.

NOTE

After an initial power ramp up, all flags in the LVSC register must be cleared (by writing 0xFFFFFFFF to the LVSC register).

Because the flags are sticky bits, it is required to clear them before usage. So, in case of an unexpected MCU POR, the source of the problem can be tracked and debugged by reading the flags in the LVSC register.

44.2.3 Interrupts

PMC includes two interrupt sources:

- HVD interrupt: It combines all HVD monitors into one interrupt source. Interrupt enable is the HVDIE field in the CONFIG register.
- LVD interrupt: This is the interrupt for the LVD5A monitor. Interrupt enable is the LVDIE field in the CONFIG register.

See the PMC Configuration Register (CONFIG) and Low Voltage Status and Control Register (LVSC) registers for details.

44.3 Signals

This table describes the PMC module signals.

Table 255. Signal Description

Signal	I/O	Description
VDD_HV_A	Supply input	This is the primary high-voltage supply input to PMC. VDD_HV_A is used for the PMC internal precision references. After the VDD_HV_A

Table continues on the next page...

Table 255. Signal Description (continued)

Signal	I/O	Description
		domain is powered up, it must be kept powered at all times of operation (FPM and LPM).
VDD_HV_B	Supply input	This is the secondary high-voltage supply input supervised by the PMC. After the VDD_HV_B domain is powered up, it must be kept powered at all times of operation (FPM and LPM).
V25	Supply output	V25 power supply domain is driven by a fully integrated low-dropout linear voltage regulator. It supplies the Flash memory and (via a double bond) the clock modules.
V15	Supply input	This is the high-current input for core/logic supply that can be fed by an external BJT or the SMPS or an external source.
VRC_CTRL	Output	VRC_CTRL connects to the base of external BJT, if this option is used to generate V15.
V11	Supply output	V11 is the core/logic supply. It is driven by a fully integrated low-dropout linear voltage regulator.
VSS	Ground	VSS must be grounded. All VSS Pins need to be externally connected to the same ground node.
VDD_DCDC	Supply input	This is the power supply domain for the SMPS gate driver and must be shorted with the source voltage of the external power FET. An off-chip decoupling capacitor between VDD_DCDC and ground is required. See Datasheet for the recommended value. **NOTE** If SMPS (DCDC) mode is used and the PMC is in low power mode, it must be ensured that SMPS supply input VDD_DCDC is fully operational and loadable before wake-up from low power mode. Otherwise a low voltage reset might occur on wake-up.
VSS_DCDC	Ground	This is the power ground for the SMPS gate driver and must be shorted with the main ground node and the ground node of the external Schottky diode.
PMOS_CTRL	Output	This is the gate driver output for the external power FET for the SMPS regulator.

44.4 PMC register descriptions

44.4.1 PMC memory map

This section includes the PMC module memory map and detailed descriptions for all the registers.

PMC_S32K358 base address: 402E_8000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Low Voltage Status and Control Register (LVSC)	32	RW	See section
4h	PMC Configuration Register (CONFIG)	32	RW	0000_0083h
8h	SMPS Configuration Register (SMPSCONFIG)	32	RW	See section
Ch	Version ID register (VERID)	32	R	0400_0001h

44.4.2 Low Voltage Status and Control Register (LVSC)

Offset

Register	Offset
LVSC	0h

Function

This register contains status and control bits to support the low-voltage reset and low- or high-voltage detect function. When the PMC is in LPM, the low- or high-voltage detect systems are disabled.

NOTE

For all flags that are not affected by reset (POR flag, all LVR flags, all GNG flags), in case a reset occurs at the same time while trying to clear the flags (by writing 1), the flag value is not defined appropriately. In this case, you need to clear the flag again after exit from reset.

NOTE

For the GNG flags, in case a GNG event occurs at the same time as the GNG flag is read, the flag value might not be defined appropriately. In such case, you need to read the flag twice and consider the value only if it both read values match.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PORF	0			GNG11OS...	GNG25OS...	GNG11OS...	GNG25OS...	LVR11LPF	LVR11F	LVR25LPF	LVR25F	LVRBLPF	LVRBF	LVRALPF	LVRAF
W	W1C				W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	u	0	0	0	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	HVD15S	Reserved	LVD5AS	HVD11S	HVD25S	HVDBS	HVDAS	0	HVD15F	0	LVD5AF	HVD11F	HVD25F	HVDBF	HVDAF
W										W1C		W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 PORF	POR flag Indicates that a power-on reset event has occurred. Writing 1 to this field clears it, and other reset sources have no effect. 0b - No power-on reset event has occurred 1b - Power-on reset event has occurred
30-28 —	Reserved
27 GNG11OSC2F	Go/NoGo detect flag on 2nd PLL part of V11 domain Indicates that the Go/NoGo sensor has detected a low voltage on the V11 domain in FPM. This applies to only that part of the domain which supplies the 2nd PLL. Writing 1 to the field clears it, and other reset sources have no effect. 0b - No event has occurred 1b - NoGo event has occurred
26 GNG25OSC2F	GO/NoGo detect flag on 2nd PLL part of V25 domain Indicates that the Go/NoGo sensor has detected a low voltage on the V25 domain in FPM. This applies to only that part of the domain which supplies the 2nd PLL. Writing 1 to the field clears it, and other reset sources have no effect. 0b - No event has occurred 1b - NoGo event has occurred
25 GNG11OSCF	Go/NoGo detect flag on Osc part of V11 domain Indicates that the Go/NoGo sensor has detected a low voltage on the V11 domain in FPM. This applies to only that part of the domain which supplies the 1.1V clocking modules (e.g. PLL, IRC). Writing 1 to the field clears it, and other reset sources have no effect. 0b - No event has occurred 1b - NoGo event has occurred
24 GNG25OSCF	GO/NoGo detect flag on Osc part of V25 domain Indicates that the Go/NoGo sensor has detected a low voltage on the V25 domain in FPM. This applies to only that part of the domain which supplies the 2.5V clocking modules (e.g. PLL, XOSC and IRC). Writing 1 to the field clears it, and other reset sources have no effect. 0b - No event has occurred 1b - NoGo event has occurred
23 LVR11LPF	LVR11LP flag on V11 domain Indicates that a low-voltage reset event has occurred on the 1.1V V11 power domain (FPM or LPM). Writing 1 to the field clears it, and other reset sources have no effect.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No low-voltage reset event has occurred 1b - Low-voltage reset event has occurred
22 LVR11F	LVR11 flag on V11 domain in FPM Indicates that a low-voltage reset event has occurred on the 1.1V V11 power domain in the FPM. Writing 1 to this field clears it, and other reset sources have no effect. 0b - No low-voltage reset event has occurred 1b - Low-voltage reset event has occurred
21 LVR25LPF	LVR25LP flag on V25 domain Indicates that a low-voltage reset event has occurred on the V25 power domain (FPM or LPM). Writing 1 to this field clears it, and other reset sources have no effect. 0b - No low-voltage reset event has occurred 1b - Low-voltage reset event has occurred
20 LVR25F	LVR25 flag on V25 domain in FPM Indicates that a low-voltage reset event has occurred on the V25 power domain in FPM. Writing 1 to this field clears it, and other reset sources have no effect. 0b - No low-voltage reset event has occurred 1b - Low-voltage reset event has occurred
19 LVRBLPF	LVRBLP flag on VDD_HV_B domain Indicates that a low-voltage reset event has occurred on the VDD_HV_B power domain (FPM or LPM). Writing 1 to this field clears it, and other reset sources have no effect. 0b - No low-voltage reset event has occurred 1b - Low-voltage reset event has occurred
18 LVRBF	LVRB flag on VDD_HV_B domain in FPM Indicates that a low-voltage reset event has occurred on the VDD_HV_B power domain in FPM. Writing 1 to this field clears it, and other reset sources have no effect. 0b - No low-voltage reset event has occurred 1b - Low-voltage reset event has occurred
17 LVRALPF	LVRALP flag on VDD_HV_A domain Indicates that a low-voltage reset event has occurred on the VDD_HV_A power domain (FPM or LPM). Writing 1 to this field clears it, and other reset sources have no effect. 0b - No low-voltage reset event has occurred 1b - Low-voltage reset event has occurred
16	LVRA flag on VDD_HV_A domain in FPM

Table continues on the next page...

Table continued from the previous page...

Field	Function
LVRAF	Indicates that a low-voltage reset event has occurred on the VDD_HV_A power domain in FPM. Writing 1 to this field clears it, and other reset sources have no effect. 0b - No low-voltage reset event has occurred 1b - Low-voltage reset event has occurred
15 —	Reserved
14 HVD15S	HVD15 status on V15 domain in FPM Shows the status of the high-voltage detect, HVD15, on the V15 power domain. This feature is only available in FPM and disabled in LPM. 0b - Voltage on V15 is below high-voltage detect threshold or LPM. 1b - Voltage on V15 is above high-voltage detect threshold and FPM.
13 —	Reserved, ignore value Reserved, ignore value
12 LVD5AS	LVD5A status on VDD_HV_A domain in FPM Shows the status of the 5V low-voltage detect, LVD5A, on the VDD_HV_A power domain. This monitor indicates if the voltage is below a certain threshold, which is set slightly below 4.5V (see Datasheet for exact value). The feature is only available in FPM and disabled in LPM. After a reset or wakeup from LPM, the software should clear the LVD5AF flag and check the status bit LVD5AS to determine voltage level on VDD_HV_A supply. 0b - Voltage on VDD_HV_A is above low-voltage detect threshold 1b - Voltage on VDD_HV_A is below low-voltage detect threshold
11 HVD11S	HVD11 status on V11 domain in FPM Shows the status of the high-voltage detect, HVD11, on the V11 power domain. This feature is only available in FPM and disabled in LPM. 0b - Voltage on V11 is below high-voltage detect threshold or LPM. 1b - Voltage on V11 is above high-voltage detect threshold and FPM.
10 HVD25S	HVD25 status on V25 domain in FPM Shows the status of the high-voltage detect, HVD25, on the V25 power domain. The feature is only available in FPM and disabled in LPM. 0b - Voltage on V25 is below high-voltage detect threshold or LPM. 1b - Voltage on V25 is above high-voltage detect threshold and FPM.
9 HVDBS	HVDB status on VDD_HV_B domain in FPM Shows the status of the high-voltage detect, HVDB, on the VDD_HV_B power domain. The feature is only available in FPM and disabled in LPM.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Voltage on VDD_HV_B is below high-voltage detect threshold or LPM. 1b - Voltage on VDD_HV_B is above high-voltage detect threshold and FPM.
8 HVDAS	HVDA status on VDD_HV_A domain in FPM Shows the status of the high-voltage detect HVDA on the VDD_HV_A power domain. The feature is only available in FPM and disabled in LPM. 0b - Voltage on VDD_HV_A is below high-voltage detect threshold or LPM. 1b - Voltage on VDD_HV_A is above high-voltage detect threshold and FPM.
7 —	Reserved
6 HVD15F	HVD15 flag on V15 domain in FPM PMC writes 1 to this field when the HVD15S status field changes. To clear HVD15F, write 1 to it. If enabled, HVD15F causes an interrupt request. 0b - HVD15S has not changed. 1b - HVD15S has changed.
5 —	Reserved
4 LVD5AF	LVD5A flag on VDD_HV_A domain in FPM PMC writes 1 to this field when LVD5AS status field changes. To clear LVD5AF, write 1 to it. If enabled, LVD5AF causes an interrupt request. 0b - LVD5AS has not changed. 1b - LVD5AS has changed.
3 HVD11F	HVD11 flag on V11 domain in FPM PMC writes 1 to this field when the HVD11S status field changes. To clear HVD11F, write 1 to it. If enabled, HVD11F causes an interrupt request. 0b - HVD11S has not changed. 1b - HVD11S has changed.
2 HVD25F	HVD25 flag on V25 domain in FPM PMC writes 1 to the HVD25F field when HVD25S status field changes. To clear HVD25F, write 1 to it. If enabled, HVD25F causes an interrupt request. 0b - HVD25S has not changed. 1b - HVD25S has changed.
1	HVDB flag on VDD_HV_B domain in FPM

Table continues on the next page...

Table continued from the previous page...

Field	Function
HVDBF	PMC writes 1 to the HVDBF field when HVDBS status field changes. To clear HVDBF, write 1 to it. If enabled, HVDBF causes an interrupt request. 0b - HVDBS has not changed. 1b - HVDBS has changed.
0 HVDAF	HVDA flag on VDD_HV_A domain in FPM PMC writes 1 to the HVDAF field when HVDAS status field changes. To clear HVDAF, write 1 to it. If enabled, HVDAF causes an interrupt request. 0b - HVDAS has not changed. 1b - HVDAS has changed.

44.4.3 PMC Configuration Register (CONFIG)

Offset

Register	Offset
CONFIG	4h

Function
This register configures the various PMC options.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0						LVDIE	HVDIE	LMSMPSEN	0		LVRBLPEN	LPM25EN	FASTREC	LMBCTLEN	LMEN
W									0						0	
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	1

Fields

Field	Function
31-10 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
9 LVDIE	Low voltage detect interrupt enable Enables hardware interrupt requests if LVD5AF is set. LVD interrupt must be disabled before going into LPM. 0b - LVD hardware interrupt is disabled (use polling). 1b - Request an LVD hardware interrupt when LVD5AF=1.
8 HVDIE	High voltage detect interrupt enable Enables hardware interrupt requests if any of the following flags is set: HVDAF, HVDBF, HVD25F, HVD11F. 0b - HVD hardware interrupt is disabled (use polling). 1b - Request an HVD hardware interrupt when HVDAF=1, HVDBF=1, HVD25F=1, or HVD11F=1.
7 LMSMPSEN	V15 Switched-mode power supply enable bit Enables the SMPS regulator (Switched-mode power supply), which regulates from VDD_DCDC down to 1.5V on V15 pin to supply the Last Mile regulator. As LMSMPSEN=1 after startup or reset, in case SMPS is not used, software must write LMBSMPSEN=0 after each reset to turn the feature off. Once LMSMPSEN=0 it can no longer be written 1 until next reset. 0b - Switched-mode power supply (SMPS) for V15 disabled 1b - Switched-mode power supply (SMPS) for V15 enabled
6-5 —	Reserved
4 LVRBLPEN	LVRBLP enable bit during LPM Controls whether the low-voltage reset detection (LVRBLP) on the VDD_HV_B power domain is active or inactive in LPM 0b - Low-voltage reset detection is disabled in LPM. 1b - Low-voltage reset detection is enabled in LPM.
3 LPM25EN	V25 domain enable bit during LPM Enables a low power V25 regulator and the low-voltage reset detection (LVR25LP) in LPM. In FPM this low power regulator is always off. 0b - V25 regulator and LVR25LP are disabled in LPM. 1b - V25 regulator and LVR25LP are enabled in LPM.
2 FASTREC	Fast recovery from LPM enable bit Controls the recovery time from LPM to FPM. This causes a higher current demand on V15, and this translates to a higher current demand on the primary side of the V15 supply. 0b - Normal recovery time from LPM 1b - Fast recovery time from LPM

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 LMBCTLEN	V15 external BJT enable bit This field must be set to 1 if external BJT between VDD_HV_A and V15 is used on the PCB. The base of this BJT must be connected to the VRC_CTRL pin and is controlled by the PMC to regulate a voltage of 1.5V on V15 pin to supply the Last Mile Regulator. As LMBCTLEN=1 after startup or reset, in case BJT is not used, software must write LMBCTLEN=0 after each reset to turn the feature off. Once LMBCTLEN=0 it can no longer be written 1 until next reset. 0b - External BCTL regulator for V15 disabled 1b - External BCTL regulator for V15 enabled
0 LMEN	Last Mile regulator enable bit Enables the Last Mile regulator, which regulates an external 1.5V voltage on V15 down to the core and logic supply (V11 power domain), which is typically 1.1V. As the Last Mile regulator is always enabled on this device, LMEN bit is read-only and always reads 1.

44.4.4 SMPS Configuration Register (SMPSCONFIG)

Offset

Register	Offset
SMPSCONFIG	8h

Function

This register configures the various options of the Switched-mode power supply (SMPS) generating the V15 (1.5V input supply to the last mile regulator).

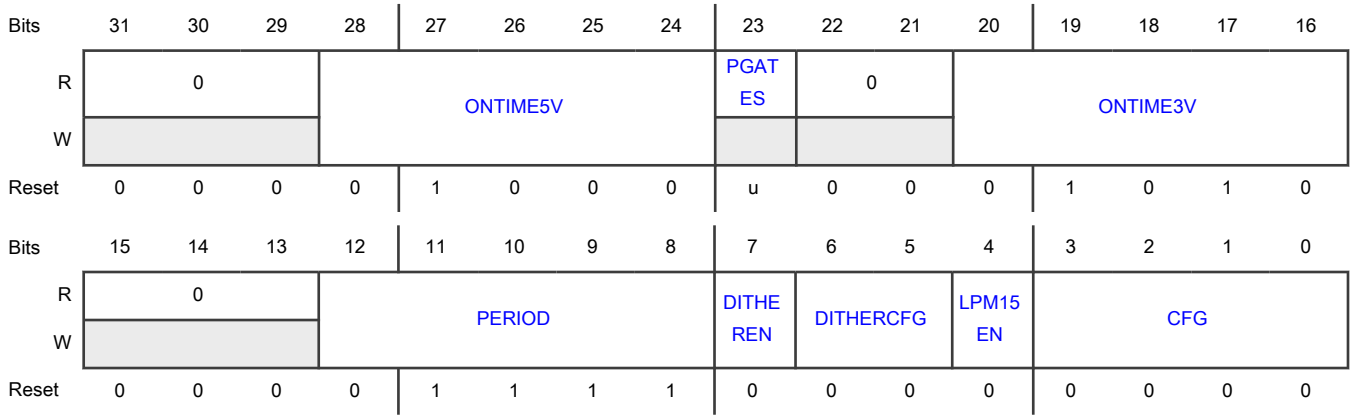
Table 256. SMPS configurations for driving PMOS_CTRL pin

CFG[3:0] bits	PMOS_CTRL Frequency	PMOS_CTRL Duty cycle 3V	PMOS_CTRL Duty cycle 5V
0000	470 kHz	58.8%	47.1%
0001	470 kHz	64.7%	52.9%
0010	470 kHz	52.9%	41.2%
0011	533 kHz	60%	46.7%
0100	533 kHz	66.7%	53.3%
0101	533 kHz	53.3%	40%
0110	421 kHz	57.9%	47.4%
0111	421 kHz	63.2%	52.6%
1000	421 kHz	52.6%	42.1%
1001 to 1110	Reserved	Reserved	Reserved
1111	8 MHz / (PERIOD[4:0]+1)	ONTIME3V[4:0] / (PERIOD[4:0]+1)	ONTIME5V[4:0] / (PERIOD[4:0]+1)

Table 257. 8 MHz IRC Dither configurations

DITHERCFG[1:0] bits	Dither Amplitude	Dither Frequency
00	+/- 0.4 MHz	400 kHz
01	+/- 0.6 MHz	286 kHz
10	+/- 0.8 MHz	222 kHz
11	+/- 1 MHz	182 kHz

Diagram



Fields

Field	Function
31-29 —	Reserved
28-24 ONTIME5V	SMPS Duty cycle for 5V range These bits determine the duty cycle of the output signal at PMOS_CTRL pin. This duty cycle is applied while VDD_DCDC is greater or equal than 4V. It calculates as follows: Duty_pmos_ctrl = ONTIME3V[4:0] / (PERIOD[4:0] + 1)
23 PGATES	PMOS_CTRL Status Bit This read-only bit reflects the status of the PMOS_CTRL pin. 0b - PMOS_CTRL pin driven to VSS_DCDC (which is connect to GND) 1b - PMOS_CTRL pin driven to VDD_DCDC voltage
22-21 —	Reserved
20-16 ONTIME3V	SMPS Duty cycle for 3V range These bits determine the duty cycle of the output signal at PMOS_CTRL pin. This duty cycle is applied while VDD_DCDC is smaller than 4V.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	It calculates as follows: Duty_pmos_ctrl = ONTIME3V[4:0] / (PERIOD[4:0] + 1)
15-13 —	Reserved
12-8 PERIOD	SMPS Period These bits determine the frequency of the output signal at PMOS_CTRL pin. The frequency calculates as follows: Freq_pmos_ctrl = 8 MHz / (PERIOD[4:0] + 1)
7 DITHEREN	IRC Dither Enable This bit enables dithering of the 8MHz IRC. The amplitude and frequency of the dithering is determined by the DITHERCFG[1:0] bits. 0b - 8MHz IRC dithering disabled 1b - 8MHz IRC dithering enabled
6-5 DITHERCFG	IRC Dither Configuration Selects the IRC dithering amplitude and frequency. This feature spreads the frequency spectrum of driving the PMOS_CTRL pin to reduce radiated emission. To change the value: 1. Write 0 to DITHEREN. 2. Write the dither configuration (DITHERCFG[1:0]). 3. Write 1 to DITHEREN. See Table 257 for configuration options.
4 LPM15EN	V15 domain enable bit during LPM Enables the V15 in LPM. This is useful for fast recovery to FPM when using the SMPS mode to generate the V15. This feature is only active in LPM, in FPM it is turned off. For this purpose there is a small LDO inside the PMC that regulates the V15 to target value during LPM. **NOTE** LPM15EN bit is only allowed to use, when using the SMPS to generate the V15. For other options to generate the V15 it is forbidden and software should always write 0 to LPM15EN. 0b - V15 not kept on target in LPM 1b - V15 kept on target in LPM
3-0 CFG	SMPS configuration select These bits select a configuration for frequency and duty cycle (ontime) of the Switched-mode power supply driving via PMOS_CTRL the external power FET according to Table 256.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE As seen in the table CFG[3:0] = 1111 allows for free customization of the PERIOD[4:0], ONTIME5V[4:0] and ONTIME3V[3:0] bits. If choosing this option consider carefully the value of these bits. To ensure the proper function of the SMPS, and to keep the EMI in check, the switching frequency and duty cycle must be programmed within a reasonable range.

44.4.5 Version ID register (VERID)

Offset

Register	Offset
VERID	Ch

Function

This register returns the major and minor version numbers of hardware implementation.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MAJOR								MINOR							
W																
Reset	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															LMFEAT
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Fields

Field	Function
31-24 MAJOR	Major version number Returns the version number for the specification
23-16 MINOR	Minor version number Returns the version number for the hardware implementation
15-1	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
0 LMFEAT	Last Mile Regulator Feature This read-only field shows if the Last Mile regulator feature is available. 0b - No Last Mile regulator 1b - Last Mile regulator (1.5V to 1.1V) is available

44.5 Glossary

FPM	Full Performance mode
HVD	High voltage detect
IRC	Internal RC oscillator
LM	Last mile regulator
LPM	Low Performance mode
LVD	Low voltage detect
LVR	Low voltage reset
NVM	Nonvolatile memory
POR	Power on reset
XOSC	External crystal oscillator
SMPS	Switched-mode power supply

Chapter 45
Power Management Controller (PMC for S32K388 and S32K389)

45.1 Overview

PMC is the power management controller for the S32K3 family of microcontrollers. It provides multiple power options to allow you to optimize power consumption for the level of functionality needed. It includes internal voltage regulators, POR, and the integrated low/high voltage detector system with reset (brown-out) capability. The voltage regulator requires a 3.3 V or 5 V input to generate all the required secondary supplies.

45.1.1 Block diagram

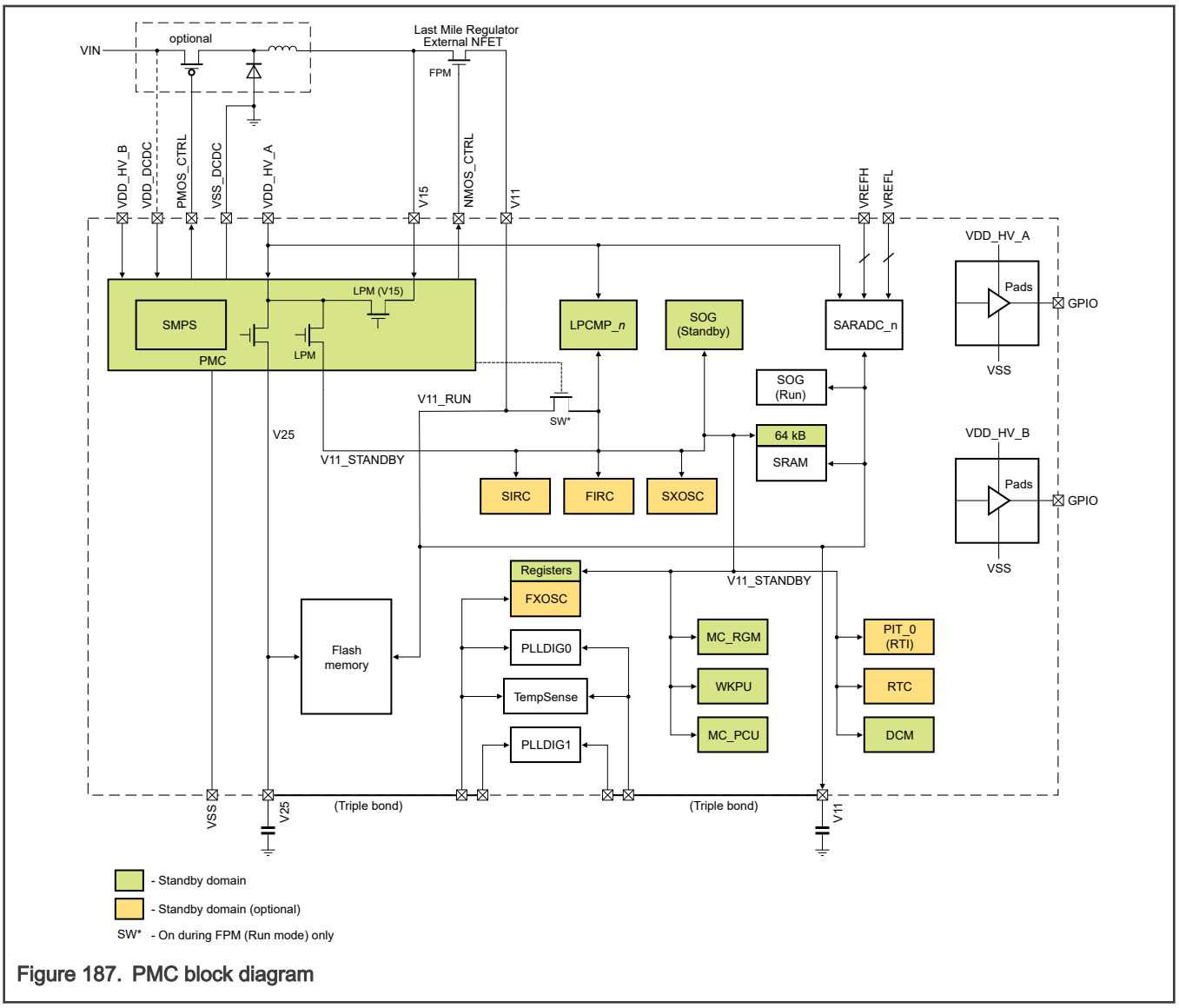


Figure 187. PMC block diagram

45.1.2 Features

- Combination of internal and external voltage regulator options, offering Run and Standby modes

- Active POR providing brown-out detect
- LVR for all system-relevant power domains
- LVD and HVD as indication for software
- Two bandgaps to enable independent supply generation and monitoring
- Two independent LVR monitors for each safety-relevant power supply domain

45.2 Functional description

45.2.1 Modes of operation

PMC provides two basic modes of operation for voltage regulators and monitors:

- FPM, which is used on chip-level in Run modes (for high-current consumption)
- LPM, which is used on chip-level for Standby modes (for low-current consumption)

45.2.2 Reset

The POR and all LVRs are combined into one single MCU POR.

After an MCU POR event, you can determine which power domain caused the event by reading LVSC[PORF] and the relevant LVR*x* field (for example, LVSC[LVR11LPF]).

After an initial power ramp-up of the MCU, PMC writes 1 to LVSC[PORF] and all LVSC[LVR*x*] fields. The Go/Nogo fields have an arbitrary value.

NOTE

After an initial power ramp-up, you must write FFFFFFFFh to LVSC to clear all flags.

Because the flags are sticky bits, you must clear them before using them. Doing so makes it possible to track and debug the source of the problem by reading the Low Voltage Status and Control (LVSC) fields, if there is an unexpected MCU POR.

45.2.3 Interrupts

PMC includes two interrupt sources:

- HVD: Combines all HVD monitors into a single interrupt source. Enable this interrupt with CONFIG[HVDIE].
- LVD: uses the LVD5A as interrupt source. Enable this interrupt with CONFIG[LVDIE].

See PMC Configuration (CONFIG) and Low Voltage Status and Control (LVSC) for details.

45.3 Signals

Table 258. Signal description

Signal	I/O	Description
VDD_HV_A	Supply input	VDD_HV_A is the primary high-voltage supply input to PMC. It is used for the PMC internal precision references. After the VDD_HV_A domain is powered up, you must keep it powered at all times of operation (FPM and LPM).
VDD_HV_B	Supply input	VDD_HV_B is the secondary high-voltage supply input that PMC supervises. After the VDD_HV_B domain is powered up, you must keep it powered at all times of operation (FPM and LPM).

Table continues on the next page...

Table 258. Signal description (continued)

Signal	I/O	Description
V25	Supply output	A fully integrated low-dropout linear voltage regulator drives this V25 power supply domain. It supplies the flash memory and (via a double bond) the clock modules.
V15	Supply input	The V15 supply is generated by the SMPS or by an external source (see Figure 187). PMC uses this pin to measure the voltage level on V15. Optionally, for SMPS, a PMC internal trickle LDO can sustain the V15 supply in LPM.
NMOS_CTRL	Output	NMOS_CTRL connects to the gate of an external NFET, which is used to regulate V15 down to V11 (core voltage). For the 1.1 V core regulator, an external NFET with low-threshold voltage (v_{th} smaller than 1.0 V) is needed, for example, Vishay SQ2310ES. Additionally 1 nF at the gate is required for stability reasons.
V11	Supply input	V11 is the core and logic supply. It is driven by the external NFET which is controlled by a linear regulator inside PMC.
VSS	Ground	VSS must be grounded. All VSS pins need to be externally connected to the same ground node.
VDD_DCDC	Supply input	VDD_DCDC is the power supply domain for the SMPS gate driver. It must be shorted with the source voltage of the external power FET. An off-chip decoupling capacitor between VDD_DCDC and ground is required. See Datasheet for the recommended value. **NOTE** If SMPS (DCDC) mode is used and the PMC is in low power mode, it must be ensured that SMPS supply input VDD_DCDC is fully operational and loadable before wake-up from low power mode. Otherwise a low voltage reset might occur on wake-up.
VSS_DCDC	Ground	VSS_DCDC is the power ground for the SMPS gate driver. It must be shorted with the main ground node and the ground node of the external Schottky diode.
PMOS_CTRL	Output	PMOS_CTRL is the gate driver output for the external power FET for the SMPS regulator.

45.4 PMC register descriptions

45.4.1 PMC memory map

This section includes the PMC module memory map and detailed descriptions for all the registers. Registers marked "Reserved" should not be modified from their reset value by a write access.

PMC base address: 402E_8000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Low Voltage Status and Control (LVSC)	32	RW	See section
4h	PMC Configuration (CONFIG)	32	RW	0000_0081h
8h	SMPS Configuration (SMPSCONFIG)	32	RW	See section
Ch	Version ID (VERID)	32	R	0600_0001h

45.4.2 Low Voltage Status and Control (LVSC)

Offset

Register	Offset
LVSC	0h

Function

Contains status and control fields that support the low-voltage reset and LVD or HVD function. When the PMC is in LPM, the LVD or HVD systems are disabled.

For flags that are not affected by reset (POR flag, all LVR flags, all GNG flags), if a reset occurs at the same time as an attempt to clear the flags (by writing 1), the flag value is not defined appropriately. In this case, you need to clear the flag again after exit from reset.

For the GNG flags, if a GNG event occurs at the same time as the GNG flag is read, the flag value might not be defined appropriately. In this case, you need to read the flag twice and consider the value only if it both read values match.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PORF	Reserved	Reserved	Reserved	GNG11OS...	GNG25OS...	GNG11OS...	GNG25OS...	LVR11LPF	LVR11F	LVR25LPF	LVR25F	LVRBLPF	LVRBF	LVRALPF	LVRAF
W	W1C				W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	HVD15S	Reserved	LVD5AS	HVD11S	HVD25S	HVDBS	HVDAS	Reserved	HVD15F	0	LVD5AF	HVD11F	HVD25F	HVDBF	HVDAF
W										W1C		W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31	POR Flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
PORF	Indicates that a POR event has occurred. Other reset sources have no effect. 0b - No POR event 1b - POR event
30 —	Reserved
29 —	Reserved
28 —	Reserved
27 GNG11OSC2F	Go/NoGo Detect Flag On Second PLL Part Of V11 Domain Indicates that the Go/NoGo sensor has detected a low voltage on the V11 domain in FPM. This applies to only the part of the domain that supplies the second PLL. Other reset sources have no effect. 0b - No event 1b - NoGo event
26 GNG25OSC2F	GO/NoGo Detect Flag On Second PLL Part Of V25 Domain Indicates that the Go/NoGo sensor has detected a low voltage on the V25 domain in FPM. This applies to only the part of the domain that supplies the second PLL. Other reset sources have no effect. 0b - No event 1b - NoGo event
25 GNG11OSCF	Go/NoGo Detect Flag On Osc Part Of V11 Domain Indicates that the Go/NoGo sensor has detected a low voltage on the V11 domain in FPM. This applies to only the part of the domain that supplies the 1.1 V clocking modules (for example, PLL or IRC). Other reset sources have no effect. 0b - No event 1b - NoGo event
24 GNG25OSCF	GO/NoGo Detect Flag On Osc Part Of V25 Domain Indicates that the Go/NoGo sensor has detected a low voltage on the V25 domain in FPM. This applies to only the part of the domain that supplies the 2.5 V clocking modules (for example, PLL, XOSC, and IRC). Other reset sources have no effect. 0b - No event 1b - NoGo event
23 LVR11LPF	LVR11LP Flag On V11 Domain

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Indicates that an LVR event has occurred on the 1.1 V V11 power domain (FPM or LPM). Other reset sources have no effect. 0b - No event 1b - Event
22 LVR11F	LVR11 Flag On V11 Domain In FPM Indicates that an LVR event has occurred on the 1.1 V V11 power domain in the FPM. Other reset sources have no effect. 0b - No event 1b - Event
21 LVR25LPF	LVR25LP Flag On V25 Domain Indicates that an LVR event has occurred on the V25 power domain (FPM or LPM). Other reset sources have no effect. 0b - No event 1b - Event
20 LVR25F	LVR25 Flag On V25 Domain In FPM Indicates that an LVR event has occurred on the V25 power domain in FPM. Other reset sources have no effect. 0b - No event 1b - Event
19 LVRBLPF	LVRBLP Flag On VDD_HV_B Domain Indicates that an LVR event has occurred on the VDD_HV_B power domain (FPM or LPM). Other reset sources have no effect. 0b - No event 1b - Event
18 LVRBF	LVRB Flag On VDD_HV_B Domain In FPM Indicates that an LVR event has occurred on the VDD_HV_B power domain in FPM. Other reset sources have no effect. 0b - No event 1b - Event
17 LVRALPF	LVRALP Flag On VDD_HV_A Domain Indicates that an LVR event has occurred on the VDD_HV_A power domain (FPM or LPM). Other reset sources have no effect. 0b - No event 1b - Event

Table continues on the next page...

Table continued from the previous page...

Field	Function
16 LVRAF	LVRA Flag On VDD_HV_A Domain In FPM Indicates that an LVR event has occurred on the VDD_HV_A power domain in FPM. Other reset sources have no effect. 0b - No event 1b - Event
15 —	Reserved
14 HVD15S	HVD15 Status On V15 Domain In FPM Shows the status of the high-voltage detect, HVD15, on the V15 power domain. This feature is available only in FPM and is disabled in LPM. 0b - Below threshold or LPM 1b - Above threshold and FPM
13 —	Reserved
12 LVD5AS	LVD5A Status On VDD_HV_A Domain In FPM Shows the status of the 5 V LVD, LVD5A, on the VDD_HV_A power domain. This monitor indicates if the voltage is below a certain threshold, which is set slightly below 4.5 V (see the data sheet for the exact value). This feature is available only in FPM and is disabled in LPM. After a reset or wake-up from LPM, you must clear the LVD5AF flag and check the status field LVD5AS to determine the voltage level on the VDD_HV_A supply. 0b - Above threshold 1b - Below threshold
11 HVD11S	HVD11 Status On V11 Domain In FPM Shows the status of HVD11 on the V11 power domain. This feature is available only in FPM and is disabled in LPM. 0b - Below threshold or LPM 1b - Above threshold and FPM
10 HVD25S	HVD25 Status On V25 Domain In FPM Shows the status of the high-voltage detect, HVD25, on the V25 power domain. This feature is available only in FPM and is disabled in LPM. 0b - Below threshold or LPM 1b - Above threshold and FPM
9 HVDBS	HVDB Status On VDD_HV_B Domain In FPM

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Shows the status of the high-voltage detect, HVDB, on the VDD_HV_B power domain. This feature is available only in FPM and is disabled in LPM. 0b - Below threshold or LPM 1b - Above threshold and FPM
8 HVDAS	HVDA Status On VDD_HV_A Domain In FPM Shows the status of the high-voltage detect HVDA on the VDD_HV_A power domain. This feature is available only in FPM and is disabled in LPM. 0b - Below threshold or LPM 1b - Above threshold and FPM
7 —	Reserved
6 HVD15F	HVD15 Flag On V15 Domain In FPM PMC writes 1 to this field when the HVD15S status field changes. To clear HVD15F, write 1 to it. If enabled, HVD15F causes an interrupt request. 0b - Not changed 1b - Changed
5 —	Reserved
4 LVD5AF	LVD5A Flag On VDD_HV_A Domain In FPM PMC writes 1 to this field when the LVD5AS status field changes. To clear LVD5AF, write 1 to it. If enabled, LVD5AF causes an interrupt request. 0b - Not changed 1b - Changed
3 HVD11F	HVD11 Flag On V11 Domain In FPM PMC writes 1 to this field when the HVD11S status field changes. To clear HVD11F, write 1 to it. If enabled, HVD11F causes an interrupt request. 0b - Not changed 1b - Changed
2 HVD25F	HVD25 Flag On V25 Domain In FPM PMC writes 1 to this field when the HVD25S status field changes. To clear HVD25F, write 1 to it. If enabled, HVD25F causes an interrupt request. 0b - Not changed 1b - Changed

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 HVDBF	HVDB Flag On VDD_HV_B Domain In FPM PMC writes 1 to this field when the HVDBS status field changes. To clear HVDBF, write 1 to it. If enabled, HVDBF causes an interrupt request. 0b - Not changed 1b - Changed
0 HVDAF	HVDA Flag On VDD_HV_A Domain In FPM PMC writes 1 to this field when the HVDAS status field changes. To clear HVDAF, write 1 to it. If enabled, HVDAF causes an interrupt request. 0b - Not changed 1b - Changed

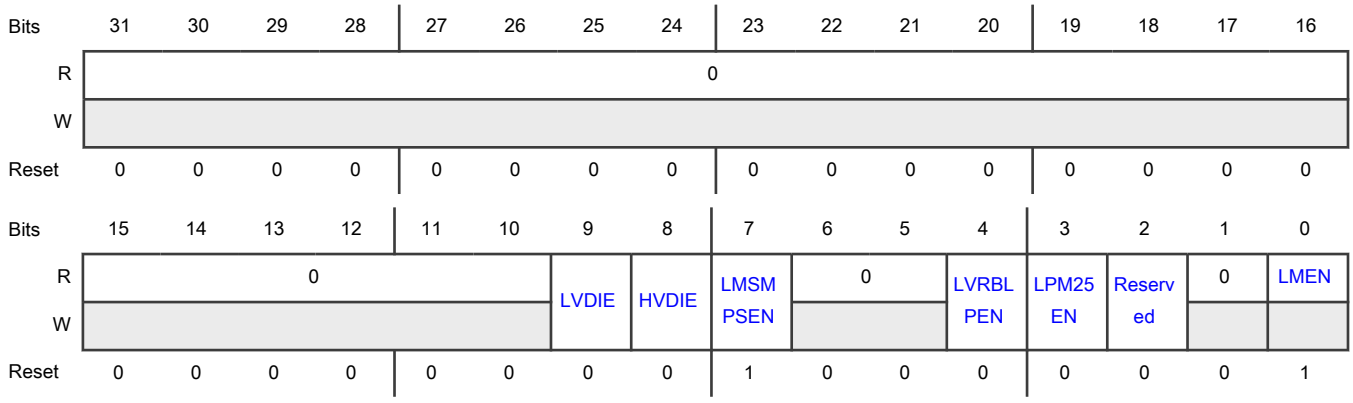
45.4.3 PMC Configuration (CONFIG)

Offset

Register	Offset
CONFIG	4h

Function
Configures the various PMC options.

Diagram



Fields

Field	Function
31-10	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
9 LVDIE	Low Voltage Detect Interrupt Enable Enables hardware interrupt requests if LVSC[LVD5AF] = 1. The LVD interrupt must be disabled before entering LPM. If enabled, there is a request for an LVD hardware interrupt. If disabled, LVD hardware interrupt is disabled (use polling). 0b - Disables 1b - Enables
8 HVDIE	HVD Interrupt Enable Enables hardware interrupt requests if any of the following fields = 1: • LVSC[HVDAF] • LVSC[HVDBF] • LVSC[HVD25F] • LVSC[HVD11F] 0b - Disables 1b - Enables
7 LMSMPSEN	V15 Switched-Mode Power Supply Enable Enables the SMPS regulator (switched-mode power supply), which regulates from VDD_DCDC down to 1.5 V on the V15 pin to supply the last mile regulator. Because LMSMPSEN = 1 after startup or reset, if SMPS is not used, you must write 0 to LMBSMPSEN after each reset to turn the feature off. When LMSMPSEN = 0, you can no longer write 1 to it until the next reset. 0b - Disables 1b - Enables
6-5 —	Reserved
4 LVRBLPEN	LVRBLP Enable During LPM Enables low-voltage reset detection (LVRBLP) on the VDD_HV_B power domain in LPM. 0b - Disables 1b - Enables
3 LPM25EN	V25 Domain Enable During LPM Enables a low-power V25 regulator and the low-voltage reset detection (LVR25LP) in LPM. In FPM, this low-power regulator is always off. 0b - Disables 1b - Enables

Table continues on the next page...

Table continued from the previous page...

Field	Function
2 —	Reserved
1 —	Reserved
0 LMEN	Last Mile Regulator Enable Enables the last mile regulator, which regulates an external 1.5 V voltage on V15 down to the core and logic supply (V11 power domain), which is typically 1.1 V. Because the last mile regulator is always enabled on this chip, LMEN always reads 1.

45.4.4 SMPS Configuration (SMPSCONFIG)

Offset

Register	Offset
SMPSCONFIG	8h

Function

Configures the various options of the switched-mode power supply (SMPS) generating V15 (1.5 V input supply to the last mile regulator).

Table 259. SMPS configurations for driving the PMOS_CTRL pin

CFG[3:0]	PMOS_CTRL frequency	PMOS_CTRL duty cycle 3 V	PMOS_CTRL duty cycle 5 V
0000	470 kHz	58.8%	47.1%
0001	470 kHz	64.7%	52.9%
0010	470 kHz	52.9%	41.2%
0011	533 kHz	60%	46.7%
0100	533 kHz	66.7%	53.3%
0101	533 kHz	53.3%	40%
0110	421 kHz	57.9%	47.4%
0111	421 kHz	63.2%	52.6%
1000	421 kHz	52.6%	42.1%
1001–1110	Reserved	Reserved	Reserved
1111	8 MHz ÷ (PERIOD[4:0] + 1)	ONTIME3V[4:0] ÷ (PERIOD[4:0] + 1)	ONTIME5V[4:0] ÷ (PERIOD[4:0] + 1)

Table 260. 8 MHz IRC dither configurations

DITHERCFG[1:0]	Dither amplitude	Dither frequency
00	± 0.4 MHz	400 kHz
01	± 0.6 MHz	286 kHz
10	± 0.8 MHz	222 kHz
11	± 1 MHz	182 kHz

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				ONTIME5V				PGATES	0			ONTIME3V			
W																
Reset	0	0	0	0	1	0	0	0	u	0	0	0	1	0	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				PERIOD				DITHEREN	DITHERCFG		LPM15EN	CFG			
W																
Reset	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	Reserved
28-24 ONTIME5V	SMPS Duty Cycle For 5 V Range Determines the duty cycle of the output signal at the PMOS_CTRL pin. This duty cycle is applied when VDD_DCDC is greater than or equal to 4 V. The duty cycle is calculated as follows: Duty_pmos_ctrl = ONTIME3V[4:0] ÷ (PERIOD[4:0] + 1)
23 PGATES	PMOS_CTRL Status Reflects the status of the PMOS_CTRL pin. If = 0, note that VSS_DCDC is connected to GND. 0b - Driven to VSS_DCDC 1b - Driven to VDD_DCDC
22-21 —	Reserved
20-16 ONTIME3V	SMPS Duty Cycle For 3 V Range Determines the duty cycle of the output signal at the PMOS_CTRL pin. This duty cycle is applied when VDD_DCDC is smaller than 4 V.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	The duty cycle is calculated as follows: Duty_pmos_ctrl = ONTIME3V[4:0] ÷ (PERIOD[4:0] + 1)
15-13 —	Reserved
12-8 PERIOD	SMPS Period Determines the frequency of the output signal at the PMOS_CTRL pin. The frequency is calculated as follows: Freq_pmos_ctrl = 8 MHz ÷ (PERIOD[4:0] + 1)
7 DITHEREN	IRC Dither Enable Enables dithering of the 8 MHz IRC. The amplitude and frequency of the dithering is determined by the DITHERCFG[1:0] bits. 0b - Disables 1b - Enables
6-5 DITHERCFG	IRC Dither Configuration Selects the IRC dithering amplitude and frequency. This feature spreads the frequency spectrum of driving the PMOS_CTRL pin to reduce radiated emission. To change the value: 1. Write 0 to DITHEREN. 2. Write the dither configuration (DITHERCFG[1:0]). 3. Write 1 to DITHEREN. See Table 260 for configuration options.
4 LPM15EN	V15 Domain Enable During LPM Enables V15 in LPM. This is useful for fast recovery to FPM when using the SMPS mode to generate V15. This feature is active only in LPM, and is turned off in FPM. For this purpose, there is a small LDO inside the PMC that regulates V15 to the target value during LPM. **NOTE** LPM15EN is only used if SMPS generates V15. All other options to generate V15 are forbidden, and in those cases you must always write 0 to LPM15EN. 0b - Not kept on target 1b - Kept on target
3-0 CFG	SMPS Configuration Select Selects a configuration for frequency and duty cycle (on-time) of the switched-mode power supply, which drives (via PMOS_CTRL) the external power FET, according to Table 259.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE As seen in the table, CFG[3:0] = 1111 allows for free customization of PERIOD[4:0], ONTIME5V[4:0], and ONTIME3V[3:0]. If you choose this option, you must carefully consider the value of these bits. To ensure proper functioning of SMPS, and to keep EMI in check, you must program the switching frequency and duty cycle within a reasonable range.

45.4.5 Version ID (VERID)

Offset

Register	Offset
VERID	Ch

Function

Returns the major and minor version numbers of hardware implementation.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MAJOR								MINOR							
W																
Reset	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															LMFEAT
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Fields

Field	Function
31-24 MAJOR	Major Version Number Returns the version number for the specification. K344 = 2, K31x = 3, K358 = 4, K396 = 5, K388 = 6
23-16 MINOR	Minor Version Number Returns the version number for the hardware implementation.
15-1	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
0 LMFEAT	Last Mile Regulator Feature Indicates whether the last mile regulator feature (1.5 V to 1.1 V) is available. 0b - Not available 1b - Available

45.5 Glossary

FPM	Full Performance mode
HVD	High voltage detector
IRC	Internal RC oscillator
LM	Last mile regulator
LPM	Low Performance mode
LVD	Low voltage detector
LVR	Low voltage reset
NVM	Nonvolatile memory
POR	Power on reset
XOSC	External crystal oscillator
SMPS	Switched-mode power supply

Chapter 46
Mode Entry Module (MC_ME)

46.1 Chip-specific MC_ME information

46.1.1 MC_ME modes

This chip implements these modes:

- Reset
- Run
- Standby

The chip always enters Run mode after exiting Reset mode wherein you can configure the chip to perform its computational and communication functions. In Run mode:

- The chip remains fully powered. The boot core is the only core enabled on Run entry.
- You can enable and disable application cores and peripherals as needed, based on functional and power requirements.
- You can configure the pins and self-test as needed. See "Self-Test programming sequence" section in the STCU2 chapter for the self-test programming sequence to be followed before initiating self-test, in which:
  - Pins are safe-stated.
  - No computational or communication activities are possible.
- The chip automatically enters Reset mode after self-test (BIST) is complete.

46.1.2 Core operation modes

The Cortex-M7 cores in the chip support these two modes of operation:

- Decoupled/independent operation
- Coupled/lockstep operation

The device configuration clients (utest_misc DCF client) control these modes of operation. See the DCF clients file attached to this document for details.

46.1.3 MC_ME partition mapping of cores and peripherals

MC_ME provides registers and interface signals to support multiple partitions. These MC_ME partitions are different from the chip's LBIST partitions described in the "Safety Overview" chapter. This chip has three MC_ME partitions:

- Partition 0: Contains application cores and the on/off-platform slots on AIPS_0 bridge
- Partition 1: Contains the on/off-platform slots on AIPS_1 bridge
- Partition 2: Contains the on/off-platform slots on AIPS_2 bridge

Table 261 and Table 262 specify the core and peripheral mapping on MC_ME partitions and their associated clock gating possibilities.

MC_ME also provides provisions to control the booting address for application cores, which can be configured to start from a nondefault address location by appropriately configuring PRTN*x*_CORE*n*_ADDR[ADDR].

46.1.4 Core clock gating

MC_ME has individual core-clock enable fields that gate the application core clocks, which can also be clock gated by executing Waiting for Interrupt (WFI).

You can enable the application cores by configuring the respective CCE fields. See Application core initialization process and Application core shutdown process for proper initialization and shutdown of application cores. There is no clock control for the HSE_B core. It needs to be only put into WFI if required to be shutdown (for example, in Standby mode).

Table 261. MC_ME partition core mapping

Core	MC_ME partition	MC_ME clock enable register field
Cortex-M7_0	0	MC_ME.PRTN0_CORE0_PCONF[CCE]
Cortex-M7_1 [1]	0	MC_ME.PRTN0_CORE1_PCONF[CCE]
Cortex-M7_2 [1], [2]	0	MC_ME.PRTN0_CORE4_PCONF[CCE]
Cortex-M7_3 [1], [3]	0	MC_ME.PRTN0_CORE3_PCONF[CCE]

1. Not present on S32K312 and S32K311
2. Only present on S32K388, S32K389, S32K358, S32K356, S32K348, S32K338, and S32K328
3. Only present on S32K388 and S32K389

46.1.5 Peripheral clock gating

See Peripheral initialization process and Peripheral shutdown process for proper initialization and shutdown of peripherals. See Table 262 for peripheral clock gating possibilities. This table details clock gating possibilities of all chips. For applicable modules/module instances see corresponding chip-specific section of the module. "MC_ME memory map" section the details clock gating possibilities of S32K388/S32K389 chip.

The application core can program the reserved configurations.

NOTE

Before accessing the registers of a peripheral to start using it, its clock must be turned on, otherwise, a Hard-Fault event will occur.

Table 262. MC_ME partition peripheral mapping and clock control

AIPS	Peripheral description	MC_ME COFB control register	MC_ME peripheral control register	MC_ME peripheral slot no. in partition	Clock gating control supported	Enabled on reset	On Platform
0	Crossbar Integrity Checker (HSE & AES_ACCEL AXBS_Lite)[1]	PRTN0_COFB0_CLKEN[REQ2]	2	2	No	Yes	Yes
0	ERM1	PRTN0_COFB0_CLKEN[REQ3][2]	3	3	Yes	No	Yes
0	Flash controller 1[3]	PRTN0_COFB0_CLKEN[REQ26]	26	26	No	Yes	Yes
0	Flash controller 1 alternate[1]	PRTN0_COFB0_CLKEN[REQ27]	27	27	No	Yes	Yes
0	Software Watchdog 3	PRTN0_COFB0_CLKEN[REQ28][1]	28	28	Yes	No	Yes
0	Trigger Multiplexing Control	PRTN0_COFB1_CLKEN[REQ32]	32	32	Yes	No	No
0	Body Cross Triggering Unit	PRTN0_COFB1_CLKEN[REQ33]	33	33	Yes	No	No
0	eMIOS 0	PRTN0_COFB1_CLKEN[REQ34]	34	34	Yes	No	No

Table continues on the next page...

Table 262. MC_ME partition peripheral mapping and clock control (continued)

AIPS	Peripheral description	MC_ME COFB control register	MC_ME peripheral control register	MC_ME peripheral slot no. in partition	Clock gating control supported	Enabled on reset	On Platform
0	eMIOS 1	PRTN0_COFB1_CLKEN[REQ35]	35	35	Yes	No	No
0	eMIOS 2[4]	PRTN0_COFB1_CLKEN[REQ36]	36	36	Yes	No	No
0	Logic Control Unit 0	PRTN0_COFB1_CLKEN[REQ38]	38	38	Yes	No	No
0	Logic Control Unit 1	PRTN0_COFB1_CLKEN[REQ39]	39	39	Yes	No	No
0	Analog-to-digital converter 0	PRTN0_COFB1_CLKEN[REQ40]	40	40	Yes	No	No
0	Analog-to-digital converter 1	PRTN0_COFB1_CLKEN[REQ41]	41	41	Yes	No	No
0	Analog-to-digital converter 2[4]	PRTN0_COFB1_CLKEN[REQ42]	42	42	Yes	No	No
0	Programmable Interrupt Timer 0	PRTN0_COFB1_CLKEN[REQ44]	44	44	Yes	Yes	No
0	Programmable Interrupt Timer 1	PRTN0_COFB1_CLKEN[REQ45]	45	45	Yes	No	No
0	MU_2_MUA[5]	PRTN0_COFB1_CLKEN[REQ46]	46	46	Yes	No	No
0	MU_2_MUB[5]	PRTN0_COFB1_CLKEN[REQ47]	47	47	Yes	No	No
0	MU_3_MUA[1]	PRTN0_COFB1_CLKEN[REQ49]	49	49	Yes	No	No
0	MU_3_MUB[1]	PRTN0_COFB1_CLKEN[REQ50]	50	50	Yes	No	No
0	MU_4_MUA[1]	PRTN0_COFB1_CLKEN[REQ51]	51	51	Yes	No	No
0	MU_4_MUB[1]	PRTN0_COFB1_CLKEN[REQ52]	52	52	Yes	No	No
1	System crossbar switch[6]	PRTN1_COFB0_CLKEN[REQ0]	128	0	No	Yes	Yes
1	Crossbar Integrity Checker (System AXBS / AXBS Lite)	PRTN1_COFB0_CLKEN[REQ1]	129	1	No	Yes	Yes
1	Crossbar Integrity Checker (Peripheral AXBS-Lite)[6]	PRTN1_COFB0_CLKEN[REQ2]	130	2	No	Yes	Yes
1	eDMA control & status (MP_CSR; MP_ES; MP_HRS)	PRTN1_COFB0_CLKEN[REQ3]	131	3	Yes	No	Yes
1	eDMA transfer control descriptor 0	PRTN1_COFB0_CLKEN[REQ4]	132	4	Yes	No	Yes
1	eDMA transfer control descriptor 1	PRTN1_COFB0_CLKEN[REQ5]	133	5	Yes	No	Yes

Table continues on the next page...

Table 262. MC_ME partition peripheral mapping and clock control (continued)

AIPS	Peripheral description	MC_ME COFB control register	MC_ME peripheral control register	MC_ME peripheral slot no. in partition	Clock gating control supported	Enabled on reset	On Platform
1	eDMA transfer control descriptor 2	PRTN1_COFB0_CLKEN[REQ6]	134	6	Yes	No	Yes
1	eDMA transfer control descriptor 3	PRTN1_COFB0_CLKEN[REQ7]	135	7	Yes	No	Yes
1	eDMA transfer control descriptor 4	PRTN1_COFB0_CLKEN[REQ8]	136	8	Yes	No	Yes
1	eDMA transfer control descriptor 5	PRTN1_COFB0_CLKEN[REQ9]	137	9	Yes	No	Yes
1	eDMA transfer control descriptor 6	PRTN1_COFB0_CLKEN[REQ10]	138	10	Yes	No	Yes
1	eDMA transfer control descriptor 7	PRTN1_COFB0_CLKEN[REQ11]	139	11	Yes	No	Yes
1	eDMA transfer control descriptor 8	PRTN1_COFB0_CLKEN[REQ12]	140	12	Yes	No	Yes
1	eDMA transfer control descriptor 9	PRTN1_COFB0_CLKEN[REQ13]	141	13	Yes	No	Yes
1	eDMA transfer control descriptor 10	PRTN1_COFB0_CLKEN[REQ14]	142	14	Yes	No	Yes
1	eDMA transfer control descriptor 11	PRTN1_COFB0_CLKEN[REQ15]	143	15	Yes	No	Yes
1	Debug APB Page0	PRTN1_COFB0_CLKEN[REQ21]	149	21	No	Yes	Yes
1	Debug APB Page1	PRTN1_COFB0_CLKEN[REQ21]	149	21	No	Yes	Yes
1	Debug APB Page2	PRTN1_COFB0_CLKEN[REQ21]	149	21	No	Yes	Yes
1	Debug APB Page3	PRTN1_COFB0_CLKEN[REQ21]	149	21	No	Yes	Yes
1	Debug APB Paged Area	PRTN1_COFB0_CLKEN[REQ21]	149	21	No	Yes	Yes
1	SDA-AP	PRTN1_COFB0_CLKEN[REQ21]	149	21	Yes	Yes	Yes
1	EIM0[7]	PRTN1_COFB0_CLKEN[REQ22]	150	22	Yes	No	Yes
1	ERM0	PRTN1_COFB0_CLKEN[REQ23]	151	23	Yes	No	Yes
1	MSCM	PRTN1_COFB0_CLKEN[REQ24]	152	24	Yes	No	Yes
1	RAM controller 0	PRTN1_COFB0_CLKEN[REQ25]	153	25	No	Yes	Yes
1	Flash controller	PRTN1_COFB0_CLKEN[REQ26]	154	26	No	Yes	Yes
1	Flash controller alternate	PRTN1_COFB0_CLKEN[REQ27]	155	27	No	Yes	Yes
1	Software Watchdog 0	PRTN1_COFB0_CLKEN[REQ28]	156	28	Yes	Yes	Yes

Table continues on the next page...

Table 262. MC_ME partition peripheral mapping and clock control (continued)

AIPS	Peripheral description	MC_ME COFB control register	MC_ME peripheral control register	MC_ME peripheral slot no. in partition	Clock gating control supported	Enabled on reset	On Platform
1	System Timer Module 0	PRTN1_COFB0_CLKEN[REQ29]	157	29	Yes	No	Yes
1	XRDC	PRTN1_COFB0_CLKEN[REQ30]	158	30	No	Yes	Yes
1	Interrupt Monitor	PRTN1_COFB0_CLKEN[REQ31]	159	31	Yes	No	Yes
1	DMA Channel Multiplexer 0	PRTN1_COFB1_CLKEN[REQ32]	160	32	Yes	No	No
1	DMA Channel Multiplexer 1	PRTN1_COFB1_CLKEN[REQ33]	161	33	Yes	No	No
1	Real-time clock	PRTN1_COFB1_CLKEN[REQ34]	162	34	Yes	Yes	No
1	Reset Generation Module	PRTN1_COFB1_CLKEN[REQ35]	163	35	No	Yes	No
1	SIUL_VIRTWRAPPER_PDAC0	PRTN1_COFB1_CLKEN[REQ36]	164	36	No	Yes	No
1	SIUL_VIRTWRAPPER_PDAC0	PRTN1_COFB1_CLKEN[REQ37]	165	37	No	Yes	No
1	SIUL_VIRTWRAPPER_PDAC1	PRTN1_COFB1_CLKEN[REQ38]	166	38	No	Yes	No
1	SIUL_VIRTWRAPPER_PDAC1	PRTN1_COFB1_CLKEN[REQ39]	167	39	No	Yes	No
1	SIUL_VIRTWRAPPER_PDAC2[5]	PRTN1_COFB1_CLKEN[REQ40]	168	40	No	Yes	No
1	SIUL_VIRTWRAPPER_PDAC2[5]	PRTN1_COFB1_CLKEN[REQ41]	169	41	No	Yes	No
1	SIUL_VIRTWRAPPER_PDAC3	PRTN1_COFB1_CLKEN[REQ42]	170	42	Yes	Yes	No
1	System Status and Configuration Module	PRTN1_COFB1_CLKEN[REQ43]	171	43	No	Yes	No
1	Wakeup Unit	PRTN1_COFB1_CLKEN[REQ45]	173	45	Yes	Yes	No
1	CMU 0-6	PRTN1_COFB1_CLKEN[REQ47]	175	47	Yes	No	No
1	Touch Sensing Coupling Controller	PRTN1_COFB1_CLKEN[REQ49]	177	49	Yes	Yes	No
1	32 kHz Slow Internal RC Oscillator	PRTN1_COFB1_CLKEN[REQ50]	178	50	No	Yes	No
1	32 kHz Slow External Crystal Oscillator [8]	PRTN1_COFB1_CLKEN[REQ51]	179	51	Yes	Yes	No
1	48 MHz Fast Internal RC Oscillator	PRTN1_COFB1_CLKEN[REQ52]	180	52	No	Yes	No

Table continues on the next page...

Table 262. MC_ME partition peripheral mapping and clock control (continued)

AIPS	Peripheral description	MC_ME COFB control register	MC_ME peripheral control register	MC_ME peripheral slot no. in partition	Clock gating control supported	Enabled on reset	On Platform
1	8-40 MHz Fast External Crystal Oscillator	PRTN1_COFB1_CLKEN[REQ53]	181	53	Yes	Yes	No
1	Clock Generation Module	PRTN1_COFB1_CLKEN[REQ54]	182	54	No	Yes	No
1	Mode Entry Module	PRTN1_COFB1_CLKEN[REQ55]	183	55	No	Yes	No
1	Frequency Modulated Phase-Locked Loop	PRTN1_COFB1_CLKEN[REQ56]	184	56	Yes	No	No
1	Frequency Modulated Phase-Locked Loop 2[9]	PRTN1_COFB1_CLKEN[REQ57]	185	57	Yes	No	No
1	Power management controller	PRTN1_COFB1_CLKEN[REQ58]	186	58	No	Yes	No
1	Flash memory	PRTN1_COFB1_CLKEN[REQ59]	187	59	No	Yes	No
1	Flash memory alternate	PRTN1_COFB1_CLKEN[REQ60]	188	60	No	Yes	No
1	SIUL_VIRTWRAPPER_PDAC4[10]	PRTN1_COFB1_CLKEN[REQ61]	189	61	No	Yes	No
1	SIUL_VIRTWRAPPER_PDAC4[10]	PRTN1_COFB1_CLKEN[REQ62]	190	62	No	Yes	No
1	Programmable Interrupt Timer 2	PRTN1_COFB1_CLKEN[REQ63]	191	63	Yes	No	No
1	Programmable Interrupt Timer 3[1]	PRTN1_COFB2_CLKEN[REQ64]	192	64	Yes	No	No
1	FlexCAN 0	PRTN1_COFB2_CLKEN[REQ65]	193	65	Yes	No	No
1	FlexCAN 1	PRTN1_COFB2_CLKEN[REQ66]	194	66	Yes	No	No
1	FlexCAN 2	PRTN1_COFB2_CLKEN[REQ67]	195	67	Yes	No	No
1	FlexCAN 3[8]	PRTN1_COFB2_CLKEN[REQ68]	196	68	Yes	No	No
1	FlexCAN 4[11]	PRTN1_COFB2_CLKEN[REQ69]	197	69	Yes	No	No
1	FlexCAN 5[11]	PRTN1_COFB2_CLKEN[REQ70]	198	70	Yes	No	No
1	FlexCAN 6[2]	PRTN1_COFB2_CLKEN[REQ71]	199	71	Yes	No	No
1	FlexCAN 7[2]	PRTN1_COFB2_CLKEN[REQ72]	200	72	Yes	No	No
1	Flexible IO	PRTN1_COFB2_CLKEN[REQ73]	201	73	Yes	No	No
1	Low Power UART 0	PRTN1_COFB2_CLKEN[REQ74]	202	74	Yes	No	No
1	Low Power UART 1	PRTN1_COFB2_CLKEN[REQ75]	203	75	Yes	No	No
1	Low Power UART 2	PRTN1_COFB2_CLKEN[REQ76]	204	76	Yes	No	No
1	Low Power UART 3	PRTN1_COFB2_CLKEN[REQ77]	205	77	Yes	No	No

Table continues on the next page...

Table 262. MC_ME partition peripheral mapping and clock control (continued)

AIPS	Peripheral description	MC_ME COFB control register	MC_ME peripheral control register	MC_ME peripheral slot no. in partition	Clock gating control supported	Enabled on reset	On Platform
1	Low Power UART 4[11]	PRTN1_COFB2_CLKEN[REQ78]	206	78	Yes	No	No
1	Low Power UART 5[11]	PRTN1_COFB2_CLKEN[REQ79]	207	79	Yes	No	No
1	Low Power UART 6[11]	PRTN1_COFB2_CLKEN[REQ80]	208	80	Yes	No	No
1	Low Power UART 7[11]	PRTN1_COFB2_CLKEN[REQ81]	209	81	Yes	No	No
1	SIUL_VIRTWRAPPER_PDAC5[1]	PRTN1_COFB2_CLKEN[REQ82]	210	82	No	Yes	No
1	SIUL_VIRTWRAPPER_PDAC5[1]	PRTN1_COFB2_CLKEN[REQ83]	211	83	No	Yes	No
1	Low Power I2C 0	PRTN1_COFB2_CLKEN[REQ84]	212	84	Yes	No	No
1	Low Power I2C 1	PRTN1_COFB2_CLKEN[REQ85]	213	85	Yes	No	No
1	Low Power SPI 0	PRTN1_COFB2_CLKEN[REQ86]	214	86	Yes	No	No
1	Low Power SPI 1	PRTN1_COFB2_CLKEN[REQ87]	215	87	Yes	No	No
1	Low Power SPI 2	PRTN1_COFB2_CLKEN[REQ88]	216	88	Yes	No	No
1	Low Power SPI 3	PRTN1_COFB2_CLKEN[REQ89]	217	89	Yes	No	No
1	Synchronous Audio Interface 0[6]	PRTN1_COFB2_CLKEN[REQ91]	219	91	Yes	No	No
1	Low Power Comparator 0	PRTN1_COFB2_CLKEN[REQ92]	220	92	Yes	Yes	No
1	Low Power Comparator 1[8]	PRTN1_COFB2_CLKEN[REQ93]	221	93	Yes	Yes	No
1	TMU Temperature Sensor Unit	PRTN1_COFB2_CLKEN[REQ95]	223	95	Yes	No	No
1	CRC	PRTN1_COFB3_CLKEN[REQ96]	224	96	Yes	No	No
1	FCCU (+FOSU)	PRTN1_COFB3_CLKEN[REQ97]	225	97	No	Yes	No
1	MU_0_MUB	PRTN1_COFB3_CLKEN[REQ99]	227	99	No	Yes	No
1	MU_1_MUB[12]	PRTN1_COFB3_CLKEN[REQ100]	228	100	No	Yes	No
1	JDC (JTAG Data Communication)	PRTN1_COFB3_CLKEN[REQ101]	229	101	No	Yes	No
1	Configuration GPR	PRTN1_COFB3_CLKEN[REQ103]	231	103	No	Yes	No
1	Self-Test Control Unit	PRTN1_COFB3_CLKEN[REQ104]	232	104	Yes	Yes	No
1	Selftest GPR[6]	PRTN1_COFB3_CLKEN[REQ108]	236	108	No	Yes	No
1	AES Accelerator [1,13]	PRTN1_COFB3_CLKEN[REQ112-REQ-115]	240	112	Yes	No	No

Table continues on the next page...

Table 262. MC_ME partition peripheral mapping and clock control (continued)

AIPS	Peripheral description	MC_ME COFB control register	MC_ME peripheral control register	MC_ME peripheral slot no. in partition	Clock gating control supported	Enabled on reset	On Platform
1	AES Application 0[1,13]	PRTN1_COFB3_CLKEN[REQ116-REQ119]	244	116	Yes	No	No
1	AES Application 1[1,13]	PRTN1_COFB3_CLKEN[REQ120-REQ123]	248	120	Yes	No	No
1	AES Application 2[1,13]	PRTN1_COFB3_CLKEN[REQ124-REQ127]	252	124	Yes	No	No
2	Crossbar Integrity Checker (TCM backdoor AHB Splitter)[14]	PRTN2_COFB0_CLKEN[REQ0]	256	0	No	Yes	Yes
2	Crossbar Integrity Checker (eDMA & STAM AXBS-Lite)[14]	PRTN2_COFB0_CLKEN[REQ1]	257	1	No	Yes	Yes
2	Crossbar Integrity Checker (PRAM2 & TCM backdoor AHB Splitter)[2]	PRTN2_COFB0_CLKEN[REQ2]	258	2	No	Yes	Yes
2	Crossbar Integrity Checker (AES_ACCEL AHB Multiplexer)[1]	PRTN2_COFB0_CLKEN[REQ3]	259	3	No	Yes	Yes
2	eDMA transfer control descriptor 12[6]	PRTN2_COFB0_CLKEN[REQ4]	260	4	Yes	No	Yes
2	eDMA transfer control descriptor 13[6]	PRTN2_COFB0_CLKEN[REQ5]	261	5	Yes	No	Yes
2	eDMA transfer control descriptor 14[6]	PRTN2_COFB0_CLKEN[REQ6]	262	6	Yes	No	Yes
2	eDMA transfer control descriptor 15[6]	PRTN2_COFB0_CLKEN[REQ7]	263	7	Yes	No	Yes
2	eDMA transfer control descriptor 16[6]	PRTN2_COFB0_CLKEN[REQ8]	264	8	Yes	No	Yes
2	eDMA transfer control descriptor 17[6]	PRTN2_COFB0_CLKEN[REQ9]	265	9	Yes	No	Yes
2	eDMA transfer control descriptor 18[6]	PRTN2_COFB0_CLKEN[REQ10]	266	10	Yes	No	Yes
2	eDMA transfer control descriptor 19[6]	PRTN2_COFB0_CLKEN[REQ11]	267	11	Yes	No	Yes
2	eDMA transfer control descriptor 20[6]	PRTN2_COFB0_CLKEN[REQ12]	268	12	Yes	No	Yes

Table continues on the next page...

Table 262. MC_ME partition peripheral mapping and clock control (continued)

AIPS	Peripheral description	MC_ME COFB control register	MC_ME peripheral control register	MC_ME peripheral slot no. in partition	Clock gating control supported	Enabled on reset	On Platform
2	eDMA transfer control descriptor 21[6]	PRTN2_COFB0_CLKEN[REQ13]	269	13	Yes	No	Yes
2	eDMA transfer control descriptor 22[6]	PRTN2_COFB0_CLKEN[REQ14]	270	14	Yes	No	Yes
2	eDMA transfer control descriptor 23[6]	PRTN2_COFB0_CLKEN[REQ15]	271	15	Yes	No	Yes
2	eDMA transfer control descriptor 24[6]	PRTN2_COFB0_CLKEN[REQ16]	272	16	Yes	No	Yes
2	eDMA transfer control descriptor 25[6]	PRTN2_COFB0_CLKEN[REQ17]	273	17	Yes	No	Yes
2	eDMA transfer control descriptor 26[6]	PRTN2_COFB0_CLKEN[REQ18]	274	18	Yes	No	Yes
2	eDMA transfer control descriptor 27[6]	PRTN2_COFB0_CLKEN[REQ19]	275	19	Yes	No	Yes
2	eDMA transfer control descriptor 28[6]	PRTN2_COFB0_CLKEN[REQ20]	276	20	Yes	No	Yes
2	eDMA transfer control descriptor 29[6]	PRTN2_COFB0_CLKEN[REQ21]	277	21	Yes	No	Yes
2	eDMA transfer control descriptor 30[6]	PRTN2_COFB0_CLKEN[REQ22]	278	22	Yes	No	Yes
2	eDMA transfer control descriptor 31[6]	PRTN2_COFB0_CLKEN[REQ23]	279	23	Yes	No	Yes
2	Semaphores2[6]	PRTN2_COFB0_CLKEN[REQ24]	280	24	Yes	No	Yes
2	RAM controller 1[4]	PRTN2_COFB0_CLKEN[REQ25]	281	25	No	Yes	Yes
2	RAM controller 2[2]	PRTN2_COFB0_CLKEN[REQ26]	282	26	No	Yes	Yes
2	Software Watchdog 1[15]	PRTN2_COFB0_CLKEN[REQ27]	283	27	Yes	No	Yes
2	Software Watchdog 2[16]	PRTN2_COFB0_CLKEN[REQ28]	284	28	Yes	No	Yes
2	System Timer Module 1[6]	PRTN2_COFB0_CLKEN[REQ29]	285	29	Yes	No	Yes
2	System Timer Module 2[2]	PRTN2_COFB0_CLKEN[REQ30]	286	30	Yes	No	Yes
2	System Timer Module 3[1]	PRTN2_COFB0_CLKEN[REQ31]	287	31	Yes	No	Yes
2	EMAC[17]	PRTN2_COFB1_CLKEN[REQ32]	288	32	Yes	No	No
2	GMAC0[2]	PRTN2_COFB1_CLKEN[REQ33]	289	33	Yes	No	No
2	GMAC1[1]	PRTN2_COFB1_CLKEN[REQ34]	290	34	Yes	No	No
2	Low Power UART 8[4]	PRTN2_COFB1_CLKEN[REQ35]	291	35	Yes	No	No

Table continues on the next page...

Table 262. MC_ME partition peripheral mapping and clock control (continued)

AIPS	Peripheral description	MC_ME COFB control register	MC_ME peripheral control register	MC_ME peripheral slot no. in partition	Clock gating control supported	Enabled on reset	On Platform
2	Low Power UART 9[4]	PRTN2_COFB1_CLKEN[REQ36]	292	36	Yes	No	No
2	Low Power UART 10[4]	PRTN2_COFB1_CLKEN[REQ37]	293	37	Yes	No	No
2	Low Power UART 11[4]	PRTN2_COFB1_CLKEN[REQ38]	294	38	Yes	No	No
2	Low Power UART 12[4]	PRTN2_COFB1_CLKEN[REQ39]	295	39	Yes	No	No
2	Low Power UART 13[4]	PRTN2_COFB1_CLKEN[REQ40]	296	40	Yes	No	No
2	Low Power UART 14[4]	PRTN2_COFB1_CLKEN[REQ41]	297	41	Yes	No	No
2	Low Power UART 15[4]	PRTN2_COFB1_CLKEN[REQ42]	298	42	Yes	No	No
2	Low Power SPI 4[4]	PRTN2_COFB1_CLKEN[REQ47]	303	47	Yes	No	No
2	Low Power SPI 5[4]	PRTN2_COFB1_CLKEN[REQ48]	304	48	Yes	No	No
2	QuadSPI[4]	PRTN2_COFB1_CLKEN[REQ51]	307	51	Yes	No	No
2	Synchronous Audio Interface 1[4]	PRTN2_COFB1_CLKEN[REQ55]	311	55	Yes	No	No
2	Ultra Secured Digital Host Controller[18]	PRTN2_COFB1_CLKEN[REQ57]	313	57	Yes	No	No
2	Low Power Comparator 2[4]	PRTN2_COFB1_CLKEN[REQ58]	314	58	Yes	Yes	No
2	MU_1_MUB[4]	PRTN2_COFB1_CLKEN[REQ59]	315	59	No	Yes	No
2	EIM0[2]	PRTN2_COFB2_CLKEN[REQ67]	323	67	Yes	No	No
2	EIM1[2]	PRTN2_COFB2_CLKEN[REQ68]	324	68	Yes	No	No
2	EIM2[2]	PRTN2_COFB2_CLKEN[REQ69]	325	69	Yes	No	No
2	EIM3[2]	PRTN2_COFB2_CLKEN[REQ70]	326	70	Yes	No	No
2	AES Application 3[1,13]	PRTN2_COFB2_CLKEN[REQ72-REQ75]	328	72	Yes	No	No
2	AES Application 4[1,13]	PRTN2_COFB2_CLKEN[REQ76-REQ79]	332	76	Yes	No	No
2	AES Application 5[1,13]	PRTN2_COFB2_CLKEN[REQ80-REQ83]	336	80	Yes	No	No
2	AES Application 6[1,13]	PRTN2_COFB2_CLKEN[REQ84-REQ87]	340	84	Yes	No	No
2	AES Application 7[1,13]	PRTN2_COFB2_CLKEN[REQ88-REQ91]	344	88	Yes	No	No
2	FlexCAN 8[3]	PRTN2_COFB2_CLKEN[REQ92]	348	92	Yes	No	No
2	FlexCAN 9[3]	PRTN2_COFB2_CLKEN[REQ93]	349	93	Yes	No	No

Table continues on the next page...

Table 262. MC_ME partition peripheral mapping and clock control (continued)

AIPS	Peripheral description	MC_ME COFB control register	MC_ME peripheral control register	MC_ME peripheral slot no. in partition	Clock gating control supported	Enabled on reset	On Platform
2	FlexCAN 10[3]	PRTN2_COFB2_CLKEN[REQ94]	350	94	Yes	No	No
2	FlexCAN 11[3]	PRTN2_COFB2_CLKEN[REQ95]	351	95	Yes	No	No
2	Flash memory 1[3]	PRTN2_COFB3_CLKEN[REQ96]	352	96	No	Yes	No
2	Flash memory 1 alternate[3]	PRTN2_COFB3_CLKEN[REQ97]	353	97	No	Yes	No
2	RAM controller 3[3]	PRTN2_COFB3_CLKEN[REQ98]	354	98	No	Yes	No

1. Applicable for S32K388 and S32K389 only.
2. Applicable for S32K328, S32K338, S32K348, S32K358, S32K356, S32K388, and S32K389 only.
3. Applicable for S32K389 only.
4. Applicable for S32K314, S32K324, S32K344, S32K328,S32K338, S32K348, S32K358, S32K356, S32K388, and S32K389 only.
5. Applicable for S32K322, S32K324, S32K328, S32K338, S32K358, S32K356, S32K388, and S32K389 only.
6. Reserved for S32K310, S32K311, and S32K312.
7. Applicable for S32K310, S32K311, S32K312, S32K322, S32K341, S32K342, S32K314, S32K324, and S32K344

Table 263. PRTN1_COFB0_CLKEN[REQ22]

Bit filed	Description
REQ22	Clock enable This bit provides the clock enable control for block 22 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.

8. Reserved for S32K310 and S32K311.
9. Applicable for S32K328, S32K338, S32K348, S32K358, S32K356, S32K388, and S32K389 only.
10. Applicable for S32K338, S32K388 and S32K389 only.
11. Reserved for S32K310, S32K311, S32K322, S32K341, and S32K342.
12. Applicable for S32K310, S32K311, and S32K312 only.
13. All the corresponding MC_ME slots must be configured to access the peripheral. For example, PRTN1_COFB3_CLKEN[REQ112], PRTN1_COFB3_CLKEN[REQ113], PRTN1_COFB3_CLKEN[REQ114], and PRTN1_COFB3_CLKEN[REQ115] must be configured to access AES Accelerator.
14. Applicable for S32K322, S32K341, S32K342, S32K314, S32K324, S32K344, S32K388, and S32K389
15. Applicable for S32K322, S32K324, S32K328, S32K338, S32K358, S32K356, S32K388, and S32K389 only.
16. Applicable for S32K338, S32K388, and S32K389 only.
17. Applicable for S32K322, S32K341, S32K342, S32K314, S32K324, and S32K344 only.

Table 264. PRTN2_COFB1_CLKEN[REQ32]

Bit filed	Description
REQ32	Clock enable This bit provides the clock enable control for block 32 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.

18. Applicable for S32K328, S32K338, S32K348, S32K356, and S32K358 only.

Table 265. PRTN2_COFB1_CLKEN[REQ57]

Bit filed	Description
REQ57	Clock enable This bit provides the clock enable control for block 57 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.

46.1.6 Application core initialization process

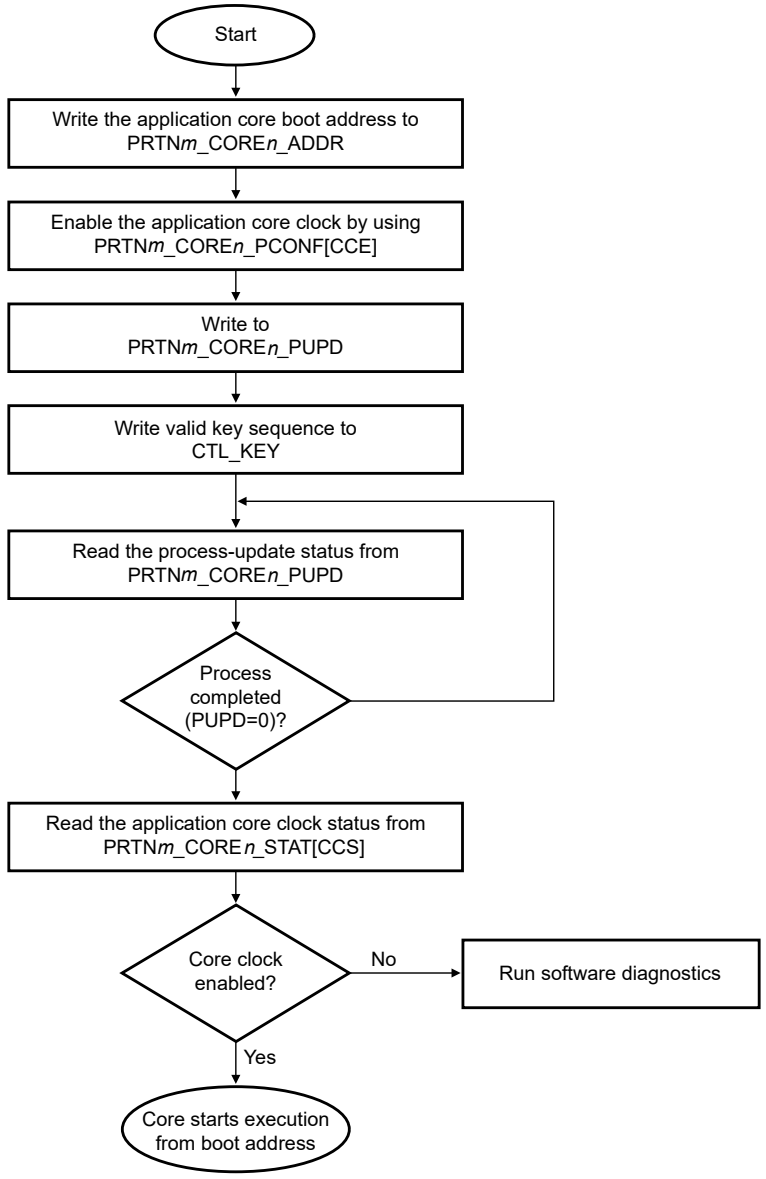


Figure 188. Application core initialization process

46.1.7 Application core shutdown process

If a debugger is attached to the chip and application debug is enabled, the application core continues running if you write 0 to MDM_AP.MDMAPCT[CM7_*n*_CORE_ACCESS].

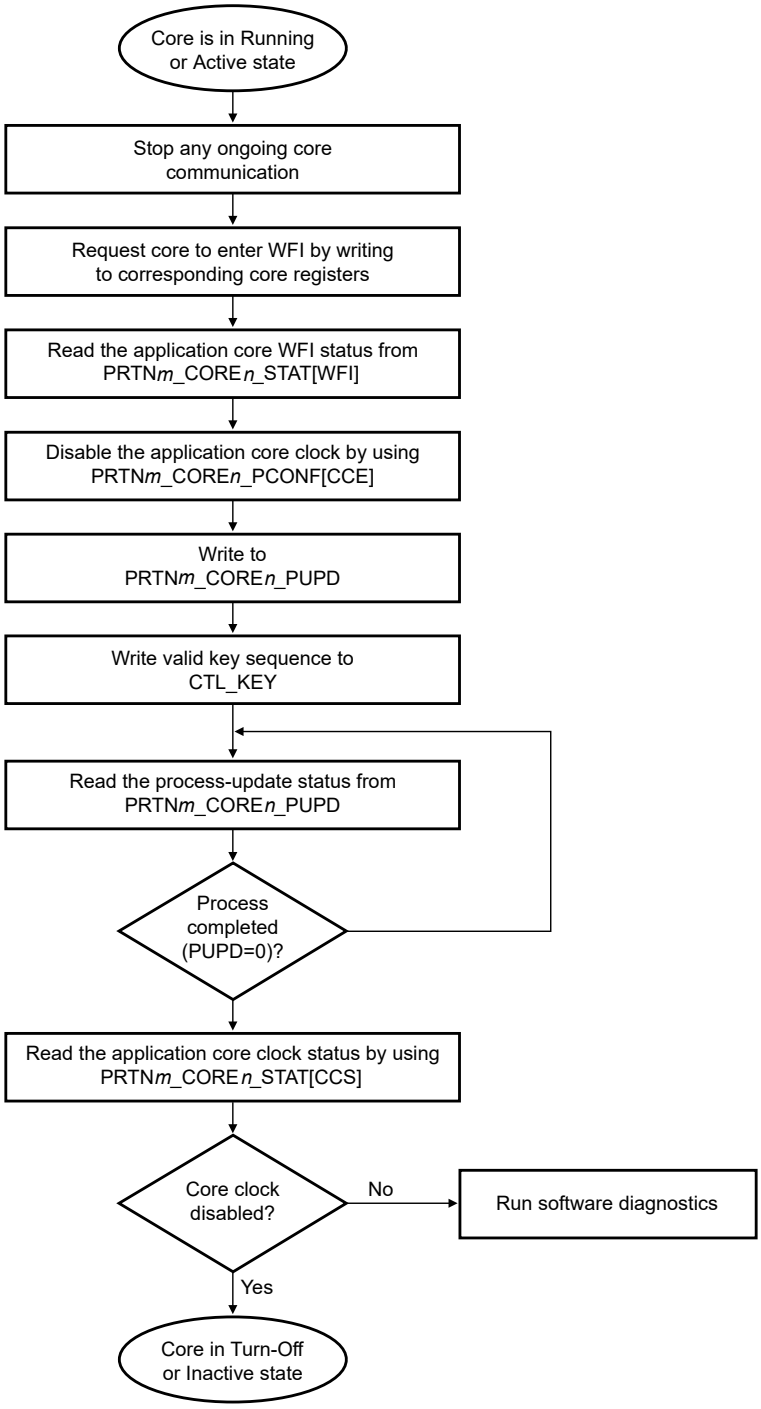


Figure 189. Application core shutdown process

46.1.8 Peripheral initialization process

You cannot control all the peripherals. For example, you cannot turn on and turn off the peripherals required for chip functionality across reset or power-up. They always remain on.

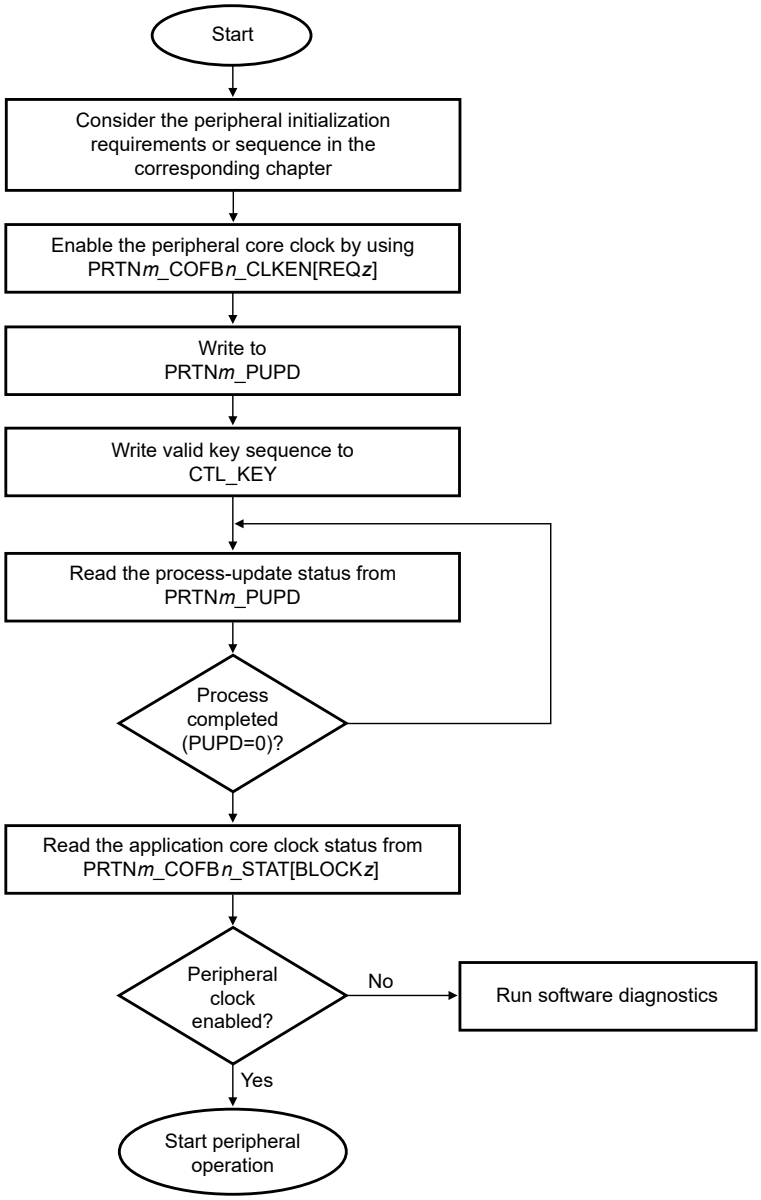


Figure 190. Peripheral initialization process

46.1.9 Peripheral shutdown process

You cannot control all the peripherals. For example, you cannot turn on and turn off the peripherals required for chip functionality across reset or power-up. They always remain on.

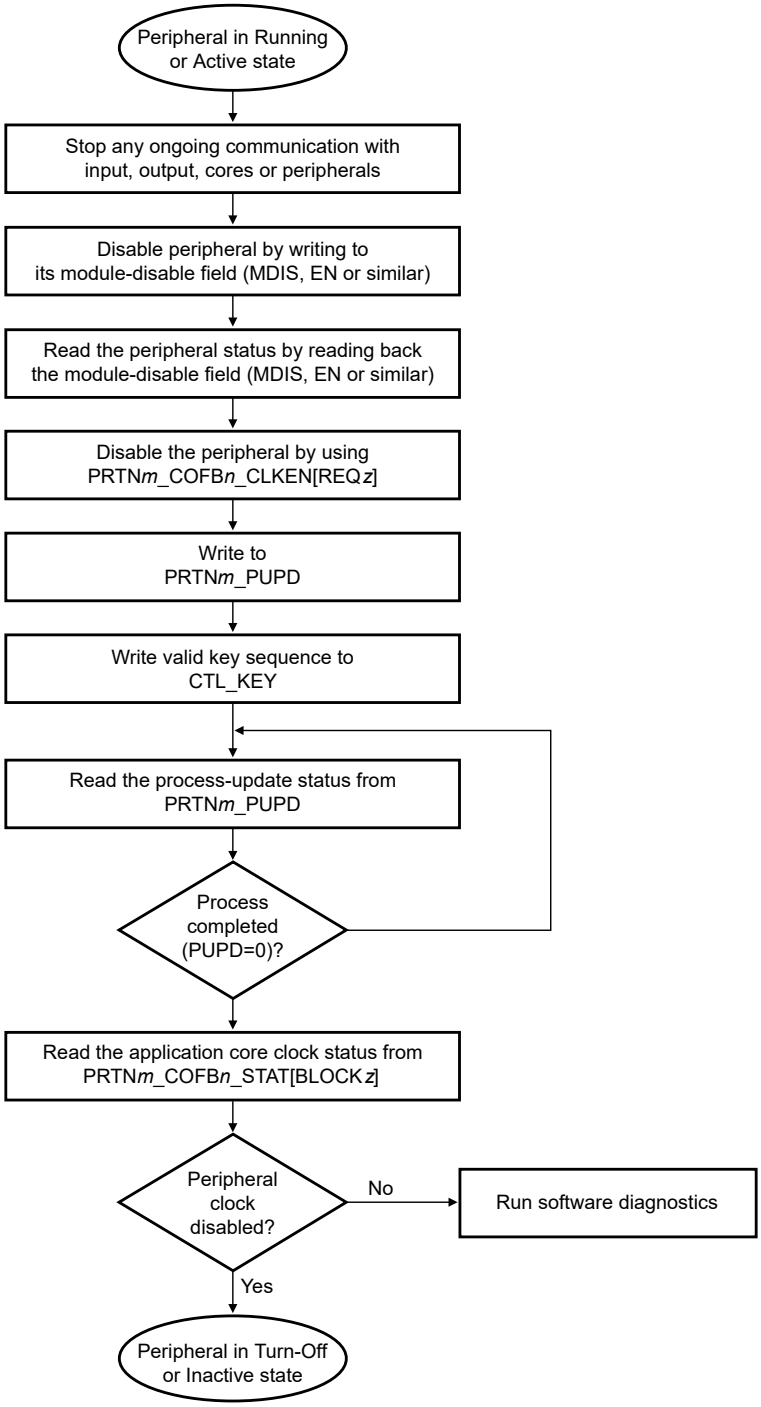


Figure 191. Peripheral shutdown process

46.2 Introduction

The MC_ME module generates control signals for a set of modules of the SoC. The set of signals are defined in corresponding 'Partition Configuration Registers'. It also implements a software-based mechanism for initiating a functional and destructive reset sequence and standby entry handshake with power management of SoC. See Figure 192 for the MC_ME block diagram.

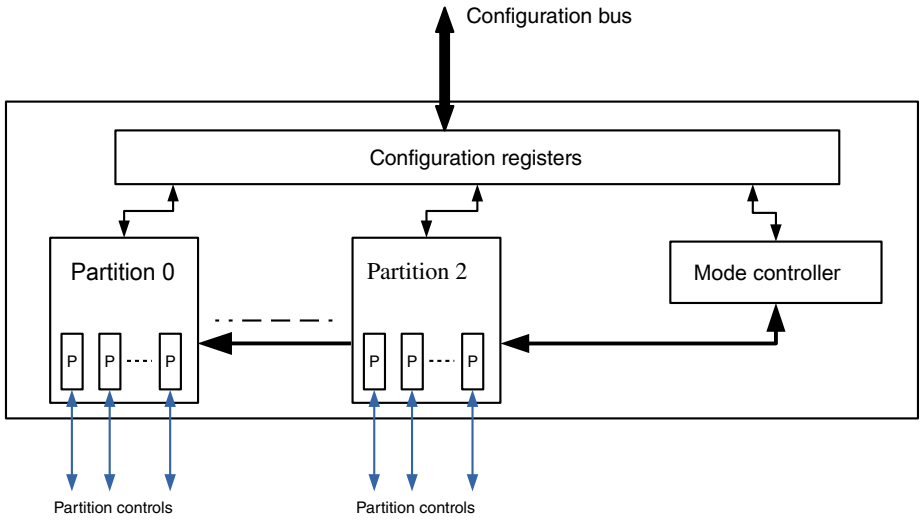


Figure 192. MC_ME block diagram

46.3 Features

MC_ME includes the following features:

- 3 logic partitions implementation and their controls
- Core clock controls
- Partition clock control
- Control mechanism for initiating a destructive or functional reset sequence to MC_RGM
- Control mechanism for initiating standby mode entry for SoC

The logic partition inside MC_ME refers to a certain group of on-chip resources (or IP blocks) that are clubbed together to represent a single 'Partition' inside MC_ME. The MC_ME partition can be the same or different than an LBIST partition. Each of the MC_ME partitions implements a certain number of hardware processes. These hardware processes provide a mechanism to regulate various control signals provided to or received from the IP blocks. The corresponding status signals can also be monitored from MC_ME register(s). Each of the hardware processes is bound to finish in 512 cycles of the MC_ME register configuration clocks. Therefore, the hardware processes are non blocking in nature. Mismatch in the expected versus actual status of any hardware process is controlled by a pre-defined software.

46.4 Partition processes

Each of the processes inside the partition controls register space and corresponds to a control signal provided to that partition. A partition can include a core, or COFBs, or both. The MC_ME hardware processes provide control and status via signals provided to partitions. Each partition can be assigned a signal for control and a signal for status. Each of the control signals implements functionality for the partition. For example, clock gating and peripheral control.

The hardware process can be triggered and monitored using a set of three registers:

- Configuration register; for example, Partition n Process Configuration register
- Update register; for example, Partition n Process Update register
- Status register; for example, Partition n Status register

Similar registers exist for cores inside the partition.

The process setup and triggering procedure is shown in Figure 193. Each of the processes is independent of others and can be triggered or re-triggered in parallel or sequential to other processes. The triggering or re-triggering mechanism remains the same.

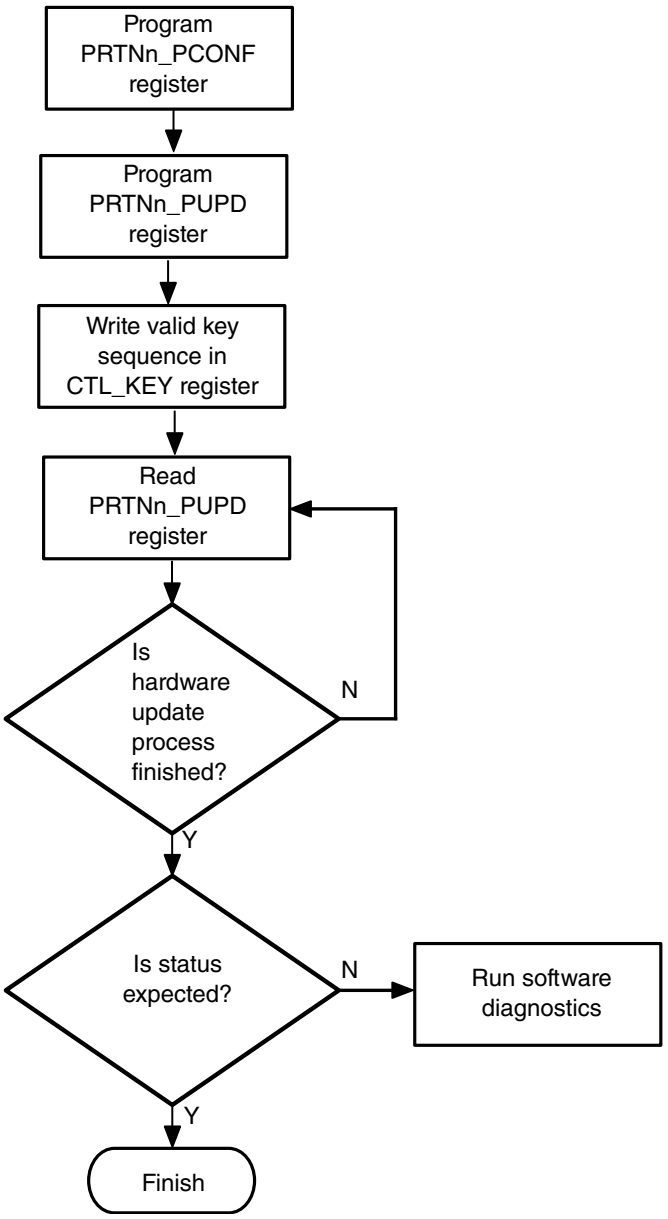


Figure 193. Partition process setup procedure

All the hardware processes are bound to finish in 512 cycles of the MC_ME configuration clock. If the actual and the expected status for a process does not match, then the diagnostics is left as a software responsibility. The software diagnostic can include further wait cycles for the status to match.

46.5 Mode transition

MC_ME implements a mode transition mechanism, whereby the mode of operation for SoC can be changed. Then module implements a mechanism that can lead to:

- Destructive reset
- Functional reset
- Standby mode entry

Destructive reset and functional reset requests from MC_ME are non-retractable transitions. After it is initiated, the other MC_ME functionality is rendered unusable and bus errors are provided for upcoming access to the MC_ME register until a reset sequence is executed by MC_RGM. Hence, it is vital that MC_RGM should never ignore or gate the reset requests from MC_ME.

For transition into the standby mode, the software should ensure that required IP blocks such as clock sources and I/O communication are in their respective inactive states before initiating a standby mode transition to MC_ME. After MC_ME initiates a power down sequence request, it cannot be retracted. The SoC enters a standby power down sequence and then reenter power-up sequence even for cases where the standby wakeup happens right at the time of initiating a power-down sequence request.

Steps for initiating the MC_ME mode transition:

1. Setup the MODE_CONF register with the corresponding target mode bit set to logic-1.
2. Perform the same update as done in the MODE_CONF register on the CONF_UPD register.
3. Write the valid control key (0x5AF0) on the CTL_KEY register.
4. Write the valid invert control key (0xA50F) on the CTL_KEY register.

Mode transition to MC_ME is initiated, after the sequence mentioned above is completed.

In step 1, if both FUNC_RST and DEST_RST in Mode Configuration Register (MODE_CONF) are 1:

- After step 4 is complete, MC_ME initiates a mode transition to a destructive (not functional) reset.
- After the chip exits reset, MC_RGM records that both MC_ME's destructive reset and MC_ME's functional reset were the reset source.

NOTE

Any hardware partition processes setup, along with mode transition, is executed in parallel to the mode transition of MC_ME.

46.6 Standby entry

MC_ME provides hardware processes that implement shutdown sequencing of on-chip resources, such as cores and COFBs. The standby entry sequencing can be achieved or implemented using these hardware processes. The order of the hardware process is determined by the software and MC_ME. It requires no restriction in sequencing of the operation. Following is an example sequence for initiating a power-down sequence for entering the standby mode for SoC. The standby entry sequence should include (but not limited to) the following steps:

1. Setting up wakeup lines
2. Shutting down cores and COFBs
3. Switching all MC_CGM muxes to FIRC with PCFS
4. Powering down all clock sources except FIRC
5. Setting up MC_ME using the main core and initiating a standby mode transition
6. Executing WFI instruction on the main core (per Arm specification)

46.6.1 Application core shutdown

This section describes a mechanism for shutting down an application core. The sequence proposed here is extendible with the housekeeping tasks required for other IPs. Each of the tasks mentioned in the following sequence, can be further integrated with an SoC-specific task.

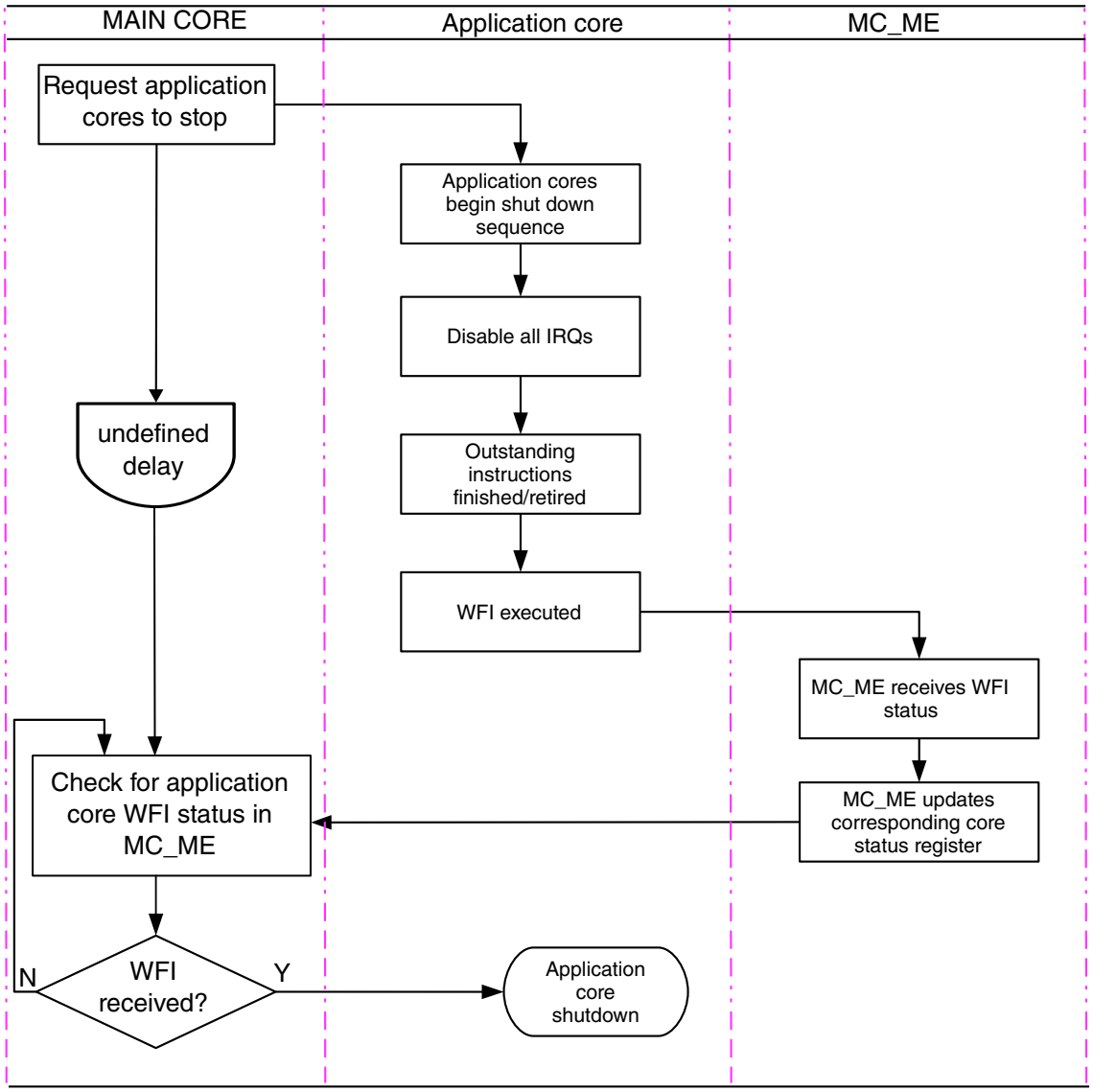


Figure 194. Application core shutdown

After the application core is shutdown, the main core can optionally decide to gate the respective core clock using the corresponding core clock hardware process.

46.6.2 Main core shutdown and standby entry

This section describes standby entry sequence along with the main core shutdown. This sequence should only be initiated after SoC is ready for entering standby and has completed all the housekeeping activities. It is necessary that the main core has completed all the operations pertaining to other (application cores) and is the last active core before initiating the standby entry sequence. See Figure 195.

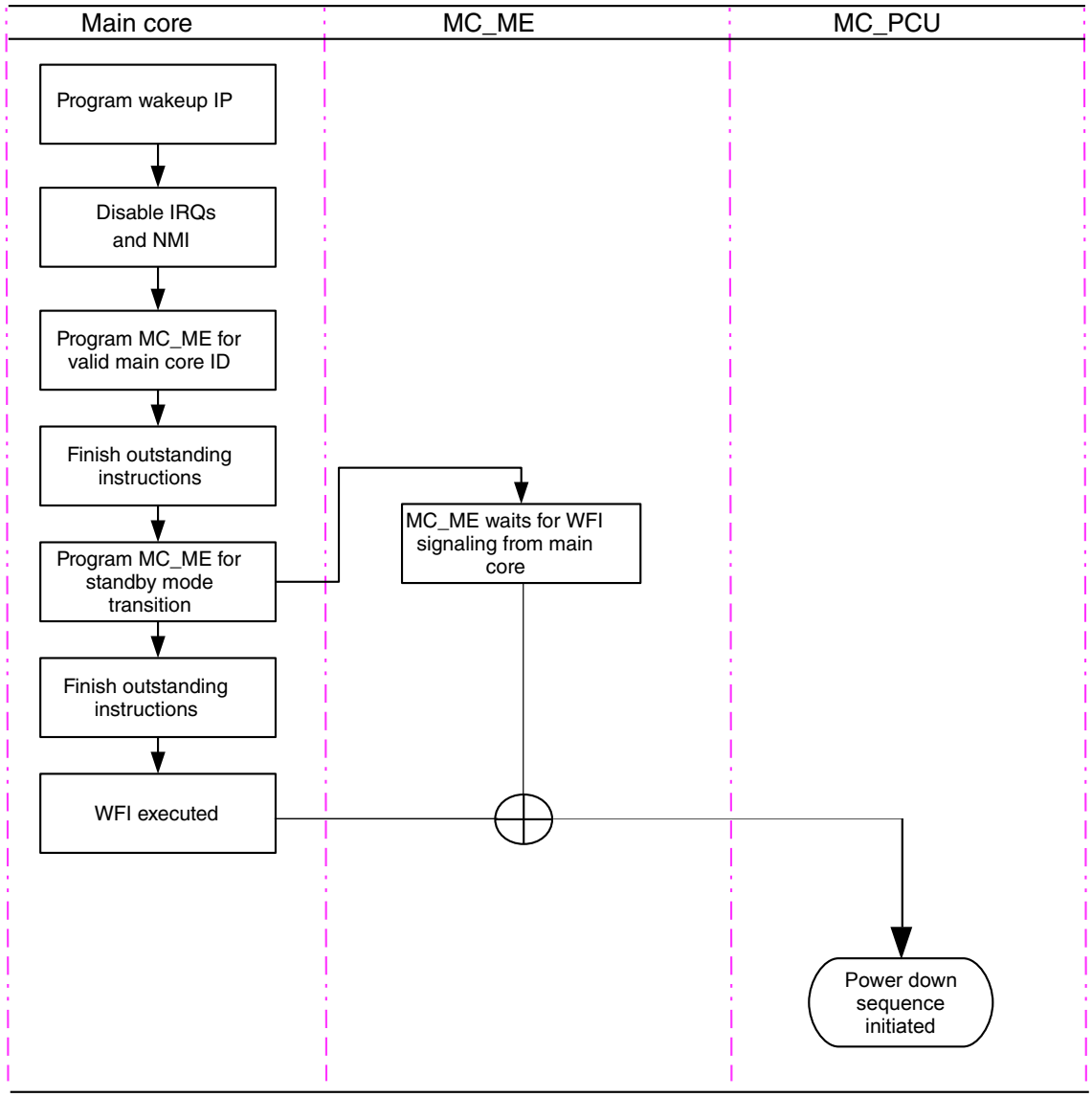


Figure 195. Standby entry sequence along with main core shutdown

NOTE

- MC_ME initiates the power sequence to MC_PCU. This enables the main core to remain inactive (WFI state) until it is reset and power-up again at standby exit.

46.7 MC_ME register descriptions

MC_ME implements set hardware processes that can be used by the software for changing the mode of operation for a partition. Following are the features of MC_ME registers:

- All registers are 32-bit wide.
- Only 32-bit read and write accesses are supported.
- Read/write accesses of less than 32 bits terminate with an error.
- Writes to read-only register fields in writable registers are ignored and do not provide an error message.

- Writes to read-only registers are aborted with an error message.

46.7.1 MC_ME memory map

MC_ME base address: 402D_C000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Control Key Register (CTL_KEY)	32	RW	0000_5AF0h
4h	Mode Configuration Register (MODE_CONF)	32	RW	0000_0000h
8h	Mode Update Register (MODE_UPD)	32	RW	0000_0000h
Ch	Mode Status Register (MODE_STAT)	32	R	0000_0000h
10h	Main Core ID Register (MAIN_COREID)	32	RW	0000_0000h
100h	Partition 0 Process Configuration Register (PRTN0_PCONF)	32	RW	0000_0001h
104h	Partition 0 Process Update Register (PRTN0_PUPD)	32	RW	0000_0000h
108h	Partition 0 Status Register (PRTN0_STAT)	32	R	0000_0001h
10Ch	Partition 0 Core Lockstep Control Register (PRTN0_CORE_LOCKSTEP)	32	RW	0000_0000h
110h	Partition 0 COFB Set 0 Clock Status Register (PRTN0_COFB0_STAT)	32	R	0C00_0004h
114h	Partition 0 COFB Set 1 Clock Status Register (PRTN0_COFB1_STAT)	32	R	0000_1000h
130h	Partition 0 COFB Set 0 Clock Enable Register (PRTN0_COFB0_CLKEN)	32	RW	0C00_0004h
134h	Partition 0 COFB Set 1 Clock Enable Register (PRTN0_COFB1_CLKEN)	32	RW	0000_1000h
140h	Partition 0 Core 0 Process Configuration Register (PRTN0_CORE0_PCONF)	32	RW	0000_0000h
144h	Partition 0 Core 0 Process Update Register (PRTN0_CORE0_PUPD)	32	RW	0000_0000h
148h	Partition 0 Core 0 Status Register (PRTN0_CORE0_STAT)	32	R	0000_0000h
14Ch	Partition 0 Core 0 Address Register (PRTN0_CORE0_ADDR)	32	RW	0040_0000h
160h	Partition 0 Core 1 Process Configuration Register (PRTN0_CORE1_PCONF)	32	RW	0000_0000h
164h	Partition 0 Core 1 Process Update Register (PRTN0_CORE1_PUPD)	32	RW	0000_0000h
168h	Partition 0 Core 1 Status Register (PRTN0_CORE1_STAT)	32	R	0000_0000h
16Ch	Partition 0 Core 1 Address Register (PRTN0_CORE1_ADDR)	32	RW	0041_0000h
188h	Partition 0 Core 2 Status Register (PRTN0_CORE2_STAT)	32	R	0000_0001h
18Ch	Partition 0 Core 2 Address Register (PRTN0_CORE2_ADDR)	32	R	00FF_FC00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1A0h	Partition 0 Core 3 Process Configuration Register (PRTN0_CORE3_PCONF)	32	RW	0000_0000h
1A4h	Partition 0 Core 3 Process Update Register (PRTN0_CORE3_PUPD)	32	RW	0000_0000h
1A8h	Partition 0 Core 3 Status Register (PRTN0_CORE3_STAT)	32	R	0000_0000h
1ACh	Partition 0 Core 3 Address Register (PRTN0_CORE3_ADDR)	32	RW	0043_0000h
1C0h	Partition 0 Core 4 Process Configuration Register (PRTN0_CORE4_PCONF)	32	RW	0000_0000h
1C4h	Partition 0 Core 4 Process Update Register (PRTN0_CORE4_PUPD)	32	RW	0000_0000h
1C8h	Partition 0 Core 4 Status Register (PRTN0_CORE4_STAT)	32	R	0000_0000h
1CCh	Partition 0 Core 4 Address Register (PRTN0_CORE4_ADDR)	32	RW	0042_0000h
1E0h	Partition 0 Core 5 Process Configuration Register (PRTN0_CORE5_PCONF)	32	RW	0000_0000h
1E4h	Partition 0 Core 5 Process Update Register (PRTN0_CORE5_PUPD)	32	RW	0000_0000h
1E8h	Partition 0 Core 5 Status Register (PRTN0_CORE5_STAT)	32	R	0000_0000h
1ECh	Partition 0 Core 5 Address Register (PRTN0_CORE5_ADDR)	32	RW	0042_0000h
300h	Partition 1 Process Configuration Register (PRTN1_PCONF)	32	RW	0000_0001h
304h	Partition 1 Process Update Register (PRTN1_PUPD)	32	RW	0000_0000h
308h	Partition 1 Status Register (PRTN1_STAT)	32	R	0000_0001h
310h	Partition 1 COFB Set 0 Clock Status Register (PRTN1_COFB0_STAT)	32	R	5E3F_0007h
314h	Partition 1 COFB Set 1 Clock Status Register (PRTN1_COFB1_STAT)	32	R	7CFE_2FFCh
318h	Partition 1 COFB Set 2 Clock Status Register (PRTN1_COFB2_STAT)	32	R	300C_0000h
31Ch	Partition 1 COFB Set 3 Clock Status Register (PRTN1_COFB3_STAT)	32	R	0000_5FEEh
330h	Partition 1 COFB Set 0 Clock Enable Register (PRTN1_COFB0_CLKEN)	32	RW	5E3F_0007h
334h	Partition 1 COFB Set 1 Clock Enable Register (PRTN1_COFB1_CLKEN)	32	RW	7CFE_2FFCh
338h	Partition 1 COFB Set 2 Clock Enable Register (PRTN1_COFB2_CLKEN)	32	RW	300C_0000h
33Ch	Partition 1 COFB Set 3 Clock Enable Register (PRTN1_COFB3_CLKEN)	32	RW	0000_5FEEh

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
500h	Partition 2 Process Configuration Register (PRTN2_PCONF)	32	RW	0000_0001h
504h	Partition 2 Process Update Register (PRTN2_PUPD)	32	RW	0000_0000h
508h	Partition 2 Status Register (PRTN2_STAT)	32	R	0000_0001h
510h	Partition 2 COFB Set 0 Clock Status Register (PRTN2_COFB0_STAT)	32	R	0600_000Fh
514h	Partition 2 COFB Set 1 Clock Status Register (PRTN2_COFB1_STAT)	32	R	CC00_0000h
518h	Partition 2 COFB Set 2 Clock Status Register (PRTN2_COFB2_STAT)	32	R	0000_0003h
51Ch	Partition 2 COFB Set 3 Clock Status Register (PRTN2_COFB3_STAT)	32	R	0000_0007h
530h	Partition 2 COFB Set 0 Clock Enable Register (PRTN2_COFB0_CLKEN)	32	RW	0600_000Fh
534h	Partition 2 COFB Set 1 Clock Enable Register (PRTN2_COFB1_CLKEN)	32	RW	CC00_0000h
538h	Partition 2 COFB Set 2 Clock Enable Register (PRTN2_COFB2_CLKEN)	32	RW	0000_0003h

46.7.2 Control Key Register (CTL_KEY)

Offset

Register	Offset
CTL_KEY	0h

Function

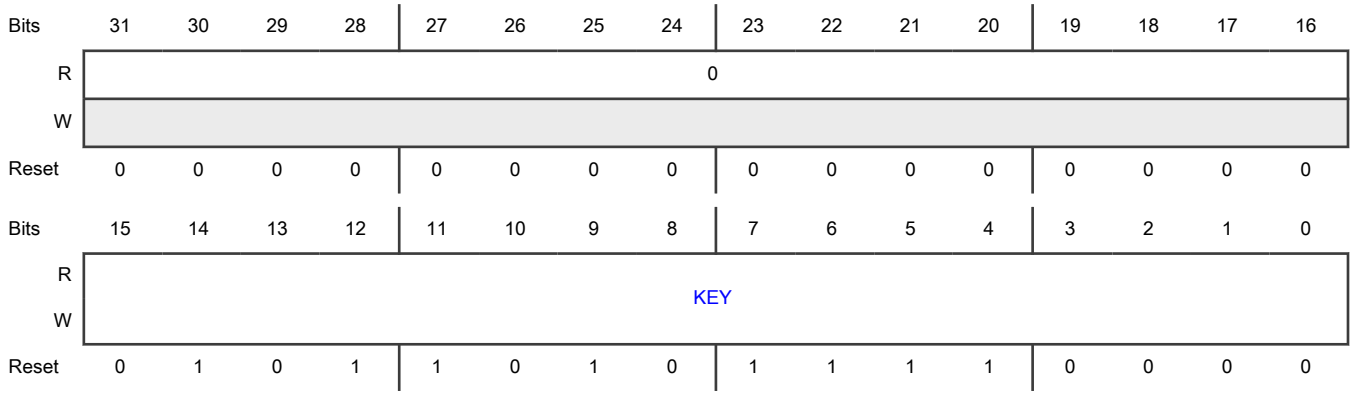
This register provides the mechanism to MC_ME for starting the hardware processes for the partition(s) and standby entry sequence. The hardware processes for partitions are triggered through the corresponding PRTNn_PCONF register. The mechanism to trigger the hardware processes of the respective partitions require two write operations: first time with key and second time with inverted key. The hexadecimal value of key is 0x5AF0 whereas for inverted key is 0xA50F.

For initiating a standby entry sequence, the MODE_CONF register is used for providing a standby entry request along with a valid key combination.

NOTE

Reads from this register return a valid key value to be written next.

Diagram



Fields

Field	Function
31-16 —	Reserved This field is reserved and read returns zeros.
15-0 KEY	Control key Key for starting the hardware processes. Writes with a value other than key or inverted key are ignored. Reads return bit inverted value corresponding to last write.

46.7.3 Mode Configuration Register (MODE_CONF)

Offset

Register	Offset
MODE_CONF	4h

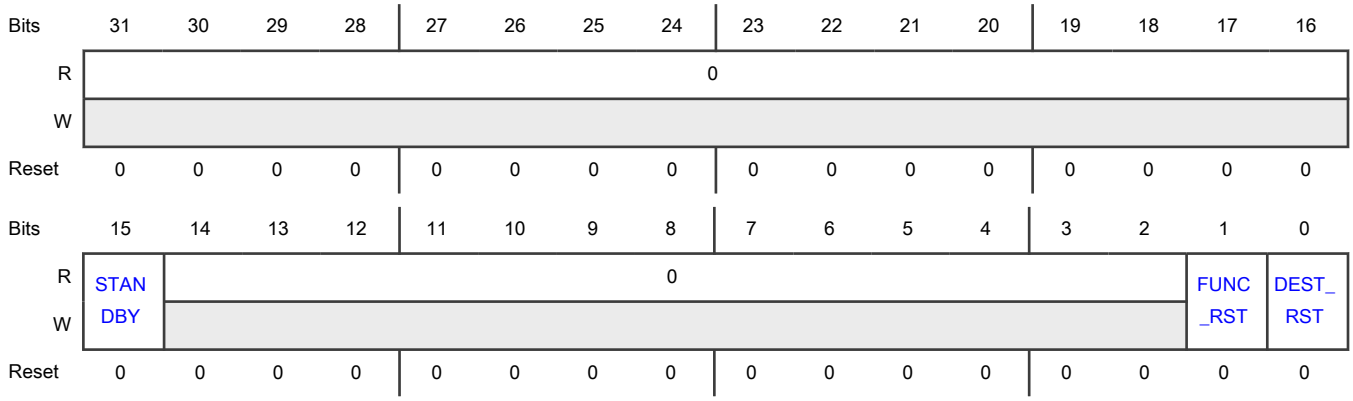
Function

This register is used for initiating a standby request or a reset event (destructive or functional) for the chip. The functional or destructive events are signaled to MC_RGM for further handling.

NOTE

Software must not enable mode entry if the value of multiple fields is 1 in the MODE_CONF register.

Diagram



Fields

Field	Function
31-16 —	Reserved This field is reserved and read returns zeros.
15 STANDBY	Standby request Writing a logic-1 to this bit along with the MODE_UPD register configuration and followed with a valid key combination makes a standby entry sequence request to MC_ME.
14-2 —	Reserved This field is reserved and read returns zeros.
1 FUNC_RST	Functional reset request Writing a logic-1 to this bit along with the MODE_UPD register configuration and followed with a valid key combination makes a functional reset event signaling to MC_RGM.
0 DEST_RST	Destructive reset request Writing a logic-1 to this bit along with the MODE_UPD register configuration and followed with a valid key combination makes a destructive reset event signaling to MC_RGM.

46.7.4 Mode Update Register (MODE_UPD)

Offset

Register	Offset
MODE_UPD	8h

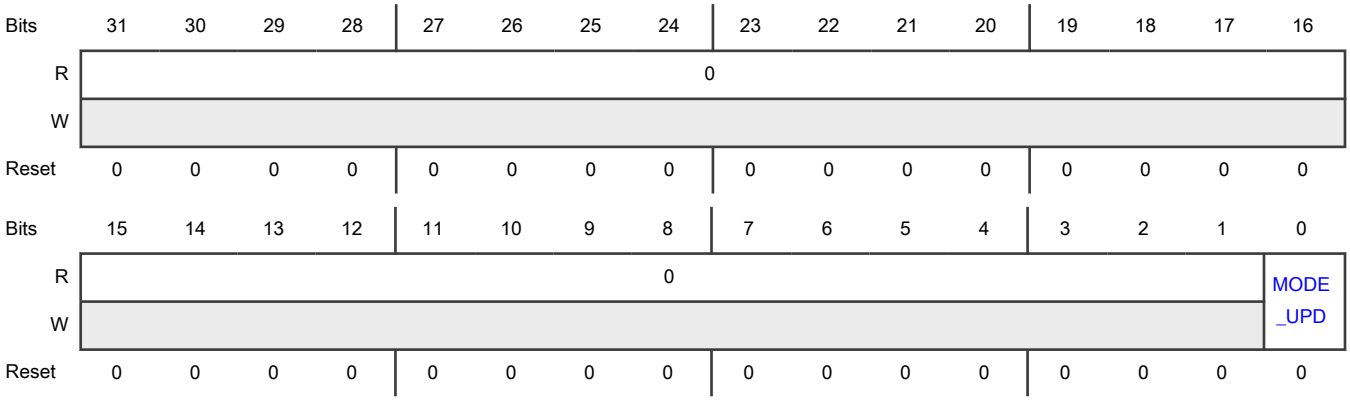
Function

This register is used for initiating a mode change. Mode change refers to initiating a standby request, or generating a destructive or functional reset event to MC_RGM. Setting mode update field to logic-1, along with programming MODE_CONF registers and then followed by a valid key combination will generate a mode transition request.

NOTE

The MODE_UPD register is implemented to make mode transition programming model the same as partition programming model. This is for future expansion inside MC_ME.

Diagram



Fields

Field	Function
31-1 —	Reserved This field is reserved and read returns zeros.
0 MODE_UPD	Mode update Writing a logic-1 to this bit, followed by a valid key combination initiates a mode change as per the MODE_CONF register.

46.7.5 Mode Status Register (MODE_STAT)

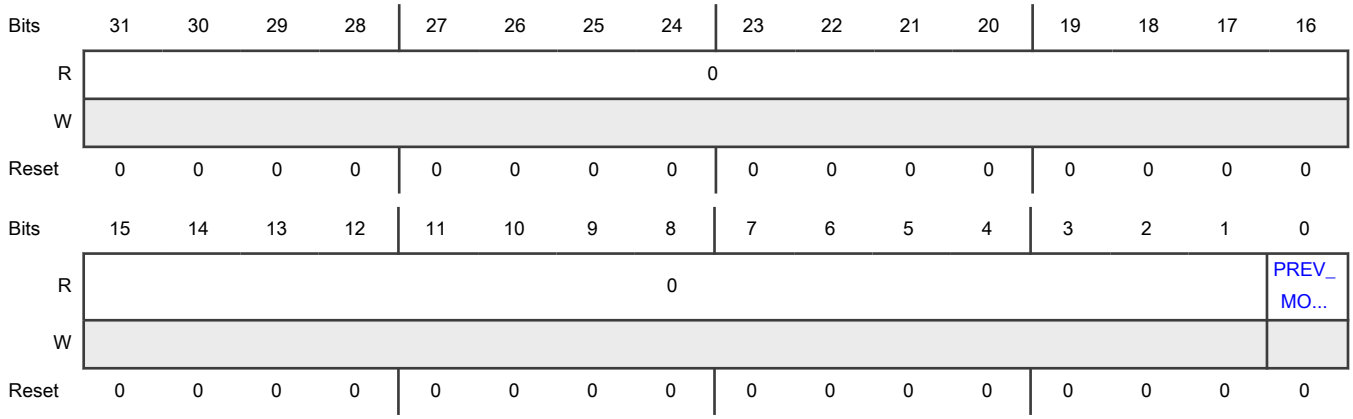
Offset

Register	Offset
MODE_STAT	Ch

Function

This register provides the status of the previous mode. In case of standby exit, if the reset event status register of MC_RGM are set, then contents of this register should be ignored.

Diagram



Fields

Field	Function
31-1 —	Reserved This field is reserved and read returns zeros.
0 PREV_MODE	Previous mode This bit shows the status of the previous mode. 0b - The previous mode was reset (any reset). 1b - The previous mode was standby.

46.7.6 Main Core ID Register (MAIN_COREID)

Offset

Register	Offset
MAIN_COREID	10h

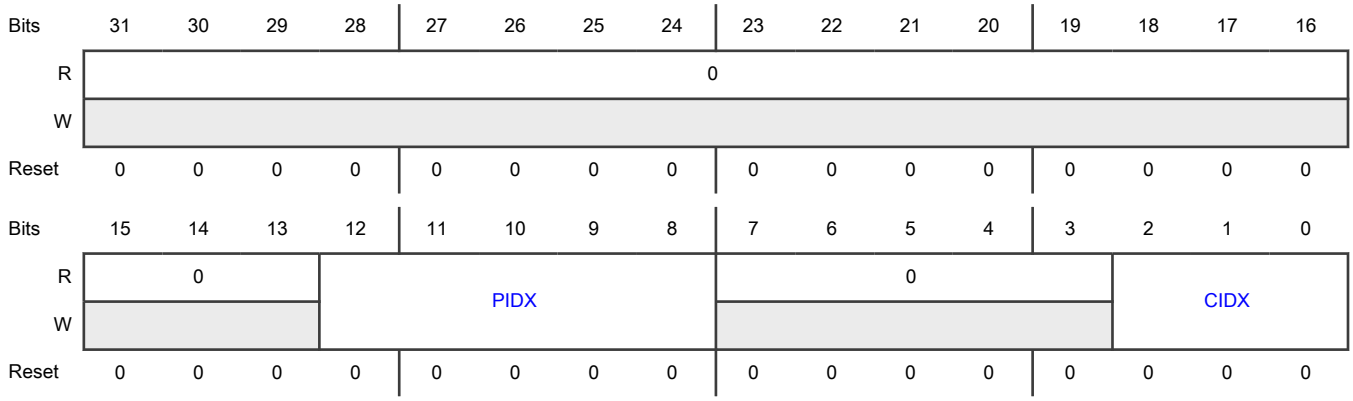
Function

This register provides the ID of the main core sequencing the operation for the standby sequence. Core ID is required for entering in the standby mode, and using this MC_ME locates the WFI instruction execution of the main core. The core ID in this register is specified by the partition index along with the core index.

NOTE

Before initiating a standby entry sequence, the contents of this register should point to the correct main core. Providing non-existing or incorrect core ID leads to unpredictable hardware behavior.

Diagram



Fields

Field	Function
31-13 —	Reserved This field is reserved and read returns zeros.
12-8 PIDX	Partition index Provides the partition index of the main core. Only values 0 - 2 can be written.
7-3 —	Reserved This field is reserved and read returns zeros.
2-0 CIDX	Core index Provides the core index of the main core inside the partition.

46.7.7 Partition 0 Process Configuration Register (PRTN0_PCONF)

Offset

Register	Offset
PRTN0_PCONF	100h

Function

This register provides a configuration for the hardware processes corresponding to partition 0. Each of the configuration bit corresponds to the 'nature' of the processes; for example, enabling/disabling and the trigger is controlled by the corresponding field in the PRTN0_PUPD register. When valid KEY combinations are written onto the CTL_KEY register, the PRTN0_PCONF and PRTN0_PUPD registers are used to determine the hardware processes to be executed. These are triggered in parallel and independent of each other. All dependent processes should be requested one after another from the software.

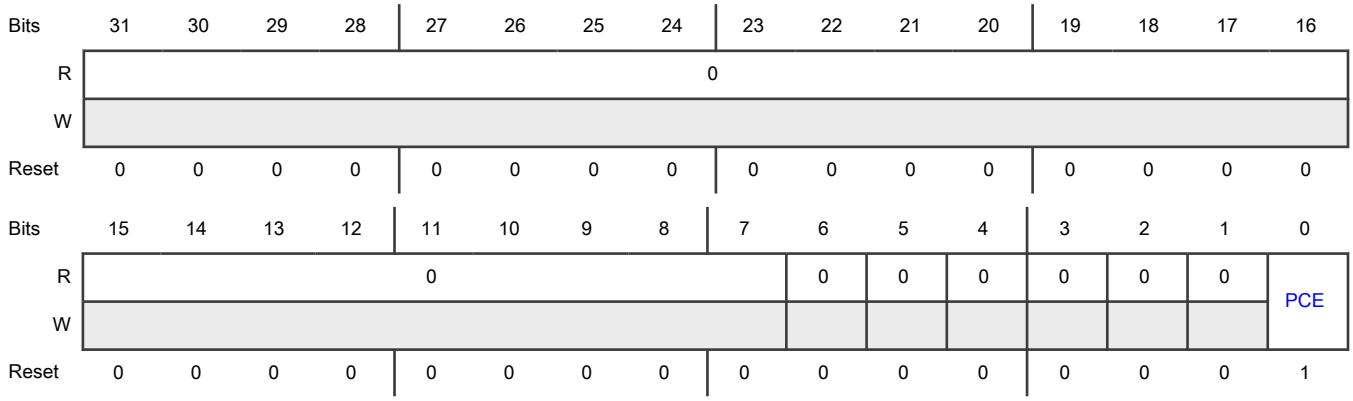
NOTE

The partition clock enable/disable are not standalone and must be done coherently in a fixed sequence. For details, see Software Reset Partition Turn-On Flow Chart and Software reset partition turn-off flowchart in Reset chapter.

NOTE

See chip-specific MC_ME information to check if this register is implemented on chip.

Diagram



Fields

Field	Function
31-7 —	Reserved This field is reserved and read returns zeros.
6 —	Reserved This field is reserved and read returns zeros.
5 —	Reserved This field is reserved and read returns zeros.
4 —	Reserved This field is reserved and read returns zeros.
3 —	Reserved This field is reserved and read returns zeros.
2 —	Reserved This field is reserved and read returns zeros.
1 —	Reserved This field is reserved and read returns zeros.
0 PCE	Partition clock enable This bit controls whether the clock to IPs (other than core(s)) in the partition should be enabled or disabled. 0b - Disable the clock to IPs 1b - Enable the clock to IPs

46.7.8 Partition 0 Process Update Register (PRTN0_PUPD)

Offset

Register	Offset
PRTN0_PUPD	104h

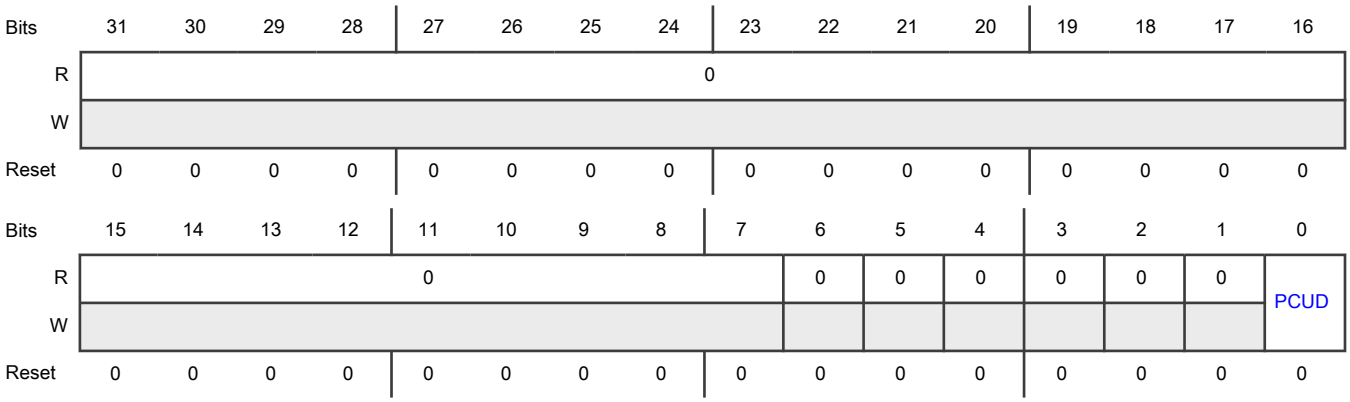
Function

This register provides trigger signaling for the hardware processes corresponding to partition 0. Each of the control bit acts as a trigger for the corresponding hardware processes. When valid KEY combinations are written onto the CTL_KEY register, the hardware checks the bit fields that are programmed as logic-1 in this register, and then triggers the hardware process per the value in the corresponding bit field in the PRTN0_PCONF register. When the hardware process is finished the corresponding bit in this register is auto-cleared to logic-0.

NOTE

See chip-specific MC_ME information to check if this register is implemented on chip.

Diagram



Fields

Field	Function
31-7 —	Reserved This field is reserved and read returns zeros.
6 —	Reserved This field is reserved and read returns zeros.
5 —	Reserved This field is reserved and read returns zeros.
4 —	Reserved This field is reserved and read returns zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
3 —	Reserved This field is reserved and read returns zeros.
2 —	Reserved This field is reserved and read returns zeros.
1 —	Reserved This field is reserved and read returns zeros.
0 PCUD	Partition clock update This bit controls whether the hardware processes for enabling/disabling the clock to IPs (other than core(s)) in the partition should be triggered or not. 0b - Do not trigger the hardware process 1b - Trigger the hardware process

46.7.9 Partition 0 Status Register (PRTN0_STAT)

Offset

Register	Offset
PRTN0_STAT	108h

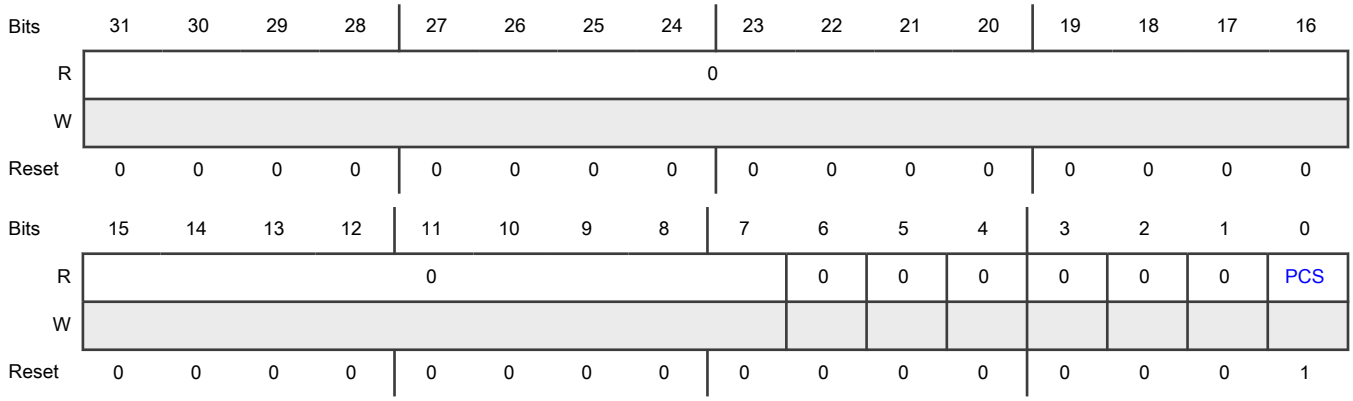
Function

This register provides the current status of the control signals from the partition 0.

NOTE

See chip-specific MC_ME information to check if this register is implemented on chip.

Diagram



Fields

Field	Function
31-7 —	Reserved This field is reserved and read returns zeros.
6 —	Reserved This field is reserved and read returns zeros.
5 —	Reserved This field is reserved and read returns zeros.
4 —	Reserved This field is reserved and read returns zeros.
3 —	Reserved This field is reserved and read returns zeros.
2 —	Reserved This field is reserved and read returns zeros.
1 —	Reserved This field is reserved and read returns zeros.
0 PCS	Partition clock status This bit provides the status of the clock to partition. 0b - Clock is inactive 1b - Clock is active

46.7.10 Partition 0 Core Lockstep Control Register (PRTN0_CORE_LOCKSTEP)

Offset

Register	Offset
PRTN0_CORE_LOCKSTEP	10Ch

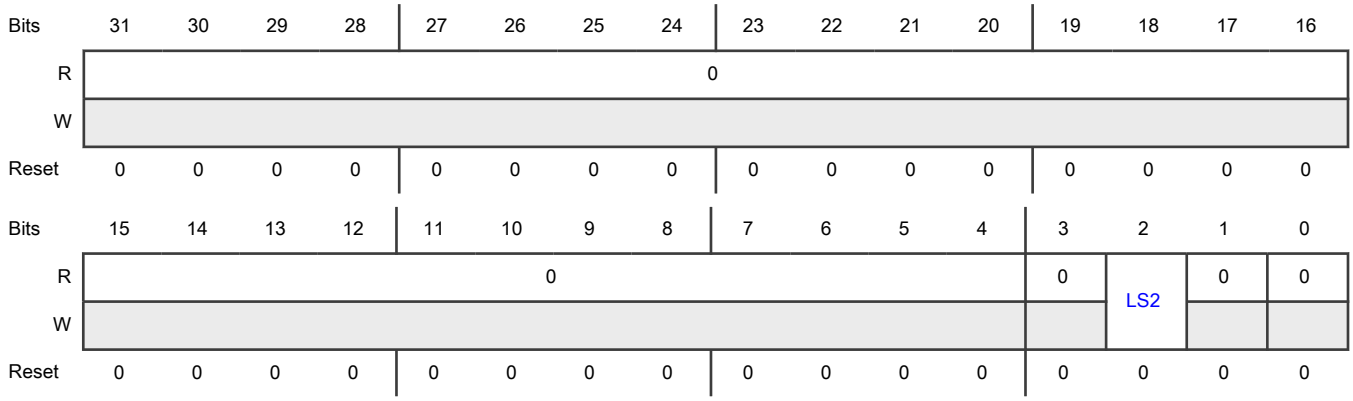
Function

This register provides the control for indicating a set of cores for lockstep execution in partition 0. Writes to this register immediately provide the corresponding bit to the described cores. No hardware process updates are required.

NOTE

See chip-specific MC_ME information to check if this register is implemented on chip.

Diagram



Fields

Field	Function
31-4 —	Reserved This field is reserved and read returns zeros.
3 —	Reserved
2 LS2	Lockstep 2 This bit provides the lockstep indication to Core 4 & Core 5 in partition 0. 0b - Lockstep disabled 1b - Lockstep enabled
1 —	Reserved
0 —	Reserved

46.7.11 Partition 0 COFB Set 0 Clock Status Register (PRTN0_COFB0_STAT)

Offset

Register	Offset
PRTN0_COFB0_STAT	110h

Function

This register provides the status of set 0 of COFBs inside partition 0.

NOTE

The reset value of this register can vary depending on the availability of active clock pulses inside partition 0.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	BLOCK28	BLOCK27	BLOCK26	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	BLOCK3	BLOCK2	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Fields

Field	Function
31 —	Reserved This field is reserved and read returns zeros.
30 —	Reserved This field is reserved and read returns zeros.
29 —	Reserved This field is reserved and read returns zeros.
28 BLOCK28	IP block status This bit provides the clock status of SWT 3 in partition 0. 0b - Clock is not running. 1b - Clock is running.
27 BLOCK27	IP block status This bit provides the clock status of block 27 in partition 0. 0b - Clock is not running. 1b - Clock is running.
26 BLOCK26	IP block status This bit provides the clock status of block 26 in partition 0. 0b - Clock is not running. 1b - Clock is running.
25 —	Reserved This field is reserved and read returns zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
24 —	Reserved This field is reserved and read returns zeros.
23 —	Reserved This field is reserved and read returns zeros.
22 —	Reserved This field is reserved and read returns zeros.
21 —	Reserved This field is reserved and read returns zeros.
20 —	Reserved This field is reserved and read returns zeros.
19 —	Reserved This field is reserved and read returns zeros.
18 —	Reserved This field is reserved and read returns zeros.
17 —	Reserved This field is reserved and read returns zeros.
16 —	Reserved This field is reserved and read returns zeros.
15 —	Reserved This field is reserved and read returns zeros.
14 —	Reserved This field is reserved and read returns zeros.
13 —	Reserved This field is reserved and read returns zeros.
12 —	Reserved This field is reserved and read returns zeros.
11 —	Reserved This field is reserved and read returns zeros.
10	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	This field is reserved and read returns zeros.
9 —	Reserved This field is reserved and read returns zeros.
8 —	Reserved This field is reserved and read returns zeros.
7 —	Reserved This field is reserved and read returns zeros.
6 —	Reserved This field is reserved and read returns zeros.
5 —	Reserved This field is reserved and read returns zeros.
4 —	Reserved This field is reserved and read returns zeros.
3 BLOCK3	IP block status This bit provides the clock status of ERM1 in partition 0. 0b - Clock is not running. 1b - Clock is running.
2 BLOCK2	IP block status This bit provides the clock status of block 2 in partition 0. 0b - Clock is not running. 1b - Clock is running.
1 —	Reserved This field is reserved and read returns zeros.
0 —	Reserved This field is reserved and read returns zeros.

46.7.12 Partition 0 COFB Set 1 Clock Status Register (PRTN0_COFB1_STAT)

Offset

Register	Offset
PRTN0_COFB1_STAT	114h

Function

This register provides the status of set 1 of COFBs inside partition 0.

NOTE

The reset value of this register can vary depending on the availability of active clock pulses inside partition 0.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	BLOCK52	BLOCK51	BLOCK50	BLOCK49	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BLOCK47	BLOCK46	BLOCK45	BLOCK44	0	BLOCK42	BLOCK41	BLOCK40	BLOCK39	BLOCK38	0	BLOCK36	BLOCK35	BLOCK34	BLOCK33	BLOCK32
W																
Reset	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved This field is reserved and read returns zeros.
30 —	Reserved This field is reserved and read returns zeros.
29 —	Reserved This field is reserved and read returns zeros.
28 —	Reserved This field is reserved and read returns zeros.
27 —	Reserved This field is reserved and read returns zeros.
26 —	Reserved This field is reserved and read returns zeros.
25 —	Reserved This field is reserved and read returns zeros.
24	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	This field is reserved and read returns zeros.
23 —	Reserved This field is reserved and read returns zeros.
22 —	Reserved This field is reserved and read returns zeros.
21 —	Reserved This field is reserved and read returns zeros.
20 BLOCK52	IP block status This bit provides the clock status of MU_4_MUB in partition 0. 0b - Clock is not running. 1b - Clock is running.
19 BLOCK51	IP block status This bit provides the clock status of MU_4_MUA in partition 0. 0b - Clock is not running. 1b - Clock is running.
18 BLOCK50	IP block status This bit provides the clock status of MU_3_MUB in partition 0. 0b - Clock is not running. 1b - Clock is running.
17 BLOCK49	IP block status This bit provides the clock status of MU_3_MUA in partition 0. 0b - Clock is not running. 1b - Clock is running.
16 —	Reserved This field is reserved and read returns zeros.
15 BLOCK47	IP block status This bit provides the clock status of MU_2_MUB in partition 0. 0b - Clock is not running. 1b - Clock is running.
14	IP block status

Table continues on the next page...

Table continued from the previous page...

Field	Function
BLOCK46	This bit provides the clock status of MU_2_MUA in partition 0. 0b - Clock is not running. 1b - Clock is running.
13 BLOCK45	IP block status This bit provides the clock status of PIT_1 in partition 0. 0b - Clock is not running. 1b - Clock is running.
12 BLOCK44	IP block status This bit provides the clock status of PIT_0 in partition 0. 0b - Clock is not running. 1b - Clock is running.
11 —	Reserved This field is reserved and read returns zeros.
10 BLOCK42	IP block status This bit provides the clock status of ADC_2 in partition 0. 0b - Clock is not running. 1b - Clock is running.
9 BLOCK41	IP block status This bit provides the clock status of ADC_1 in partition 0. 0b - Clock is not running. 1b - Clock is running.
8 BLOCK40	IP block status This bit provides the clock status of ADC_0 in partition 0. 0b - Clock is not running. 1b - Clock is running.
7 BLOCK39	IP block status This bit provides the clock status of LCU_1 in partition 0. 0b - Clock is not running. 1b - Clock is running.
6 BLOCK38	IP block status This bit provides the clock status of LCU_0 in partition 0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Clock is not running. 1b - Clock is running.
5 —	Reserved This field is reserved and read returns zeros.
4 BLOCK36	IP block status This bit provides the clock status of eMIOS_2 in partition 0. 0b - Clock is not running. 1b - Clock is running.
3 BLOCK35	IP block status This bit provides the clock status of eMIOS_1 in partition 0. 0b - Clock is not running. 1b - Clock is running.
2 BLOCK34	IP block status This bit provides the clock status of eMIOS_0 in partition 0. 0b - Clock is not running. 1b - Clock is running.
1 BLOCK33	IP block status This bit provides the clock status of BCTU in partition 0. 0b - Clock is not running. 1b - Clock is running.
0 BLOCK32	IP block status This bit provides the clock status of TRGMUX in partition 0. 0b - Clock is not running. 1b - Clock is running.

46.7.13 Partition 0 COFB Set 0 Clock Enable Register (PRTN0_COFB0_CLKEN)

Offset

Register	Offset
PRTN0_COFB0_CLKEN	130h

Function

This register provides clock control signaling to the individual COFBs in set 0 inside partition 0. Whenever a partition clock enable (non-core) hardware process is initiated, the value of logic-1 in the corresponding bit locations of this register enables the clock to the corresponding block in the partition.

NOTE

The reset value of this register is not defined and is as per the availability of the clock source. See Chip-specific MC_ME information for clock source availability.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	REQ28	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	REQ3	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Fields

Field	Function
31 —	Reserved This field is reserved and read returns zeros.
30 —	Reserved This field is reserved and read returns zeros.
29 —	Reserved This field is reserved and read returns zeros.
28 REQ28	Clock enable This bit provides the clock enable control for SWT 3 in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.
27 —	Reserved This field is reserved and read returns zeros.
26 —	Reserved This field is reserved and read returns zeros.
25	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	This field is reserved and read returns zeros.
24 —	Reserved This field is reserved and read returns zeros.
23 —	Reserved This field is reserved and read returns zeros.
22 —	Reserved This field is reserved and read returns zeros.
21 —	Reserved This field is reserved and read returns zeros.
20 —	Reserved This field is reserved and read returns zeros.
19 —	Reserved This field is reserved and read returns zeros.
18 —	Reserved This field is reserved and read returns zeros.
17 —	Reserved This field is reserved and read returns zeros.
16 —	Reserved This field is reserved and read returns zeros.
15 —	Reserved This field is reserved and read returns zeros.
14 —	Reserved This field is reserved and read returns zeros.
13 —	Reserved This field is reserved and read returns zeros.
12 —	Reserved This field is reserved and read returns zeros.
11 —	Reserved This field is reserved and read returns zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
10 —	Reserved This field is reserved and read returns zeros.
9 —	Reserved This field is reserved and read returns zeros.
8 —	Reserved This field is reserved and read returns zeros.
7 —	Reserved This field is reserved and read returns zeros.
6 —	Reserved This field is reserved and read returns zeros.
5 —	Reserved This field is reserved and read returns zeros.
4 —	Reserved This field is reserved and read returns zeros.
3 REQ3	Clock enable This bit provides the clock enable control for ERM1 in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.
2 —	Reserved This field is reserved and read returns zeros.
1 —	Reserved This field is reserved and read returns zeros.
0 —	Reserved This field is reserved and read returns zeros.

46.7.14 Partition 0 COFB Set 1 Clock Enable Register (PRTN0_COFB1_CLKEN)

Offset

Register	Offset
PRTN0_COFB1_CLKEN	134h

Function

This register provides clock control signaling to the individual COFBs in set 1 inside partition 0. Whenever a partition clock enable (non-core) hardware process is initiated, the value of logic-1 in the corresponding bit locations of this register enables the clock to the corresponding block in the partition.

NOTE

The reset value of this register is not defined and is as per the availability of the clock source. See Chip-specific MC_ME information for clock source availability.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	REQ52	REQ51	REQ50	REQ49	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	REQ47	REQ46	REQ45	REQ44	0	REQ42	REQ41	REQ40	REQ39	REQ38	0	REQ36	REQ35	REQ34	REQ33	REQ32
W																
Reset	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved This field is reserved and read returns zeros.
30 —	Reserved This field is reserved and read returns zeros.
29 —	Reserved This field is reserved and read returns zeros.
28 —	Reserved This field is reserved and read returns zeros.
27 —	Reserved This field is reserved and read returns zeros.
26 —	Reserved This field is reserved and read returns zeros.
25 —	Reserved This field is reserved and read returns zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
24 —	Reserved This field is reserved and read returns zeros.
23 —	Reserved This field is reserved and read returns zeros.
22 —	Reserved This field is reserved and read returns zeros.
21 —	Reserved This field is reserved and read returns zeros.
20 REQ52	Clock enable This bit provides the clock enable control for MU_4_MUB in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.
19 REQ51	Clock enable This bit provides the clock enable control for MU_4_MUA in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.
18 REQ50	Clock enable This bit provides the clock enable control for MU_3_MUB in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.
17 REQ49	Clock enable This bit provides the clock enable control for MU_3_MUA in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.
16 —	Reserved This field is reserved and read returns zeros.
15 REQ47	Clock enable This bit provides the clock enable control for MU_2_MUB in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.

Table continues on the next page...

Table continued from the previous page...

Field	Function
14 REQ46	Clock enable This bit provides the clock enable control for MU_2_MUA in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.
13 REQ45	Clock enable This bit provides the clock enable control for PIT_1 in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.
12 REQ44	Clock enable This bit provides the clock enable control for PIT_0 in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.
11 —	Reserved This field is reserved and read returns zeros.
10 REQ42	Clock enable This bit provides the clock enable control for ADC_2 in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.
9 REQ41	Clock enable This bit provides the clock enable control for ADC_1 in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.
8 REQ40	Clock enable This bit provides the clock enable control for ADC_0 in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.
7 REQ39	Clock enable This bit provides the clock enable control for LCU_1 in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.
6	Clock enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
REQ38	This bit provides the clock enable control for LCU_0 in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.
5 —	Reserved This field is reserved and read returns zeros.
4 REQ36	Clock enable This bit provides the clock enable control for eMIOS_2 in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.
3 REQ35	Clock enable This bit provides the clock enable control for eMIOS_1 in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.
2 REQ34	Clock enable This bit provides the clock enable control for eMIOS_0 in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.
1 REQ33	Clock enable This bit provides the clock enable control for BCTU in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.
0 REQ32	Clock enable This bit provides the clock enable control for TRGMUX in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.

46.7.15 Partition 0 Core 0 Process Configuration Register (PRTN0_CORE0_PCONF)

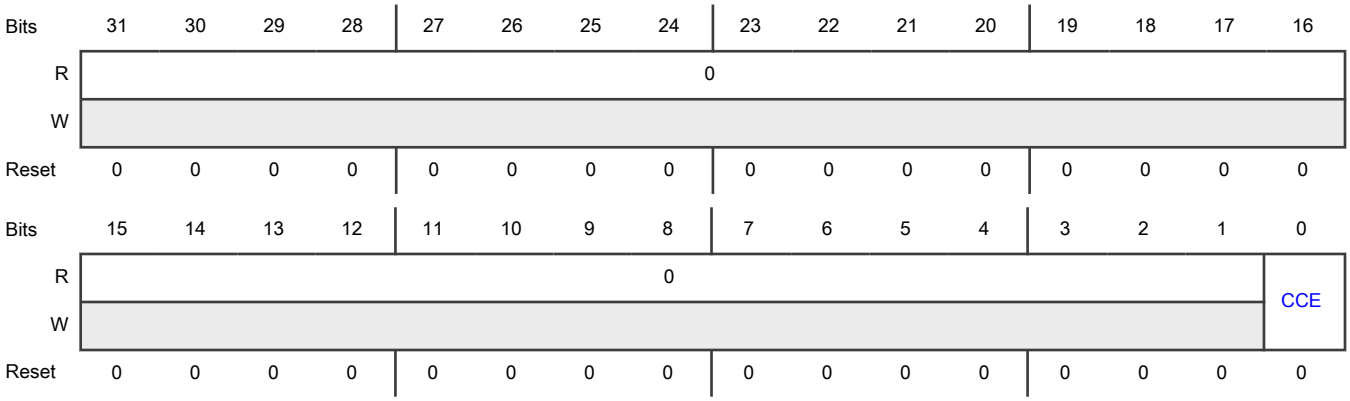
Offset

Register	Offset
PRTN0_CORE0_PCONF	140h

Function

This register provides configurations for the Core 0 hardware processes corresponding to partition 0. Each of the configuration bit corresponds to the 'nature' of the processes; for example, enabling/disabling and the trigger is controlled by the corresponding field in the PRTN0_CORE0_PUPD register. When valid KEY combinations are written onto the CTL_KEY register, the PRTN0_CORE0_PUPD and PRTN0_CORE0_PCONF registers are used to determine the hardware processes to be executed. These processes are triggered in parallel and are independent of each other. All dependent processes should be requested one after another from the software.

Diagram



Fields

Field	Function
31-1 —	Reserved This field is reserved and read returns zeros.
0 CCE	Core 0 clock enable This bit controls whether the clock to Core 0 in partition 0 should be enabled or disabled. 0b - Disable the core clock 1b - Enable the core clock

46.7.16 Partition 0 Core 0 Process Update Register (PRTN0_CORE0_PUPD)

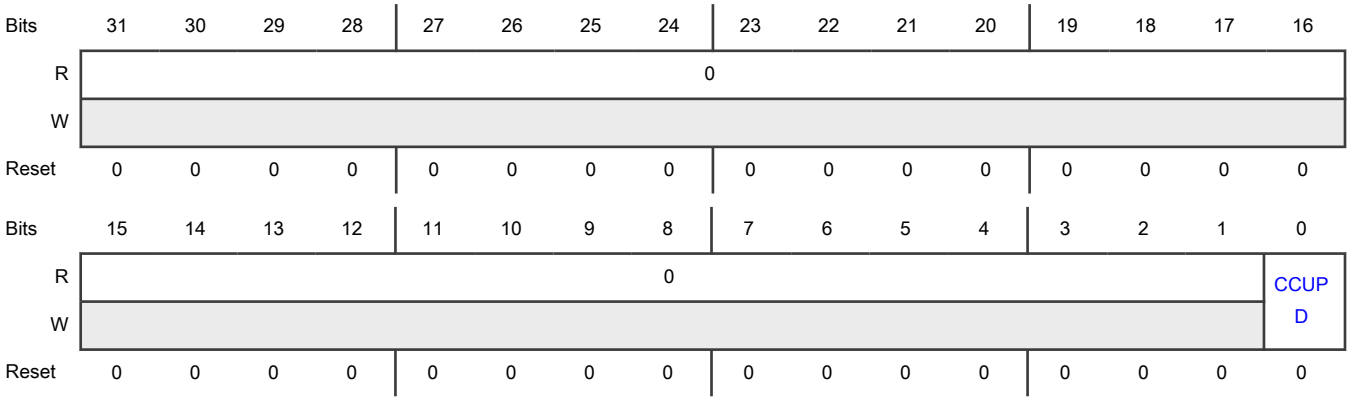
Offset

Register	Offset
PRTN0_CORE0_PUPD	144h

Function

This register provides trigger signaling for the core hardware processes corresponding to partition 0. Each of the control bit acts as a trigger for the corresponding hardware processes. When valid KEY combinations are written onto the CTL_KEY register, the hardware checks the bit fields that are programmed as logic-1 in this register, and then triggers the hardware process per the value in the corresponding bit field in the PRTN0_CORE0_PCONF register. When the hardware process is finished, the corresponding bit in this register is auto-cleared to logic-0.

Diagram



Fields

Field	Function
31-1 —	Reserved This field is reserved and read returns zeros.
0 CCUPD	Core 0 clock update This bit controls whether the hardware processes for enabling/disabling the clock to Core 0 in the partition 0 should be triggered or not. 0b - Do not trigger the hardware process 1b - Trigger the hardware process

46.7.17 Partition 0 Core 0 Status Register (PRTN0_CORE0_STAT)

Offset

Register	Offset
PRTN0_CORE0_STAT	148h

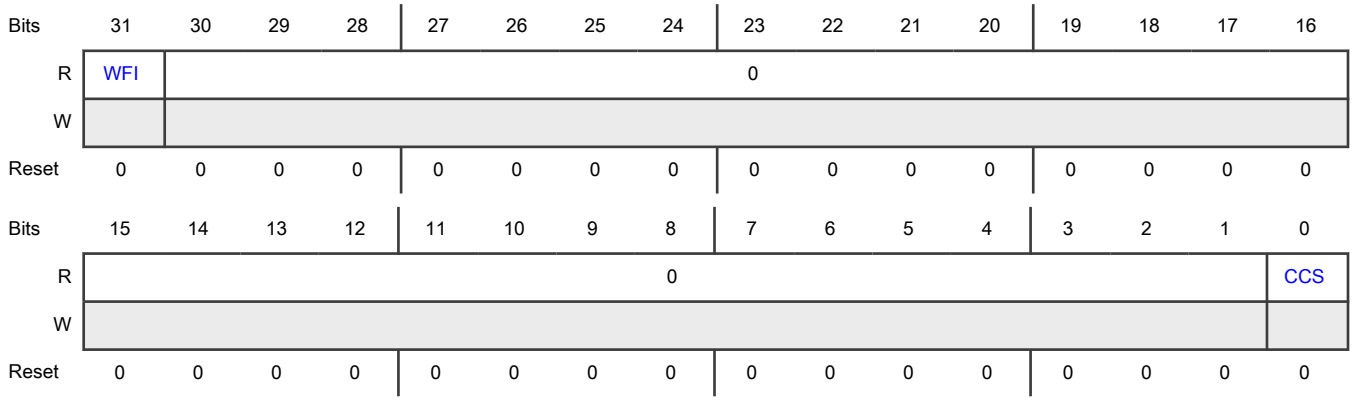
Function

This register provides the status corresponding to Core 0 in partition 0. The status signal corresponds to clock states and the WFI signal included from Core 0.

NOTE

The value held in WFI field of this STATUS register is "current" value of the WFISTANDBY signal from the core. Hence out-of-reset, the reset value of this field will depend on the status of the core (core is running or in low power mode). So, simple reset read sweep will always return current value (different than other register reads such as on control registers).

Diagram



Fields

Field	Function
31 WFI	Wait for interrupt status This bit provides the WFI status approaching from Core 0 in partition 0. 0b - No WFI executed 1b - WFI executed
30-1 —	Reserved This field is reserved and read returns zeros.
0 CCS	Core 0 clock process status This bit provides the status of the clock corresponding to core clock enablement/disablement. 0b - Clock is inactive. 1b - Clock is active.

46.7.18 Partition 0 Core 0 Address Register (PRTN0_CORE0_ADDR)

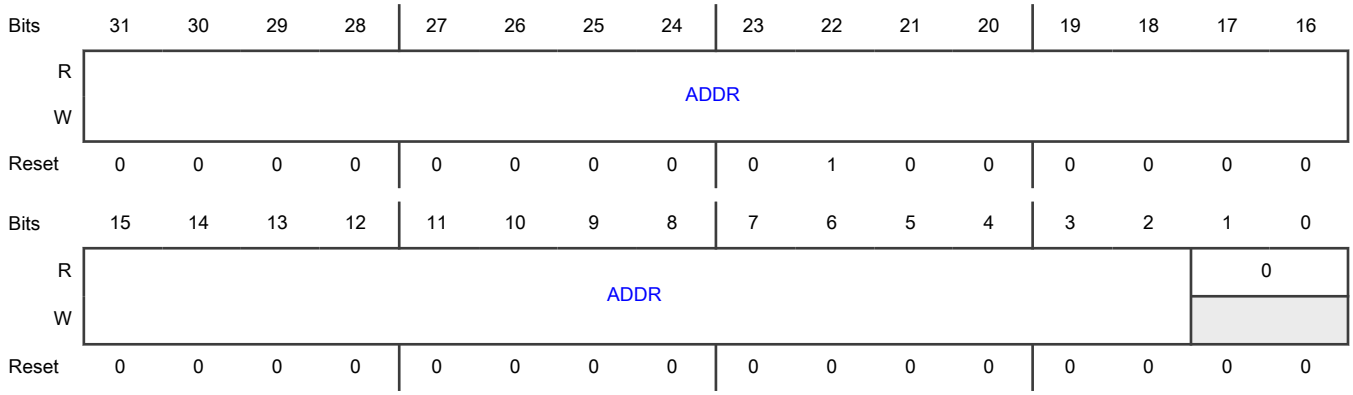
Offset

Register	Offset
PRTN0_CORE0_ADDR	14Ch

Function

This register contains the boot address for Core 0 in partition 0.

Diagram



Fields

Field	Function
31-2 ADDR	Address Core 0 boot address
1-0 —	Reserved This field is reserved and read returns zeros.

46.7.19 Partition 0 Core 1 Process Configuration Register (PRTN0_CORE1_PCONF)

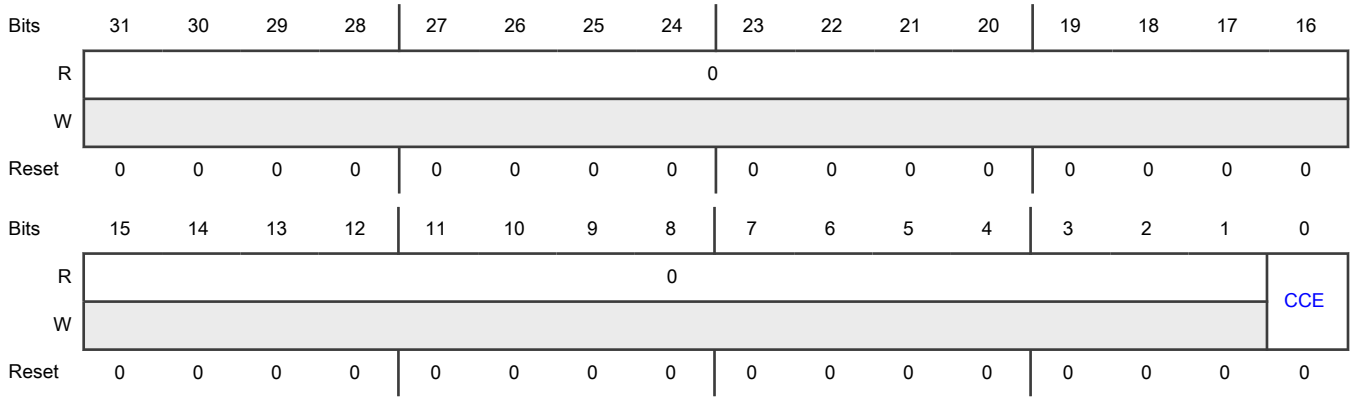
Offset

Register	Offset
PRTN0_CORE1_PCONF	160h

Function

This register provides configurations for the Core 1 hardware processes corresponding to partition 0. Each of the configuration bit corresponds to the 'nature' of the processes; for example, enabling/disabling and the trigger is controlled by the corresponding field in the PRTN0_CORE1_PUPD register. When valid KEY combinations are written onto the CTL_KEY register, the PRTN0_CORE1_PUPD and PRTN0_CORE1_PCONF registers are used to determine the hardware processes to be executed. These processes are triggered in parallel and are independent of each other. All dependent processes should be requested one after another from the software.

Diagram



Fields

Field	Function
31-1 —	Reserved This field is reserved and read returns zeros.
0 CCE	Core 1 clock enable This bit controls whether the clock to Core 1 in partition 0 should be enabled or disabled. 0b - Disable the core clock 1b - Enable the core clock

46.7.20 Partition 0 Core 1 Process Update Register (PRTN0_CORE1_PUPD)

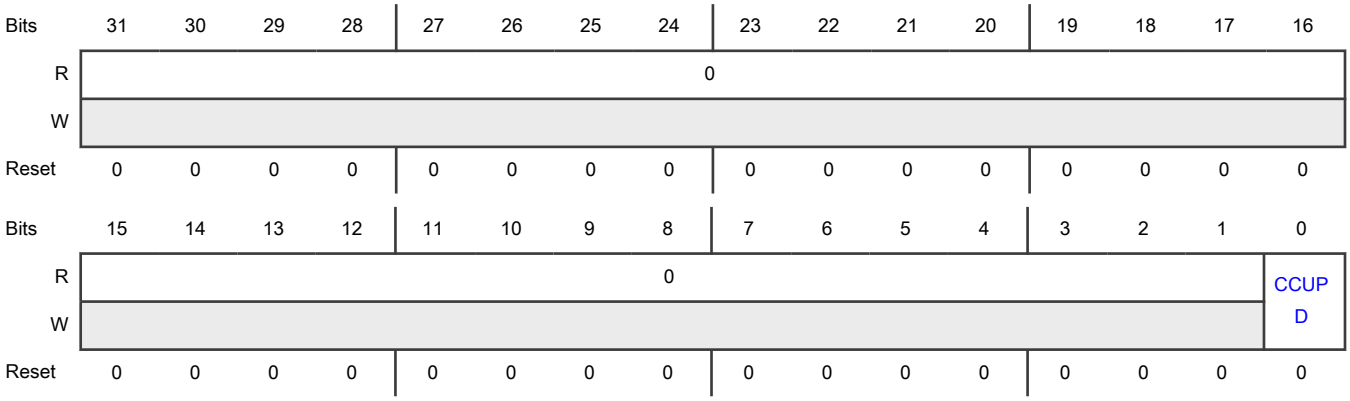
Offset

Register	Offset
PRTN0_CORE1_PUPD	164h

Function

This register provides trigger signaling for the core hardware processes corresponding to partition 0. Each of the control bit acts as a trigger for the corresponding hardware processes. When valid KEY combinations are written onto the CTL_KEY register, the hardware checks the bit fields that are programmed as logic-1 in this register, and then triggers the hardware process per the value in the corresponding bit field in the PRTN0_CORE1_PCONF register. When the hardware process is finished, the corresponding bit in this register is auto-cleared to logic-0.

Diagram



Fields

Field	Function
31-1 —	Reserved This field is reserved and read returns zeros.
0 CCUPD	Core 1 clock update This bit controls whether the hardware processes for enabling/disabling the clock to Core 1 in the partition 0 should be triggered or not. 0b - Do not trigger the hardware process 1b - Trigger the hardware process

46.7.21 Partition 0 Core 1 Status Register (PRTN0_CORE1_STAT)

Offset

Register	Offset
PRTN0_CORE1_STAT	168h

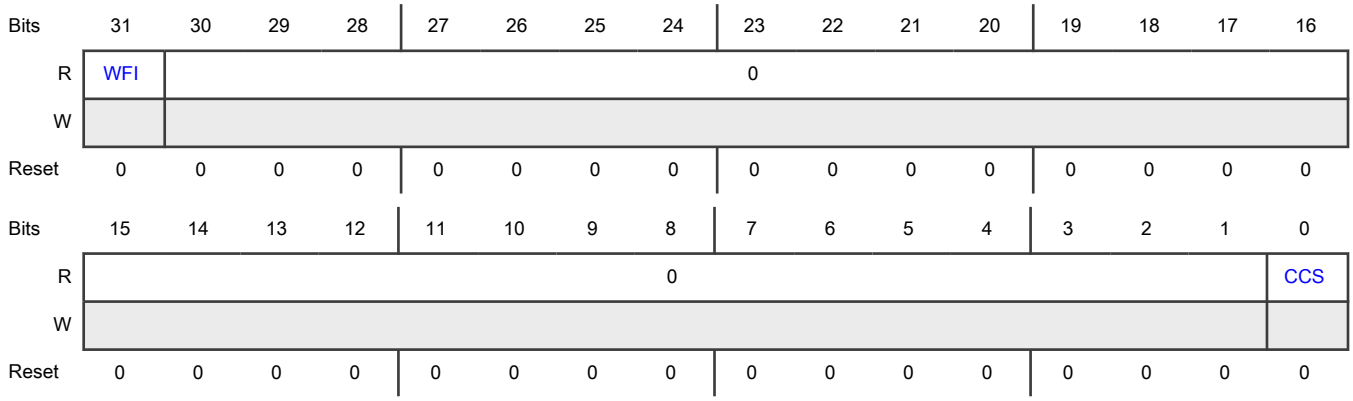
Function

This register provides the status corresponding to Core 1 in partition 0. The status signal corresponds to clock states and the WFI signal included from Core 1.

NOTE

The value held in WFI field of this STATUS register is "current" value of the WFISTANDBY signal from the core. Hence out-of-reset, the reset value of this field will depend on the status of the core (core is running or in low power mode). So, simple reset read sweep will always return current value (different than other register reads such as on control registers).

Diagram



Fields

Field	Function
31 WFI	Wait for interrupt status This bit provides the WFI status approaching from Core 1 in partition 0. 0b - No WFI executed 1b - WFI executed
30-1 —	Reserved This field is reserved and read returns zeros.
0 CCS	Core 1 clock process status This bit provides the status of the clock corresponding to core clock enablement/disablement. 0b - Clock is inactive. 1b - Clock is active.

46.7.22 Partition 0 Core 1 Address Register (PRTN0_CORE1_ADDR)

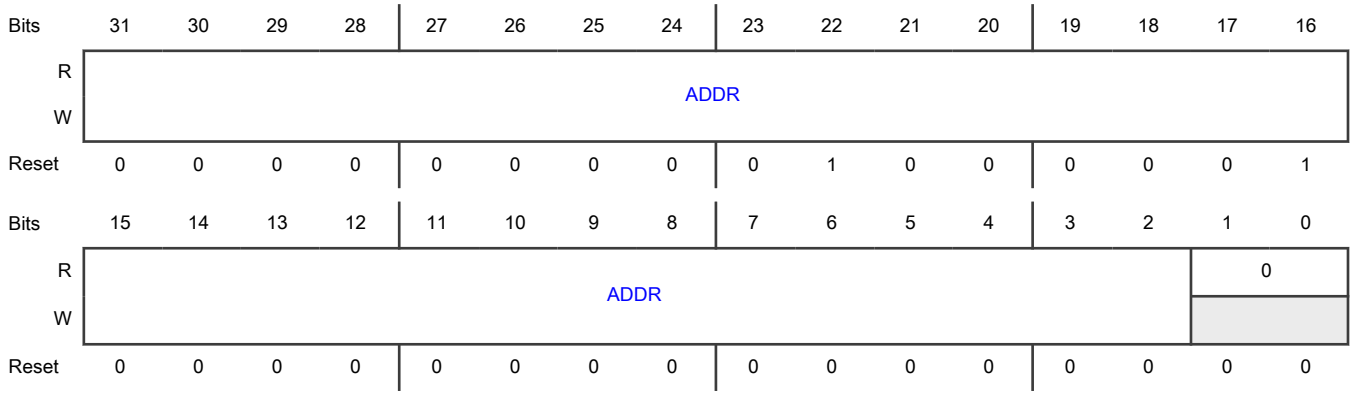
Offset

Register	Offset
PRTN0_CORE1_ADDR	16Ch

Function

This register contains the boot address for Core 1 in partition 0.

Diagram



Fields

Field	Function
31-2 ADDR	Address Core 1 boot address
1-0 —	Reserved This field is reserved and read returns zeros.

46.7.23 Partition 0 Core 2 Status Register (PRTN0_CORE2_STAT)

Offset

Register	Offset
PRTN0_CORE2_STAT	188h

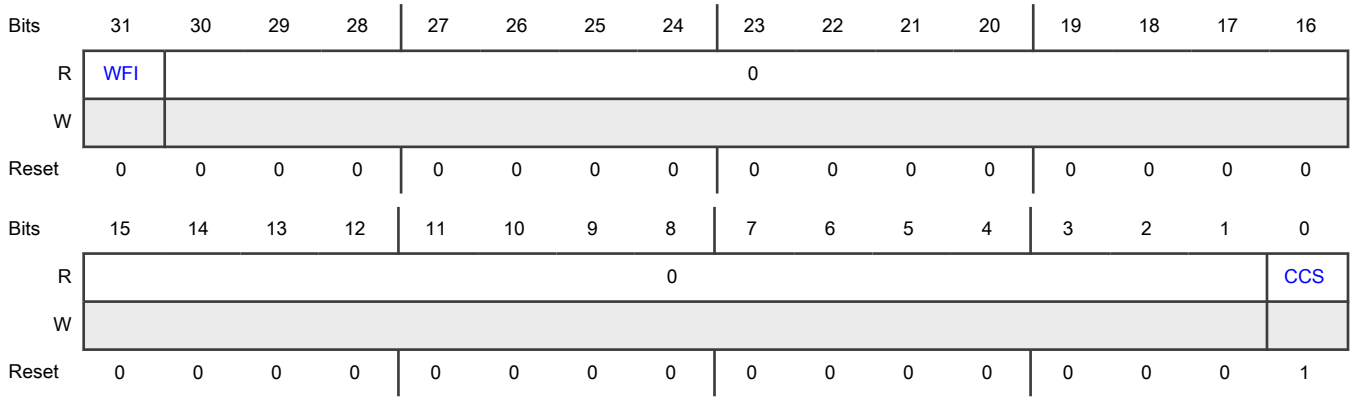
Function

This register provides the status corresponding to Core 2 in partition 0. The status signal corresponds to clock states and the WFI signal included from Core 2.

NOTE

The value held in WFI field of this STATUS register is "current" value of the WFISTANDBY signal from the core. Hence out-of-reset, the reset value of this field will depend on the status of the core (core is running or in low power mode). So, simple reset read sweep will always return current value (different than other register reads such as on control registers).

Diagram



Fields

Field	Function
31 WFI	Wait for interrupt status This bit provides the WFI status approaching from Core 2 in partition 0. 0b - No WFI executed 1b - WFI executed
30-1 —	Reserved This field is reserved and read returns zeros.
0 CCS	Core 2 clock process status This bit provides the status of the clock corresponding to core clock enablement/disablement. 1b - Clock is active.

46.7.24 Partition 0 Core 2 Address Register (PRTN0_CORE2_ADDR)

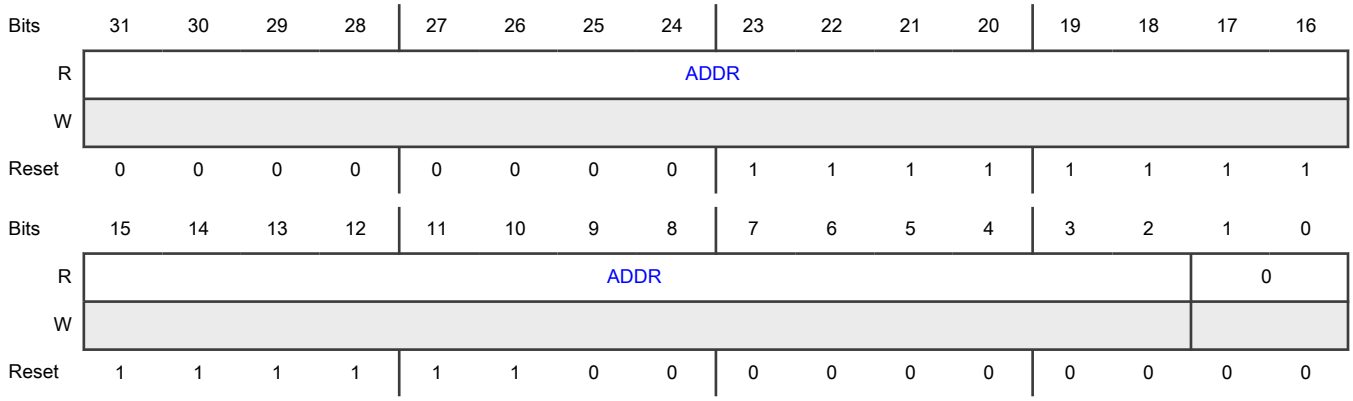
Offset

Register	Offset
PRTN0_CORE2_ADDR	18Ch

Function

This register contains the boot address for Core 2 in partition 0.

Diagram



Fields

Field	Function
31-2 ADDR	Address Core 2 boot address
1-0 —	Reserved This field is reserved and read returns zeros.

46.7.25 Partition 0 Core 3 Process Configuration Register (PRTN0_CORE3_PCONF)

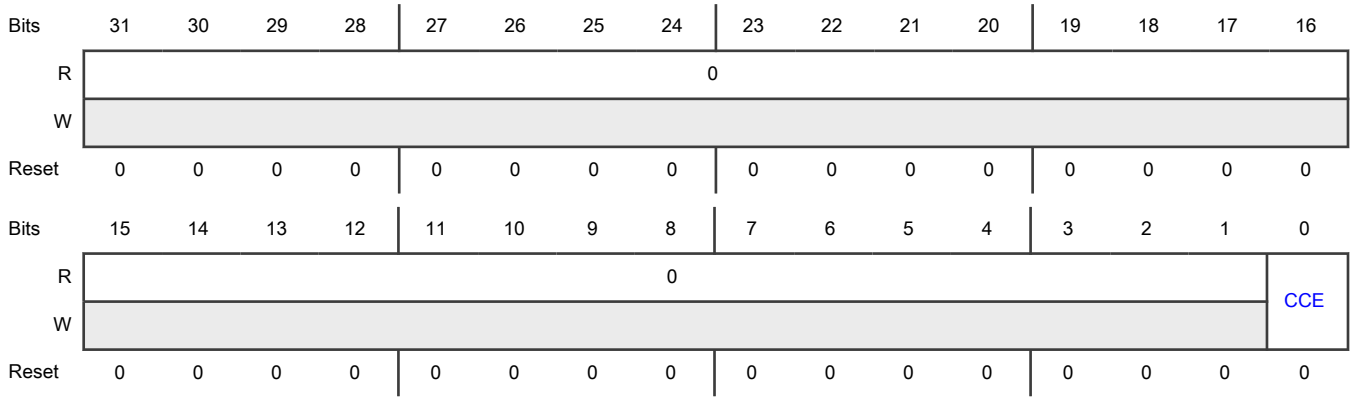
Offset

Register	Offset
PRTN0_CORE3_PCONF	1A0h

Function

This register provides configurations for the Core 3 hardware processes corresponding to partition 0. Each of the configuration bit corresponds to the 'nature' of the processes; for example, enabling/disabling and the trigger is controlled by the corresponding field in the PRTN0_CORE3_PUPD register. When valid KEY combinations are written onto the CTL_KEY register, the PRTN0_CORE3_PUPD and PRTN0_CORE3_PCONF registers are used to determine the hardware processes to be executed. These processes are triggered in parallel and are independent of each other. All dependent processes should be requested one after another from the software.

Diagram



Fields

Field	Function
31-1 —	Reserved This field is reserved and read returns zeros.
0 CCE	Core 3 clock enable This bit controls whether the clock to Core 3 in partition 0 should be enabled or disabled. 0b - Disable the core clock 1b - Enable the core clock

46.7.26 Partition 0 Core 3 Process Update Register (PRTN0_CORE3_PUPD)

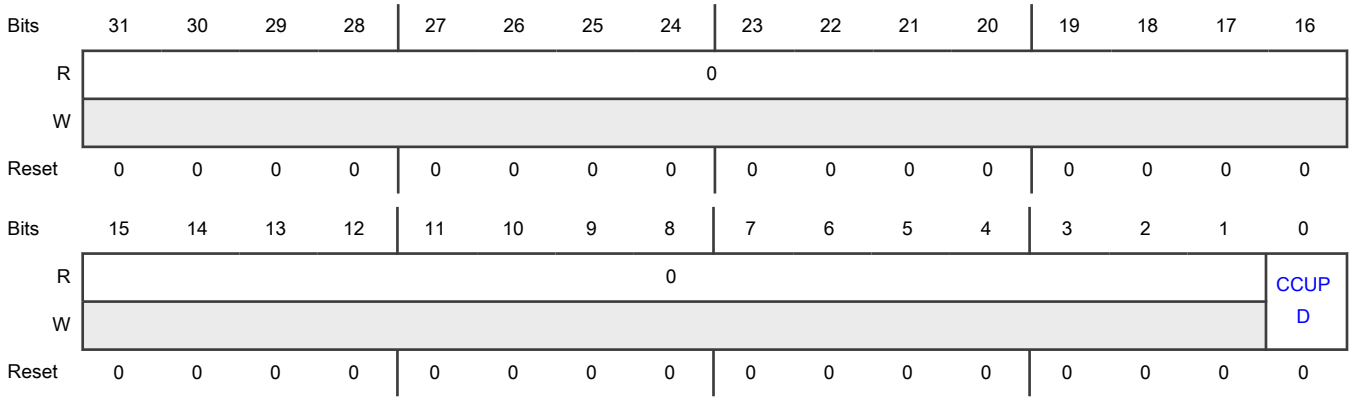
Offset

Register	Offset
PRTN0_CORE3_PUPD	1A4h

Function

This register provides trigger signaling for the core hardware processes corresponding to partition 0. Each of the control bit acts as a trigger for the corresponding hardware processes. When valid KEY combinations are written onto the CTL_KEY register, the hardware checks the bit fields that are programmed as logic-1 in this register, and then triggers the hardware process per the value in the corresponding bit field in the PRTN0_CORE3_PCONF register. When the hardware process is finished, the corresponding bit in this register is auto-cleared to logic-0.

Diagram



Fields

Field	Function
31-1 —	Reserved This field is reserved and read returns zeros.
0 CCUPD	Core 3 clock update This bit controls whether the hardware processes for enabling/disabling the clock to Core 3 in the partition 0 should be triggered or not. 0b - Do not trigger the hardware process 1b - Trigger the hardware process

46.7.27 Partition 0 Core 3 Status Register (PRTN0_CORE3_STAT)

Offset

Register	Offset
PRTN0_CORE3_STAT	1A8h

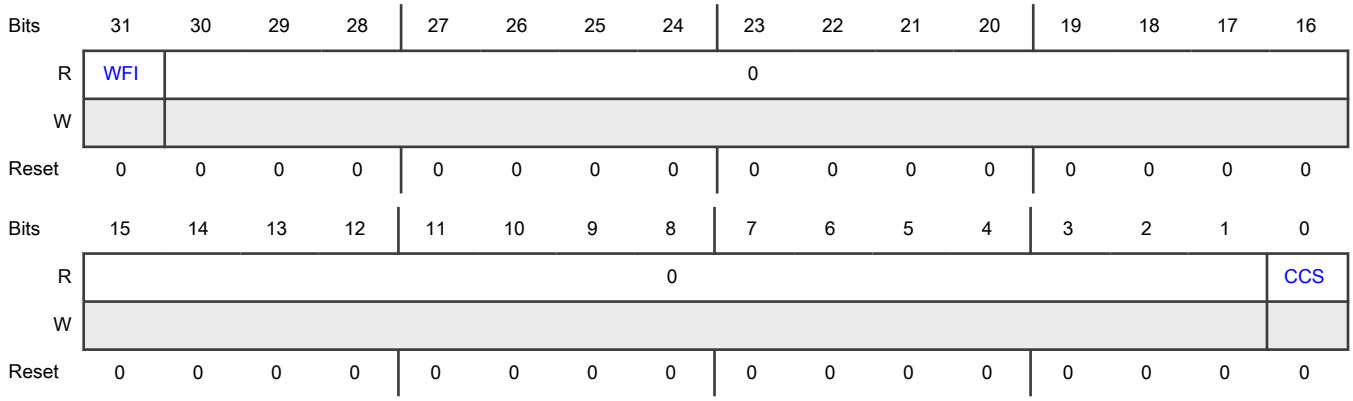
Function

This register provides the status corresponding to Core 3 in partition 0. The status signal corresponds to clock states and the WFI signal included from Core 3.

NOTE

The value held in WFI field of this STATUS register is "current" value of the WFISTANDBY signal from the core. Hence out-of-reset, the reset value of this field will depend on the status of the core (core is running or in low power mode). So, simple reset read sweep will always return current value (different than other register reads such as on control registers).

Diagram



Fields

Field	Function
31 WFI	Wait for interrupt status This bit provides the WFI status approaching from Core 3 in partition 0. 0b - No WFI executed 1b - WFI executed
30-1 —	Reserved This field is reserved and read returns zeros.
0 CCS	Core 3 clock process status This bit provides the status of the clock corresponding to core clock enablement/disablement. 0b - Clock is inactive. 1b - Clock is active.

46.7.28 Partition 0 Core 3 Address Register (PRTN0_CORE3_ADDR)

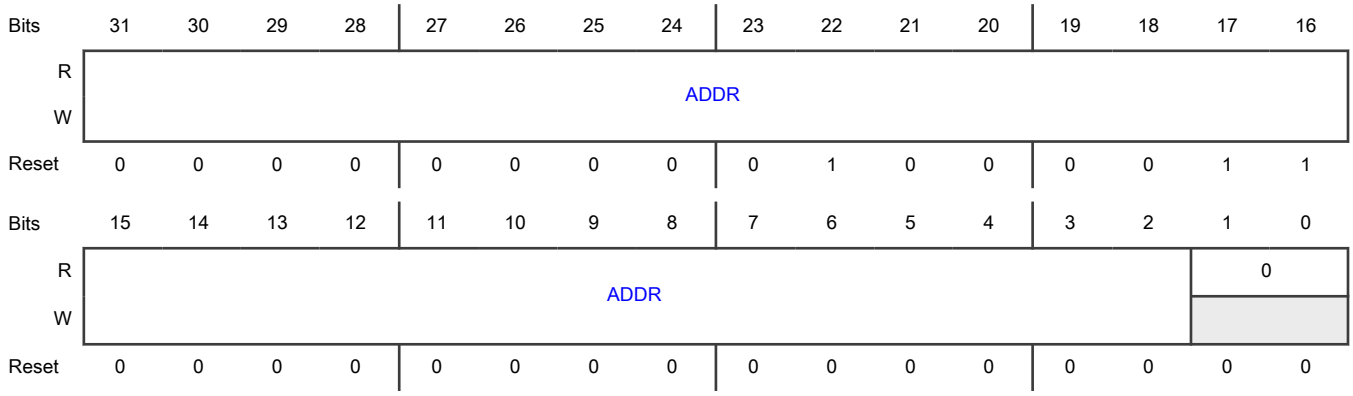
Offset

Register	Offset
PRTN0_CORE3_ADDR	1ACh

Function

This register contains the boot address for Core 3 in partition 0.

Diagram



Fields

Field	Function
31-2 ADDR	Address Core 3 boot address
1-0 —	Reserved This field is reserved and read returns zeros.

46.7.29 Partition 0 Core 4 Process Configuration Register (PRTN0_CORE4_PCONF)

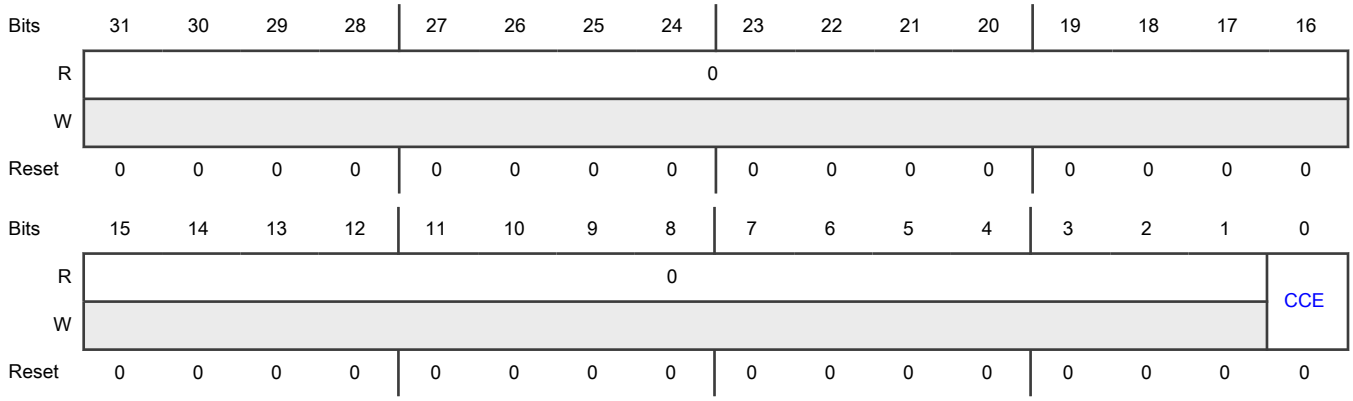
Offset

Register	Offset
PRTN0_CORE4_PCONF	1C0h

Function

This register provides configurations for the Core 4 hardware processes corresponding to partition 0. Each of the configuration bit corresponds to the 'nature' of the processes; for example, enabling/disabling and the trigger is controlled by the corresponding field in the PRTN0_CORE4_PUPD register. When valid KEY combinations are written onto the CTL_KEY register, the PRTN0_CORE4_PUPD and PRTN0_CORE4_PCONF registers are used to determine the hardware processes to be executed. These processes are triggered in parallel and are independent of each other. All dependent processes should be requested one after another from the software.

Diagram



Fields

Field	Function
31-1 —	Reserved This field is reserved and read returns zeros.
0 CCE	Core 4 clock enable This bit controls whether the clock to Core 4 in partition 0 should be enabled or disabled. 0b - Disable the core clock 1b - Enable the core clock

46.7.30 Partition 0 Core 4 Process Update Register (PRTN0_CORE4_PUPD)

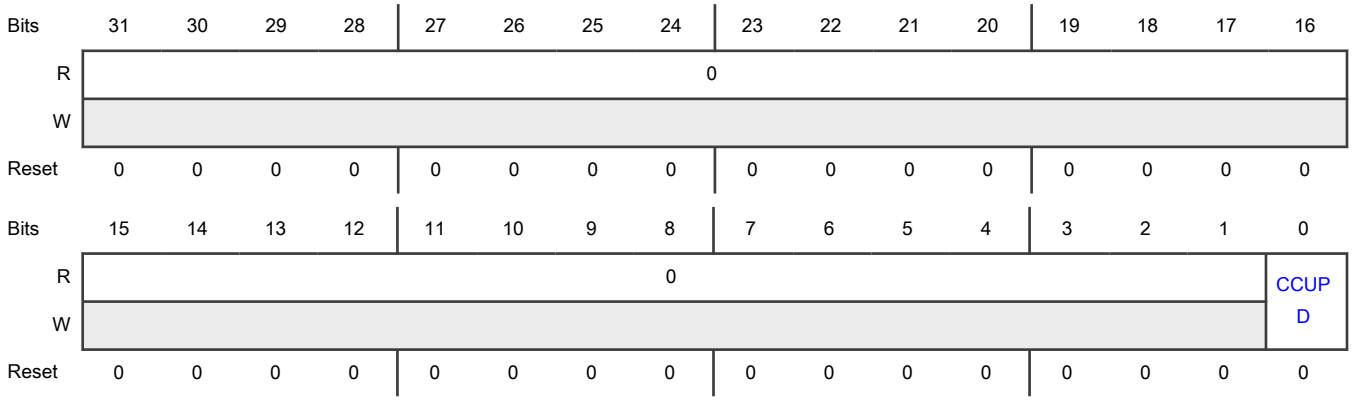
Offset

Register	Offset
PRTN0_CORE4_PUPD	1C4h

Function

This register provides trigger signaling for the core hardware processes corresponding to partition 0. Each of the control bit acts as a trigger for the corresponding hardware processes. When valid KEY combinations are written onto the CTL_KEY register, the hardware checks the bit fields that are programmed as logic-1 in this register, and then triggers the hardware process per the value in the corresponding bit field in the PRTN0_CORE4_PCONF register. When the hardware process is finished, the corresponding bit in this register is auto-cleared to logic-0.

Diagram



Fields

Field	Function
31-1 —	Reserved This field is reserved and read returns zeros.
0 CCUPD	Core 4 clock update This bit controls whether the hardware processes for enabling/disabling the clock to Core 4 in the partition 0 should be triggered or not. 0b - Do not trigger the hardware process 1b - Trigger the hardware process

46.7.31 Partition 0 Core 4 Status Register (PRTN0_CORE4_STAT)

Offset

Register	Offset
PRTN0_CORE4_STAT	1C8h

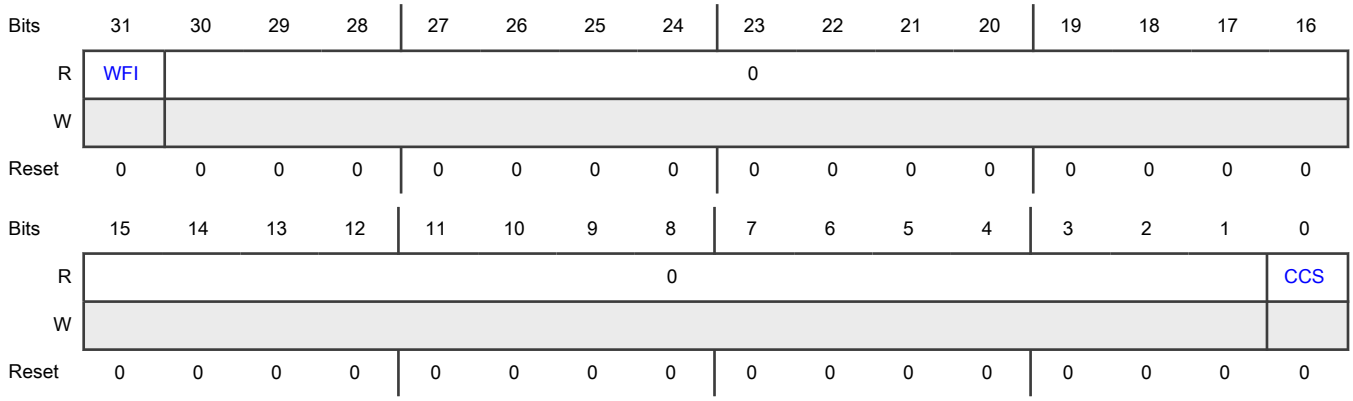
Function

This register provides the status corresponding to Core 4 in partition 0. The status signal corresponds to clock states and the WFI signal included from Core 4.

NOTE

The value held in WFI field of this STATUS register is "current" value of the WFISTANDBY signal from the core. Hence out-of-reset, the reset value of this field will depend on the status of the core (core is running or in low power mode). So, simple reset read sweep will always return current value (different than other register reads such as on control registers).

Diagram



Fields

Field	Function
31 WFI	Wait for interrupt status This bit provides the WFI status approaching from Core 4 in partition 0. 0b - No WFI executed 1b - WFI executed
30-1 —	Reserved This field is reserved and read returns zeros.
0 CCS	Core 4 clock process status This bit provides the status of the clock corresponding to core clock enablement/disablement. 0b - Clock is inactive. 1b - Clock is active.

46.7.32 Partition 0 Core 4 Address Register (PRTN0_CORE4_ADDR)

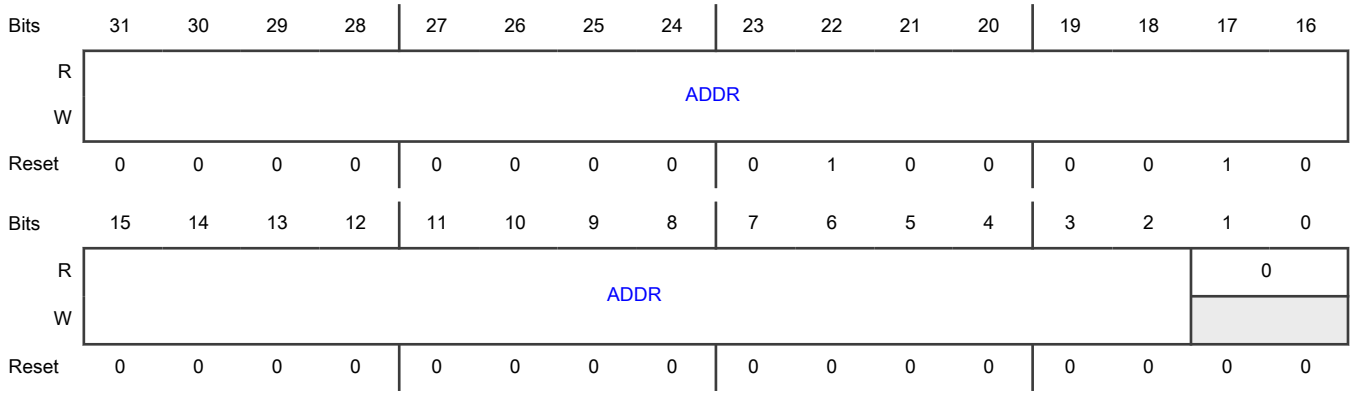
Offset

Register	Offset
PRTN0_CORE4_ADDR	1CCh

Function

This register contains the boot address for Core 4 in partition 0.

Diagram



Fields

Field	Function
31-2 ADDR	Address Core 4 boot address
1-0 —	Reserved This field is reserved and read returns zeros.

46.7.33 Partition 0 Core 5 Process Configuration Register (PRTN0_CORE5_PCONF)

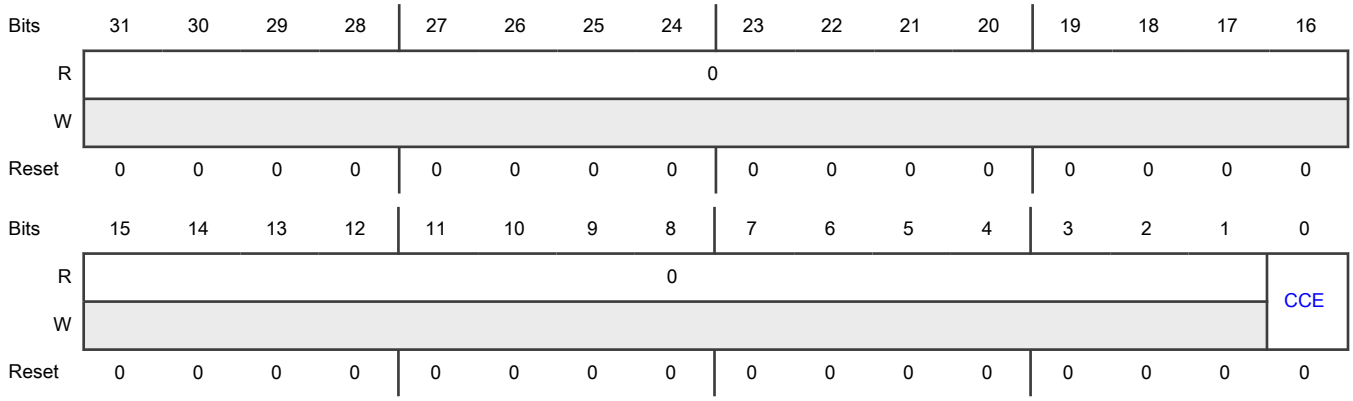
Offset

Register	Offset
PRTN0_CORE5_PCONF	1E0h

Function

This register provides configurations for the Core 5 hardware processes corresponding to partition 0. Each of the configuration bit corresponds to the 'nature' of the processes; for example, enabling/disabling and the trigger is controlled by the corresponding field in the PRTN0_CORE5_PUPD register. When valid KEY combinations are written onto the CTL_KEY register, the PRTN0_CORE5_PUPD and PRTN0_CORE5_PCONF registers are used to determine the hardware processes to be executed. These processes are triggered in parallel and are independent of each other. All dependent processes should be requested one after another from the software.

Diagram



Fields

Field	Function
31-1 —	Reserved This field is reserved and read returns zeros.
0 CCE	Core 5 clock enable This bit controls whether the clock to Core 5 in partition 0 should be enabled or disabled. 0b - Disable the core clock 1b - Enable the core clock

46.7.34 Partition 0 Core 5 Process Update Register (PRTN0_CORE5_PUPD)

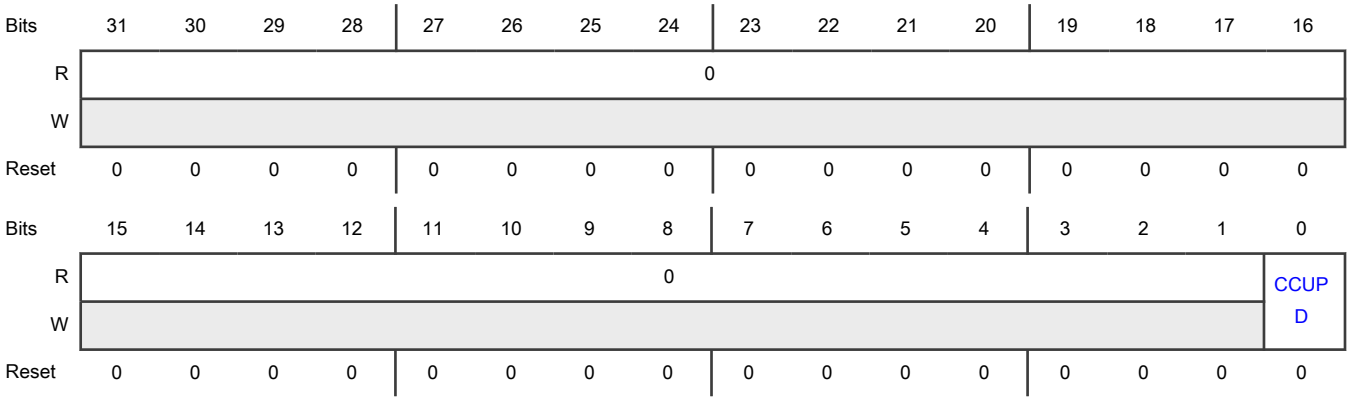
Offset

Register	Offset
PRTN0_CORE5_PUPD	1E4h

Function

This register provides trigger signaling for the core hardware processes corresponding to partition 0. Each of the control bit acts as a trigger for the corresponding hardware processes. When valid KEY combinations are written onto the CTL_KEY register, the hardware checks the bit fields that are programmed as logic-1 in this register, and then triggers the hardware process per the value in the corresponding bit field in the PRTN0_CORE5_PCONF register. When the hardware process is finished, the corresponding bit in this register is auto-cleared to logic-0.

Diagram



Fields

Field	Function
31-1 —	Reserved This field is reserved and read returns zeros.
0 CCUPD	Core 5 clock update This bit controls whether the hardware processes for enabling/disabling the clock to Core 5 in the partition 0 should be triggered or not. 0b - Do not trigger the hardware process 1b - Trigger the hardware process

46.7.35 Partition 0 Core 5 Status Register (PRTN0_CORE5_STAT)

Offset

Register	Offset
PRTN0_CORE5_STAT	1E8h

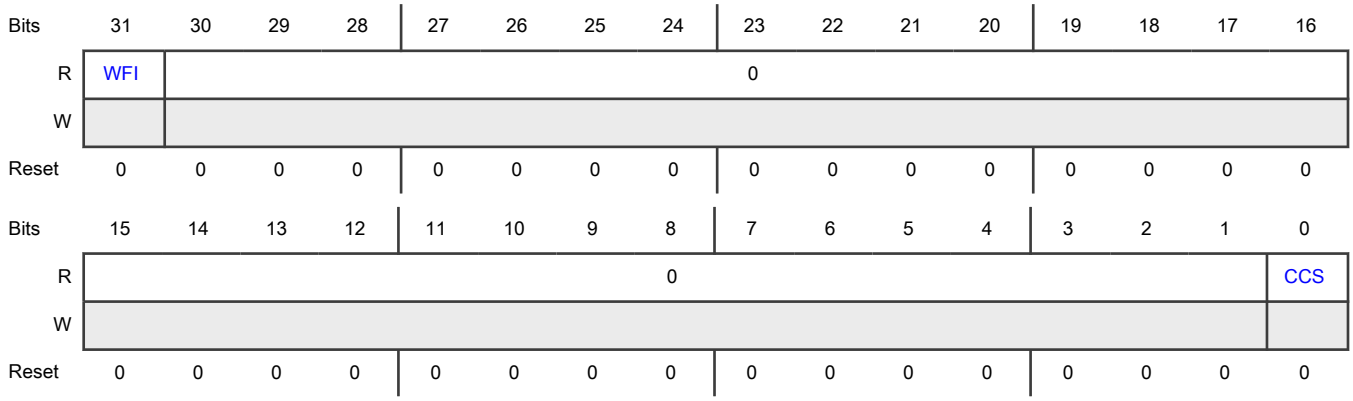
Function

This register provides the status corresponding to Core 5 in partition 0. The status signal corresponds to clock states and the WFI signal included from Core 5.

NOTE

The value held in WFI field of this STATUS register is "current" value of the WFISTANDBY signal from the core. Hence out-of-reset, the reset value of this field will depend on the status of the core (core is running or in low power mode). So, simple reset read sweep will always return current value (different than other register reads such as on control registers).

Diagram



Fields

Field	Function
31 WFI	Wait for interrupt status This bit provides the WFI status approaching from Core 5 in partition 0. 0b - No WFI executed 1b - WFI executed
30-1 —	Reserved This field is reserved and read returns zeros.
0 CCS	Core 5 clock process status This bit provides the status of the clock corresponding to core clock enablement/disablement. 0b - Clock is inactive. 1b - Clock is active.

46.7.36 Partition 0 Core 5 Address Register (PRTN0_CORE5_ADDR)

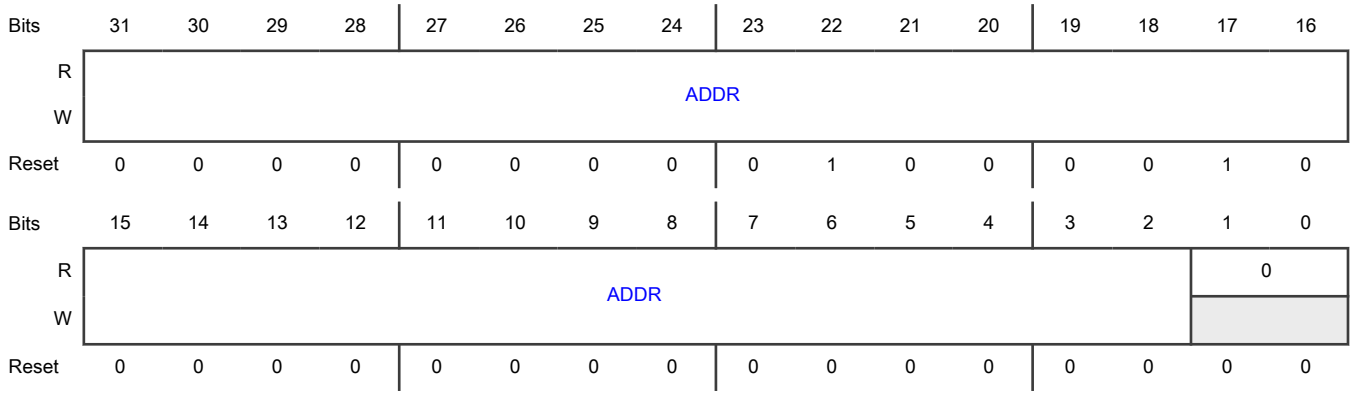
Offset

Register	Offset
PRTN0_CORE5_ADDR	1ECh

Function

This register contains the boot address for Core 5 in partition 0.

Diagram



Fields

Field	Function
31-2 ADDR	Address Core 5 boot address
1-0 —	Reserved This field is reserved and read returns zeros.

46.7.37 Partition 1 Process Configuration Register (PRTN1_PCONF)

Offset

Register	Offset
PRTN1_PCONF	300h

Function

This register provides a configuration for the hardware processes corresponding to partition 1. Each of the configuration bit corresponds to the 'nature' of the processes; for example, enabling/disabling and the trigger is controlled by the corresponding field in the PRTN1_PUPD register. When valid KEY combinations are written onto the CTL_KEY register, the PRTN1_PCONF and PRTN1_PUPD registers are used to determine the hardware processes to be executed. These are triggered in parallel and independent of each other. All dependent processes should be requested one after another from the software.

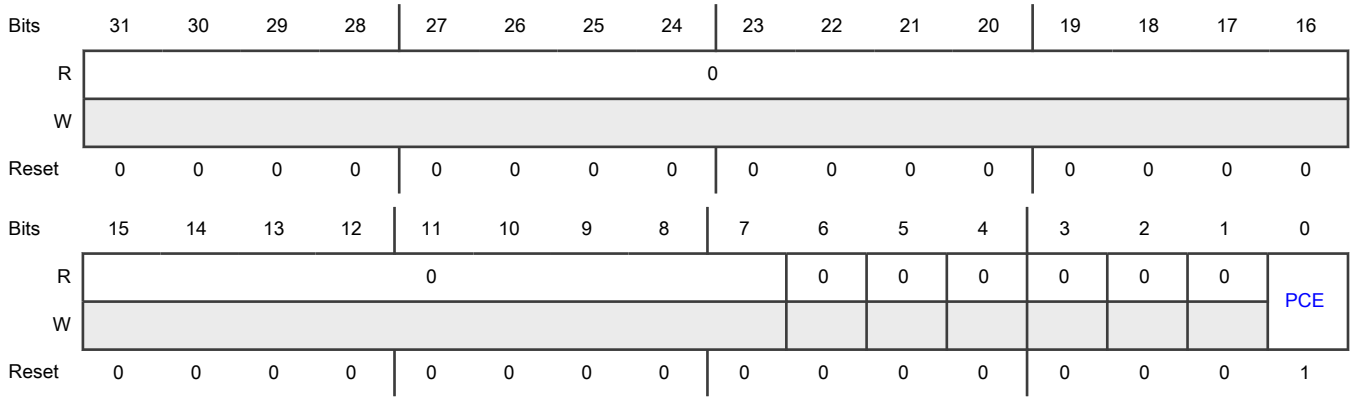
NOTE

The partition clock enable/disable are not standalone and must be done coherently in a fixed sequence. For details, see Software Reset Partition Turn-On Flow Chart and Software reset partition turn-off flowchart in Reset chapter.

NOTE

See chip-specific MC_ME information to check if this register is implemented on chip.

Diagram



Fields

Field	Function
31-7 —	Reserved This field is reserved and read returns zeros.
6 —	Reserved This field is reserved and read returns zeros.
5 —	Reserved This field is reserved and read returns zeros.
4 —	Reserved This field is reserved and read returns zeros.
3 —	Reserved This field is reserved and read returns zeros.
2 —	Reserved This field is reserved and read returns zeros.
1 —	Reserved This field is reserved and read returns zeros.
0 PCE	Partition clock enable This bit controls whether the clock to IPs (other than core(s)) in the partition should be enabled or disabled. 0b - Disable the clock to IPs 1b - Enable the clock to IPs

46.7.38 Partition 1 Process Update Register (PRTN1_PUPD)

Offset

Register	Offset
PRTN1_PUPD	304h

Function

This register provides trigger signaling for the hardware processes corresponding to partition 1. Each of the control bit acts as a trigger for the corresponding hardware processes. When valid KEY combinations are written onto the CTL_KEY register, the hardware checks the bit fields that are programmed as logic-1 in this register, and then triggers the hardware process per the value in the corresponding bit field in the PRTN1_PCONF register. When the hardware process is finished the corresponding bit in this register is auto-cleared to logic-0.

NOTE

See chip-specific MC_ME information to check if this register is implemented on chip.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0									0	0	0	0	0	0	PCUD
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-7 —	Reserved This field is reserved and read returns zeros.
6 —	Reserved This field is reserved and read returns zeros.
5 —	Reserved This field is reserved and read returns zeros.
4 —	Reserved This field is reserved and read returns zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
3 —	Reserved This field is reserved and read returns zeros.
2 —	Reserved This field is reserved and read returns zeros.
1 —	Reserved This field is reserved and read returns zeros.
0 PCUD	Partition clock update This bit controls whether the hardware processes for enabling/disabling the clock to IPs (other than core(s)) in the partition should be triggered or not. 0b - Do not trigger the hardware process 1b - Trigger the hardware process

46.7.39 Partition 1 Status Register (PRTN1_STAT)

Offset

Register	Offset
PRTN1_STAT	308h

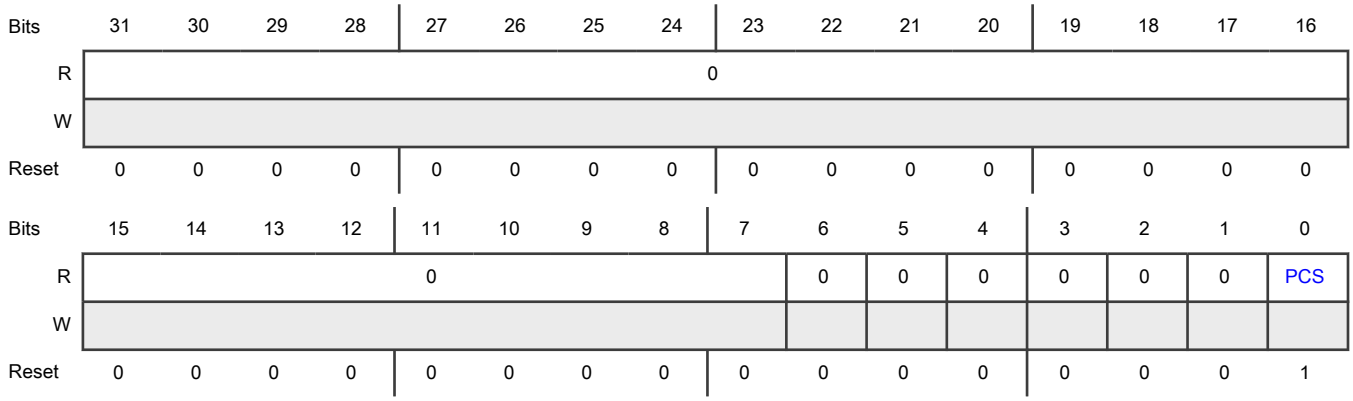
Function

This register provides the current status of the control signals from the partition 1.

NOTE

See chip-specific MC_ME information to check if this register is implemented on chip.

Diagram



Fields

Field	Function
31-7 —	Reserved This field is reserved and read returns zeros.
6 —	Reserved This field is reserved and read returns zeros.
5 —	Reserved This field is reserved and read returns zeros.
4 —	Reserved This field is reserved and read returns zeros.
3 —	Reserved This field is reserved and read returns zeros.
2 —	Reserved This field is reserved and read returns zeros.
1 —	Reserved This field is reserved and read returns zeros.
0 PCS	Partition clock status This bit provides the status of the clock to partition. 0b - Clock is inactive 1b - Clock is active

46.7.40 Partition 1 COFB Set 0 Clock Status Register (PRTN1_COFB0_STAT)

Offset

Register	Offset
PRTN1_COFB0_STAT	310h

Function

This register provides the status of set 0 of COFBs inside partition 1.

NOTE

The reset value of this register can vary depending on the availability of active clock pulses inside partition 1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BLOCK31	BLOCK30	BLOCK29	BLOCK28	BLOCK27	BLOCK26	BLOCK25	BLOCK24	BLOCK23	0	BLOCK21	BLOCK20	BLOCK19	BLOCK18	BLOCK17	BLOCK16
W																
Reset	0	1	0	1	1	1	1	0	0	0	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BLOCK15	BLOCK14	BLOCK13	BLOCK12	BLOCK11	BLOCK10	BLOCK9	BLOCK8	BLOCK7	BLOCK6	BLOCK5	BLOCK4	BLOCK3	BLOCK2	BLOCK1	BLOCK0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1

Fields

Field	Function
31 BLOCK31	IP block status This bit provides the clock status of INTM in partition 1. 0b - Clock is not running. 1b - Clock is running.
30 BLOCK30	IP block status This bit provides the clock status of block 30 in partition 1. 0b - Clock is not running. 1b - Clock is running.
29 BLOCK29	IP block status This bit provides the clock status of STM 0 in partition 1. 0b - Clock is not running. 1b - Clock is running.
28 BLOCK28	IP block status This bit provides the clock status of SWT 0 in partition 1. 0b - Clock is not running. 1b - Clock is running.
27 BLOCK27	IP block status This bit provides the clock status of block 27 in partition 1. 0b - Clock is not running. 1b - Clock is running.
26	IP block status

Table continues on the next page...

Table continued from the previous page...

Field	Function
BLOCK26	This bit provides the clock status of block 26 in partition 1. 0b - Clock is not running. 1b - Clock is running.
25 BLOCK25	IP block status This bit provides the clock status of block 25 in partition 1. 0b - Clock is not running. 1b - Clock is running.
24 BLOCK24	IP block status This bit provides the clock status of MSCM in partition 1. 0b - Clock is not running. 1b - Clock is running.
23 BLOCK23	IP block status This bit provides the clock status of ERM_0 in partition 1. 0b - Clock is not running. 1b - Clock is running.
22 —	Reserved This field is reserved and read returns zeros.
21 BLOCK21	IP block status This bit provides the clock status of SDA-AP and Debug APB Paged Area in partition 1. 0b - Clock is not running. 1b - Clock is running.
20 BLOCK20	IP block status This bit provides the clock status of block 20 in partition 1. 0b - Clock is not running. 1b - Clock is running.
19 BLOCK19	IP block status This bit provides the clock status of block 19 in partition 1. 0b - Clock is not running. 1b - Clock is running.
18 BLOCK18	IP block status This bit provides the clock status of block 18 in partition 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Clock is not running. 1b - Clock is running.
17 BLOCK17	IP block status This bit provides the clock status of block 17 in partition 1. 0b - Clock is not running. 1b - Clock is running.
16 BLOCK16	IP block status This bit provides the clock status of block 16 in partition 1. 0b - Clock is not running. 1b - Clock is running.
15 BLOCK15	IP block status This bit provides the clock status of eDMA TCD 11 in partition 1. 0b - Clock is not running. 1b - Clock is running.
14 BLOCK14	IP block status This bit provides the clock status of eDMA TCD 10 in partition 1. 0b - Clock is not running. 1b - Clock is running.
13 BLOCK13	IP block status This bit provides the clock status of eDMA TCD 9 in partition 1. 0b - Clock is not running. 1b - Clock is running.
12 BLOCK12	IP block status This bit provides the clock status of eDMA TCD 8 in partition 1. 0b - Clock is not running. 1b - Clock is running.
11 BLOCK11	IP block status This bit provides the clock status of eDMA TCD 7 in partition 1. 0b - Clock is not running. 1b - Clock is running.
10	IP block status

Table continues on the next page...

Table continued from the previous page...

Field	Function
BLOCK10	This bit provides the clock status of eDMA TCD 6 in partition 1. 0b - Clock is not running. 1b - Clock is running.
9 BLOCK9	IP block status This bit provides the clock status of eDMA TCD 5 in partition 1. 0b - Clock is not running. 1b - Clock is running.
8 BLOCK8	IP block status This bit provides the clock status of eDMA TCD 4 in partition 1. 0b - Clock is not running. 1b - Clock is running.
7 BLOCK7	IP block status This bit provides the clock status of eDMA TCD 3 in partition 1. 0b - Clock is not running. 1b - Clock is running.
6 BLOCK6	IP block status This bit provides the clock status of eDMA TCD 2 in partition 1. 0b - Clock is not running. 1b - Clock is running.
5 BLOCK5	IP block status This bit provides the clock status of eDMA TCD 1 in partition 1. 0b - Clock is not running. 1b - Clock is running.
4 BLOCK4	IP block status This bit provides the clock status of eDMA TCD 0 in partition 1. 0b - Clock is not running. 1b - Clock is running.
3 BLOCK3	IP block status This bit provides the clock status of eDMA_Control_and_Status in partition 1. 0b - Clock is not running. 1b - Clock is running.

Table continues on the next page...

Table continued from the previous page...

Field	Function
2 BLOCK2	IP block status This bit provides the clock status of block 2 in partition 1. 0b - Clock is not running. 1b - Clock is running.
1 BLOCK1	IP block status This bit provides the clock status of block 1 in partition 1. 0b - Clock is not running. 1b - Clock is running.
0 BLOCK0	IP block status This bit provides the clock status of block 0 in partition 1. 0b - Clock is not running. 1b - Clock is running.

46.7.41 Partition 1 COFB Set 1 Clock Status Register (PRTN1_COFB1_STAT)

Offset

Register	Offset
PRTN1_COFB1_STAT	314h

Function

This register provides the status of set 1 of COFBs inside partition 1.

NOTE

The reset value of this register can vary depending on the availability of active clock pulses inside partition 1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BLOCK63	BLOCK62	BLOCK61	BLOCK60	BLOCK59	BLOCK58	BLOCK57	BLOCK56	BLOCK55	BLOCK54	BLOCK53	BLOCK52	BLOCK51	BLOCK50	BLOCK49	0
W																
Reset	0	1	1	1	1	1	0	0	1	1	1	1	1	1	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BLOCK47	0	BLOCK45	0	BLOCK43	BLOCK42	BLOCK41	BLOCK40	BLOCK39	BLOCK38	BLOCK37	BLOCK36	BLOCK35	BLOCK34	BLOCK33	BLOCK32
W																
Reset	0	0	1	0	1	1	1	1	1	1	1	1	1	1	0	0

Fields

Field	Function
31 BLOCK63	IP block status This bit provides the clock status of PIT 2 in partition 1. 0b - Clock is not running. 1b - Clock is running.
30 BLOCK62	IP block status This bit provides the clock status of block 62 in partition 1. 0b - Clock is not running. 1b - Clock is running.
29 BLOCK61	IP block status This bit provides the clock status of block 61 in partition 1. 0b - Clock is not running. 1b - Clock is running.
28 BLOCK60	IP block status This bit provides the clock status of block 60 in partition 1. 0b - Clock is not running. 1b - Clock is running.
27 BLOCK59	IP block status This bit provides the clock status of block 59 in partition 1. 0b - Clock is not running. 1b - Clock is running.
26 BLOCK58	IP block status This bit provides the clock status of block 58 in partition 1. 0b - Clock is not running. 1b - Clock is running.
25 BLOCK57	IP block status This bit provides the clock status of PLL 2 in partition 1. 0b - Clock is not running. 1b - Clock is running.
24 BLOCK56	IP block status This bit provides the clock status of PLL in partition 1. 0b - Clock is not running.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Clock is running.
23 BLOCK55	IP block status This bit provides the clock status of block 55 in partition 1. 0b - Clock is not running. 1b - Clock is running.
22 BLOCK54	IP block status This bit provides the clock status of block 54 in partition 1. 0b - Clock is not running. 1b - Clock is running.
21 BLOCK53	IP block status This bit provides the clock status of FXOSC in partition 1. 0b - Clock is not running. 1b - Clock is running.
20 BLOCK52	IP block status This bit provides the clock status of block 52 in partition 1. 0b - Clock is not running. 1b - Clock is running.
19 BLOCK51	IP block status This bit provides the clock status of SXOSC in partition 1. 0b - Clock is not running. 1b - Clock is running.
18 BLOCK50	IP block status This bit provides the clock status of block 50 in partition 1. 0b - Clock is not running. 1b - Clock is running.
17 BLOCK49	IP block status This bit provides the clock status of TSPC in partition 1. 0b - Clock is not running. 1b - Clock is running.
16 —	Reserved This field is reserved and read returns zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
15 BLOCK47	IP block status This bit provides the clock status of CMU_0-5 in partition 1. 0b - Clock is not running. 1b - Clock is running.
14 —	Reserved This field is reserved and read returns zeros.
13 BLOCK45	IP block status This bit provides the clock status of WKPU in partition 1. 0b - Clock is not running. 1b - Clock is running.
12 —	Reserved This field is reserved and read returns zeros.
11 BLOCK43	IP block status This bit provides the clock status of block 43 in partition 1. 0b - Clock is not running. 1b - Clock is running.
10 BLOCK42	IP block status This bit provides the clock status of SIUL_VIRTWRAPPER_PDAC3 in partition 1. 0b - Clock is not running. 1b - Clock is running.
9 BLOCK41	IP block status This bit provides the clock status of block 41 in partition 1. 0b - Clock is not running. 1b - Clock is running.
8 BLOCK40	IP block status This bit provides the clock status of block 40 in partition 1. 0b - Clock is not running. 1b - Clock is running.
7 BLOCK39	IP block status This bit provides the clock status of block 39 in partition 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Clock is not running. 1b - Clock is running.
6 BLOCK38	IP block status This bit provides the clock status of block 38 in partition 1. 0b - Clock is not running. 1b - Clock is running.
5 BLOCK37	IP block status This bit provides the clock status of block 37 in partition 1. 0b - Clock is not running. 1b - Clock is running.
4 BLOCK36	IP block status This bit provides the clock status of block 36 in partition 1. 0b - Clock is not running. 1b - Clock is running.
3 BLOCK35	IP block status This bit provides the clock status of block 35 in partition 1. 0b - Clock is not running. 1b - Clock is running.
2 BLOCK34	IP block status This bit provides the clock status of RTC in partition 1. 0b - Clock is not running. 1b - Clock is running.
1 BLOCK33	IP block status This bit provides the clock status of DMAMUX 1 in partition 1. 0b - Clock is not running. 1b - Clock is running.
0 BLOCK32	IP block status This bit provides the clock status of DMAMUX 0 in partition 1. 0b - Clock is not running. 1b - Clock is running.

46.7.42 Partition 1 COFB Set 2 Clock Status Register (PRTN1_COFB2_STAT)

Offset

Register	Offset
PRTN1_COFB2_STAT	318h

Function

This register provides the status of set 2 of COFBs inside partition 1.

NOTE

The reset value of this register can vary depending on the availability of active clock pulses inside partition 1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BLOCK95	0	BLOCK93	BLOCK92	BLOCK91	0	BLOCK89	BLOCK88	BLOCK87	BLOCK86	BLOCK85	BLOCK84	BLOCK83	BLOCK82	BLOCK81	BLOCK80
W																
Reset	0	0	1	1	0	0	0	0	0	0	0	0	1	1	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BLOCK79	BLOCK78	BLOCK77	BLOCK76	BLOCK75	BLOCK74	BLOCK73	BLOCK72	BLOCK71	BLOCK70	BLOCK69	BLOCK68	BLOCK67	BLOCK66	BLOCK65	BLOCK64
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 BLOCK95	IP block status This bit provides the clock status of TMU in partition 1. 0b - Clock is not running. 1b - Clock is running.
30 —	Reserved This field is reserved and read returns zeros.
29 BLOCK93	IP block status This bit provides the clock status of LPCMP 1 in partition 1. 0b - Clock is not running. 1b - Clock is running.

Table continues on the next page...

Table continued from the previous page...

Field	Function
28 BLOCK92	IP block status This bit provides the clock status of LPCMP 0 in partition 1. 0b - Clock is not running. 1b - Clock is running.
27 BLOCK91	IP block status This bit provides the clock status of SAI_0 in partition 1. 0b - Clock is not running. 1b - Clock is running.
26 —	Reserved This field is reserved and read returns zeros.
25 BLOCK89	IP block status This bit provides the clock status of LPSPI 3 in partition 1. 0b - Clock is not running. 1b - Clock is running.
24 BLOCK88	IP block status This bit provides the clock status of LPSPI 2 in partition 1. 0b - Clock is not running. 1b - Clock is running.
23 BLOCK87	IP block status This bit provides the clock status of LPSPI 1 in partition 1. 0b - Clock is not running. 1b - Clock is running.
22 BLOCK86	IP block status This bit provides the clock status of LPSPI 0 in partition 1. 0b - Clock is not running. 1b - Clock is running.
21 BLOCK85	IP block status This bit provides the clock status of LPI2C 1 in partition 1. 0b - Clock is not running. 1b - Clock is running.
20	IP block status

Table continues on the next page...

Table continued from the previous page...

Field	Function
BLOCK84	This bit provides the clock status of LPI2C 0 in partition 1. 0b - Clock is not running. 1b - Clock is running.
19 BLOCK83	IP block status This bit provides the clock status of block 83 in partition 1. 0b - Clock is not running. 1b - Clock is running.
18 BLOCK82	IP block status This bit provides the clock status of block 82 in partition 1. 0b - Clock is not running. 1b - Clock is running.
17 BLOCK81	IP block status This bit provides the clock status of LPUART 7 in partition 1. 0b - Clock is not running. 1b - Clock is running.
16 BLOCK80	IP block status This bit provides the clock status of LPUART 6 in partition 1. 0b - Clock is not running. 1b - Clock is running.
15 BLOCK79	IP block status This bit provides the clock status of LPUART 5 in partition 1. 0b - Clock is not running. 1b - Clock is running.
14 BLOCK78	IP block status This bit provides the clock status of LPUART 4 in partition 1. 0b - Clock is not running. 1b - Clock is running.
13 BLOCK77	IP block status This bit provides the clock status of LPUART 3 in partition 1. 0b - Clock is not running. 1b - Clock is running.

Table continues on the next page...

Table continued from the previous page...

Field	Function
12 BLOCK76	IP block status This bit provides the clock status of LPUART 2 in partition 1. 0b - Clock is not running. 1b - Clock is running.
11 BLOCK75	IP block status This bit provides the clock status of LPUART 1 in partition 1. 0b - Clock is not running. 1b - Clock is running.
10 BLOCK74	IP block status This bit provides the clock status of LPUART 0 in partition 1. 0b - Clock is not running. 1b - Clock is running.
9 BLOCK73	IP block status This bit provides the clock status of FlexIO in partition 1. 0b - Clock is not running. 1b - Clock is running.
8 BLOCK72	IP block status This bit provides the clock status of FlexCAN 7 in partition 1. 0b - Clock is not running. 1b - Clock is running.
7 BLOCK71	IP block status This bit provides the clock status of FlexCAN 6 in partition 1. 0b - Clock is not running. 1b - Clock is running.
6 BLOCK70	IP block status This bit provides the clock status of FlexCAN 5 in partition 1. 0b - Clock is not running. 1b - Clock is running.
5 BLOCK69	IP block status This bit provides the clock status of FlexCAN 4 in partition 1. 0b - Clock is not running.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Clock is running.
4 BLOCK68	IP block status This bit provides the clock status of FlexCAN 3 in partition 1. 0b - Clock is not running. 1b - Clock is running.
3 BLOCK67	IP block status This bit provides the clock status of FlexCAN 2 in partition 1. 0b - Clock is not running. 1b - Clock is running.
2 BLOCK66	IP block status This bit provides the clock status of FlexCAN 1 in partition 1. 0b - Clock is not running. 1b - Clock is running.
1 BLOCK65	IP block status This bit provides the clock status of FlexCAN 0 in partition 1. 0b - Clock is not running. 1b - Clock is running.
0 BLOCK64	IP block status This bit provides the clock status of PIT 3 in partition 1. 0b - Clock is not running. 1b - Clock is running.

46.7.43 Partition 1 COFB Set 3 Clock Status Register (PRTN1_COFB3_STAT)

Offset

Register	Offset
PRTN1_COFB3_STAT	31Ch

Function

This register provides the status of set 3 of COFBs inside partition 1.

NOTE

The reset value of this register can vary depending on the availability of active clock pulses inside partition 1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BLOCK127	BLOCK126	BLOCK125	BLOCK124	BLOCK123	BLOCK122	BLOCK121	BLOCK120	BLOCK119	BLOCK118	BLOCK117	BLOCK116	BLOCK115	BLOCK114	BLOCK113	BLOCK112
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	BLOCK110	0	BLOCK108	BLOCK107	BLOCK106	BLOCK105	BLOCK104	BLOCK103	BLOCK102	BLOCK101	0	BLOCK99	BLOCK98	BLOCK97	BLOCK96
W																
Reset	0	1	0	1	1	1	1	1	1	1	1	0	1	1	1	0

Fields

Field	Function
31 BLOCK127	IP block status This bit provides the clock status of AES Application 2 in partition 1. 0b - Clock is not running. 1b - Clock is running.
30 BLOCK126	IP block status This bit provides the clock status of AES Application 2 in partition 1. 0b - Clock is not running. 1b - Clock is running.
29 BLOCK125	IP block status This bit provides the clock status of AES Application 2 in partition 1. 0b - Clock is not running. 1b - Clock is running.
28 BLOCK124	IP block status This bit provides the clock status of AES Application 2 in partition 1. 0b - Clock is not running. 1b - Clock is running.
27 BLOCK123	IP block status This bit provides the clock status of AES Application 1 in partition 1. 0b - Clock is not running. 1b - Clock is running.
26	IP block status

Table continues on the next page...

Table continued from the previous page...

Field	Function
BLOCK122	This bit provides the clock status of AES Application 1 in partition 1. 0b - Clock is not running. 1b - Clock is running.
25 BLOCK121	IP block status This bit provides the clock status of AES Application 1 in partition 1. 0b - Clock is not running. 1b - Clock is running.
24 BLOCK120	IP block status This bit provides the clock status of AES Application 1 in partition 1. 0b - Clock is not running. 1b - Clock is running.
23 BLOCK119	IP block status This bit provides the clock status of AES Application 0 in partition 1. 0b - Clock is not running. 1b - Clock is running.
22 BLOCK118	IP block status This bit provides the clock status of AES Application 0 in partition 1. 0b - Clock is not running. 1b - Clock is running.
21 BLOCK117	IP block status This bit provides the clock status of AES Application 0 in partition 1. 0b - Clock is not running. 1b - Clock is running.
20 BLOCK116	IP block status This bit provides the clock status of AES Application 0 in partition 1. 0b - Clock is not running. 1b - Clock is running.
19 BLOCK115	IP block status This bit provides the clock status of AES Accelerator in partition 1. 0b - Clock is not running. 1b - Clock is running.

Table continues on the next page...

Table continued from the previous page...

Field	Function
18 BLOCK114	IP block status This bit provides the clock status of AES Accelerator in partition 1. 0b - Clock is not running. 1b - Clock is running.
17 BLOCK113	IP block status This bit provides the clock status of AES Accelerator in partition 1. 0b - Clock is not running. 1b - Clock is running.
16 BLOCK112	IP block status This bit provides the clock status of AES Accelerator in partition 1. 0b - Clock is not running. 1b - Clock is running.
15 —	Reserved This field is reserved and read returns zeros.
14 BLOCK110	IP block status This bit provides the clock status of block 110 in partition 1. 0b - Clock is not running. 1b - Clock is running.
13 —	Reserved This field is reserved and read returns zeros.
12 BLOCK108	IP block status This bit provides the clock status of block 108 in partition 1. 0b - Clock is not running. 1b - Clock is running.
11 BLOCK107	IP block status This bit provides the clock status of block 107 in partition 1. 0b - Clock is not running. 1b - Clock is running.
10 BLOCK106	IP block status This bit provides the clock status of block 106 in partition 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Clock is not running. 1b - Clock is running.
9 BLOCK105	IP block status This bit provides the clock status of block 105 in partition 1. 0b - Clock is not running. 1b - Clock is running.
8 BLOCK104	IP block status This bit provides the clock status of STCU in partition 1. 0b - Clock is not running. 1b - Clock is running.
7 BLOCK103	IP block status This bit provides the clock status of block 103 in partition 1. 0b - Clock is not running. 1b - Clock is running.
6 BLOCK102	IP block status This bit provides the clock status of block 102 in partition 1. 0b - Clock is not running. 1b - Clock is running.
5 BLOCK101	IP block status This bit provides the clock status of block 101 in partition 1. 0b - Clock is not running. 1b - Clock is running.
4 —	Reserved This field is reserved and read returns zeros.
3 BLOCK99	IP block status This bit provides the clock status of block 99 in partition 1. 0b - Clock is not running. 1b - Clock is running.
2 BLOCK98	IP block status This bit provides the clock status of block 98 in partition 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Clock is not running. 1b - Clock is running.
1 BLOCK97	IP block status This bit provides the clock status of block 97 in partition 1. 0b - Clock is not running. 1b - Clock is running.
0 BLOCK96	IP block status This bit provides the clock status of CRC in partition 1. 0b - Clock is not running. 1b - Clock is running.

46.7.44 Partition 1 COFB Set 0 Clock Enable Register (PRTN1_COFB0_CLKEN)

Offset

Register	Offset
PRTN1_COFB0_CLKEN	330h

Function

This register provides clock control signaling to the individual COFBs in set 0 inside partition 1. Whenever a partition clock enable (non-core) hardware process is initiated, the value of logic-1 in the corresponding bit locations of this register enables the clock to the corresponding block in the partition.

NOTE

The reset value of this register is not defined and is as per the availability of the clock source. See Chip-specific MC_ME information for clock source availability.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	REQ31	0	REQ29	REQ28	0	0	0	REQ24	REQ23	0	REQ21	0	0	0	0	0
W																
Reset	0	1	0	1	1	1	1	0	0	0	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	REQ15	REQ14	REQ13	REQ12	REQ11	REQ10	REQ9	REQ8	REQ7	REQ6	REQ5	REQ4	REQ3	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1

Fields

Field	Function
31 REQ31	Clock enable This bit provides the clock enable control for INTM in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
30 —	Reserved This field is reserved and read returns zeros.
29 REQ29	Clock enable This bit provides the clock enable control for STM 0 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
28 REQ28	Clock enable This bit provides the clock enable control for SWT 0 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
27 —	Reserved This field is reserved and read returns zeros.
26 —	Reserved This field is reserved and read returns zeros.
25 —	Reserved This field is reserved and read returns zeros.
24 REQ24	Clock enable This bit provides the clock enable control for MSCM in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
23 REQ23	Clock enable This bit provides the clock enable control for ERM_0 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
22 —	Reserved This field is reserved and read returns zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
21 REQ21	Clock enable This bit provides the clock enable control for SDA-AP and Debug APB Paged Area in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
20 —	Reserved This field is reserved and read returns zeros.
19 —	Reserved This field is reserved and read returns zeros.
18 —	Reserved This field is reserved and read returns zeros.
17 —	Reserved This field is reserved and read returns zeros.
16 —	Reserved This field is reserved and read returns zeros.
15 REQ15	Clock enable This bit provides the clock enable control for eDMA TCD 11 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
14 REQ14	Clock enable This bit provides the clock enable control for eDMA TCD 10 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
13 REQ13	Clock enable This bit provides the clock enable control for eDMA TCD 9 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
12 REQ12	Clock enable This bit provides the clock enable control for eDMA TCD 8 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.

Table continues on the next page...

Table continued from the previous page...

Field	Function
11 REQ11	Clock enable This bit provides the clock enable control for eDMA TCD 7 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
10 REQ10	Clock enable This bit provides the clock enable control for eDMA TCD 6 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
9 REQ9	Clock enable This bit provides the clock enable control for eDMA TCD 5 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
8 REQ8	Clock enable This bit provides the clock enable control for eDMA TCD 4 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
7 REQ7	Clock enable This bit provides the clock enable control for eDMA TCD 3 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
6 REQ6	Clock enable This bit provides the clock enable control for eDMA TCD 2 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
5 REQ5	Clock enable This bit provides the clock enable control for eDMA TCD 1 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
4 REQ4	Clock enable This bit provides the clock enable control for eDMA TCD 0 in partition 1. 0b - Clock is turned off.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Clock is turned on.
3 REQ3	Clock enable This bit provides the clock enable control for eDMA_Control_and_Status in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
2 —	Reserved This field is reserved and read returns zeros.
1 —	Reserved This field is reserved and read returns zeros.
0 —	Reserved This field is reserved and read returns zeros.

46.7.45 Partition 1 COFB Set 1 Clock Enable Register (PRTN1_COFB1_CLKEN)

Offset

Register	Offset
PRTN1_COFB1_CLKEN	334h

Function

This register provides clock control signaling to the individual COFBs in set 1 inside partition 1. Whenever a partition clock enable (non-core) hardware process is initiated, the value of logic-1 in the corresponding bit locations of this register enables the clock to the corresponding block in the partition.

NOTE

The reset value of this register is not defined and is as per the availability of the clock source. See Chip-specific MC_ME information for clock source availability.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	REQ63	0	0	0	0	0	REQ57	REQ56	0	0	REQ53	0	REQ51	0	REQ49	0
W																
Reset	0	1	1	1	1	1	0	0	1	1	1	1	1	1	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	REQ47	0	REQ45	0	0	REQ42	0	0	0	0	0	0	0	REQ34	REQ33	REQ32
W																
Reset	0	0	1	0	1	1	1	1	1	1	1	1	1	1	0	0

Fields

Field	Function
31 REQ63	Clock enable This bit provides the clock enable control for PIT 2 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
30 —	Reserved This field is reserved and read returns zeros.
29 —	Reserved This field is reserved and read returns zeros.
28 —	Reserved This field is reserved and read returns zeros.
27 —	Reserved This field is reserved and read returns zeros.
26 —	Reserved This field is reserved and read returns zeros.
25 REQ57	Clock enable This bit provides the clock enable control for PLL 2 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
24 REQ56	Clock enable This bit provides the clock enable control for PLL in partition 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Clock is turned off. 1b - Clock is turned on.
23 —	Reserved This field is reserved and read returns zeros.
22 —	Reserved This field is reserved and read returns zeros.
21 REQ53	Clock enable This bit provides the clock enable control for FXOSC in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
20 —	Reserved This field is reserved and read returns zeros.
19 REQ51	Clock enable This bit provides the clock enable control for SXOSC in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
18 —	Reserved This field is reserved and read returns zeros.
17 REQ49	Clock enable This bit provides the clock enable control for TSPC in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
16 —	Reserved This field is reserved and read returns zeros.
15 REQ47	Clock enable This bit provides the clock enable control for CMU_0-5 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
14 —	Reserved This field is reserved and read returns zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
13 REQ45	Clock enable This bit provides the clock enable control for WKPU in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
12 —	Reserved This field is reserved and read returns zeros.
11 —	Reserved This field is reserved and read returns zeros.
10 REQ42	Clock enable This bit provides the clock enable control for SIUL_VIRTWRAPPER_PDAC3 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
9 —	Reserved This field is reserved and read returns zeros.
8 —	Reserved This field is reserved and read returns zeros.
7 —	Reserved This field is reserved and read returns zeros.
6 —	Reserved This field is reserved and read returns zeros.
5 —	Reserved This field is reserved and read returns zeros.
4 —	Reserved This field is reserved and read returns zeros.
3 —	Reserved This field is reserved and read returns zeros.
2 REQ34	Clock enable This bit provides the clock enable control for RTC in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 REQ33	Clock enable This bit provides the clock enable control for DMAMUX 1 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
0 REQ32	Clock enable This bit provides the clock enable control for DMAMUX 0 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.

46.7.46 Partition 1 COFB Set 2 Clock Enable Register (PRTN1_COFB2_CLKEN)

Offset

Register	Offset
PRTN1_COFB2_CLKEN	338h

Function

This register provides clock control signaling to the individual COFBs in set 2 inside partition 1. Whenever a partition clock enable (non-core) hardware process is initiated, the value of logic-1 in the corresponding bit locations of this register enables the clock to the corresponding block in the partition.

NOTE

The reset value of this register is not defined and is as per the availability of the clock source. See Chip-specific MC_ME information for clock source availability.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	REQ95	0	REQ93	REQ92	REQ91	0	REQ89	REQ88	REQ87	REQ86	REQ85	REQ84	0	0	REQ81	REQ80
W																
Reset	0	0	1	1	0	0	0	0	0	0	0	0	1	1	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	REQ79	REQ78	REQ77	REQ76	REQ75	REQ74	REQ73	REQ72	REQ71	REQ70	REQ69	REQ68	REQ67	REQ66	REQ65	REQ64
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 REQ95	Clock enable This bit provides the clock enable control for TMU in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
30 —	Reserved This field is reserved and read returns zeros.
29 REQ93	Clock enable This bit provides the clock enable control for LPCMP 1 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
28 REQ92	Clock enable This bit provides the clock enable control for LPCMP 0 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
27 REQ91	Clock enable This bit provides the clock enable control for SAI_0 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
26 —	Reserved This field is reserved and read returns zeros.
25 REQ89	Clock enable This bit provides the clock enable control for LPSPI 3 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
24 REQ88	Clock enable This bit provides the clock enable control for LPSPI 2 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
23 REQ87	Clock enable This bit provides the clock enable control for LPSPI 1 in partition 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Clock is turned off. 1b - Clock is turned on.
22 REQ86	Clock enable This bit provides the clock enable control for LPSPI 0 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
21 REQ85	Clock enable This bit provides the clock enable control for LPI2C 1 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
20 REQ84	Clock enable This bit provides the clock enable control for LPI2C 0 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
19 —	Reserved This field is reserved and read returns zeros.
18 —	Reserved This field is reserved and read returns zeros.
17 REQ81	Clock enable This bit provides the clock enable control for LPUART 7 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
16 REQ80	Clock enable This bit provides the clock enable control for LPUART 6 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
15 REQ79	Clock enable This bit provides the clock enable control for LPUART 5 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
14	Clock enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
REQ78	This bit provides the clock enable control for LPUART 4 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
13 REQ77	Clock enable This bit provides the clock enable control for LPUART 3 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
12 REQ76	Clock enable This bit provides the clock enable control for LPUART 2 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
11 REQ75	Clock enable This bit provides the clock enable control for LPUART 1 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
10 REQ74	Clock enable This bit provides the clock enable control for LPUART 0 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
9 REQ73	Clock enable This bit provides the clock enable control for FlexIO in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
8 REQ72	Clock enable This bit provides the clock enable control for FlexCAN 7 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
7 REQ71	Clock enable This bit provides the clock enable control for FlexCAN 6 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.

Table continues on the next page...

Table continued from the previous page...

Field	Function
6 REQ70	Clock enable This bit provides the clock enable control for FlexCAN 5 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
5 REQ69	Clock enable This bit provides the clock enable control for FlexCAN 4 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
4 REQ68	Clock enable This bit provides the clock enable control for FlexCAN 3 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
3 REQ67	Clock enable This bit provides the clock enable control for FlexCAN 2 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
2 REQ66	Clock enable This bit provides the clock enable control for FlexCAN 1 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
1 REQ65	Clock enable This bit provides the clock enable control for FlexCAN 0 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
0 REQ64	Clock enable This bit provides the clock enable control for PIT 3 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.

46.7.47 Partition 1 COFB Set 3 Clock Enable Register (PRTN1_COFB3_CLKEN)

Offset

Register	Offset
PRTN1_COFB3_CLKEN	33Ch

Function

This register provides clock control signaling to the individual COFBs in set 3 inside partition 1. Whenever a partition clock enable (non-core) hardware process is initiated, the value of logic-1 in the corresponding bit locations of this register enables the clock to the corresponding block in the partition.

NOTE

The reset value of this register is not defined and is as per the availability of the clock source. See Chip-specific MC_ME information for clock source availability.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	REQ127	REQ126	REQ125	REQ124	REQ123	REQ122	REQ121	REQ120	REQ119	REQ118	REQ117	REQ116	REQ115	REQ114	REQ113	REQ112
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	REQ104	0	REQ102	0	0	0	0	0	REQ96
W																
Reset	0	1	0	1	1	1	1	1	1	1	1	0	1	1	1	0

Fields

Field	Function
31 REQ127	Clock enable This bit provides the clock enable control for AES Application 2 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
30 REQ126	Clock enable This bit provides the clock enable control for AES Application 2 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
29 REQ125	Clock enable This bit provides the clock enable control for AES Application 2 in partition 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Clock is turned off. 1b - Clock is turned on.
28 REQ124	Clock enable This bit provides the clock enable control for AES Application 2 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
27 REQ123	Clock enable This bit provides the clock enable control for AES Application 1 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
26 REQ122	Clock enable This bit provides the clock enable control for AES Application 1 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
25 REQ121	Clock enable This bit provides the clock enable control for AES Application 1 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
24 REQ120	Clock enable This bit provides the clock enable control for AES Application 1 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
23 REQ119	Clock enable This bit provides the clock enable control for AES Application 0 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
22 REQ118	Clock enable This bit provides the clock enable control for AES Application 0 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
21	Clock enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
REQ117	This bit provides the clock enable control for AES Application 0 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
20 REQ116	Clock enable This bit provides the clock enable control for AES Application 0 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
19 REQ115	Clock enable This bit provides the clock enable control for AES Accelerator in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
18 REQ114	Clock enable This bit provides the clock enable control for AES Accelerator in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
17 REQ113	Clock enable This bit provides the clock enable control for AES Accelerator in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
16 REQ112	Clock enable This bit provides the clock enable control for AES Accelerator in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
15 —	Reserved This field is reserved and read returns zeros.
14 —	Reserved This field is reserved and read returns zeros.
13 —	Reserved This field is reserved and read returns zeros.
12	Reserved This field is reserved and read returns zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
11 —	Reserved This field is reserved and read returns zeros.
10 —	Reserved This field is reserved and read returns zeros.
9 —	Reserved This field is reserved and read returns zeros.
8 REQ104	Clock enable This bit provides the clock enable control for STCU in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
7 —	Reserved This field is reserved and read returns zeros.
6 REQ102	Clock enable This bit provides the clock enable control for block 102 in partition 1. 0b - Clock is turned off. 1b - Clock is turned on.
5 —	Reserved This field is reserved and read returns zeros.
4 —	Reserved This field is reserved and read returns zeros.
3 —	Reserved This field is reserved and read returns zeros.
2 —	Reserved This field is reserved and read returns zeros.
1 —	Reserved This field is reserved and read returns zeros.
0 REQ96	Clock enable This bit provides the clock enable control for CRC in partition 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Clock is turned off. 1b - Clock is turned on.

46.7.48 Partition 2 Process Configuration Register (PRTN2_PCONF)

Offset

Register	Offset
PRTN2_PCONF	500h

Function

This register provides a configuration for the hardware processes corresponding to partition 2. Each of the configuration bit corresponds to the 'nature' of the processes; for example, enabling/disabling and the trigger is controlled by the corresponding field in the PRTN2_PUPD register. When valid KEY combinations are written onto the CTL_KEY register, the PRTN2_PCONF and PRTN2_PUPD registers are used to determine the hardware processes to be executed. These are triggered in parallel and independent of each other. All dependent processes should be requested one after another from the software.

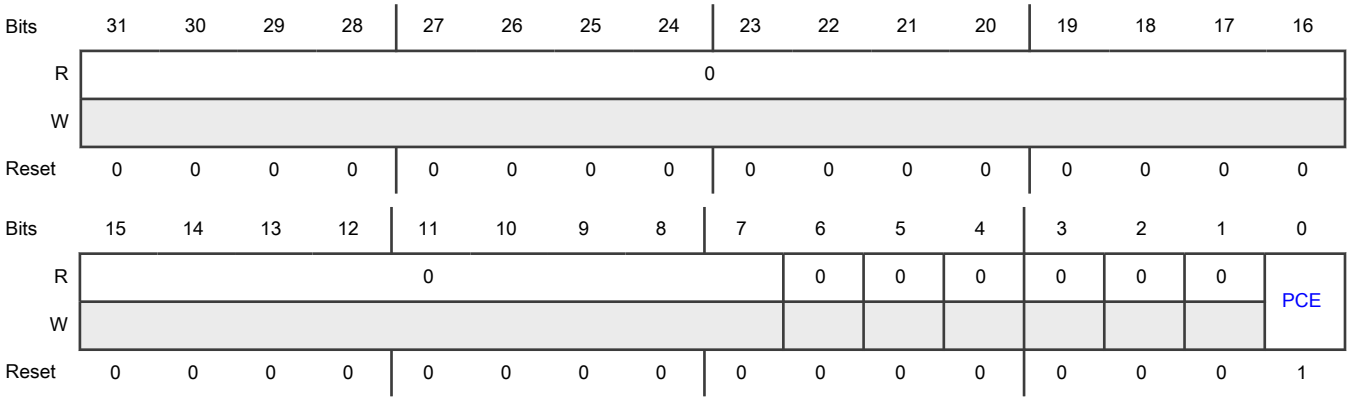
NOTE

The partition clock enable/disable are not standalone and must be done coherently in a fixed sequence. For details, see Software Reset Partition Turn-On Flow Chart and Software reset partition turn-off flowchart in Reset chapter.

NOTE

See chip-specific MC_ME information to check if this register is implemented on chip.

Diagram



Fields

Field	Function
31-7	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	This field is reserved and read returns zeros.
6 —	Reserved This field is reserved and read returns zeros.
5 —	Reserved This field is reserved and read returns zeros.
4 —	Reserved This field is reserved and read returns zeros.
3 —	Reserved This field is reserved and read returns zeros.
2 —	Reserved This field is reserved and read returns zeros.
1 —	Reserved This field is reserved and read returns zeros.
0 PCE	Partition clock enable This bit controls whether the clock to IPs (other than core(s)) in the partition should be enabled or disabled. 0b - Disable the clock to IPs 1b - Enable the clock to IPs

46.7.49 Partition 2 Process Update Register (PRTN2_PUPD)

Offset

Register	Offset
PRTN2_PUPD	504h

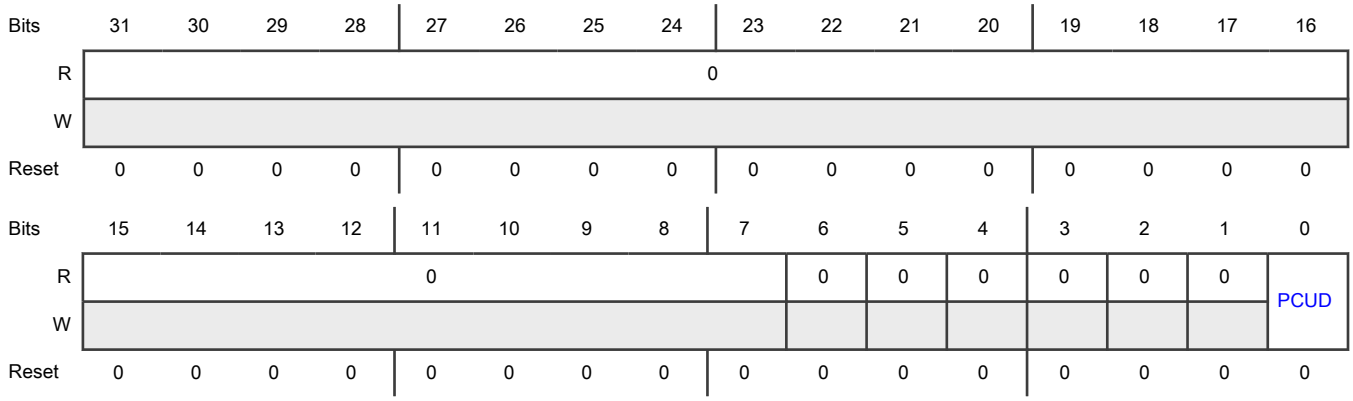
Function

This register provides trigger signaling for the hardware processes corresponding to partition 2. Each of the control bit acts as a trigger for the corresponding hardware processes. When valid KEY combinations are written onto the CTL_KEY register, the hardware checks the bit fields that are programmed as logic-1 in this register, and then triggers the hardware process per the value in the corresponding bit field in the PRTN2_PCONF register. When the hardware process is finished the corresponding bit in this register is auto-cleared to logic-0.

NOTE

See chip-specific MC_ME information to check if this register is implemented on chip.

Diagram



Fields

Field	Function
31-7 —	Reserved This field is reserved and read returns zeros.
6 —	Reserved This field is reserved and read returns zeros.
5 —	Reserved This field is reserved and read returns zeros.
4 —	Reserved This field is reserved and read returns zeros.
3 —	Reserved This field is reserved and read returns zeros.
2 —	Reserved This field is reserved and read returns zeros.
1 —	Reserved This field is reserved and read returns zeros.
0 PCUD	Partition clock update This bit controls whether the hardware processes for enabling/disabling the clock to IPs (other than core(s)) in the partition should be triggered or not. 0b - Do not trigger the hardware process 1b - Trigger the hardware process

46.7.50 Partition 2 Status Register (PRTN2_STAT)

Offset

Register	Offset
PRTN2_STAT	508h

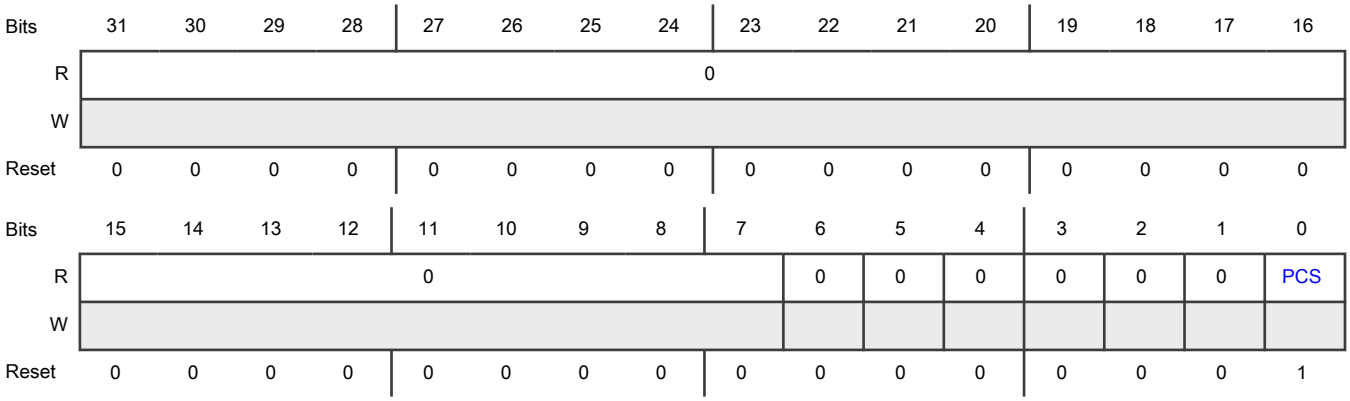
Function

This register provides the current status of the control signals from the partition 2.

NOTE

See chip-specific MC_ME information to check if this register is implemented on chip.

Diagram



Fields

Field	Function
31-7 —	Reserved This field is reserved and read returns zeros.
6 —	Reserved This field is reserved and read returns zeros.
5 —	Reserved This field is reserved and read returns zeros.
4 —	Reserved This field is reserved and read returns zeros.
3 —	Reserved This field is reserved and read returns zeros.
2	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	This field is reserved and read returns zeros.
1 —	Reserved This field is reserved and read returns zeros.
0 PCS	Partition clock status This bit provides the status of the clock to partition. 0b - Clock is inactive 1b - Clock is active

46.7.51 Partition 2 COFB Set 0 Clock Status Register (PRTN2_COFB0_STAT)

Offset

Register	Offset
PRTN2_COFB0_STAT	510h

Function

This register provides the status of set 0 of COFBs inside partition 2.

NOTE

The reset value of this register can vary depending on the availability of active clock pulses inside partition 2.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BLOCK31	BLOCK30	BLOCK29	BLOCK28	BLOCK27	BLOCK26	BLOCK25	BLOCK24	BLOCK23	BLOCK22	BLOCK21	BLOCK20	BLOCK19	BLOCK18	BLOCK17	BLOCK16
W																
Reset	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BLOCK15	BLOCK14	BLOCK13	BLOCK12	BLOCK11	BLOCK10	BLOCK9	BLOCK8	BLOCK7	BLOCK6	BLOCK5	BLOCK4	BLOCK3	BLOCK2	BLOCK1	BLOCK0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1

Fields

Field	Function
31 BLOCK31	IP block status This bit provides the clock status of STM 3 in partition 2. 0b - Clock is not running. 1b - Clock is running.
30 BLOCK30	IP block status This bit provides the clock status of STM 2 in partition 2. 0b - Clock is not running. 1b - Clock is running.
29 BLOCK29	IP block status This bit provides the clock status of STM 1 in partition 2. 0b - Clock is not running. 1b - Clock is running.
28 BLOCK28	IP block status This bit provides the clock status of SWT 2 in partition 2. 0b - Clock is not running. 1b - Clock is running.
27 BLOCK27	IP block status This bit provides the clock status of SWT 1 in partition 2. 0b - Clock is not running. 1b - Clock is running.
26 BLOCK26	IP block status This bit provides the clock status of block 26 in partition 2. 0b - Clock is not running. 1b - Clock is running.
25 BLOCK25	IP block status This bit provides the clock status of block 25 in partition 2. 0b - Clock is not running. 1b - Clock is running.
24 BLOCK24	IP block status This bit provides the clock status of SEMA42 in partition 2. 0b - Clock is not running.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Clock is running.
23 BLOCK23	IP block status This bit provides the clock status of eDMA TCD 31 in partition 2. 0b - Clock is not running. 1b - Clock is running.
22 BLOCK22	IP block status This bit provides the clock status of eDMA TCD 30 in partition 2. 0b - Clock is not running. 1b - Clock is running.
21 BLOCK21	IP block status This bit provides the clock status of eDMA TCD 29 in partition 2. 0b - Clock is not running. 1b - Clock is running.
20 BLOCK20	IP block status This bit provides the clock status of eDMA TCD 28 in partition 2. 0b - Clock is not running. 1b - Clock is running.
19 BLOCK19	IP block status This bit provides the clock status of eDMA TCD 27 in partition 2. 0b - Clock is not running. 1b - Clock is running.
18 BLOCK18	IP block status This bit provides the clock status of eDMA TCD 26 in partition 2. 0b - Clock is not running. 1b - Clock is running.
17 BLOCK17	IP block status This bit provides the clock status of eDMA TCD 25 in partition 2. 0b - Clock is not running. 1b - Clock is running.
16 BLOCK16	IP block status This bit provides the clock status of eDMA TCD 24 in partition 2.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Clock is not running. 1b - Clock is running.
15 BLOCK15	IP block status This bit provides the clock status of eDMA TCD 23 in partition 2. 0b - Clock is not running. 1b - Clock is running.
14 BLOCK14	IP block status This bit provides the clock status of eDMA TCD 22 in partition 2. 0b - Clock is not running. 1b - Clock is running.
13 BLOCK13	IP block status This bit provides the clock status of eDMA TCD 21 in partition 2. 0b - Clock is not running. 1b - Clock is running.
12 BLOCK12	IP block status This bit provides the clock status of eDMA TCD 20 in partition 2. 0b - Clock is not running. 1b - Clock is running.
11 BLOCK11	IP block status This bit provides the clock status of eDMA TCD 19 in partition 2. 0b - Clock is not running. 1b - Clock is running.
10 BLOCK10	IP block status This bit provides the clock status of eDMA TCD 18 in partition 2. 0b - Clock is not running. 1b - Clock is running.
9 BLOCK9	IP block status This bit provides the clock status of eDMA TCD 17 in partition 2. 0b - Clock is not running. 1b - Clock is running.
8	IP block status

Table continues on the next page...

Table continued from the previous page...

Field	Function
BLOCK8	This bit provides the clock status of eDMA TCD 16 in partition 2. 0b - Clock is not running. 1b - Clock is running.
7 BLOCK7	IP block status This bit provides the clock status of eDMA TCD 15 in partition 2. 0b - Clock is not running. 1b - Clock is running.
6 BLOCK6	IP block status This bit provides the clock status of eDMA TCD 14 in partition 2. 0b - Clock is not running. 1b - Clock is running.
5 BLOCK5	IP block status This bit provides the clock status of eDMA TCD 13 in partition 2. 0b - Clock is not running. 1b - Clock is running.
4 BLOCK4	IP block status This bit provides the clock status of eDMA TCD 12 in partition 2. 0b - Clock is not running. 1b - Clock is running.
3 BLOCK3	IP block status This bit provides the clock status of block 3 in partition 2. 0b - Clock is not running. 1b - Clock is running.
2 BLOCK2	IP block status This bit provides the clock status of block 2 in partition 2. 0b - Clock is not running. 1b - Clock is running.
1 BLOCK1	IP block status This bit provides the clock status of block 1 in partition 2. 0b - Clock is not running. 1b - Clock is running.

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 BLOCK0	IP block status This bit provides the clock status of block 0 in partition 2. 0b - Clock is not running. 1b - Clock is running.

46.7.52 Partition 2 COFB Set 1 Clock Status Register (PRTN2_COFB1_STAT)

Offset

Register	Offset
PRTN2_COFB1_STAT	514h

Function

This register provides the status of set 1 of COFBs inside partition 2.

NOTE

The reset value of this register can vary depending on the availability of active clock pulses inside partition 2.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BLOCK63	BLOCK62	0	0	BLOCK59	BLOCK58	0	0	BLOCK55	0	0	0	BLOCK51	0	0	BLOCK48
W																
Reset	1	1	0	0	1	1	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BLOCK47	0	0	0	0	BLOCK42	BLOCK41	BLOCK40	BLOCK39	BLOCK38	BLOCK37	BLOCK36	BLOCK35	BLOCK34	BLOCK33	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 BLOCK63	IP block status This bit provides the clock status of block 63 in partition 2. 0b - Clock is not running.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Clock is running.
30 BLOCK62	IP block status This bit provides the clock status of block 62 in partition 2. 0b - Clock is not running. 1b - Clock is running.
29 —	Reserved This field is reserved and read returns zeros.
28 —	Reserved This field is reserved and read returns zeros.
27 BLOCK59	IP block status This bit provides the clock status of block 59 in partition 2. 0b - Clock is not running. 1b - Clock is running.
26 BLOCK58	IP block status This bit provides the clock status of LPCMP 2 in partition 2. 0b - Clock is not running. 1b - Clock is running.
25 —	Reserved This field is reserved and read returns zeros.
24 —	Reserved This field is reserved and read returns zeros.
23 BLOCK55	IP block status This bit provides the clock status of SAI 1 in partition 2. 0b - Clock is not running. 1b - Clock is running.
22 —	Reserved This field is reserved and read returns zeros.
21 —	Reserved This field is reserved and read returns zeros.
20	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	This field is reserved and read returns zeros.
19 BLOCK51	IP block status This bit provides the clock status of QuadSPI in partition 2. 0b - Clock is not running. 1b - Clock is running.
18 —	Reserved This field is reserved and read returns zeros.
17 —	Reserved This field is reserved and read returns zeros.
16 BLOCK48	IP block status This bit provides the clock status of LPSPI 5 in partition 2. 0b - Clock is not running. 1b - Clock is running.
15 BLOCK47	IP block status This bit provides the clock status of LPSPI 4 in partition 2. 0b - Clock is not running. 1b - Clock is running.
14 —	Reserved This field is reserved and read returns zeros.
13 —	Reserved This field is reserved and read returns zeros.
12 —	Reserved This field is reserved and read returns zeros.
11 —	Reserved This field is reserved and read returns zeros.
10 BLOCK42	IP block status This bit provides the clock status of LPUART 15 in partition 2. 0b - Clock is not running. 1b - Clock is running.
9	IP block status

Table continues on the next page...

Table continued from the previous page...

Field	Function
BLOCK41	This bit provides the clock status of LPUART 14 in partition 2. 0b - Clock is not running. 1b - Clock is running.
8 BLOCK40	IP block status This bit provides the clock status of LPUART 13 in partition 2. 0b - Clock is not running. 1b - Clock is running.
7 BLOCK39	IP block status This bit provides the clock status of LPUART 12 in partition 2. 0b - Clock is not running. 1b - Clock is running.
6 BLOCK38	IP block status This bit provides the clock status of LPUART 11 in partition 2. 0b - Clock is not running. 1b - Clock is running.
5 BLOCK37	IP block status This bit provides the clock status of LPUART 10 in partition 2. 0b - Clock is not running. 1b - Clock is running.
4 BLOCK36	IP block status This bit provides the clock status of LPUART 9 in partition 2. 0b - Clock is not running. 1b - Clock is running.
3 BLOCK35	IP block status This bit provides the clock status of LPUART 8 in partition 2. 0b - Clock is not running. 1b - Clock is running.
2 BLOCK34	IP block status This bit provides the clock status of GMAC 1 in partition 2. 0b - Clock is not running. 1b - Clock is running.

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 BLOCK33	IP block status This bit provides the clock status of GMAC 0 in partition 2. 0b - Clock is not running. 1b - Clock is running.
0 —	Reserved This field is reserved and read returns zeros.

46.7.53 Partition 2 COFB Set 2 Clock Status Register (PRTN2_COFB2_STAT)

Offset

Register	Offset
PRTN2_COFB2_STAT	518h

Function

This register provides the status of set 2 of COFBs inside partition 2.

NOTE

The reset value of this register can vary depending on the availability of active clock pulses inside partition 2.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BLOCK95	BLOCK94	BLOCK93	BLOCK92	BLOCK91	BLOCK90	BLOCK89	BLOCK88	BLOCK87	BLOCK86	BLOCK85	BLOCK84	BLOCK83	BLOCK82	BLOCK81	BLOCK80
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BLOCK79	BLOCK78	BLOCK77	BLOCK76	BLOCK75	BLOCK74	BLOCK73	BLOCK72	0	BLOCK70	BLOCK69	BLOCK68	BLOCK67	0	BLOCK65	BLOCK64
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Fields

Field	Function
31	IP block status

Table continues on the next page...

Table continued from the previous page...

Field	Function
BLOCK95	This bit provides the clock status of FlexCAN 11 in partition 2. 0b - Clock is not running. 1b - Clock is running.
30 BLOCK94	IP block status This bit provides the clock status of FlexCAN 10 in partition 2. 0b - Clock is not running. 1b - Clock is running.
29 BLOCK93	IP block status This bit provides the clock status of FlexCAN 9 in partition 2. 0b - Clock is not running. 1b - Clock is running.
28 BLOCK92	IP block status This bit provides the clock status of FlexCAN 8 in partition 2. 0b - Clock is not running. 1b - Clock is running.
27 BLOCK91	IP block status This bit provides the clock status of AES Application 7 in partition 2. 0b - Clock is not running. 1b - Clock is running.
26 BLOCK90	IP block status This bit provides the clock status of AES Application 7 in partition 2. 0b - Clock is not running. 1b - Clock is running.
25 BLOCK89	IP block status This bit provides the clock status of AES Application 7 in partition 2. 0b - Clock is not running. 1b - Clock is running.
24 BLOCK88	IP block status This bit provides the clock status of AES Application 7 in partition 2. 0b - Clock is not running. 1b - Clock is running.

Table continues on the next page...

Table continued from the previous page...

Field	Function
23 BLOCK87	IP block status This bit provides the clock status of AES Application 6 in partition 2. 0b - Clock is not running. 1b - Clock is running.
22 BLOCK86	IP block status This bit provides the clock status of AES Application 6 in partition 2. 0b - Clock is not running. 1b - Clock is running.
21 BLOCK85	IP block status This bit provides the clock status of AES Application 6 in partition 2. 0b - Clock is not running. 1b - Clock is running.
20 BLOCK84	IP block status This bit provides the clock status of AES Application 6 in partition 2. 0b - Clock is not running. 1b - Clock is running.
19 BLOCK83	IP block status This bit provides the clock status of AES Application 5 in partition 2. 0b - Clock is not running. 1b - Clock is running.
18 BLOCK82	IP block status This bit provides the clock status of AES Application 5 in partition 2. 0b - Clock is not running. 1b - Clock is running.
17 BLOCK81	IP block status This bit provides the clock status of AES Application 5 in partition 2. 0b - Clock is not running. 1b - Clock is running.
16 BLOCK80	IP block status This bit provides the clock status of AES Application 5 in partition 2. 0b - Clock is not running.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Clock is running.
15 BLOCK79	IP block status This bit provides the clock status of AES Application 4 in partition 2. 0b - Clock is not running. 1b - Clock is running.
14 BLOCK78	IP block status This bit provides the clock status of AES Application 4 in partition 2. 0b - Clock is not running. 1b - Clock is running.
13 BLOCK77	IP block status This bit provides the clock status of AES Application 4 in partition 2. 0b - Clock is not running. 1b - Clock is running.
12 BLOCK76	IP block status This bit provides the clock status of AES Application 4 in partition 2. 0b - Clock is not running. 1b - Clock is running.
11 BLOCK75	IP block status This bit provides the clock status of AES Application 3 in partition 2. 0b - Clock is not running. 1b - Clock is running.
10 BLOCK74	IP block status This bit provides the clock status of AES Application 3 in partition 2. 0b - Clock is not running. 1b - Clock is running.
9 BLOCK73	IP block status This bit provides the clock status of AES Application 3 in partition 2. 0b - Clock is not running. 1b - Clock is running.
8 BLOCK72	IP block status This bit provides the clock status of AES Application 3 in partition 2.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Clock is not running. 1b - Clock is running.
7 —	Reserved This field is reserved and read returns zeros.
6 BLOCK70	IP block status This bit provides the clock status of EIM 3 in partition 2. 0b - Clock is not running. 1b - Clock is running.
5 BLOCK69	IP block status This bit provides the clock status of EIM 2 in partition 2. 0b - Clock is not running. 1b - Clock is running.
4 BLOCK68	IP block status This bit provides the clock status of EIM 1 in partition 2. 0b - Clock is not running. 1b - Clock is running.
3 BLOCK67	IP block status This bit provides the clock status of EIM 0 in partition 2. 0b - Clock is not running. 1b - Clock is running.
2 —	Reserved This field is reserved and read returns zeros.
1 BLOCK65	IP block status This bit provides the clock status of block 65 in partition 2. 0b - Clock is not running. 1b - Clock is running.
0 BLOCK64	IP block status This bit provides the clock status of block 64 in partition 2. 0b - Clock is not running. 1b - Clock is running.

46.7.54 Partition 2 COFB Set 3 Clock Status Register (PRTN2_COFB3_STAT)

Offset

Register	Offset
PRTN2_COFB3_STAT	51Ch

Function

This register provides the status of set 3 of COFBs inside partition 2.

NOTE

The reset value of this register can vary depending on the availability of active clock pulses inside partition 2.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	BLOCK98	BLOCK97	BLOCK96
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1

Fields

Field	Function
31 —	Reserved This field is reserved and read returns zeros.
30 —	Reserved This field is reserved and read returns zeros.
29 —	Reserved This field is reserved and read returns zeros.
28 —	Reserved This field is reserved and read returns zeros.
27 —	Reserved This field is reserved and read returns zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
26 —	Reserved This field is reserved and read returns zeros.
25 —	Reserved This field is reserved and read returns zeros.
24 —	Reserved This field is reserved and read returns zeros.
23 —	Reserved This field is reserved and read returns zeros.
22 —	Reserved This field is reserved and read returns zeros.
21 —	Reserved This field is reserved and read returns zeros.
20 —	Reserved This field is reserved and read returns zeros.
19 —	Reserved This field is reserved and read returns zeros.
18 —	Reserved This field is reserved and read returns zeros.
17 —	Reserved This field is reserved and read returns zeros.
16 —	Reserved This field is reserved and read returns zeros.
15 —	Reserved This field is reserved and read returns zeros.
14 —	Reserved This field is reserved and read returns zeros.
13 —	Reserved This field is reserved and read returns zeros.
12	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	This field is reserved and read returns zeros.
11 —	Reserved This field is reserved and read returns zeros.
10 —	Reserved This field is reserved and read returns zeros.
9 —	Reserved This field is reserved and read returns zeros.
8 —	Reserved This field is reserved and read returns zeros.
7 —	Reserved This field is reserved and read returns zeros.
6 —	Reserved This field is reserved and read returns zeros.
5 —	Reserved This field is reserved and read returns zeros.
4 —	Reserved This field is reserved and read returns zeros.
3 —	Reserved This field is reserved and read returns zeros.
2 BLOCK98	IP block status This bit provides the clock status of block 98 in partition 2. 0b - Clock is not running. 1b - Clock is running.
1 BLOCK97	IP block status This bit provides the clock status of block 97 in partition 2. 0b - Clock is not running. 1b - Clock is running.
0 BLOCK96	IP block status This bit provides the clock status of block 96 in partition 2.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Clock is not running. 1b - Clock is running.

46.7.55 Partition 2 COFB Set 0 Clock Enable Register (PRTN2_COFB0_CLKEN)

Offset

Register	Offset
PRTN2_COFB0_CLKEN	530h

Function

This register provides clock control signaling to the individual COFBs in set 0 inside partition 2. Whenever a partition clock enable (non-core) hardware process is initiated, the value of logic-1 in the corresponding bit locations of this register enables the clock to the corresponding block in the partition.

NOTE

The reset value of this register is not defined and is as per the availability of the clock source. See Chip-specific MC_ME information for clock source availability.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	REQ31	REQ30	REQ29	REQ28	REQ27	0	0	REQ24	REQ23	REQ22	REQ21	REQ20	REQ19	REQ18	REQ17	REQ16
W																
Reset	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	REQ15	REQ14	REQ13	REQ12	REQ11	REQ10	REQ9	REQ8	REQ7	REQ6	REQ5	REQ4	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1

Fields

Field	Function
31 REQ31	Clock enable This bit provides the clock enable control for STM 3 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.

Table continues on the next page...

Table continued from the previous page...

Field	Function
30 REQ30	Clock enable This bit provides the clock enable control for STM 2 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
29 REQ29	Clock enable This bit provides the clock enable control for STM 1 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
28 REQ28	Clock enable This bit provides the clock enable control for SWT 2 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
27 REQ27	Clock enable This bit provides the clock enable control for SWT 1 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
26 —	Reserved This field is reserved and read returns zeros.
25 —	Reserved This field is reserved and read returns zeros.
24 REQ24	Clock enable This bit provides the clock enable control for SEMA42 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
23 REQ23	Clock enable This bit provides the clock enable control for eDMA TCD 31 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
22 REQ22	Clock enable This bit provides the clock enable control for eDMA TCD 30 in partition 2.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Clock is turned off. 1b - Clock is turned on.
21 REQ21	Clock enable This bit provides the clock enable control for eDMA TCD 29 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
20 REQ20	Clock enable This bit provides the clock enable control for eDMA TCD 28 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
19 REQ19	Clock enable This bit provides the clock enable control for eDMA TCD 27 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
18 REQ18	Clock enable This bit provides the clock enable control for eDMA TCD 26 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
17 REQ17	Clock enable This bit provides the clock enable control for eDMA TCD 25 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
16 REQ16	Clock enable This bit provides the clock enable control for eDMA TCD 24 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
15 REQ15	Clock enable This bit provides the clock enable control for eDMA TCD 23 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
14	Clock enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
REQ14	This bit provides the clock enable control for eDMA TCD 22 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
13 REQ13	Clock enable This bit provides the clock enable control for eDMA TCD 21 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
12 REQ12	Clock enable This bit provides the clock enable control for eDMA TCD 20 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
11 REQ11	Clock enable This bit provides the clock enable control for eDMA TCD 19 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
10 REQ10	Clock enable This bit provides the clock enable control for eDMA TCD 18 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
9 REQ9	Clock enable This bit provides the clock enable control for eDMA TCD 17 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
8 REQ8	Clock enable This bit provides the clock enable control for eDMA TCD 16 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
7 REQ7	Clock enable This bit provides the clock enable control for eDMA TCD 15 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.

Table continues on the next page...

Table continued from the previous page...

Field	Function
6 REQ6	Clock enable This bit provides the clock enable control for eDMA TCD 14 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
5 REQ5	Clock enable This bit provides the clock enable control for eDMA TCD 13 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
4 REQ4	Clock enable This bit provides the clock enable control for eDMA TCD 12 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
3 —	Reserved This field is reserved and read returns zeros.
2 —	Reserved This field is reserved and read returns zeros.
1 —	Reserved This field is reserved and read returns zeros.
0 —	Reserved This field is reserved and read returns zeros.

46.7.56 Partition 2 COFB Set 1 Clock Enable Register (PRTN2_COFB1_CLKEN)

Offset

Register	Offset
PRTN2_COFB1_CLKEN	534h

Function

This register provides clock control signaling to the individual COFBs in set 1 inside partition 2. Whenever a partition clock enable (non-core) hardware process is initiated, the value of logic-1 in the corresponding bit locations of this register enables the clock to the corresponding block in the partition.

NOTE

The reset value of this register is not defined and is as per the availability of the clock source. See Chip-specific MC_ME information for clock source availability.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	REQ63	REQ62	0	0	0	REQ58	0	0	REQ55	0	0	0	REQ51	0	0	REQ48
W																
Reset	1	1	0	0	1	1	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	REQ47	0	0	0	0	REQ42	REQ41	REQ40	REQ39	REQ38	REQ37	REQ36	REQ35	REQ34	REQ33	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 REQ63	Clock enable This bit provides the clock enable control for block 63 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
30 REQ62	Clock enable This bit provides the clock enable control for block 62 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
29 —	Reserved This field is reserved and read returns zeros.
28 —	Reserved This field is reserved and read returns zeros.
27 —	Reserved This field is reserved and read returns zeros.
26 REQ58	Clock enable This bit provides the clock enable control for LPCMP 2 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.

Table continues on the next page...

Table continued from the previous page...

Field	Function
25 —	Reserved This field is reserved and read returns zeros.
24 —	Reserved This field is reserved and read returns zeros.
23 REQ55	Clock enable This bit provides the clock enable control for SAI 1 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
22 —	Reserved This field is reserved and read returns zeros.
21 —	Reserved This field is reserved and read returns zeros.
20 —	Reserved This field is reserved and read returns zeros.
19 REQ51	Clock enable This bit provides the clock enable control for QuadSPI in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
18 —	Reserved This field is reserved and read returns zeros.
17 —	Reserved This field is reserved and read returns zeros.
16 REQ48	Clock enable This bit provides the clock enable control for LPSPI 5 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
15 REQ47	Clock enable This bit provides the clock enable control for LPSPI 4 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.

Table continues on the next page...

Table continued from the previous page...

Field	Function
14 —	Reserved This field is reserved and read returns zeros.
13 —	Reserved This field is reserved and read returns zeros.
12 —	Reserved This field is reserved and read returns zeros.
11 —	Reserved This field is reserved and read returns zeros.
10 REQ42	Clock enable This bit provides the clock enable control for LPUART 15 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
9 REQ41	Clock enable This bit provides the clock enable control for LPUART 14 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
8 REQ40	Clock enable This bit provides the clock enable control for LPUART 13 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
7 REQ39	Clock enable This bit provides the clock enable control for LPUART 12 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
6 REQ38	Clock enable This bit provides the clock enable control for LPUART 11 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
5 REQ37	Clock enable This bit provides the clock enable control for LPUART 10 in partition 2.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Clock is turned off. 1b - Clock is turned on.
4 REQ36	Clock enable This bit provides the clock enable control for LPUART 9 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
3 REQ35	Clock enable This bit provides the clock enable control for LPUART 8 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
2 REQ34	Clock enable This bit provides the clock enable control for GMAC 1 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
1 REQ33	Clock enable This bit provides the clock enable control for GMAC 0 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
0 —	Reserved This field is reserved and read returns zeros.

46.7.57 Partition 2 COFB Set 2 Clock Enable Register (PRTN2_COFB2_CLKEN)

Offset

Register	Offset
PRTN2_COFB2_CLKEN	538h

Function

This register provides clock control signaling to the individual COFBs in set 2 inside partition 2. Whenever a partition clock enable (non-core) hardware process is initiated, the value of logic-1 in the corresponding bit locations of this register enables the clock to the corresponding block in the partition.

NOTE

The reset value of this register is not defined and is as per the availability of the clock source. See Chip-specific MC_ME information for clock source availability.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	REQ95	REQ94	REQ93	REQ92	REQ91	REQ90	REQ89	REQ88	REQ87	REQ86	REQ85	REQ84	REQ83	REQ82	REQ81	REQ80
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	REQ79	REQ78	REQ77	REQ76	REQ75	REQ74	REQ73	REQ72	0	REQ70	REQ69	REQ68	REQ67	0	REQ65	REQ64
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Fields

Field	Function
31 REQ95	Clock enable This bit provides the clock enable control for FlexCAN 11 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
30 REQ94	Clock enable This bit provides the clock enable control for FlexCAN 10 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
29 REQ93	Clock enable This bit provides the clock enable control for FlexCAN 9 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
28 REQ92	Clock enable This bit provides the clock enable control for FlexCAN 8 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
27 REQ91	Clock enable This bit provides the clock enable control for AES Application 7 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
26 REQ90	Clock enable This bit provides the clock enable control for AES Application 7 in partition 2.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Clock is turned off. 1b - Clock is turned on.
25 REQ89	Clock enable This bit provides the clock enable control for AES Application 7 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
24 REQ88	Clock enable This bit provides the clock enable control for AES Application 7 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
23 REQ87	Clock enable This bit provides the clock enable control for AES Application 6 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
22 REQ86	Clock enable This bit provides the clock enable control for AES Application 6 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
21 REQ85	Clock enable This bit provides the clock enable control for AES Application 6 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
20 REQ84	Clock enable This bit provides the clock enable control for AES Application 6 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
19 REQ83	Clock enable This bit provides the clock enable control for AES Application 5 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
18	Clock enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
REQ82	This bit provides the clock enable control for AES Application 5 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
17 REQ81	Clock enable This bit provides the clock enable control for AES Application 5 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
16 REQ80	Clock enable This bit provides the clock enable control for AES Application 5 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
15 REQ79	Clock enable This bit provides the clock enable control for AES Application 4 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
14 REQ78	Clock enable This bit provides the clock enable control for AES Application 4 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
13 REQ77	Clock enable This bit provides the clock enable control for AES Application 4 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
12 REQ76	Clock enable This bit provides the clock enable control for AES Application 4 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
11 REQ75	Clock enable This bit provides the clock enable control for AES Application 3 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.

Table continues on the next page...

Table continued from the previous page...

Field	Function
10 REQ74	Clock enable This bit provides the clock enable control for AES Application 3 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
9 REQ73	Clock enable This bit provides the clock enable control for AES Application 3 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
8 REQ72	Clock enable This bit provides the clock enable control for AES Application 3 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
7 —	Reserved This field is reserved and read returns zeros.
6 REQ70	Clock enable This bit provides the clock enable control for EIM 3 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
5 REQ69	Clock enable This bit provides the clock enable control for EIM 2 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
4 REQ68	Clock enable This bit provides the clock enable control for EIM 1 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
3 REQ67	Clock enable This bit provides the clock enable control for EIM 0 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
2	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	This field is reserved and read returns zeros.
1 REQ65	Clock enable This bit provides the clock enable control for block 65 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
0 REQ64	Clock enable This bit provides the clock enable control for block 64 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.

46.8 Glossary

WFI Wait for interrupt

COFB Collection of functional blocks also referred as number of peripherals

Chapter 47
Power Control Unit (MC_PCU)

47.1 Introduction

The power control unit (MC_PCU) is used for initiating a Standby mode entry that reduces the overall chip power consumption. Power can be saved by disconnecting parts of the chip from the power supply. The blocks inside the chip are grouped into multiple parts having this capability, which are called "power domains".

When a power domain is disconnected from the supply, the power consumption is reduced to zero and the configuration of the every core and module that belongs to such power domain is completely lost. When you reconnect a power domain to the supply voltage, the domain draws an increased current until the power domain reaches its operational voltage. Maximum power saving is achieved by entering the Standby mode.

After the MC_ME asserts a standby entry request, MC_PCU initiates the power sequence, which is non-retractable and includes the handshake with the chip power management controller. The power-up/down sequences are handled by FSMs to ensure a smooth and safe transition into and out of the Standby mode. Exiting the Standby mode can only be done through a system wakeup event, power-on reset, destructive reset, or a functional reset.

47.2 Power sequence FSM

MC_PCU implements an FSM to initiate the power sequencing of the Standby mode entry/exit sequence for the chip.

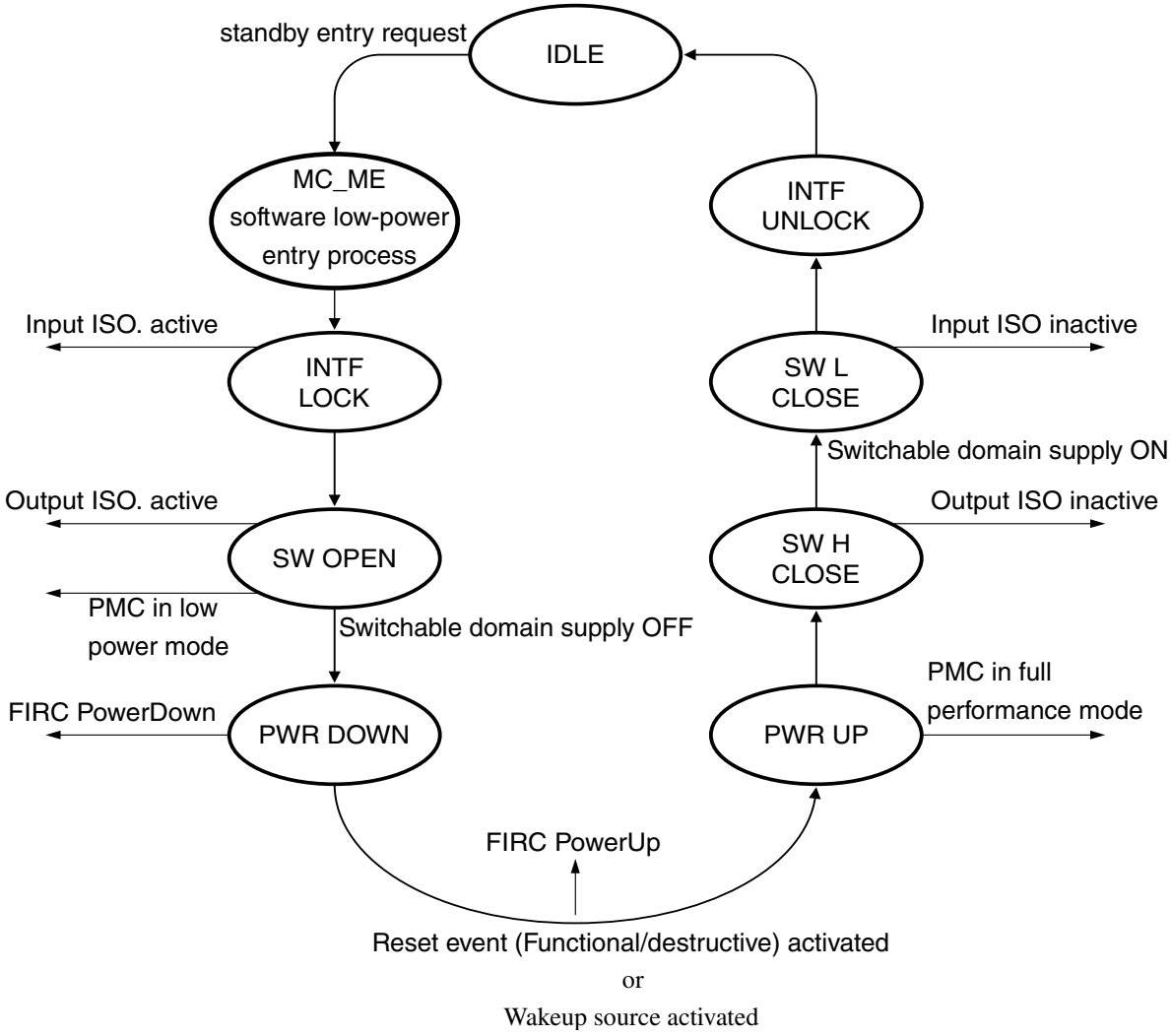


Figure 196. MC_PCU FSM

NOTE

When destructive reset is asserted from MC_RGM, MC_PCU FSM moves to the IDLE state immediately. Indication of this is not shown in the above figure.

Table 266. MC_PCU FSM transition description

State	Name	Exit condition	Signal controlled	Signal monitored
IDLE	Idle state	Standby request received from MC_ME	-	-
MC_ME software low power entry process	MC_ME software low power entry process	Software entry sequence completed (SW4 process completed)	-	-
INTF LOCK	Interface lock state	Input isolation active	Input isolation activation	Input isolation active

Table continues on the next page...

Table 266. MC_PCU FSM transition description (continued)

State	Name	Exit condition	Signal controlled	Signal monitored
SW OPEN	Switch open state	Output isolation active, Switchable domain supply turned off	Output isolation activation, low power mode request to PMC	Output isolation active, Switchable domain supply
PWR DOWN	Power down state	Wakeup/functional reset occurrence	FIRC power down	Wakeup and functional reset
PWR UP	Power up state	PMC in full performance mode	-	-
SW H CLOSE	Switch H close state	Switchable domain supply turned on	-	Switchable domain supply
SW L CLOSE	Switch L close state	Input isolation inactive	Input isolation inactive	Input isolation active
INTF UNLOCK	Interface unlock	-	-	-

47.3 Glossary

FSM Finite state machine

Chapter 48
Wakeup Unit (WKPU)

48.1 Chip-specific WKPU information

48.1.1 WKPU configuration on the chip

WKPU provides the following features:

- A configurable, low-power wake-up capability to the chip from multiple configurable asynchronous wake-up events.
- Support for four internal and up to 60 external source that can generate interrupts or wake-up events.
- Support for an NMI input.

Table 267. WKPU configuration on the chip

Number of sources, interrupts, and vectors	S32K311/S32K310	S32K312/ S32K342/ S32K341/S32K322	S32K344/ S32K324/S32K314	S32K388/ S32K358/ S32K356/ S32K348/ S32K338/ S32K328/S32K389	Description
NMI sources	1	1	1	1	Single NMI pin routed to all application cores
Internal wake-up sources[1]	4	4	4	4	• SWT_0 wake-up, RTC-API wake-up • RTC timeout • Analog comparator round robin wake-up (from LPCMP_0, 1, and 2 round robin wake-up)[2] • RTI wake-up
External wake-up sources	33	59	60	67 [3]	See the IOMUX file attached to this document for details about the number of wake up pins and their assignment to each wake up source.

Table continues on the next page...

Table 267. WKPU configuration on the chip (continued)

Number of sources, interrupts, and vectors	S32K311/S32K310	S32K312/ S32K342/ S32K341/S32K322	S32K344/ S32K324/S32K314	S32K388/ S32K358/ S32K356/ S32K348/ S32K338/ S32K328/S32K389	Description
Glitch filters for external interrupts	33	59	60	67	Glitch filter on each external wake-up source
External interrupt vectors	8	8	8	8	—

1. Internal wakeup sources have only positive polarity, therefore, must not be configured for negedge triggered functionality.
2. Both wake-up sources 0 (RTC-API) and 2 (CMP_*x* trigger mode wake-up) use RTC-API wake-up, with Trigger mode having a higher priority. This means, if you configure LPCMP_*x*.RRCR0[ERR_EN] for either of the comparators, the RTC_API wake-up is used only for the CMP_*x* trigger mode operation. The RTC-API wake-up does not cause wake-up from Standby mode in this scenario.
3. Some of external sources are sharing the same WKPU channels. Please see DCM_GPR chapter to select one from multiple external sources.

All of the aforementioned wake-up sources can be enabled or disabled. Also, you can configure these wake-up sources, by using WKPU configuration registers, to provide wake-up on rising or falling events. See the WKPU register memory map for details.

NOTE

You must use SIUL2 to perform WKPU pin configurations (PUE, PUS, and IBE). For this, you must first configure SIUL2.MSCR[IBE] for the corresponding pin. This chip does not support WKPU-provided pin configurations; it supports only bypass control.

Two kinds of BYPASS connectivity for alternate glitch filters

Below figure is related to the BYPASS functionality of the Analog glitch filters associated with external wakeup pins.

48.1.2 WKPU register fields and their applicability

Table 268. WKPU register fields and applicability

Register	Field	Chips where applicable
NSR	NIF1	S32K324, S32K322
NSR	NOVF1	S32K324, S32K322
NSR	NIF3, NOVF3	S32K388/S32K389
NCR	NLOCK1	S32K324, S32K322
NCR	NDSS1	S32K324, S32K322
NCR	NWRE1	S32K324, S32K322
NCR	NREE1	S32K324, S32K322
NCR	NFEE1	S32K324, S32K322
NCR	NFE1	S32K324, S32K322
NCR	NLOCK3, NDSS3, NWRE3, NREE3, NFEE3, NFE3	S32K388/S32K389

48.1.3 WKPU NMI configuration

WKPU supports one external source that can cause non-maskable interrupts to on-chip cores and wake-up events to the system.

The following figure shows applicable cores, some family chips may not have cores Cortex-M7_1, Cortex-M7_2 or Coretex-M7_3. In case of a wake-up event (internal or external), WKPU initiates the recovery of the chip and feeds an interrupt to the core(s) depending on the configuration. The sections that follow provide details related to the associated configurations.

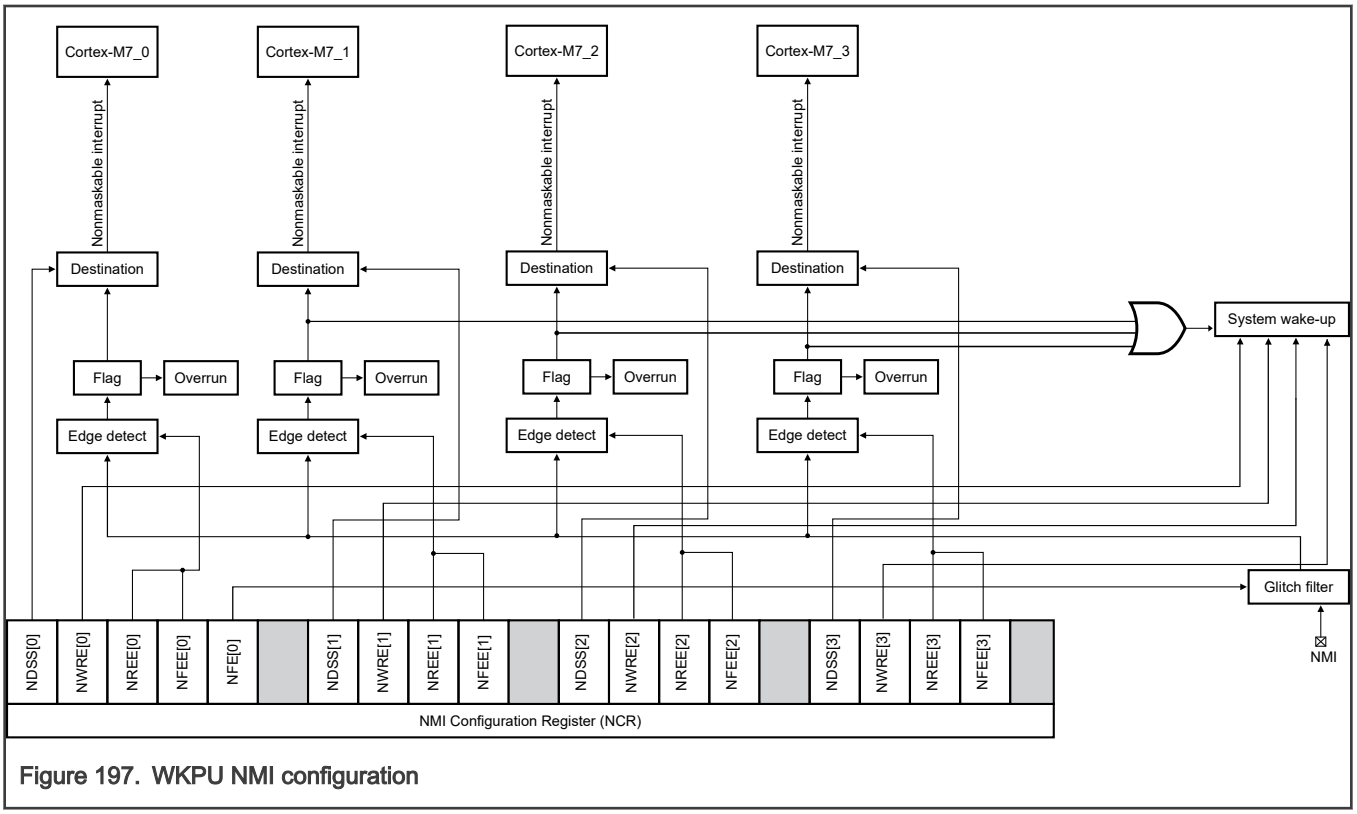


Figure 197. WKPU NMI configuration

48.1.4 WKPU wake-up source connectivity

WKPU allows the external NMI pin to assert the core NMIs on the chip. NMI supports NSR's status and overrun flags. This is what you could do using NCR:

- Control the NMI destination interrupt by configuring NCR[NDSS].
- Control the rising edge, falling edge, or either of the edge reactions to the NMI pin by using bits 2:0 of NCR[NFEE] and NCR[NREE]. The enabling of these edge reactions to the NMI pin is independent of each core.

WKPU supports the capturing of a second event per NMI input before the interrupt is cleared, thus reducing the chance of losing an NMI event.

The following figure shows routing of external wake-up events or interrupts with WKPU and the system interrupt controller.

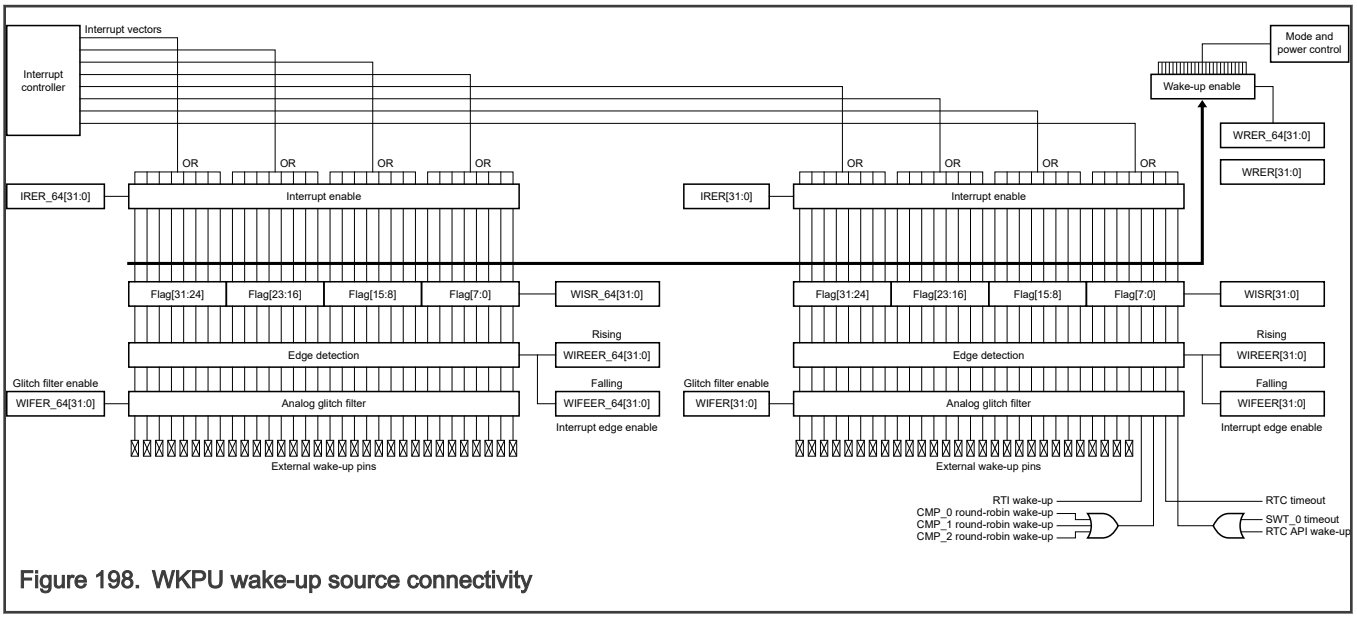


Figure 198. WKPU wake-up source connectivity

This is the wake-up source mapping to WKPU:

- Wake-up source 0 : SWT_0 timeout, RTC-API API wake-up
- Wake-up source 1 : RTC-API RTC timeout
- Wake-up source 2 : Round robin wake-up interrupt (Trigger mode interrupt) from LPCMP_0, LPCMP_1, or LPCMP_2
- Wake-up source 3 : RTI wake-up
- Wake-up source 4 : Wake-up source up to 60-external pin wake-up sources, WKPU[0]-WKPU[59]

If you configure any or all of the LPCMP_*x*.RRCR0[RR_EN] fields to be active, the corresponding CMP_*x* pins must be dedicated for the CMP_*x* operation. In case you are not using any of the CMPs, you can use SIUL2.MSCR*x*[SSS] to configure the pins for digital functionalities.

NOTE

You must enable WKPU (by using MC_ME.PRTN*x*_COFB*y*_CLKEN) before entering any of the chip's low-power modes.

48.2 Overview

WKPU supports 64 external sources that can generate interrupts or wake-up events, and 0 external sources that can cause nonmaskable interrupt requests or wake-up events. Additionally, it combines its wake-up events with those generated by other wake-up sources to supply a single wake-up to the system.

48.2.1 Block diagram

The following figure shows WKPU and its interfaces with the other system components.

NOTE

The signal widths in the following diagram do not depict a particular configuration of this chip. See the chip-specific WKPU information for details.

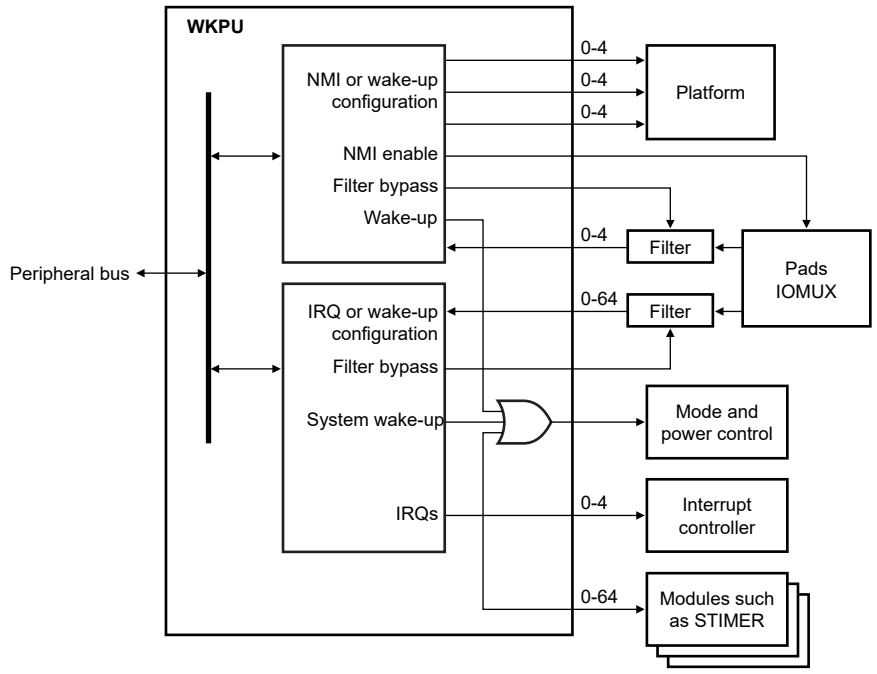


Figure 199. Block diagram

48.2.2 Features

- Supports nonmaskable interrupts including:
  - 4 NMI sources
  - 4 analog glitch filters
  - Independent interrupt destination for each core:
    - Nonmaskable interrupt
    - Critical interrupt
  - Active edge selection control (rise and fall) for events
  - Configurable system wake-up triggering from NMI sources
- Supports external wake-up and interrupts that include the following:
  - One System interrupt vector for interrupt sources
  - 64 analog glitch filters
  - Independent interrupt mask
  - Edge detection
  - Configurable system wake-up triggering from all interrupt sources

48.3 Functional description

48.3.1 Nonmaskable interrupts

WKPU supports the capturing of the second event according to the NMI input before the interrupt becomes 0. This reduces the chance of losing an NMI event, although it creates an overrun condition.

Each NMI passes through a bypassable analog glitch filter.

NOTE

When NMI is disabled, glitch filter control and pad configuration perform. This prevents erroneous triggering by glitches caused by the configuration process itself.

NOTE

The following figure represents a generic configuration and does not represent the configuration of this specific chip. See the chip-specific information for details on this chip's WKPU.

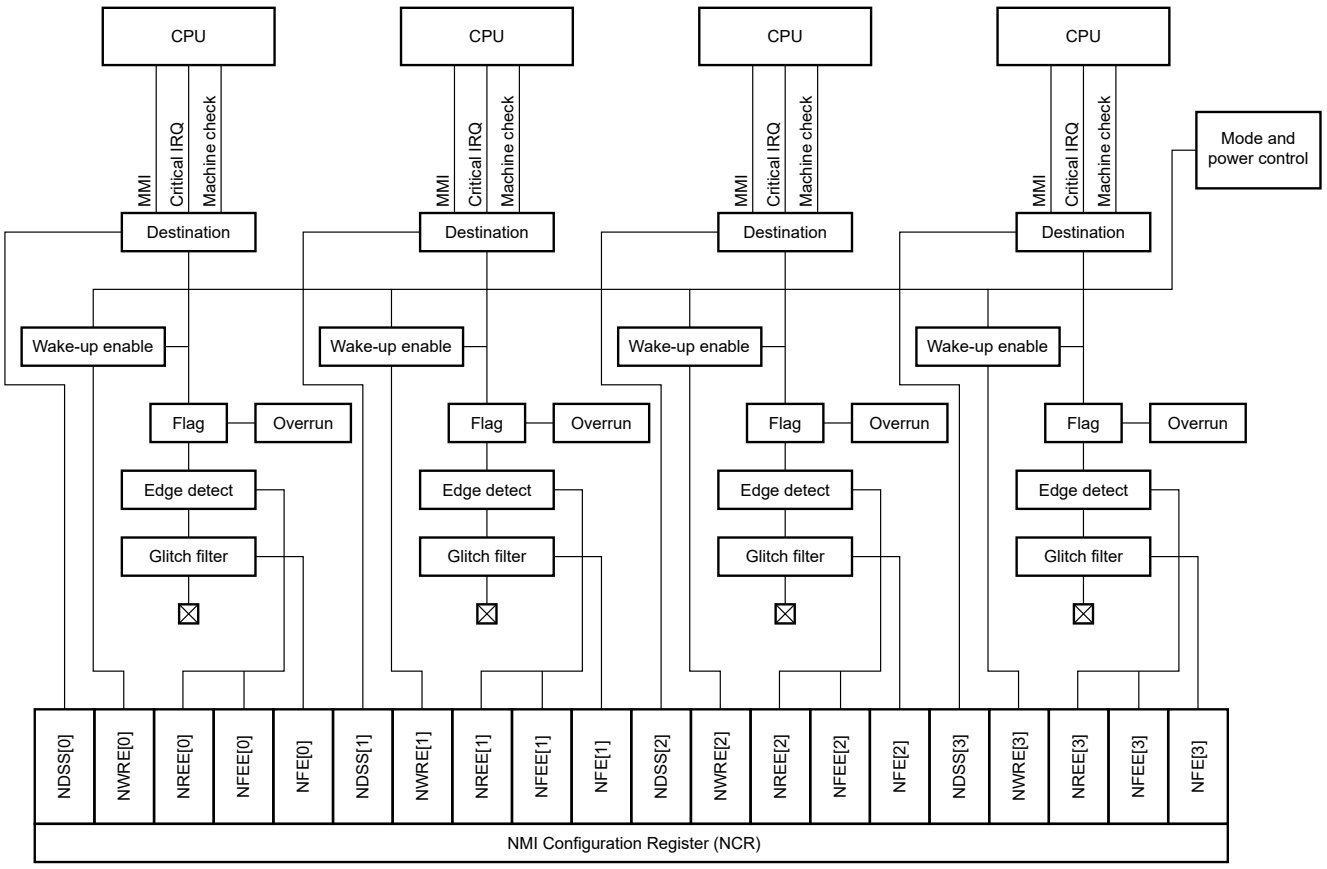


Figure 200. NMI pad diagram

48.3.1.1 NMI management

You can enable or disable each NMI independently. You can perform this by using the registers, NCR which are laid out to contain all configuration bits for a given NMI in a single byte (see NMI Configuration (NCR)). You can configure a pad defined as an NMI to recognize interrupts with an active rising edge, falling edge, or both edges being active. If both the edge events are disabled, it results in no interrupt being detected and does not get configured.

You can also control an active NMI edge through the configuration of the fields, NCR[NREE*n*] and NCR[NFEE*n*].

NOTE

After a reset, NREE and NFEE are set to 0, which disables the NMI functionality and software enables it explicitly.

After a pad's NMI functionality is enabled, the pad cannot be reconfigured to override or disable the NMI. See the chip-specific WKPU information for details of NMI implementation.

Additionally, you can control the NMI destination interrupt through the configuration of the field, NCR[NDSS*n*]. See NMI Configuration (NCR) for details.

Each NMI supports a status flag and an overrun flag, both of which are located in NMI Status Flag (NSR). This register prevents an inadvertent overwriting of the other flags in the same register. The status flag is set whenever an NMI event is detected. The overrun flag is set whenever an NMI event is detected when the status flag is still set (that is, the status flag has not yet been cleared).

NOTE

The overrun flag is cleared by writing 1 to the appropriate overrun field in NMI Status Flag (NSR). If the status field is 0 but the overrun field is still, the pending interrupt is not cleared.

During an NMI ISR, on wake-up of the chip from an NMI, any writes to the ECC-protected memory must have the correct ECC.

48.3.2 Clocking

This module has no clocking considerations.

WKPU has ipg_clk and ipg_clk_s as input clocks having the same clock frequency for register configuration and internal logic.

48.3.3 Interrupts

WKPU supports 4 interrupt vectors to the interrupt controller of the chip. Each interrupt vector supports multiple external interrupt sources from the device pads, with the total across all vectors being equal to the number of external interrupt sources. Each external interrupt source is assigned to exactly one interrupt vector. The interrupt vector assignment follows a sequence: one interrupt vector is for external interrupt sources 0 to N-1, the next is for N to N+M-1, and so on.

See the following figure for an overview of the external interrupt implementation, showing an example of four interrupt vectors with eight external interrupt sources each.

NOTE

The following figure represents a generic WKPU configuration. For details on this chip's configuration, see the chip-specific WKPU information.

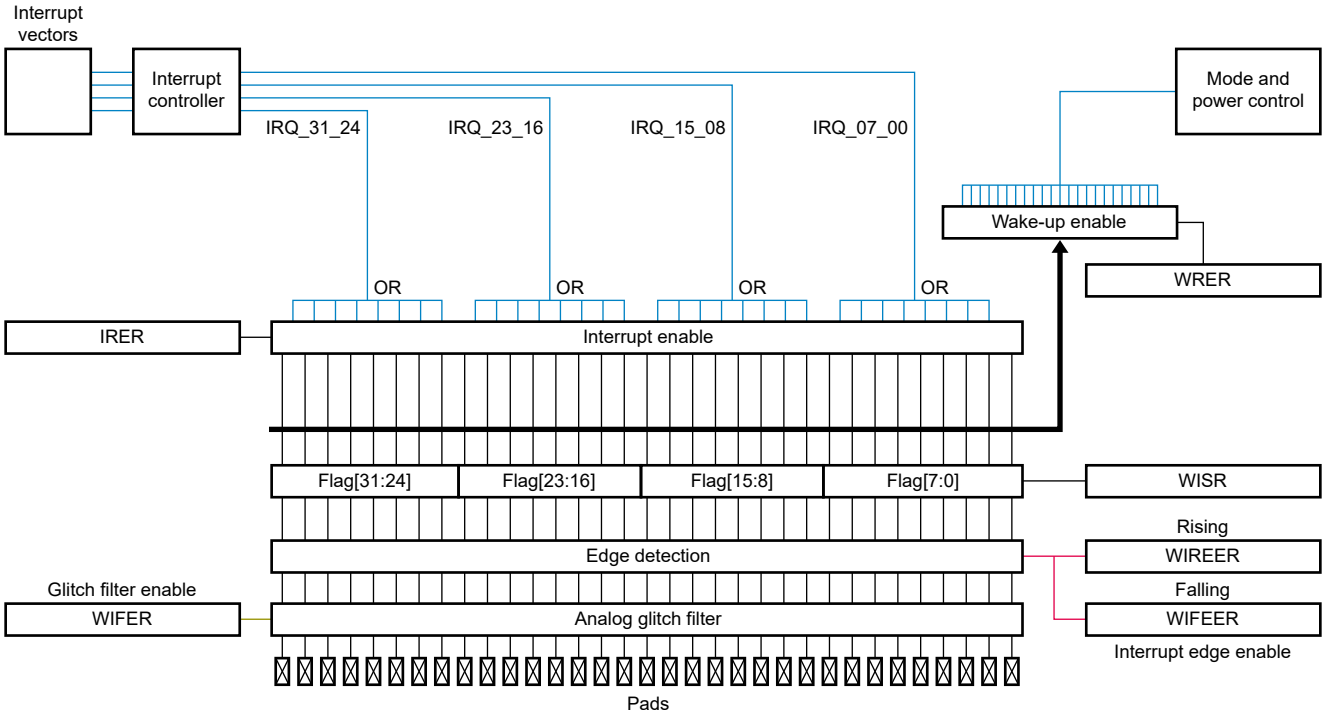


Figure 201. External interrupt pad diagram

All the external interrupt pads within a single group have equal priority. You must search through the group of sources in the most appropriate way for their application.

The priority of the vectors used by the external interrupt pads is set based on the platform and the priority levels of the interrupt controller. However, the chip can allow an independent configuration of pad allocation to each group of interrupts.

The external interrupt lines have a digital glitch filter applied to them.

48.3.3.1 External interrupt management

You can enable or disable each external interrupt independently using a single rolled-up register (Interrupt Request Enable (IRER)). You can configure a pad defined as an external interrupt to recognize interrupts with an active rising edge, an active falling edge, or both edges being active.

NOTE

Writing 0 to both IREE[*n*] and IFEE[*n*] disables the external interrupt functionality for that pad completely (means no system wake-up or interrupt is generated from any activity on that pad).

You can control an active IRQ edge through the configuration of the registers, WIREER and WIFEER.

Each external interrupt supports an individual flag, which is held in the flag register, WISR. This W1C register prevents inadvertent overwriting of other flags in the same register.

48.3.4 External signals

This module has no external signals.

48.4 Initialization

To initialize this module, you must perform the following configuration:

- Glitch filter and pad configuration

- Nonmaskable interrupts
- Wake-up event

48.4.1 Glitch filter and pad configuration

You can perform glitch filter control and pad configuration when the NMI is disabled to avoid erroneous triggering by glitches caused by the configuration process.

When enabling the glitch filter, do not enable the rising and falling-edge events bits (that is, the NCR[NREE], NCR[NFEE], NCR[NREE], and NCR[NFEE]) in the same register write.

48.4.2 Nonmaskable interrupts

If IBE of NMI is tied off to 1, no false interrupt is expected.

48.4.3 Wake-up event

See the chip-specific WKPU information for wake-up event initialization.

48.5 WKPU memory map and registers

48.5.1 WKPU register descriptions

This section provides a detailed description of all the registers accessible in the WKPU module.

NOTE

Reserved registers read as zero and writes have no effect. A transfer error is generated when trying to access a completely reserved register space. The field length of external pad control registers depends on the number of WKPU channels implemented in a chip.

48.5.1.1 WKPU memory map

WKPU base address: 402B_4000h

Offset	Register	Width (In bits)	Access	Reset value
0h	NMI Status Flag (NSR)	32	RW	0000_0000h
8h	NMI Configuration (NCR)	32	RW	6060_6060h
14h	Wake-Up and Interrupt Status Flag (WISR)	32	RW	0000_0000h
18h	Interrupt Request Enable (IRER)	32	RW	0000_0000h
1Ch	Wake-Up Request Enable (WRER)	32	RW	0000_0000h
28h	Wake-Up and Interrupt Rising-Edge Event Enable (WIREER)	32	RW	0000_0000h
2Ch	Wake-Up and Interrupt Falling-Edge Event Enable (WIFEER)	32	RW	0000_0000h
30h	Wake-Up and Interrupt Filter Enable (WIFER)	32	RW	0000_0000h
54h	Wake-Up and Interrupt Status Flag (WISR_64)	32	RW	0000_0000h
58h	Interrupt Request Enable (IRER_64)	32	RW	0000_0000h
5Ch	Wake-Up Request Enable (WRER_64)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
68h	Wake-Up and Interrupt Rising-Edge Event Enable (WIREER_64)	32	RW	0000_0000h
6Ch	Wake-Up and Interrupt Falling-Edge Event Enable (WIFEER_64)	32	RW	0000_0000h
70h	Wake-Up and Interrupt Filter Enable (WIFER_64)	32	RW	0000_0000h

48.5.1.2 NMI Status Flag (NSR)

Offset

Register	Offset
NSR	0h

Function

Holds the nonmaskable interrupt status flags.

NOTE

This register is accessible by 8-, 16-, and 32-bit read/write operations.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	NIF0	NOVF0	Reserved						NIF1	NOVF1	Reserved					
W	W1C	W1C							W1C	W1C						
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	NIF2	NOVF2	Reserved						NIF3	NOVF3	Reserved					
W	W1C	W1C							W1C	W1C						
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 NIF0	NMI Status Flag 0 Causes an interrupt request when an event, as defined by NREE0 and NFEE0, has occurred. If NREE0 or NFEE0 is 1, this flag causes an interrupt request.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field behaves differently for register reads and writes. When reading 0b - No event occurred on the pad 1b - An event occurred When writing 0b - No effect 1b - Clear the flag
30 NOVF0	NMI Overrun Status Flag 0 Indicates whether an overrun has occurred on NMI input 0. This flag has the same current value as of NIF0 (when the NMI event occurs), indicating that the NMI occurred when the previous one was not serviced. If NREE0 or NFEE0 is 1, this flag causes an interrupt request. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No overrun occurred 1b - An overrun occurred When writing 0b - No effect 1b - Clear the flag
29-24 —	Reserved
23 NIF1	NMI Status Flag 1 Causes an interrupt request when an event, as defined by NREE1 and NFEE1, has occurred. If NREE1 or NFEE1 is 1, this flag causes an interrupt request. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No event occurred on the pad 1b - An event occurred When writing

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No effect 1b - Clear the flag
22 NOVF1	NMI Overrun Status Flag 1 Indicates whether an overrun has occurred on NMI input 1. This flag has the same current value as of NIF1 (when the NMI event occurs), indicating that the NMI occurred when the previous one was not serviced. If NREE1 or NFEE1 is 1, this flag causes an interrupt request. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No overrun occurred 1b - An overrun occurred When writing 0b - No effect 1b - Clear the flag
21-16 —	Reserved
15 NIF2	NMI Status Flag 2 Causes an interrupt request when an event, as defined by NREE2 and NFEE2, has occurred. If NREE2 or NFEE2 is 1, this flag causes an interrupt request. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No event occurred on the pad 1b - An event occurred When writing 0b - No effect 1b - Clear the flag
14 NOVF2	NMI Overrun Status Flag 2 Indicates whether an overrun has occurred on NMI input 2. This flag has the same current value as of NIF2 (when the NMI event occurs), indicating that the NMI ocuurred when the previous one was not serviced. If NREE2 or NFEE2 is 1, this flag causes an interrupt request.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field behaves differently for register reads and writes. When reading 0b - No overrun occurred 1b - An overrun occurred When writing 0b - No effect 1b - Clear the flag
13-8 —	Reserved
7 NIF3	NMI Status Flag 3 Causes an interrupt request when an event, as defined by NREE3 and NFEE3, has occurred. If NREE3 or NFEE3 is 1, this flag causes an interrupt request. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No event occurred on the pad 1b - An event occurred When writing 0b - No effect 1b - Clear the flag
6 NOVF3	NMI Overrun Status Flag 3 Indicates whether an overrun has occurred on NMI input 3. This flag has the same current value as of NIF3 (when the NMI event occurs), indicating that the NMI occurred when the previous one was not serviced. If NREE3 or NFEE3 is 1, this flag causes an interrupt request. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No overrun occurred 1b - An overrun occurred When writing

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No effect 1b - Clear the flag
5-0 —	Reserved

48.5.1.3 NMI Configuration (NCR)

Offset

Register	Offset
NCR	8h

Function

Holds the configuration fields for the nonmaskable interrupt settings.

NOTE

- This register is accessible by 8-, 16-, and 32-bit read/write operations.
- Writing 0 to both NREE[*n*] and NFEE[*n*] disables the NMI functionality completely (means no nonmaskable interrupt is generated on any pad activity).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	NLOCK0	NDSS0		NWRE0	0	NREE0	NFEE0	NFE0	NLOCK1	NDSS1		NWRE1	0	NREE1	NFEE1	NFE1
W																
Reset	0	1	1	0	0	0	0	0	0	1	1	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	NLOCK2	NDSS2		NWRE2	0	NREE2	NFEE2	NFE2	NLOCK3_...	NDSS3_RDSS		NWRE3_R...	0	NREE3_R...	NFEE3_R...	NFE3
W																
Reset	0	1	1	0	0	0	0	0	0	1	1	0	0	0	0	0

Fields

Field	Function
31 NLOCK0	NMI Configuration Lock Register 0 Specifies the lock configuration for the NMI.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When you write 1 to this field, it locks the configuration for the NMI until it is unlocked by a system reset or Standby0 mode exit. 0b - No effect 1b - Locks the configuration for the NMI
30-29 NDSS0	NMI Destination Source Select 0 Specifies the NMI destination interrupt to platform. **NOTE** As wake-up interrupt does not support another interrupt than NMI, the destination source select signal bits are reserved and always retain their reset value. This means no other request other than NMI can be generated. 00b - Nonmaskable interrupt 01b - Reserved 10b - Reserved 11b - Reserved. Reserved
28 NWRE0	NMI Wake-Up Request Enable 0 Enables a system wake-up request when the corresponding NIF0 = 1 or NOVFO = 1. 0b - Disable 1b - Enable
27 —	Reserved
26 NREE0	NMI Rising-Edge Events Enable 0 Enables the NMI rising-edge event. 0b - Disable 1b - Enable
25 NFEE0	NMI Falling-Edge Events Enable 0 Enables the NMI falling-edge event. 0b - Disable 1b - Enable
24 NFE0	NMI Filter Enable 0 Enables analog glitch filter on the NMI pad input. 0b - Disable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Enable
23 NLOCK1	NMI Configuration Lock 1 Specifies the lock configuration for the NMI. When you write 1 to this field, it locks the configuration for the NMI until it is unlocked by a system reset or Standby0 mode exit. 0b - No effect 1b - Locks the configuration for the NMI
22-21 NDSS1	NMI Destination Source Select 1 Specifies the NMI destination interrupt to platform. **NOTE** As wake-up does not support another interrupt than NMI, the destination source select signal bits are reserved and always retain their reset value. This means no other request other than NMI can be generated. 00b - Nonmaskable interrupt 01b - Reserved 10b - Reserved 11b - Reserved
20 NWRE1	NMI Wake-Up Request Enable 1 Enables a system wake-up request when the corresponding NIF1 = 1 or NOVF1 = 1. 0b - Disable 1b - Enable
19 —	Reserved
18 NREE1	NMI Rising-Edge Events Enable 1 Enables the NMI rising-edge event. 0b - Disable 1b - Enable
17 NFEE1	NMI Falling-Edge Events Enable 1 Enables the NMI falling-edge event. 0b - Disable 1b - Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
16 NFE1	NMI Filter Enable 1 Enables analog glitch filter on the NMI pad input. 0b - Disable 1b - Enable
15 NLOCK2	NMI Configuration Lock 2 Specifies the lock configuration for the NMI. When you write 1 to this field, it locks the configuration for the NMI until it is unlocked by a system reset. 0b - No effect 1b - Locks the configuration for the NMI
14-13 NDSS2	NMI Destination Source Select 2 Specifies the NMI destination interrupt to platform. **NOTE** Because wake-up does not support any other interrupt other than NMI, the destination source select signal bits are reserved and always retain their reset value. This means no other request other than NMI can be generated. 00b - Nonmaskable interrupt 01b - Reserved 10b - Reserved 11b - Reserved
12 NWRE2	NMI Wake-Up Request Enable 2 Enables a system wake-up request when the corresponding NIF2 = 1 or NOVF2 =1. 0b - Disable 1b - Enable
11 —	Reserved
10 NREE2	NMI Rising-Edge Events Enable 2 Enables the NMI rising-edge event. 0b - Disable 1b - Enable
9 NFEE2	NMI Falling-Edge Events Enable 2 Enables the NMI falling-edge event.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disable 1b - Enable
8 NFE2	NMI Filter Enable 2 Enables analog glitch filter on the NMI pad input. 0b - Disable 1b - Enable
7 NLOCK3_RLOCK	NMI Configuration Lock Register 3 Specifies the lock configuration for the NMI. When you write 1 to this field, it locks the configuration for the NMI until it is unlocked by a system reset . 0b - No effect 1b - Locks the configuration for the NMI
6-5 NDSS3_RDSS	NMI Destination Source Select 3. . 00b - Nonmaskable interrupt 01b - Reserved 10b - Reserved 11b - Reserved
4 NWRE3_RWRE	NMI Wake-Up Request Enable 3 Enables system wake-up requests. 0b - System wake-up requests from the corresponding NIF3 bit are disabled 1b - NIF3 or NOVF3 being set causes a system wake-up request
3 —	Reserved
2 NREE3_RREE	NMI Rising-Edge Events Enable 3. Enables the NMI rising-edge event. 0b - Disable 1b - Enable
1 NFEE3_RFEE	NMI Falling-Edge Events Enable 3 Enables the NMI falling-edge event. 0b - Disable 1b - Enable
0	NMI Filter Enable 3

Table continues on the next page...

Table continued from the previous page...

Field	Function
NFE3	Enables analog glitch filter on the NMI pad input. 0b - Disable 1b - Enable

48.5.1.4 Wake-Up and Interrupt Status Flag (WISR)

Offset

Register	Offset
WISR	14h

Function

Holds the wake-up and interrupt flags.

NOTE

- This register is accessible only by 32-bit read/write operations.
- The status fields associated with on-chip wake-up sources are located to the left of the external wake-up and interrupt status fields and are read-only. The wake-up for these sources must be configured and cleared at the on-chip wake-up source. Also, the configuration registers for the external interrupts/wake-ups do not have corresponding bits.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	EIF															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EIF															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 EIF	External Wake-Up and Interrupt Status Flag

Table continues on the next page...

Field	Function
	Specifies whether an event, as defined by WIREER and WIFEER, has occurred on the pad. If enabled (IRER[*n*]), EIF[*n*] causes an interrupt request. 0b - No event occurred 1b - An event occurred

48.5.1.5 Interrupt Request Enable (IRER)

Offset

Register	Offset
IRER	18h

Function

Enables the interrupt messaging from the wake-up and interrupt pads to the interrupt controller.

NOTE

This register is accessible only by 32-bit read/write operations.

If a pin is disabled through this register, you must write 0 to the corresponding fields in WIFEER and WIREER to ensure that the pin does not respond to any change.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	EIRE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	EIRE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 EIRE	External Interrupt Request Enable Enables interrupt requests from the corresponding field, EIF[*n*]. 0b - Disable 1b - Enable (with EIF[*n*] set)

48.5.1.6 Wake-Up Request Enable (WRER)

Offset

Register	Offset
WRER	1Ch

Function

Enables the system wake-up messaging from the wake-up and interrupt pads to the mode entry and power control modules.

NOTE

This register is accessible only by 32-bit read/write operations.

If a pin is disabled through this register, you must write 0 to the corresponding fields in WIFEER and WIREER to ensure that the pin does not respond to any change.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	WRE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	WRE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 WRE	External Wake-Up Request Enable Enables system wake-up requests from the corresponding field, EIF[*n*]. 0b - Disable 1b - Enable (with EIF[*n*] set)

48.5.1.7 Wake-Up and Interrupt Rising-Edge Event Enable (WIREER)

Offset

Register	Offset
WIREER	28h

Function

Enables rising-edge triggered events on the corresponding wake-up/interrupt pads.

NOTE

- This register is accessible only by 32-bit read/write operations.
- WIREER or WIFEER is configured for rising or falling edge after WRER to enable wake-up source.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	IREE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	IREE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 IREE	External Interrupt Rising-Edge Events Enable Enables interrupt rising-edge event. 0b - Disable 1b - Enable

48.5.1.8 Wake-Up and Interrupt Falling-Edge Event Enable (WIFEER)

Offset

Register	Offset
WIFEER	2Ch

Function

Enables falling-edge triggered events on the corresponding wake-up and interrupt pads.

NOTE

- This register is accessible only by 32-bit read/write operations.
- WIREER or WIFEER is configured for rising or falling edge after WRER to enable wake-up source.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	IFEEx															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	IFEEx															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 IFEEx	External Interrupt Falling-Edge Events Enable Enables interrupt falling-edge event. 0b - Disable 1b - Enable

48.5.1.9 Wake-Up and Interrupt Filter Enable (WIFER)

Offset

Register	Offset
WIFER	30h

Function

Enables an analog filter on the corresponding interrupt pads to filter out glitches on the inputs.

NOTE

This register is accessible only by 32-bit read/write operations.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	IFE															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	IFE															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 IFE	External Interrupt Filter Enable Enables analog glitch filter on the external interrupt pad input. 0b - Disable 1b - Enable

48.5.1.10 Wake-Up and Interrupt Status Flag (WISR_64)

Offset

Register	Offset
WISR_64	54h

Function

Holds the wake-up and interrupt flags.

NOTE

- This register is accessible only by 32-bit read/write operations.
- Status fields associated with on-chip wake-up sources are located to the left of the external wake-up/interrupt status fields and are read-only. The wake-up for these sources must be configured and cleared at the on-chip wake-up source. Also, the configuration registers for the external interrupts/wake-ups do not have corresponding fields.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	EIF_1															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EIF_1															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 EIF_1	External Wake-Up and Interrupt Status Flag Specifies whether an event, as defined by WIREER and WIFEER, has occurred on the pad. If enabled (IRER[*n*]), EIF[*n*] causes an interrupt request. 0b - No event occurred 1b - An event occurred

48.5.1.11 Interrupt Request Enable (IRER_64)

Offset

Register	Offset
IRER_64	58h

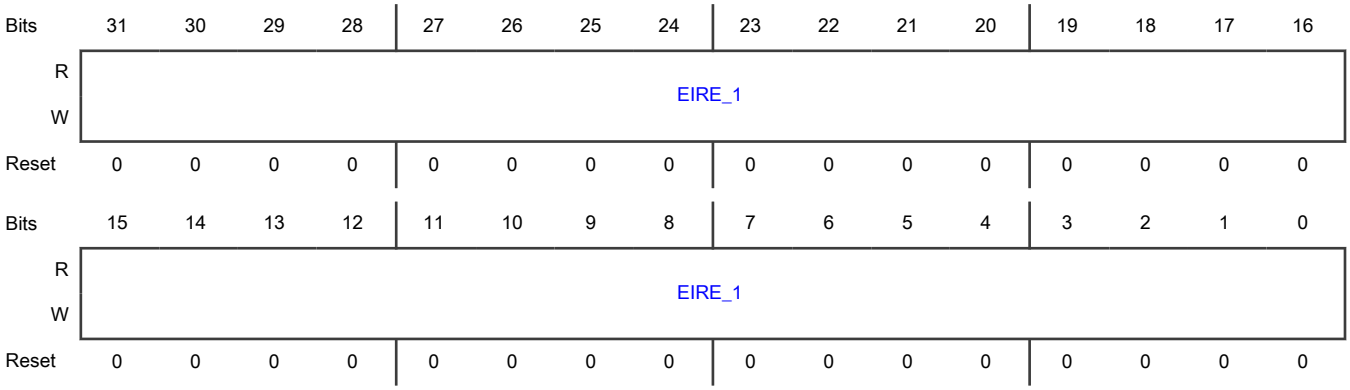
Function

Enables interrupt messaging from the wake-up and interrupt pads to the interrupt controller.

NOTE

This register is accessible only by 32-bit read/write operations.

Diagram



Fields

Field	Function
31-0 EIRE_1	External Interrupt Request Enable Enables interrupt requests from the corresponding field, EIF[*n*]. 0b - Disable 1b - Enable (with EIF[*n*] set)

48.5.1.12 Wake-Up Request Enable (WRER_64)

Offset

Register	Offset
WRER_64	5Ch

Function

Enables system wake-up messaging from the wake-up and interrupt pads to the mode entry and power control modules.

NOTE

This register is accessible only by 32-bit read/write operations.

If a pin is disabled through this register, you must write 0 to the corresponding fields in WIFEER and WIREER to ensure that the pin does not respond to any change.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	WRE_1															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	WRE_1															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 WRE_1	External Wake-Up Request Enable Enables system wake-up requests from the corresponding field, EIF[*n*]. 0b - Disable 1b - Enable (with EIF[*n*] set)

48.5.1.13 Wake-Up and Interrupt Rising-Edge Event Enable (WIREER_64)

Offset

Register	Offset
WIREER_64	68h

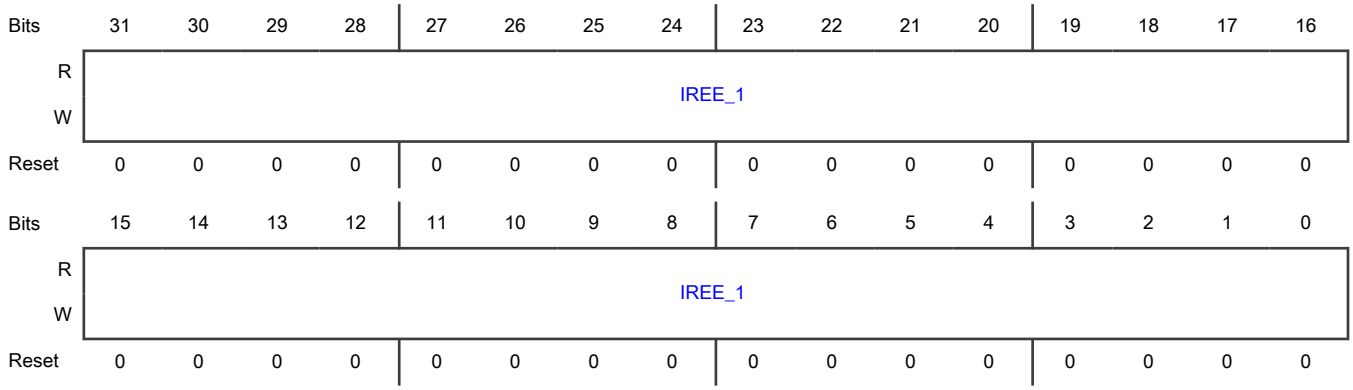
Function

Enables rising-edge triggered events on the corresponding wake-up and interrupt pads.

NOTE

- This register is accessible only by 32-bit read/write operations.
- WIREER or WIFEER is configured for rising or falling-edge after WRER to enable wake-up source.

Diagram



Fields

Field	Function
31-0 IREE_1	External Interrupt Rising-edge Events Enable Enables an external interrupt rising-edge event. 0b - Disable 1b - Enable

48.5.1.14 Wake-Up and Interrupt Falling-Edge Event Enable (WIFEER_64)

Offset

Register	Offset
WIFEER_64	6Ch

Function

Enables falling-edge triggered events on the corresponding wake-up and interrupt pads.

NOTE

- This register is accessible only by 32-bit read/write operations.
- WIREER or WIFEER is configured for rising or falling-edge after WRER to enable a wake-up source.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	IFEEx_1															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	IFEEx_1															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 IFEEx_1	External Interrupt Falling-Edge Events Enable Enables an external interrupt falling-edge event. 0b - Disable 1b - Enable

48.5.1.15 Wake-Up and Interrupt Filter Enable (WIFER_64)

Offset

Register	Offset
WIFER_64	70h

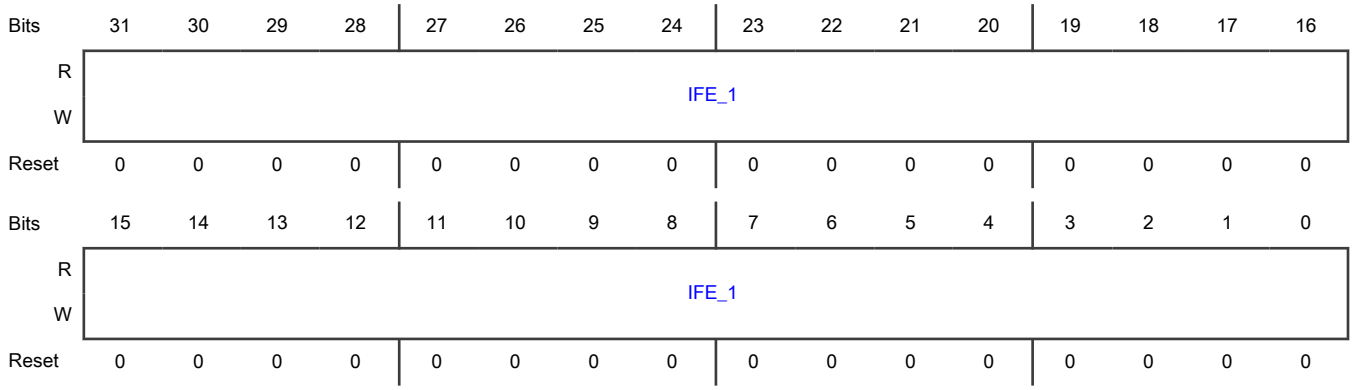
Function

Enables an analog filter on the corresponding interrupt pads to filter out glitches on the inputs.

NOTE

This register is accessible only by 32-bit read/write operations.

Diagram



Fields

Field	Function
31-0 IFE_1	External Interrupt Filter Enable Enables an analog glitch filter on the external interrupt pad input. 0b - Disable 1b - Enable

48.6 Glossary

NMI Nonmaskable interrupts

Chapter 49
Safety Overview

49.1 Introduction

This chip family is developed following the ISO 26262 standards, with derivatives targeting to be operable in a system that fulfills the requirements of the ASIL D or ASIL B safety integrity levels. The S32K388, S32K389, and S32K358 can target an ASIL B safety integrity level with one portion of the chip, while targeting an ASIL D safety integrity level with another portion.

Table 269. ASIL levels

Chip	ASIL level
S32K388, S32K389, S32K358, S32K356, S32K348, S32K344, S32K342, S32K341	D and B
S32K338, S32K328, S32K324, S32K314, S32K322, S32K312, S32K311, S32K310	B

The ASIL level targets the safety processing, which means the software functions are executed as intended:

1. Read instructions from memory.
2. Execute instructions.
3. Read data from memory.
4. Process data.
5. Write back the result data into memory.

Some elements of the safety concept are based on the assumption that the chip is connected to an external SBC or PMIC. The SBC or PMIC performs observations and control functionalities that are essential to fulfill some related functional safety requirements. If you do not use an SBC or PMIC, you must ensure that your S32K-family chip provides an equivalent functionality that properly manages the corresponding interface(s).

The following figure illustrates the safety interface between the chip and the SBC or PMIC.

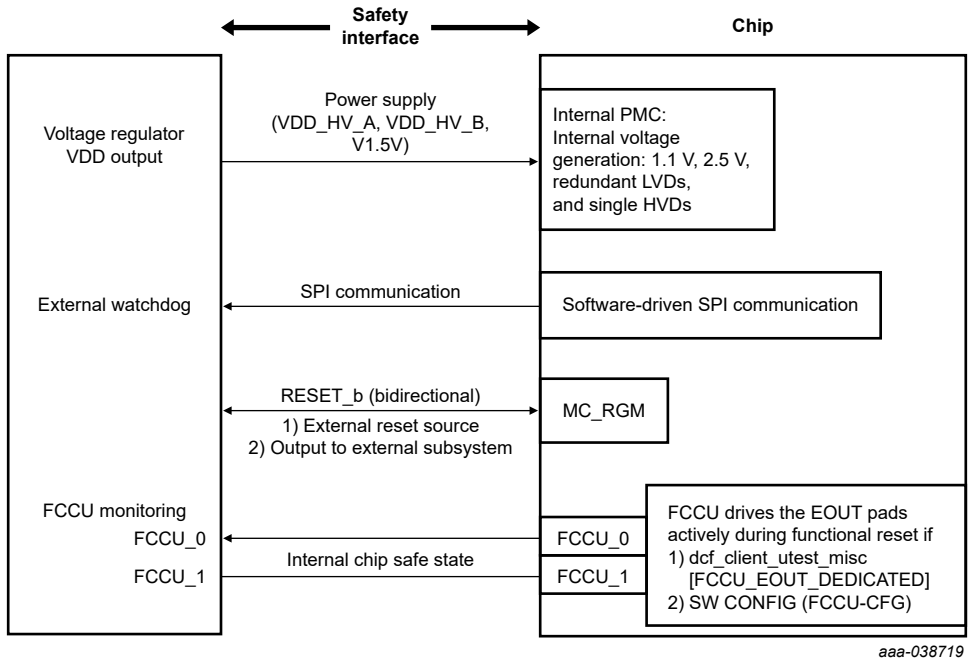


Figure 202. Safety interface between the chip and the SBC or PMIC

In the safety context, the chip interfaces can be classified into the groups shown in the table.

Table 270. Chip interface groups

Interface	Description
Power supply	This interface is between the SBC and the chip. It ensures that the supply to the chip is in the correct range. In case of any low-voltage event, the chip has POR, LVR, and LVD circuits in place and in case of any HVD event, the chip raises an interrupt. The SBC must ensure that the voltage regulator outputs never exceed the allowed range (more specifically for HVD range).
Communication	Responsible for communication between the SBC and the chip. Though the chip supports multiple communication protocols (UART, LIN, SPI, I2C, FlexCAN, and so on), SPI is the preferred communication with SBC. This is relevant to initialize an external watchdog when the chip is inoperative for a considerably long time (this is indicated by the pin states of the communication interface).
Reset	The chip consists of a reset bidirectional pin interfaced with SBC. The SBC can initiate a chip reset via this pin as a safety reaction in case of: • Extreme critical faults • An inoperative chip • Stuck cases based on the criticality and the application requirements The SBC also samples the reset pin state to identify the chip condition (whether in running state or in reset).
FCCU	The chip indicates the chip faults to the SBC via FCCU EOUT pins through this interface.

The interfaces in Table 270 ensure the chip's operational safety and integrity.

49.2 Safety architecture elements

The chip safety architecture consists of following elements that operate in an interconnected way to meet the ASIL requirements:

- Cortex-M7 core complex (including the cache controllers) operating in delayed lockstep (ASIL D only)
- eDMA controller
- Internal windowed watchdogs with independent clock sources
- Power supply monitoring with redundant low-voltage detectors, single high-voltage detectors, and internal ADC connection to check the internal voltages during application
- Robust clock monitoring, including PLL loss of lock detection
- Embedded flash memory with ECC SECDED and address encoding (parallel address path) check
- System RAM with ECC SECDED and error detection
- Cortex-M7 cache memories with ECC SECDED
- Peripherals memories (EMAC, FlexCAN) with ECC SECDED

- End-to-End EDC (E2E EDC), with address encoding and monitoring of the control signals done by a dedicated module (EDC interface gaskets and XBIC). This ensures the safety of storage operations and of the data path to the internal storage (RAM, flash memory, core cache) and peripherals across the crossbar switch. See the XBIC chapter for details.
- Hardware CRC module that supports end-to-end data check integrity for any data transfer in the system (SRAM to peripherals via DMA transfer, external interfaces to SRAM/peripherals, and so on)
- Cortex-M7 MPUs
- XRDC for memory and peripheral protection
- AIPS_Lite peripheral protection with trusted master-slave connection
- Register protection mechanism for safety-critical registers
- On-chip temperature sensor for temperature monitoring
- Hardware self-test that can be triggered by software:
  - LBIST to detect latent faults in functional logic as well as in safety integrity mechanisms

NOTE

LBIST is not supported in S32K312, S32K311, and S32K310.

  - MBIST to ensure integrity of the memories in the chip (SRAM, ITCM, DTCM, peripheral memories, and so on)
- ADC self-test
- FCCU for error collection and reaction, including reporting error status to system; FCCU supports these programmable reaction types:
  - Interrupt
  - Functional reset
- Error pads indicate the chip's internal state to the external chip interface or SBC.
- EIM to inject errors into the memories and interface gaskets to verify the error-detection features of the memory controllers and the interface gaskets
- ERM to collect diagnostic information from memory controllers in case of an error event

These hardware elements can be supported by software safety measures, for example a structural core self-test or the check-the-checker software library.

49.3 I/O peripherals

The arrangement of I/O peripherals across peripheral bridges allows redundant use of peripherals while limiting possible causes of CCF. Redundant use includes using equivalent peripherals in a replicated way as well as using functionally different peripherals in, for example, feedback measurement loops. Comparison of redundant operation is the responsibility of the application software, not the safety hardware mechanism.

The peripherals are distributed evenly across the peripheral bridges (AIPS*n*), except for singular modules like EMAC, QuadSPI, and so on.

NOTE

EMAC and QuadSPI are not present in S32K312 and S32K311.

49.4 Self-test

The chip supports a self-test operation. Your safety software must initiate the self-test operation by configuring STCU2; the chip does not initiate it. STCU2 then controls the self-test operation. Self-test supports both MBIST and LBIST. After the self-test operation completes, the chip enters a reset sequence. Self-test results are stored in STCU2 and your safety software can read the results after the reset sequence.

Figure 203 depicts the processing steps related to the self-test operation, and its linkage to the chip reset and an application start. You decide whether to run a self-test or to skip it before starting an application. You must also specify the self-test configuration before relinquishing control to STCU2 for performing the self-test operation. This processing finishes by reentering the chip reset sequence, depending on whether the chip encountered an unrecoverable fault during the self-test operation. When an unrecoverable fault is encountered during the self-test operation, the chip enters a reset sequence by performing a destructive reset. When no such fault is encountered, the self-test operation completes by entering the chip sequence with a functional reset sequence. You can prevent reset cycling by limiting the amount of resets permitted; the chip shuts down when this limit has been reached. See the "Functional reset escalation" and "Destructive reset escalation" sections in the MC_RGM chapter.

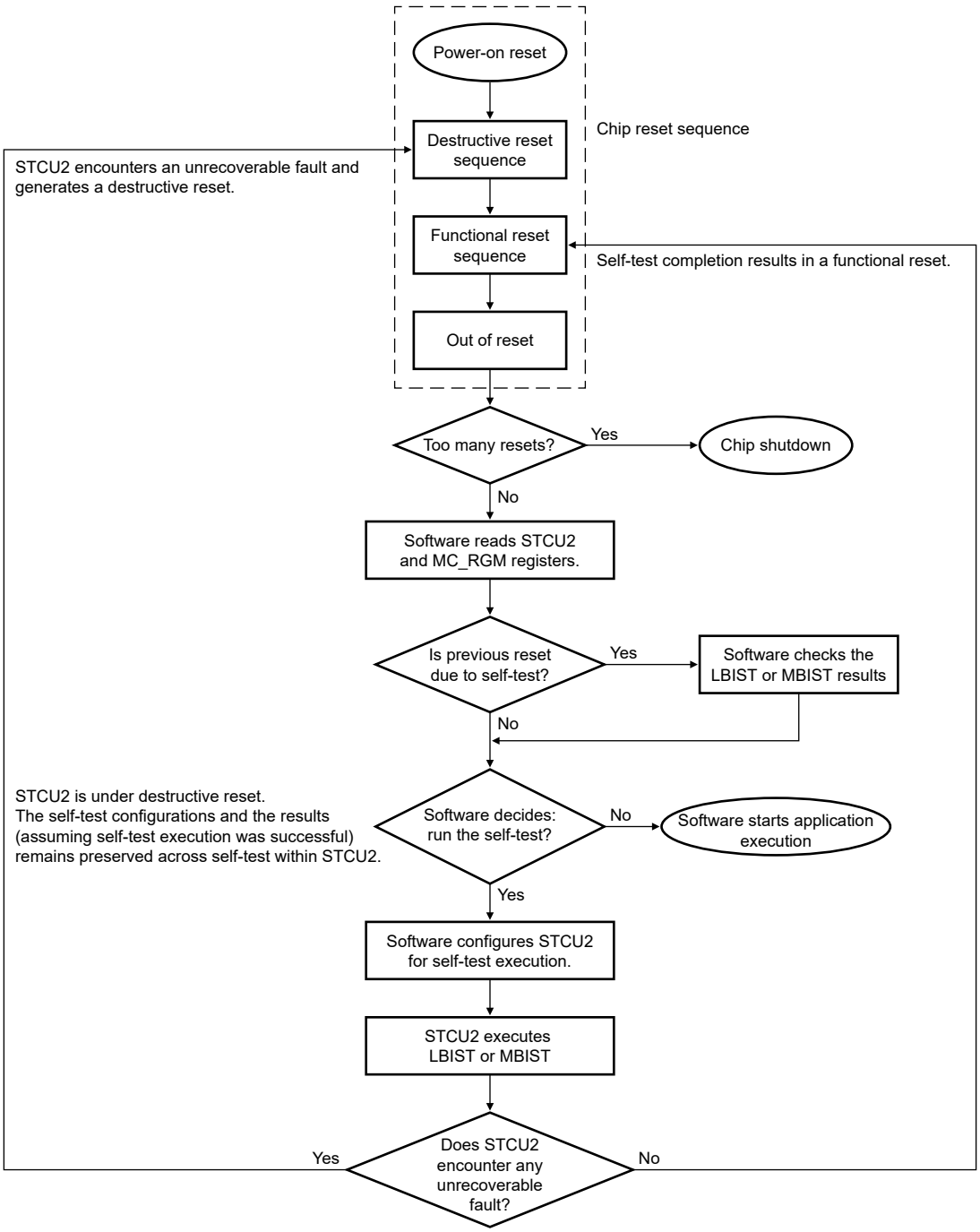


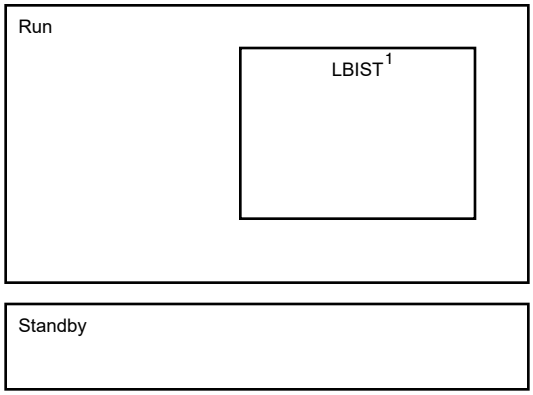
Figure 203. LBIST/MBIST execution

Figure 204 visualizes the top-level partitioning of the chip. The chip consists of two partitions, Run and Standby.

- Run: This partition consists of logic which is present in switchable domain and is shut off (has no supply) while the chip operates in low-power (Standby) mode.
- Standby: This partition consists of always-on logic which is functional even while the chip operates In low-power (Standby) mode.

The Run partition also contains the control logic that is essential for the chip self-test operation, as well as blocks that undergo self-test. The chip consists of a single LBIST partition, which is a subpartition within the Run partition. The LBIST subpartition contains the logic over which self-test is executed. The logic outside the LBIST subpartition and within the Run partition consists of the LBIST control logic as well as logic which does not undergo LBIST.

The Standby partition cannot participate in the LBIST of the Run partition because it is a different partition of the chip. Any module within the Standby partition is therefore excluded from the LBIST. The components within the Standby partition are listed within the section "Chip power domain partitioning" in the "Power Management" chapter.



1. LBIST is not supported on S32K312, S32K311, and S32K310

Figure 204. S32K3xx chip top-level partition view

See "STCU2 LBIST/MBIST mapping" in the STCU chip-specific section for modules participating in the LBIST operation.

NOTE

REG_PROT of an IP undergoes (or does not undergo) LBIST in conjunction with the protected module that undergoes (or does not undergo) LBIST.

The modules that are vital to the self-test operation are excluded from LBIST regions to allow LBIST to execute successfully.

You must run self-test with PLLDIG configured as the system clock. The LBIST clock controller controls the clock during serial shift, but returns clock control to the functional nodes during the self-test.

49.5 Glossary

ASIL	Automotive safety integrity level. This is a risk classification scheme as defined by ISO 26262 for automotive standard.
CCF	Common cause failure
DTCM	Data tightly coupled memory
ECC	Error correction code
EDC	Error detection code
ITCM	Instruction tightly coupled memory
LBIST	Logic built-in self-test
MBIST	Memory built-in self-test

PLL	Phase-locked loop oscillator
PMIC	Power management integrated chip
SECDED	Single error correction, Double error detection
SBC	System basis chip

Chapter 50
Error Injection Module (EIM)

50.1 Chip-specific EIM information

50.1.1 EIM instances

This chip supports up to four instances of EIM:

- EIM_0
- EIM_1
- EIM_2
- EIM_3

Table 271. EIM instances

Instances	S32K388/S32K389	S32K358/S32K356/S32K348/ S32K338/S32K328	S32K322/S32K324/S32K344/S32K342/S32K341/ S32K314/S32K312/S32K311/S32K310
EIM_0	Yes	Yes	Yes
EIM_1	Yes	Yes	No
EIM_2	Yes	Yes	No
EIM_3	Yes	No	No

50.1.2 EIM0 base address

Table 272. EIM0 base address

EIM0 base address	Variants
0x4025_8000	S32K322/S32K324/S32K344/S32K342/S32K341/S32K314/S32K312/S32K311/ S32K310
0x4050_C000	S32K388/S32K389/S32K358/S32K356/S32K348/S32K338/S32K328

50.1.3 EIM channel mapping

EIM integrates with the memory controller and memory array to enable error injection in a controlled way. Each memory controller has its own EIM channel.

Cortex-M7_1, EMAC, AIPS2 gasket, Cortex-M7 lockstep, and QuadSPI gasket are not available in the S32K312 and S32K311 product variants of the S32K3 family.

Table 273. EIM channel mapping - S32K3x1, S32K3x2, S32K344/S32K324/S32K314

Channel #	Target	Data bits [1]	Check bits[1]	# of data bits	# of check bits
0	SRAM0	Word1[31:0] – SRAM0 read data[63:32]	Word0[31:24] – SRAM0 read data ECC[7:0]	64	8
		Word2[31:0] – SRAM0 read data[31:0]			

Table continues on the next page...

Table 273. EIM channel mapping - S32K3x1, S32K3x2, S32K344/S32K324/S32K314 (continued)

Channel #	Target	Data bits [1]	Check bits[1]	# of data bits	# of check bits
1	SRAM1[2]	Word1[31:0] – SRAM1 read data[63:32]	Word0[31:24] SRAM1 read data ECC[7:0]	64	8
		Word2[31:0] – SRAM1 read data[31:0]			
2	DMA TCD	Word1[31:0] – DMA TCD RAM read data[63:32]	Word0[31:24] DMA TCD RAM read data checkbits[7:0]	64	8
		Word2[31:0] – DMA TCD RAM read data[31:0]			
3	Cortex-M7_0 IC tag	Word1[12:0] – Cortex-M7_0 IC tag read data1[28:16]	Word0[31:25] – Cortex-M7_0 IC tag read data1[6:0]	44	14
		Word2[31:22] – Cortex-M7_0 IC tag read data1[15:7]			
		Word2[21:0] – Cortex-M7_0 IC tag read data0[28:7]	Word0[24:18] – Cortex-M7_0 IC tag read data1[6:0]		
4	Cortex-M7_0 IC data	Word1[31:0] – Cortex-M7_0 IC data read data1[71:40]	Word0[31:24] – Cortex-M7_0 IC data data1[7:0]	128	16
		Word2[31:0] – Cortex-M7_0 IC data read data1[39:8]			
		Word3[31:0] – Cortex-M7_0 IC data read data0[71:40]	Word0[23:16] – Cortex-M7_0 IC data read data0[7:0]		
		Word4[31:0] – Cortex-M7_0 IC data read data0[39:8]			
5	Cortex-M7_0 DC tag	Word1[7:0] – Cortex-M7_0 DC tag read data3[32:25]	Word0[31:25] – Cortex-M7_0 DC tag read data3[6:0]	104	28
		Word2[31:14] – Cortex-M7_0 DC tag read data3[24:7]			
		Word2[13:0] – Cortex-M7_0 DC tag read data2[32:19]	Word0[24:18] – Cortex-M7_0 DC tag read data2[6:0]		
		Word3[31:20] – Cortex-M7_0 DC tag read data2[18:7]			
		Word3[19:0] – Cortex-M7_0 DC tag read data1[32:13]	Word0[17:11] – Cortex-M7_0 DC tag read data1[6:0]		
		Word4[31:26] – Cortex-M7_0 DC tag read data1[12:7]			
		Word4[25:0] – Cortex-M7_0 DC tag read data0[32:7]	Word0[10:4] – Cortex-M7_0 DC tag read data0[6:0]		

Table continues on the next page...

Table 273. EIM channel mapping - S32K3x1, S32K3x2, S32K344/S32K324/S32K314 (continued)

Channel #	Target	Data bits [1]	Check bits[1]	# of data bits	# of check bits
6	Cortex-M7_0 DC data0	Word1[31:0] – Cortex-M7_0 DC data0 read data3[38:7]	Word0[31:25] – Cortex-M7_0 DC data0 read data3[6:0]	128	28
		Word2[31:0] – Cortex-M7_0 DC data0 read data2[38:7]	Word0[24:18] – Cortex-M7_0 DC data0 read data2[6:0]		
		Word3[31:0] – Cortex-M7_0 DC data0 read data1[38:7]	Word0[17:11] – Cortex-M7_0 DC data0 read data1[6:0]		
		Word4[31:0] – Cortex-M7_0 DC data0 read data0[38:7]	Word0[10:4] – Cortex-M7_0 DC data0 read data0[6:0]		
7	Cortex-M7_0 DC data1	Word1[31:0] – Cortex-M7_0 DC data1 read data3[38:7]	Word0[31:25] – Cortex-M7_0 DC data1 read data3[6:0]	128	28
		Word2[31:0] – Cortex-M7_0 DC data1 read data2[38:7]	Word0[24:18] – Cortex-M7_0 DC data1 read data2[6:0]		
		Word3[31:0] – Cortex-M7_0 DC data1 read data1[38:7]	Word0[17:11] – Cortex-M7_0 DC data1 read data1[6:0]		
		Word4[31:0] – Cortex-M7_0 DC data1 read data0[38:7]	Word0[10:4] – Cortex-M7_0 DC data1 read data0[6:0]		
8	Cortex-M7_1 IC tag	Word1[12:0] – Cortex-M7_1 IC tag read data1[28:16]	Word0[31:25] – Cortex-M7_1 IC tag read data1[6:0]	44	14
		Word2[31:22] – Cortex-M7_1 IC tag read data1[15:7]			
		Word2[21:0] – Cortex-M7_1 IC tag read data0[28:7]	Word0[24:18] – Cortex-M7_1 IC tag read data0[6:0]		
9	Cortex-M7_1 IC tag	Word1[31:0] – Cortex-M7_1 IC data read data0[71:40]	Word0[31:24] – Cortex-M7_1 IC data read data1[7:0]	128	16
		Word2[31:0] – Cortex-M7_1 IC data read data0[39:8]			
		Word3[31:0] – Cortex-M7_1 IC data read data0[71:40]	Word0[23:16] – Cortex-M7_1 IC data read data0[7:0]		
		Word4[31:0] – Cortex-M7_1 IC data read data0[39:8]			
10	Cortex-M7_1 DC tag	Word1[7:0] – Cortex-M7_1 DC tag read data3[32:25]	Word0[31:25] – Cortex-M7_1 DC tag read data3[6:0]	104	28

Table continues on the next page...

Table 273. EIM channel mapping - S32K3x1, S32K3x2, S32K344/S32K324/S32K314 (continued)

Channel #	Target	Data bits [1]	Check bits[1]	# of data bits	# of check bits
		Word2[31:14] – Cortex-M7_1 DC tag read data3[24:7]			
		Word2[31:0] – Cortex-M7_1 DC tag read data2[32:19]	Word0[24:18] – Cortex-M7_1 DC tag read data2[6:0]		
		Word3[31:20] – Cortex-M7_1 DC tag read data1[18:7]			
		Word3[19:0] – Cortex-M7_1 DC tag read data1[32:13]	Word0[17:11] – Cortex-M7_1 DC tag read data1[6:0]		
		Word4[31:26] – Cortex-M7_1 DC tag read data1[12:7]			
		Word4[25:0] – Cortex-M7_1 DC tag read data0[32:7]	Word0[10:4] – Cortex-M7_1 DC tag read data0[6:0]		
11	Cortex-M7_1 DC data0	Word1[31:0] – Cortex-M7_1 DC data0 read data3[38:7]	Word0[31:25] – Cortex-M7_1 DC data0 read data3[6:0]	128	28
		Word2[31:0] – Cortex-M7_1 DC data0 read data2[38:7]	Word0[24:18] – Cortex-M7_1 DC data0 read data2[6:0]		
		Word3[31:0] – Cortex-M7_1 DC data0 read data1[38:7]	Word0[17:11] – Cortex-M7_1 DC data0 read data1[6:0]		
		Word4[31:0] – Cortex-M7_1 DC data0 read data0[38:7]	Word0[10:4] – Cortex-M7_1 DC data0 read data0[6:0]		
12	Cortex-M7_1 DC data1	Word1[31:0] – Cortex-M7_1 DC data1 read data0[38:7]	Word0[31:25] – Cortex-M7_1 DC data1 read data3[6:0]	128	28
		Word2[31:0] – Cortex-M7_1 DC data1 read data0[38:7]	Word0[24:18] – Cortex-M7_1 DC data1 read data2[6:0]		
		Word3[31:0] – Cortex-M7_1 DC data1 read data0[38:7]	Word0[17:11] – Cortex-M7_1 DC data1 read data1[6:0]		
		Word4[31:0] – Cortex-M7_1 DC data1 read data0[38:7]	Word0[10:4] – Cortex-M7_1 DC data1 read data0[6:0]		
13	Cortex-M7_0 ITCM	Word1[31:0] – Cortex-M7_0 ITCM read data[63:32]	Word0[31:24] – Cortex-M7_0 ITCM read data ECC[7:0]	64	8
		Word2[31:0] – Cortex-M7_0 ITCM read data[31:0]			

Table continues on the next page...

Table 273. EIM channel mapping - S32K3x1, S32K3x2, S32K344/S32K324/S32K314 (continued)

Channel #	Target	Data bits [1]	Check bits[1]	# of data bits	# of check bits
14	Cortex-M7_0 D0TCM	Word1[31:0] – Cortex-M7_0 D0TCM read data[31:0]	Word0[31:24] – Cortex-M7_0 D0TCM read data ECC[7:0]	32	8
15	Cortex-M7_0 D1TCM	Word1[31:0] – Cortex-M7_0 D1TCM read data[31:0]	Word0[31:24] – Cortex-M7_0 D1TCM read data ECC[7:0]	32	8
16	Cortex-M7_1 ITCM	Word1[31:0] – Cortex-M7_1 ITCM read data[63:32]	Word0[31:24] – Cortex-M7_1 ITCM read data ECC[7:0]	64	8
		Word2[31:0] – Cortex-M7_1 ITCM read data[31:0]			
17	Cortex-M7_1 D0TCM	Word1[31:0] – Cortex-M7_1 D0TCM read data[31:0]	Word0[31:24] – Cortex-M7_1 D0TCM read data ECC[7:0]	32	8
18	Cortex-M7_1 D1TCM	Word1[31:0] – Cortex-M7_1 D1TCM read data[31:0]	Word0[31:24] – Cortex-M7_1 D1TCM read data ECC[7:0]	32	8
19	EMAC gasket	Word1[27:0] – EMAC AHB write data[63:36]	—	188	0
		Word2[31:28] – EMAC AHB write data[35:32]			
		Word2[27:0] – EMAC AHB write data[31:4]			
		Word3[31:28] – EMAC AHB write data[3:0]			
		Word3[27:0] – EMAC AHB read data[63:36]			
		Word4[31:28] – EMAC AHB read data[35:32]			
		Word4[27:0] – EMAC AHB read data[31:4]			
		Word5[31:28] – EMAC AHB read data[3:0]			
		Word5[27:0] – EMAC gasket monitor error injection[59:32]			
		Word6[31:0] – EMAC gasket monitor error injection[31:0]			
20	Cortex-M7 TCM gasket	Word1[27:0] – TCM AHB write data[63:36]	—	188	0

Table continues on the next page...

Table 273. EIM channel mapping - S32K3x1, S32K3x2, S32K344/S32K324/S32K314 (continued)

Channel #	Target	Data bits [1]	Check bits[1]	# of data bits	# of check bits
		Word2[31:28] – TCM AHB write data[35:32]			
		Word2[27:0] – TCM AHB write data[31:4]			
		Word3[31:28] – TCM AHB write data[3:0]			
		Word3[27:0] – TCM AHB read data[63:36]			
		Word4[31:28] – TCM AHB read data[35:32]			
		Word4[27:0] – TCM AHB read data[31:4]			
		Word5[31:28] – TCM AHB read data[3:0]			
		Word5[27:0] – TCM gasket monitor error injection[59:32]			
		Word6[31:0] – TCM gasket monitor error injection[31:0]			
21	DMA AXBS S0 gasket	Word1[27:0] – DMA AXBS S0 gasket monitor error injection[59:32]	—	60	0
		Word2[31:0] – DMA AXBS S0 gasket monitor error injection[0:31]			
22	DMA AXBS S1 gasket	Word1[27:0] – DMA AXBS S1 gasket monitor error injection[59:32]	—	60	0
		Word2[31:0] – DMA AXBS S1 gasket monitor error injection[31:0]			
23	HSE gasket	Word1[27:0] – HSE gasket monitor error injection[59:32]	—	60	0
		Word2[31:0] – HSE gasket monitor error injection[31:0]			
24	QuadSPI gasket	Word1[27:0] – QuadSPI gasket monitor error injection[59:32]	—	60	0

Table continues on the next page...

Table 273. EIM channel mapping - S32K3x1, S32K3x2, S32K344/S32K324/S32K314 (continued)

Channel #	Target	Data bits [1]	Check bits[1]	# of data bits	# of check bits
		Word2[31:0] – QuadSPI gasket monitor error injection[31:0]			
25	AIPS1 gasket	Word1[27:0] – AIPS1 gasket monitor error injection[59:32]	—	60	0
		Word2[31:0] – AIPS1 gasket monitor error injection[31:0]			
26	AIPS2 gasket	Word1[27:0] – AIPS2 gasket monitor error injection[59:32]	—	60	0
		Word2[31:0] – AIPS2 gasket monitor error injection[31:0]			
27	Cortex-M7 lockstep	Word1[29:0] – Cortex-M7 error injection[29:0]	—	30	0
28	ECC checking address	Word1[1:0] – Inject error on flash memory controller port 0 address checker	—	24	0
		Word1[3:2] – Inject error on flash memory controller port 1 address checker			
		Word1[4:4] – Inject error on flash memory controller port 2 address checker			
		Word1[7:6] – Inject error on PRAM0 controller address checker			
		Word1[9:8] – Inject error on PRAM1 controller address checker			
		Word1[11:10] – Inject error on 64-bit TCM bus address checker			
		Word1[13:12] – Inject error on QuadSPI path address checker			
		Word1[15:14] – Inject error on AIPS0 address checker			
		Word1[17:16] – Inject error on AIPS1 address checker			

Table continues on the next page...

Table 273. EIM channel mapping - S32K3x1, S32K3x2, S32K344/S32K324/S32K314 (continued)

Channel #	Target	Data bits [1]	Check bits[1]	# of data bits	# of check bits
		Word1[19:18] – Inject error on AIPS2 address checker			
		Word1[21:20] – Inject error on 32-bit TCM Cortex-M7_0 path address checker			
		Word1[23:22] – Inject error on 32-bit TCM Cortex-M7_1 path address checker			
		Word1[25:24] – Inject error on DMA AXBS S0 address parity checker[3]			
		Word1[27:26] – Inject error on DMA AXBS S1 address parity checker[3]			
29	EDC checking wdata	Word1[1:0] – Inject error on PRAM0 controller write data checker	—	18	0
		Word1[3:2] – Inject error on PRAM1 controller write data checker			
		Word1[5:4] – Inject error on 64-bit TCM bus write data checker			
		Word1[7:6] – Reserved			
		Word1[9:8] – Inject error on AIPS0 write data checker			
		Word1[11:10] – Inject error on AIPS1 write data checker			
		Word1[13:12] – Inject error on AIPS2 write data checker			
		Word1[15:14] – Inject error on 32-bit TCM Cortex-M7_0 path write data checker			
		Word1[17:16] – Inject error on 32-bit TCM Cortex-M7_1 path write data checker			
30	EDC checking rdata	Word1[1:0] – Inject error on Cortex-M7_0 AHBM read data checker	—	18	0

Table continues on the next page...

Table 273. EIM channel mapping - S32K3x1, S32K3x2, S32K344/S32K324/S32K314 (continued)

Channel #	Target	Data bits [1]	Check bits[1]	# of data bits	# of check bits
		Word1[3:2] – Inject error on Cortex-M7_0 AHBP read data checker			
		Word1[5:4] – Inject error on DMA read data checker			
		Word1[7:6] – Inject error on STAM read data checker			
		Word1[9:8] – Inject error on HSE read data checker			
		Word1[11:10] – Inject error on EMAC read data checker			
		Word1[13:12] – Inject error on Cortex-M7_1 AHBM read data checker			
		Word1[15:14] – Inject error on Cortex-M7_1 AHBP read data checker			
		Word1[17:16] – Inject error on 32-bit TCM bus path read data checker			

1. You must write to EICHDi_WORDj registers to inject errors in the desired data and check bits. For details, see tables "Error injection channel descriptor: DATA_MASK details" and "DATA_MASK bit: Channel-word mapping" in this chapter.
2. SRAM1 is not available for the S32K342/S32K322/S32K341, S32K312, and S32K311 variants.
3. Applicable for S32K342, S32K341, and S32K322 only.

The two enables, GEIEN and EICHEN*n*, enable the error injection functionality. The former enables it globally and the latter does it for a particular channel. This double-layer enable provides protection against accidental enabling and reconfiguration of the error injection function for each channel.

EIM provides support for inducing single-bit and multi-bit inversions on read data when accessing peripheral RAMs through its data mask registers.

NOTE

For enabling error injection on EDC gaskets (corresponding to channel 28, channel 29, and channel 30), you must also enable the fields corresponding to the required EDC gasket in the MSCM_ENEDC register before enabling the EIM channel.

EIM_EICHD1_CH01, EIM_EICHD1_CH08, EIM_EICHD1_CH09, EIM_EICHD1_CH10, EIM_EICHD1_CH11, EIM_EICHD1_CH12, EIM_EICHD1_CH16, EIM_EICHD1_CH17, EIM_EICHD1_CH18, EIM_EICHD1_CH19, EIM_EICHD1_CH21, EIM_EICHD1_CH22, EIM_EICHD1_CH24, EIM_EICHD1_CH26, EIM_EICHD1_CH27 are not present in S32K312 and S32K311, hence the registers corresponding to these channels are also not present in S32K312 and S32K311.

Table 274. EIM_0 channel mapping - S32K358/S32K356/S32K348/S32K338/S32K328

Channel Number	Target	Data bits	Check bits	Num of data bits	Num of check bits
0	Cortex-M7_0 IC tag	Word1[12:0] – Cortex-M7_0 IC tag read data1[28:16] Word2[31:22] – Cortex-M7_0 IC tag read data1[15:7] Word2[21:0] – Cortex-M7_0 IC tag read data0[28:7]	Word0[31:25] – Cortex- M7_0 IC tag read data1[6:0] Word0[24:18] – Cortex- M7_0 IC tag read data1[6:0]	44	14
1	Cortex-M7_0 IC data	Word1[31:0] – Cortex-M7_0 IC data read data1[71:40] Word2[31:0] – Cortex-M7_0 IC data read data1[39:8] Word3[31:0] – Cortex-M7_0 IC data read data0[71:40] Word4[31:0] – Cortex-M7_0 IC data read data0[39:8]	Word0[31:24] – Cortex- M7_0 IC data data1[7:0] Word0[23:16] – Cortex-M7_0 IC data read data0[7:0]	128	16
2	Cortex-M7_0 DC tag	Word1[7:0] – Cortex-M7_0 DC tag read data3[32:25] Word2[31:14] – Cortex-M7_0 DC tag read data3[24:7] Word2[13:0] – Cortex-M7_0 DC tag read data2[32:19] Word3[31:20] – Cortex-M7_0 DC tag read data2[18:7] Word3[19:0] – Cortex-M7_0 DC tag read data1[32:13] Word4[31:26] – Cortex-M7_0 DC tag read data1[12:7] Word4[25:0] – Cortex-M7_0 DC tag read data0[32:7]	Word0[31:25] – Cortex-M7_0 DC tag read data3[6:0] Word0[24:18] – Cortex-M7_0 DC tag read data2[6:0] Word0[17:11] – Cortex-M7_0 DC tag read data1[6:0]	104	28
3	Cortex-M7_0 DC data0	Word1[31:0] – Cortex-M7_0 DC data0 read data3[38:7] Word2[31:0] – Cortex-M7_0 DC data0 read data2[38:7] Word3[31:0] – Cortex-M7_0 DC data0 read data1[38:7] Word4[31:0] – Cortex-M7_0 DC data0 read data0[38:7]	Word0[31:25] – Cortex-M7_0 DC data0 read data3[6:0] Word0[24:18] – Cortex-M7_0 DC data0 read data2[6:0] Word0[17:11] – Cortex-M7_0 DC data0 read data1[6:0] Word0[10:4] – Cortex-M7_0 DC data0 read data0[6:0]	128	28
4	Cortex-M7_0 DC data1	Word1[31:0] – Cortex-M7_0 DC data1 read data3[38:7]	Word0[31:25] – Cortex-M7_0 DC data1 read data3[6:0]	128	28

Table continues on the next page...

Table 274. EIM_0 channel mapping - S32K358/S32K356/S32K348/S32K338/S32K328 (continued)

Channel Number	Target	Data bits	Check bits	Num of data bits	Num of check bits
		Word2[31:0] – Cortex-M7_0 DC data1 read data2[38:7] Word3[31:0] – Cortex-M7_0 DC data1 read data1[38:7] Word4[31:0] – Cortex-M7_0 DC data1 read data0[38:7]	Word0[24:18] – Cortex-M7_0 DC data1 read data2[6:0] Word0[17:11] – Cortex-M7_0 DC data1 read data1[6:0] Word0[10:4] – Cortex-M7_0 DC data1 read data0[6:0]		
5	Cortex-M7_1 IC tag	Word1[12:0] – Cortex-M7_1 IC tag read data1[28:16] Word2[31:22] – Cortex-M7_1 IC tag read data1[15:7] Word2[21:0] – Cortex-M7_1 IC tag read data0[28:7]	Word0[31:25] – Cortex- M7_1 IC tag read data1[6:0] Word0[24:18] – Cortex- M7_1 IC tag read data1[6:0]	44	14
6	Cortex-M7_1 IC data	Word1[31:0] – Cortex-M7_1 IC data read data1[71:40] Word2[31:0] – Cortex-M7_1 IC data read data1[39:8] Word3[31:0] – Cortex-M7_1 IC data read data0[71:40] Word4[31:0] – Cortex-M7_1 IC data read data0[39:8]	Word0[31:24] – Cortex- M7_1 IC data data1[7:0] Word0[23:16] – Cortex-M7_1 IC data read data0[7:0]	128	16
7	Cortex-M7_1 DC tag	Word1[7:0] – Cortex-M7_1 DC tag read data3[32:25] Word2[31:14] – Cortex-M7_1 DC tag read data3[24:7] Word2[13:0] – Cortex-M7_1 DC tag read data2[32:19] Word3[31:20] – Cortex-M7_1 DC tag read data2[18:7] Word3[19:0] – Cortex-M7_1 DC tag read data1[32:13] Word4[31:26] – Cortex-M7_1 DC tag read data1[12:7] Word4[25:0] – Cortex-M7_1 DC tag read data0[32:7]	Word0[31:25] – Cortex-M7_1 DC tag read data3[6:0] Word0[24:18] – Cortex-M7_1 DC tag read data2[6:0] Word0[17:11] – Cortex-M7_1 DC tag read data1[6:0]	104	28
8	Cortex-M7_1 DC data0	Word1[31:0] – Cortex-M7_1 DC data0 read data3[38:7] Word2[31:0] – Cortex-M7_1 DC data0 read data2[38:7]	Word0[31:25] – Cortex-M7_1 DC data0 read data3[6:0] Word0[24:18] – Cortex-M7_1 DC data0 read data2[6:0]	128	28

Table continues on the next page...

Table 274. EIM_0 channel mapping - S32K358/S32K356/S32K348/S32K338/S32K328 (continued)

Channel Number	Target	Data bits	Check bits	Num of data bits	Num of check bits
		Word3[31:0] – Cortex-M7_1 DC data0 read data1[38:7] Word4[31:0] – Cortex-M7_1 DC data0 read data0[38:7]	Word0[17:11] – Cortex-M7_1 DC data0 read data1[6:0] Word0[10:4] – Cortex-M7_1 DC data0 read data0[6:0]		
9	Cortex-M7_1 DC data1	Word1[31:0] – Cortex-M7_1 DC data1 read data3[38:7] Word2[31:0] – Cortex-M7_1 DC data1 read data2[38:7] Word3[31:0] – Cortex-M7_1 DC data1 read data1[38:7] Word4[31:0] – Cortex-M7_1 DC data1 read data0[38:7]	Word0[31:25] – Cortex-M7_1 DC data1 read data3[6:0] Word0[24:18] – Cortex-M7_1 DC data1 read data2[6:0] Word0[17:11] – Cortex-M7_1 DC data1 read data1[6:0] Word0[10:4] – Cortex-M7_1 DC data1 read data0[6:0]	128	28
10	Cortex-M7_0 ITCM	Word1[31:0] – Cortex-M7_0 ITCM read data[63:32] Word2[31:0] – Cortex-M7_0 ITCM read data[31:0]	Word0[31:24] – Cortex-M7_0 ITCM read data ECC[7:0]	64	8
11	Cortex-M7_0 D0TCM	Word1[31:0] – Cortex-M7_0 D0TCM read data[31:0]	Word0[31:24] – Cortex- M7_0 D0TCM read data ECC[7:0]	32	8
12	Cortex-M7_0 d1tcm	Word1[31:0] – Cortex-M7_0 D1TCM read data[31:0]	Word0[31:24] – Cortex- M7_0 D1TCM read data ECC[7:0]	32	8
13	Cortex-M7_1 ITCM	Word1[31:0] – Cortex-M7_1 ITCM read data[63:32] Word2[31:0] – Cortex-M7_1 ITCM read data[31:0]	Word0[31:24] – Cortex-M7_1 ITCM read data ECC[7:0]	64	8
14	Cortex-M7_1 D0TCM	Word1[31:0] – Cortex-M7_1 D0TCM read data[31:0]	Word0[31:24] – Cortex- M7_1 D0TCM read data ECC[7:0]	32	8
15	Cortex-M7_1 d1tcm	Word1[31:0] – Cortex-M7_1 D1TCM read data[31:0]	Word0[31:24] – Cortex- M7_1 D1TCM read data ECC[7:0]	32	8
16	Cortex-M7 lockstep	Word1[29:0] – Cortex-M7 error injection[29:0]	-	30	0
17-31	Unused				

Table 275. EIM_1 channel mapping - S32K358/S32K356/S32K348/S32K338/S32K328

Channel Number	Target	Data bits	Check bits	Num of data bits	Num of check bits
0	Cortex-M7_2 IC tag	Word1[12:0] – Cortex-M7_2 IC tag read data1[28:16]	Word0[31:25] – Cortex- M7_2 IC tag read data1[6:0]	44	14

Table continues on the next page...

Table 275. EIM_1 channel mapping - S32K358/S32K356/S32K348/S32K338/S32K328 (continued)

Channel Number	Target	Data bits	Check bits	Num of data bits	Num of check bits
		Word2[31:22] – Cortex-M7_2 IC tag read data1[15:7] Word2[21:0] – Cortex-M7_2 IC tag read data0[28:7]	Word0[24:18] – Cortex- M7_2 IC tag read data1[6:0]		
1	Cortex-M7_2 IC data	Word1[31:0] – Cortex-M7_2 IC data read data1[71:40] Word2[31:0] – Cortex-M7_2 IC data read data1[39:8] Word3[31:0] – Cortex-M7_2 IC data read data0[71:40] Word4[31:0] – Cortex-M7_2 IC data read data0[39:8]	Word0[31:24] – Cortex- M7_2 IC data data1[7:0] Word0[23:16] – Cortex-M7_2 IC data read data0[7:0]	128	16
2	Cortex-M7_2 DC tag	Word1[7:0] – Cortex-M7_2 DC tag read data3[32:25] Word2[31:14] – Cortex-M7_3 DC tag read data3[24:7] Word2[13:0] – Cortex-M7_2 DC tag read data2[32:19] Word3[31:20] – Cortex-M7_3 DC tag read data2[18:7] Word3[19:0] – Cortex-M7_2 DC tag read data1[32:13] Word4[31:26] – Cortex-M7_3 DC tag read data1[12:7] Word4[25:0] – Cortex-M7_2 DC tag read data0[32:7]	Word0[31:25] – Cortex-M7_2 DC tag read data3[6:0] Word0[24:18] – Cortex-M7_2 DC tag read data2[6:0] Word0[17:11] – Cortex-M7_2 DC tag read data1[6:0]	104	28
3	Cortex-M7_2 DC data0	Word1[31:0] – Cortex-M7_2 DC data0 read data3[38:7] Word2[31:0] – Cortex-M7_2 DC data0 read data2[38:7] Word3[31:0] – Cortex-M7_2 DC data0 read data1[38:7] Word4[31:0] – Cortex-M7_2 DC data0 read data0[38:7]	Word0[31:25] – Cortex-M7_2 DC data0 read data3[6:0] Word0[24:18] – Cortex-M7_2 DC data0 read data2[6:0] Word0[17:11] – Cortex-M7_2 DC data0 read data1[6:0] Word0[10:4] – Cortex-M7_2 DC data0 read data0[6:0]	128	28
4	Cortex-M7_2 DC data1	Word1[31:0] – Cortex-M7_2 DC data1 read data3[38:7] Word2[31:0] – Cortex-M7_2 DC data1 read data2[38:7]	Word0[31:25] – Cortex-M7_2 DC data1 read data3[6:0] Word0[24:18] – Cortex-M7_2 DC data1 read data2[6:0]	128	28

Table continues on the next page...

Table 275. EIM_1 channel mapping - S32K358/S32K356/S32K348/S32K338/S32K328 (continued)

Channel Number	Target	Data bits	Check bits	Num of data bits	Num of check bits
		Word3[31:0] – Cortex-M7_2 DC data1 read data1[38:7] Word4[31:0] – Cortex-M7_2 DC data1 read data0[38:7]	Word0[17:11] – Cortex-M7_2 DC data1 read data1[6:0] Word0[10:4] – Cortex-M7_2 DC data1 read data0[6:0]		
5-9	Unused				
10	Cortex-M7_2 ITCM	Word1[31:0] – Cortex-M7_2 ITCM read data[63:32] Word2[31:0] – Cortex-M7_2 ITCM read data[31:0]	Word0[31:24] – Cortex-M7_2 ITCM read data ECC[7:0]	64	8
11	Cortex-M7_2 D0TCM	Word1[31:0] – Cortex-M7_2 D0TCM read data[31:0]	Word0[31:24] – Cortex- M7_2 D0TCM read data ECC[7:0]	32	8
12	Cortex-M7_2 d1tcm	Word1[31:0] – Cortex-M7_2 D1TCM read data[31:0]	Word0[31:24] – Cortex- M7_2 D1TCM read data ECC[7:0]	32	8
13-31	Unused				

Table 276. EIM_2 channel mapping - S32K358/S32K356/S32K348/S32K338/S32K328

Channel Number	Target	Data bits	Check bits	Num of data bits	Num of check bits
0	sram0	Word1[31:0] – SRAM0 read data[63:32] Word2[31:0] – SRAM0 read data[31:0]	Word0[31:24] – SRAM0 read data ECC[7:0]	64	8
1	sram1	Word1[31:0] – SRAM1 read data[63:32] Word2[31:0] – SRAM1 read data[31:0]	Word0[31:24] – SRAM1 read data ECC[7:0]	64	8
2	sram2	Word1[31:0] – SRAM2 read data[63:32] Word2[31:0] – SRAM2 read data[31:0]	Word0[31:24] – SRAM1 read data ECC[7:0]	64	8
3	DMA TCD	Word1[31:0] – DMA TCD RAM read data[63:32] Word2[31:0] – DMA TCD RAM read data[31:0]	Word0[31:24] DMA TCD RAM read data checkbits[7:0]	64	8
4	Unused				
5	GMAC gasket	Word1[27:0] – GMAC gasket monitor error injection[59:32] Word2[31:0] – GMAC gasket monitor error injection[31:0]	-	60	0
6	CM7 TCM gasket[1]	Word1[27:0] – TCM AHB write data[63:36] Word2[31:28] – TCM AHB write data[35:32]	-	188	0

Table continues on the next page...

Table 276. EIM_2 channel mapping - S32K358/S32K356/S32K348/S32K338/S32K328 (continued)

Channel Number	Target	Data bits	Check bits	Num of data bits	Num of check bits
		Word2[27:0] – TCM AHB write data[31:4] Word3[31:28] – TCM AHB write data[3:0] Word3[27:0] – TCM AHB read data[63:36] Word4[31:28] – TCM AHB read data[35:32] Word4[27:0] – TCM AHB read data[31:4] Word5[31:28] – TCM AHB read data[3:0] Word5[27:0] – TCM gasket monitor error injection[59:32] Word6[31:0] – TCM gasket monitor error injection[31:0]			
7	DMA AXBS S0 gasket	Word1[27:0] – DMA AXBS S0 gasket monitor error injection[59:32] Word2[31:0] – DMA AXBS S0 gasket monitor error injection[31:0]	-	60	0
8	DMA AXBS S1 gasket	Word1[27:0] – DMA AXBS S1 gasket monitor error injection[59:32] Word2[31:0] – DMA AXBS S1 gasket monitor error injection[31:0]	-	60	0
9	pram0_gasket	Word1[27:0] – PRAM0 gasket monitor error injection[59:32] Word2[31:0] – PRAM0 gasket monitor error injection[31:0]	-	60	0
10	pram1_gasket	Word1[27:0] – PRAM1 gasket monitor error injection[59:32] Word2[31:0] – PRAM1 gasket monitor error injection[31:0]	-	60	0
11	tcm_pram_gasket	Word1[27:0] – TCM_PRAM gasket monitor error injection[59:32] Word2[31:0] – TCM_PRAM gasket monitor error injection[31:0]	-	60	0
12	Cortex-M7_0_ahbs gasket	Word1[27:0] – Cortex-M7_0 AHBS write data[63:36] Word2[31:28] – Cortex-M7_0 AHBS write data[35:32] Word2[27:0] – Cortex-M7_0 AHBS write data[31:4]	-	188	0

Table continues on the next page...

Table 276. EIM_2 channel mapping - S32K358/S32K356/S32K348/S32K338/S32K328 (continued)

Channel Number	Target	Data bits	Check bits	Num of data bits	Num of check bits
		Word3[31:28] – Cortex-M7_0 AHBS write data[3:0] Word3[27:0] – Cortex-M7_0 AHBS read data[63:36] Word4[31:28] – Cortex-M7_0 AHBS read data[35:32] Word4[27:0] – Cortex-M7_0 AHBS read data[31:4] Word5[31:28] – Cortex-M7_0 AHBS read data[3:0] Word5[27:0] – Cortex-M7_0 AHBS gasket monitor error injection[59:32] Word6[31:0] – Cortex-M7_0 AHBS gasket monitor error injection[31:0]			
13	cm7_1_ahbs gasket	Word1[27:0] – Cortex-M7_1 AHBS write data[63:36] Word2[31:28] – Cortex-M7_1 AHBS write data[35:32] Word2[27:0] – Cortex-M7_1 AHBS write data[31:4] Word3[31:28] – Cortex-M7_1 AHBS write data[3:0] Word3[27:0] – Cortex-M7_1 AHBS read data[63:36] Word4[31:28] – Cortex-M7_1 AHBS read data[35:32] Word4[27:0] – Cortex-M7_1 AHBS read data[31:4] Word5[31:28] – Cortex-M7_0 AHBS read data[3:0] Word5[27:0] – Cortex-M7_1 AHBS gasket monitor error injection[59:32] Word6[31:0] – Cortex-M7_1 AHBS gasket monitor error injection[31:0]	-	188	0
14	Cortex-M7_2_ahbs gasket	Word1[27:0] – Cortex-M7_2 AHBS write data[63:36] Word2[31:28] – Cortex-M7_2 AHBS write data[35:32]	-	188	0

Table continues on the next page...

Table 276. EIM_2 channel mapping - S32K358/S32K356/S32K348/S32K338/S32K328 (continued)

Channel Number	Target	Data bits	Check bits	Num of data bits	Num of check bits
		Word2[27:0] – Cortex-M7_2 AHBS write data[31:4] Word3[31:28] – Cortex-M7_2 AHBS write data[3:0] Word3[27:0] – Cortex-M7_2 AHBS read data[63:36] Word4[31:28] – Cortex-M7_2 AHBS read data[35:32] Word4[27:0] – Cortex-M7_2 AHBS read data[31:4] Word5[31:28] – Cortex-M7_2 AHBS read data[3:0] Word5[27:0] – Cortex-M7_2 AHBS gasket monitor error injection[59:32] Word6[31:0] – Cortex-M7_2 AHBS gasket monitor error injection[31:0]			
15	pram2_gasket	Word1[27:0] – PRAM2 gasket monitor error injection[59:32] Word2[31:0] – PRAM2 gasket monitor error injection[31:0]	-	60	0
	Unused				
17	hse gasket	Word1[27:0] – HSE gasket monitor error injection[59:32] Word2[31:0] – HSE gasket monitor error injection[31:0]	-	60	0
18	QuadSPI gasket	Word1[27:0] – QuadSPI gasket monitor error injection[59:32] Word2[31:0] – QuadSPI gasket monitor error injection[31:0]	-	60	0
19	AIPS1	Word1[27:0] – AIPS1 gasket monitor error injection[59:32] Word2[31:0] – AIPS1 gasket monitor error injection[31:0]	-	60	0
20	AIPS2	Word1[27:0] – AIPS2 gasket monitor error injection[59:32] Word2[31:0] – AIPS2 gasket monitor error injection[31:0]	-	60	0

Table continues on the next page...

Table 276. EIM_2 channel mapping - S32K358/S32K356/S32K348/S32K338/S32K328 (continued)

Channel Number	Target	Data bits	Check bits	Num of data bits	Num of check bits
21	AIPS0	Word1[27:0] – AIPS0 gasket monitor error injection[59:32] Word2[31:0] – AIPS0 gasket monitor error injection[31:0]	-	60	0
22	Unused				
23	uSDHC gasket	Word1[27:0] – uSDHC write data[63:36] Word2[31:28] – uSDHC write data[35:32] Word2[27:0] – uSDHC write data[31:4] Word3[31:28] – uSDHC write data[3:0] Word3[27:0] – uSDHC read data[63:36] Word4[31:28] – uSDHC read data[35:32] Word4[27:0] – uSDHC read data[31:4] Word5[31:28] – uSDHC read data[3:0] Word5[27:0] – uSDHC gasket monitor error injection[59:32] Word6[31:0] – Cortex-M7_0 uSDHC gasket monitor error injection[31:0]	-	188	0
24-25	Unused				
25	Unused				
26	edc1 gaskets addr	Word1[1:0] - Inject error on flash controller 3 port address checker Word1[3:2] - Inject error on 32 bit CM7_2 TCM path address checker Word1 [5:4] - Inject error on PRAM2 address checker	-	6	0
27	edc1 gaskets wdata	Word1 [1:0] - Inject error on CM7_2 TCM path write data checker Word1 [3:2] - Inject error on PRAM2 write data checker	-	4	0
28	edc1 gaskets rdata	Word1 [1:0] - Inject error on CM7_2 AHBM read data checker Word1[3:2] - Inject error on CM7_2 AHBP read data checker Word1[5:4] - Inject error on uSDHC read data checker	-	6	0

Table continues on the next page...

Table 276. EIM_2 channel mapping - S32K358/S32K356/S32K348/S32K338/S32K328 (continued)

Channel Number	Target	Data bits	Check bits	Num of data bits	Num of check bits
29	EDC gaskets addr	Word1[1:0] – Inject error on flash memory controller port 0 address checker Word1[3:2] – Inject error on flash memory controller port 1 address checker Word1[4:4] – Inject error on flash memory controller port 2 address checker Word1[7:6] – Inject error on PRAM0 controller address checker Word1[9:8] – Inject error on PRAM1 controller address checker Word1[11:10] – Inject error on 64-bit TCM bus address checker Word1[13:12] – Inject error on QuadSPI path address checker Word1[15:14] – Inject error on AIPS0 address checker Word1[17:16] – Inject error on AIPS1 address checker Word1[19:18] – Inject error on AIPS2 address checker Word1[21:20] – Inject error on 32-bit TCM Cortex-M7_0 path address checker Word1[23:22] – Inject error on 32-bit TCM Cortex-M7_1 path address checker Word1[25:24] – Inject error on DMA AXBS S0 address parity checker Word1[27:26] – Inject error on DMA AXBS S1 address parity checker	-	28	0
30	EDC gaskets wdata	Word1[1:0] – Inject error on PRAM0 controller write data checker Word1[3:2] – Inject error on PRAM1 controller write data checker Word1[5:4] – Inject error on 64-bit TCM bus write data checker Word1[7:6] – Reserved Word1[9:8] – Inject error on AIPS0 write data checker Word1[11:10] – Inject error on AIPS1 write data checker	-	18	0

Table continues on the next page...

Table 276. EIM_2 channel mapping - S32K358/S32K356/S32K348/S32K338/S32K328 (continued)

Channel Number	Target	Data bits	Check bits	Num of data bits	Num of check bits
		Word1[13:12] – Inject error on AIPS2 write data checker Word1[15:14] – Inject error on 32-bit TCM Cortex-M7_0 path write data checker Word1[17:16] – Inject error on 32-bit TCM Cortex-M7_1 path write data checker			
31	EDC gaskets rdata	Word1[1:0] – Inject error on Cortex-M7_0 AHBM read data checker Word1[3:2] – Inject error on Cortex-M7_0 AHBP read data checker Word1[5:4] – Inject error on DMA read data checker Word1[7:6] – Inject error on STAM read data checker Word1[9:8] – Inject error on HSE read data checker Word1[11:10] – Inject error on EMAC read data checker Word1[13:12] – Inject error on Cortex-M7_1 AHBM read data checker Word1[15:14] – Inject error on Cortex-M7_1 AHBP read data checker Word1[17:16] – Inject error on 32-bit TCM bus path read data checker	-	18	0

1. This tcm gasket is not present for S32K358, S32K356, S32K338, S32K388, and S32K389. Hence, the channel 6 for EIM_2 is unused.

Table 277. EIM_0 channel mapping - S32K388/S32K389

Channel Number	Target	Data bits	Check bits	Num of data bits	Num of check bits
0	Cortex-M7_0 IC tag	Word1[12:0] – Cortex-M7_0 IC tag read data1[28:16] Word2[31:22] – Cortex-M7_0 IC tag read data1[15:7] Word2[21:0] – Cortex-M7_0 IC tag read data0[28:7]	Word0[31:25] – Cortex- M7_0 IC tag read data1[6:0] Word0[24:18] – Cortex- M7_0 IC tag read data1[6:0]	44	14

Table continues on the next page...

Table 277. EIM_0 channel mapping - S32K388/S32K389 (continued)

Channel Number	Target	Data bits	Check bits	Num of data bits	Num of check bits
1	Cortex-M7_0 IC data	Word1[31:0] – Cortex-M7_0 IC data read data1[71:40] Word2[31:0] – Cortex-M7_0 IC data read data1[39:8] Word3[31:0] – Cortex-M7_0 IC data read data0[71:40] Word4[31:0] – Cortex-M7_0 IC data read data0[39:8]	Word0[31:24] – Cortex- M7_0 IC data data1[7:0] Word0[23:16] – Cortex-M7_0 IC data read data0[7:0]	128	16
2	Cortex-M7_0 DC tag	Word1[7:0] – Cortex-M7_0 DC tag read data3[32:25] Word2[31:14] – Cortex-M7_0 DC tag read data3[24:7] Word2[13:0] – Cortex-M7_0 DC tag read data2[32:19] Word3[31:20] – Cortex-M7_0 DC tag read data2[18:7] Word3[19:0] – Cortex-M7_0 DC tag read data1[32:13] Word4[31:26] – Cortex-M7_0 DC tag read data1[12:7] Word4[25:0] – Cortex-M7_0 DC tag read data0[32:7]	Word0[31:25] – Cortex-M7_0 DC tag read data3[6:0] Word0[24:18] – Cortex-M7_0 DC tag read data2[6:0] Word0[17:11] – Cortex-M7_0 DC tag read data1[6:0] Word0[10:4] – Cortex-M7_0 DC tag read data0[6:0]	104	28

Table continues on the next page...

Table 277. EIM_0 channel mapping - S32K388/S32K389 (continued)

Channel Number	Target	Data bits	Check bits	Num of data bits	Num of check bits
3	Cortex-M7_0 DC data0	Word1[31:0] – Cortex-M7_0 DC data0 read data3[38:7] Word2[31:0] – Cortex-M7_0 DC data0 read data2[38:7] Word3[31:0] – Cortex-M7_0 DC data0 read data1[38:7] Word4[31:0] – Cortex-M7_0 DC data0 read data0[38:7]	Word0[31:25] – Cortex-M7_0 DC data0 read data3[6:0] Word0[24:18] – Cortex-M7_0 DC data0 read data2[6:0] Word0[17:11] – Cortex-M7_0 DC data0 read data1[6:0] Word0[10:4] – Cortex-M7_0 DC data0 read data0[6:0]	128	28
4	Cortex-M7_0 DC data1	Word1[31:0] – Cortex-M7_0 DC data1 read data3[38:7] Word2[31:0] – Cortex-M7_0 DC data1 read data2[38:7] Word3[31:0] – Cortex-M7_0 DC data1 read data1[38:7] Word4[31:0] – Cortex-M7_0 DC data1 read data0[38:7]	Word0[31:25] – Cortex-M7_0 DC data1 read data3[6:0] Word0[24:18] – Cortex-M7_0 DC data1 read data2[6:0] Word0[17:11] – Cortex-M7_0 DC data1 read data1[6:0] Word0[10:4] – Cortex-M7_0 DC data1 read data0[6:0]	128	28
5	Cortex-M7_1 IC tag	Word1[12:0] – Cortex-M7_1 IC tag read data1[28:16] Word2[31:22] – Cortex-M7_1 IC tag read data1[15:7] Word2[21:0] – Cortex-M7_1 IC tag read data0[28:7]	Word0[31:25] – Cortex- M7_1 IC tag read data1[6:0] Word0[24:18] – Cortex- M7_1 IC tag read data1[6:0]	44	14

Table continues on the next page...

Table 277. EIM_0 channel mapping - S32K388/S32K389 (continued)

Channel Number	Target	Data bits	Check bits	Num of data bits	Num of check bits
6	Cortex-M7_1 IC data	Word1[31:0] – Cortex-M7_1 IC data read data1[71:40] Word2[31:0] – Cortex-M7_1 IC data read data1[39:8] Word3[31:0] – Cortex-M7_1 IC data read data0[71:40] Word4[31:0] – Cortex-M7_1 IC data read data0[39:8]	Word0[31:24] – Cortex- M7_1 IC data data1[7:0] Word0[23:16] – Cortex-M7_1 IC data read data0[7:0]	128	16
7	Cortex-M7_1 DC tag	Word1[7:0] – Cortex-M7_1 DC tag read data3[32:25] Word2[31:14] – Cortex-M7_1 DC tag read data3[24:7] Word2[13:0] – Cortex-M7_1 DC tag read data2[32:19] Word3[31:20] – Cortex-M7_1 DC tag read data2[18:7] Word3[19:0] – Cortex-M7_1 DC tag read data1[32:13] Word4[31:26] – Cortex-M7_1 DC tag read data1[12:7] Word4[25:0] – Cortex-M7_1 DC tag read data0[32:7]	Word0[31:25] – Cortex-M7_1 DC tag read data3[6:0] Word0[24:18] – Cortex-M7_1 DC tag read data2[6:0] Word0[17:11] – Cortex-M7_1 DC tag read data1[6:0] Word0[10:4] – Cortex-M7_1 DC tag read data0[6:0]	104	28

Table continues on the next page...

Table 277. EIM_0 channel mapping - S32K388/S32K389 (continued)

Channel Number	Target	Data bits	Check bits	Num of data bits	Num of check bits
8	Cortex-M7_1 DC data0	Word1[31:0] – Cortex-M7_1 DC data0 read data3[38:7] Word2[31:0] – Cortex-M7_1 DC data0 read data2[38:7] Word3[31:0] – Cortex-M7_1 DC data0 read data1[38:7] Word4[31:0] – Cortex-M7_1 DC data0 read data0[38:7]	Word0[31:25] – Cortex-M7_1 DC data0 read data3[6:0] Word0[24:18] – Cortex-M7_1 DC data0 read data2[6:0] Word0[17:11] – Cortex-M7_1 DC data0 read data1[6:0] Word0[10:4] – Cortex-M7_1 DC data0 read data0[6:0]	128	28
9	Cortex-M7_1 DC data1	Word1[31:0] – Cortex-M7_1 DC data1 read data3[38:7] Word2[31:0] – Cortex-M7_1 DC data1 read data2[38:7] Word3[31:0] – Cortex-M7_1 DC data1 read data1[38:7] Word4[31:0] – Cortex-M7_1 DC data1 read data0[38:7]	Word0[31:25] – Cortex-M7_1 DC data1 read data3[6:0] Word0[24:18] – Cortex-M7_1 DC data1 read data2[6:0] Word0[17:11] – Cortex-M7_1 DC data1 read data1[6:0] Word0[10:4] – Cortex-M7_1 DC data1 read data0[6:0]	128	28
10	Cortex-M7_0 ITCM	Word1[31:0] – Cortex-M7_0 ITCM read data[63:32] Word2[31:0] – Cortex-M7_0 ITCM read data[31:0]	Word0[31:24] – Cortex-M7_0 ITCM read data ECC[7:0]	64	8
11	Cortex-M7_0 D0TCM	Word1[31:0] – Cortex-M7_0 D0TCM read data[31:0]	Word0[31:24] – Cortex- M7_0 D0TCM read data ECC[7:0]	32	8

Table continues on the next page...

Table 277. EIM_0 channel mapping - S32K388/S32K389 (continued)

Channel Number	Target	Data bits	Check bits	Num of data bits	Num of check bits
12	Cortex-M7_0 D1TCM	Word1[31:0] – Cortex-M7_0 D1TCM read data[31:0]	Word0[31:24] – Cortex- M7_0 D1TCM read data ECC[7:0]	32	8
13	Cortex-M7_1 ITCM	Word1[31:0] – Cortex-M7_1 ITCM read data[63:32] Word2[31:0] – Cortex-M7_1 ITCM read data[31:0]	Word0[31:24] – Cortex-M7_1 ITCM read data ECC[7:0]	64	8
14	Cortex-M7_1 D0TCM	Word1[31:0] – Cortex-M7_1 D0TCM read data[31:0]	Word0[31:24] – Cortex- M7_0 D0TCM read data ECC[7:0]	32	8
15	Cortex-M7_1 D1TCM	Word1[31:0] – Cortex-M7_0 D1TCM read data[31:0]	Word0[31:24] – Cortex- M7_0 D1TCM read data ECC[7:0]	32	8
16	Cortex-M7 lockstep	Word1[29:0] – Cortex-M7 error injection[29:0]		30	0

Table 278. EIM_1 channel mapping - S32K388/S32K389

Channel Number	Target	Data bits	Check bits	Num of data bits	Num of check bits
0	Cortex-M7_2 IC tag	Word1[12:0] – Cortex-M7_2 IC tag read data1[28:16] Word2[31:22] – Cortex-M7_2 IC tag read data1[15:7] Word2[21:0] – Cortex-M7_2 IC tag read data0[28:7]	Word0[31:25] – Cortex- M7_2 IC tag read data1[6:0] Word0[24:18] – Cortex- M7_2 IC tag read data1[6:0]	44	14
1	Cortex-M7_2 IC data	Word1[31:0] – Cortex-M7_2 IC data read data1[71:40] Word2[31:0] – Cortex-M7_2 IC data read data1[39:8]	Word0[31:24] – Cortex- M7_2 IC data data1[7:0] Word0[23:16] – Cortex-M7_2 IC data read data0[7:0]	128	16

Table continues on the next page...

Table 278. EIM_1 channel mapping - S32K388/S32K389 (continued)

Channel Number	Target	Data bits	Check bits	Num of data bits	Num of check bits
		Word3[31:0] – Cortex-M7_2 IC data read data0[71:40] Word4[31:0] – Cortex-M7_2 IC data read data0[39:8]			
2	Cortex-M7_2 DC tag	Word1[7:0] – Cortex-M7_2 DC tag read data3[32:25] Word2[31:14] – Cortex-M7_2 DC tag read data3[24:7] Word2[13:0] – Cortex-M7_2 DC tag read data2[32:19] Word3[31:20] – Cortex-M7_2 DC tag read data2[18:7] Word3[19:0] – Cortex-M7_2 DC tag read data1[32:13] Word4[31:26] – Cortex-M7_2 DC tag read data1[12:7] Word4[25:0] – Cortex-M7_2 DC tag read data0[32:7]	Word0[31:25] – Cortex-M7_2 DC tag read data3[6:0] Word0[24:18] – Cortex-M7_2 DC tag read data2[6:0] Word0[17:11] – Cortex-M7_2 DC tag read data1[6:0] Word0[10:4] – Cortex-M7_2 DC tag read data0[6:0]	104	28
3	Cortex-M7_2 DC data0	Word1[31:0] – Cortex-M7_2 DC data0 read data3[38:7] Word2[31:0] – Cortex-M7_2 DC data0 read data2[38:7]	Word0[31:25] – Cortex-M7_2 DC data0 read data3[6:0] Word0[24:18] – Cortex-M7_2 DC data0 read data2[6:0]	128	28

Table continues on the next page...

Table 278. EIM_1 channel mapping - S32K388/S32K389 (continued)

Channel Number	Target	Data bits	Check bits	Num of data bits	Num of check bits
		Word3[31:0] – Cortex-M7_2 DC data0 read data1[38:7] Word4[31:0] – Cortex-M7_2 DC data0 read data0[38:7]	Word0[17:11] – Cortex-M7_2 DC data0 read data1[6:0] Word0[10:4] – Cortex-M7_2 DC data0 read data0[6:0]		
4	Cortex-M7_2 DC data1	Word1[31:0] – Cortex-M7_2 DC data1 read data3[38:7] Word2[31:0] – Cortex-M7_2 DC data1 read data2[38:7] Word3[31:0] – Cortex-M7_2 DC data1 read data1[38:7] Word4[31:0] – Cortex-M7_2 DC data1 read data0[38:7]	Word0[31:25] – Cortex-M7_2 DC data1 read data3[6:0] Word0[24:18] – Cortex-M7_2 DC data1 read data2[6:0] Word0[17:11] – Cortex-M7_2 DC data1 read data1[6:0] Word0[10:4] – Cortex-M7_2 DC data1 read data0[6:0]	128	28
5	Cortex-M7_3 IC tag	Word1[12:0] – Cortex-M7_3 IC tag read data1[28:16] Word2[31:22] – Cortex-M7_3 IC tag read data1[15:7] Word2[21:0] – Cortex-M7_3 IC tag read data0[28:7]	Word0[31:25] – Cortex- M7_3 IC tag read data1[6:0] Word0[24:18] – Cortex- M7_3 IC tag read data1[6:0]	44	14
6	Cortex-M7_3 IC data	Word1[31:0] – Cortex-M7_3 IC data read data1[71:40] Word2[31:0] – Cortex-M7_3 IC data read data1[39:8] Word3[31:0] – Cortex-M7_3	Word0[31:24] – Cortex- M7_3 IC data data1[7:0] Word0[23:16] – Cortex-M7_3 IC data read data0[7:0]	128	16

Table continues on the next page...

Table 278. EIM_1 channel mapping - S32K388/S32K389 (continued)

Channel Number	Target	Data bits	Check bits	Num of data bits	Num of check bits
		IC data read data0[71:40] Word4[31:0] – Cortex-M7_3 IC data read data0[39:8]			
7	Cortex-M7_3 DC tag	Word1[7:0] – Cortex-M7_3 DC tag read data3[32:25] Word2[31:14] – Cortex-M7_3 DC tag read data3[24:7] Word2[13:0] – Cortex-M7_3 DC tag read data2[32:19] Word3[31:20] – Cortex-M7_3 DC tag read data2[18:7] Word3[19:0] – Cortex-M7_3 DC tag read data1[32:13] Word4[31:26] – Cortex-M7_3 DC tag read data1[12:7] Word4[25:0] – Cortex-M7_3 DC tag read data0[32:7]	Word0[31:25] – Cortex-M7_3 DC tag read data3[6:0] Word0[24:18] – Cortex-M7_3 DC tag read data2[6:0] Word0[17:11] – Cortex-M7_3 DC tag read data1[6:0] Word0[10:4] – Cortex-M7_3 DC tag read data0[6:0]	104	28
8	Cortex-M7_3 DC data0	Word1[31:0] – Cortex-M7_3 DC data0 read data3[38:7] Word2[31:0] – Cortex-M7_3 DC data0 read data2[38:7]	Word0[31:25] – Cortex-M7_3 DC data0 read data3[6:0] Word0[24:18] – Cortex-M7_3 DC data0 read data2[6:0]	128	28

Table continues on the next page...

Table 278. EIM_1 channel mapping - S32K388/S32K389 (continued)

Channel Number	Target	Data bits	Check bits	Num of data bits	Num of check bits
		Word3[31:0] – Cortex-M7_3 DC data0 read data1[38:7] Word4[31:0] – Cortex-M7_3 DC data0 read data0[38:7]	Word0[17:11] – Cortex-M7_3 DC data0 read data1[6:0] Word0[10:4] – Cortex-M7_3 DC data0 read data0[6:0]		
9	Cortex-M7_3 DC data1	Word1[31:0] – Cortex-M7_3 DC data1 read data3[38:7] Word2[31:0] – Cortex-M7_3 DC data1 read data2[38:7] Word3[31:0] – Cortex-M7_3 DC data1 read data1[38:7] Word4[31:0] – Cortex-M7_3 DC data1 read data0[38:7]	Word0[31:25] – Cortex-M7_3 DC data1 read data3[6:0] Word0[24:18] – Cortex-M7_3 DC data1 read data2[6:0] Word0[17:11] – Cortex-M7_3 DC data1 read data1[6:0] Word0[10:4] – Cortex-M7_3 DC data1 read data0[6:0]	128	28
10	Cortex-M7_2 ITCM	Word1[31:0] – Cortex-M7_2 ITCM read data[63:32] Word2[31:0] – Cortex-M7_2 ITCM read data[31:0]	Word0[31:24] – Cortex-M7_1 ITCM read data ECC[7:0]	64	8
11	Cortex-M7_2 D0TCM	Word1[31:0] – Cortex-M7_2 D0TCM read data[31:0]	Word0[31:24] – Cortex- M7_2 D0TCM read data ECC[7:0]	32	8
12	Cortex-M7_2 D1TCM	Word1[31:0] – Cortex-M7_2 D1TCM read data[31:0]	Word0[31:24] – Cortex- M7_2 D1TCM read data ECC[7:0]	32	8
13	Cortex-M7_3 ITCM	Word1[31:0] – Cortex-M7_3 ITCM read data[63:32]	Word0[31:24] – Cortex-M7_3 ITCM read data ECC[7:0]	64	8

Table continues on the next page...

Table 278. EIM_1 channel mapping - S32K388/S32K389 (continued)

Channel Number	Target	Data bits	Check bits	Num of data bits	Num of check bits
		Word2[31:0] – Cortex-M7_3 ITCM read data[31:0]			
14	Cortex-M7_3 D0TCM	Word1[31:0] – Cortex-M7_3 D0TCM read data[31:0]	Word0[31:24] – Cortex- M7_3 D0TCM read data ECC[7:0]	32	8
15	Cortex-M7_3 D1TCM	Word1[31:0] – Cortex-M7_0 D1TCM read data[31:0]	Word0[31:24] – Cortex- M7_3 D1TCM read data ECC[7:0]	32	8
16	Cortex-M7 PLS_lockstep	Word1[29:0] – Cortex-M7 PLS_error injection[29:0]		30	0
17-31	Unused				

Table 279. EIM_2 channel mapping - S32K388/S32K389

Channel Number	Target	Data bits	Check bits	Num of data bits	Num of check bits
0	SRAM0	Word1[31:0] – SRAM0 read data[63:32] Word2[31:0] – SRAM0 read data[31:0]	Word0[31:24] – SRAM0 read data ECC[7:0]	64	8
1	SRAM1	Word1[31:0] – SRAM1 read data[63:32] Word2[31:0] – SRAM1 read data[31:0]	Word0[31:24] – SRAM1 read data ECC[7:0]	64	8
2	SRAM2	Word1[31:0] – SRAM2 read data[63:32] Word2[31:0] – SRAM2 read data[31:0]	Word0[31:24] – SRAM2 read data ECC[7:0]	64	8
3	DMA TCD	Word1[31:0] – DMA TCD RAM read data[63:32]	Word0[31:24] DMA TCD RAM read data checkbits[7:0]	64	8

Table continues on the next page...

Table 279. EIM_2 channel mapping - S32K388/S32K389 (continued)

Channel Number	Target	Data bits	Check bits	Num of data bits	Num of check bits
		Word2[31:0] – DMA TCD RAM read data[31:0]			
4	GMAC_1 gasket	Word1[27:0] – GMAC_1 gasket monitor error injection[59:32] Word2[31:0] – GMAC_1 gasket monitor error injection[0:31]		60	0
5	GMAC gasket	Word1[27:0] – GMAC gasket monitor error injection[59:32] Word2[31:0] – GMAC gasket monitor error injection[0:31]		60	0
6	Unused				
7	DMA AXBS S0 gasket	Word1[27:0] – DMA AXBS S0 gasket monitor error injection[59:32] Word2[31:0] – DMA AXBS S0 gasket monitor error injection[0:31]		60	0
8	DMA AXBS S1 gasket	Word1[27:0] – DMA AXBS S1 gasket monitor error injection[59:32] Word2[31:0] – DMA AXBS S1 gasket monitor error injection[0:31]		60	0
9	Cortex-M7_3 AHBP	Word1[27:0] – Cortex-M7_3 AHBP gasket monitor error injection[59:32]		60	0

Table continues on the next page...

Table 279. EIM_2 channel mapping - S32K388/S32K389 (continued)

Channel Number	Target	Data bits	Check bits	Num of data bits	Num of check bits
		Word2[31:0] – Cortex-M7_3 AHBP gasket monitor error injection[0:31]			
10	Cortex-M7_3 AHBM	Word1[27:0] – Cortex-M7_3 AHBM gasket monitor error injection[59:32] Word2[31:0] – Cortex-M7_3 AHBM gasket monitor error injection[0:31]		60	0
11	Cortex-M7_0 AHBP	Word1[27:0] – Cortex-M7_0 AHBP gasket monitor error injection[59:32] Word2[31:0] – Cortex-M7_0 AHBP gasket monitor error injection[0:31]		60	0
12	Cortex-M7_0 AHBM	Word1[27:0] – Cortex-M7_0 AHBM gasket monitor error injection[59:32] Word2[31:0] – Cortex-M7_0 AHBM gasket monitor error injection[0:31]		60	0
13	Cortex-M7_1 AHBP	Word1[27:0] – Cortex-M7_1 AHBP gasket monitor error injection[59:32] Word2[31:0] – Cortex-M7_1 AHBP		60	0

Table continues on the next page...

Table 279. EIM_2 channel mapping - S32K388/S32K389 (continued)

Channel Number	Target	Data bits	Check bits	Num of data bits	Num of check bits
		gasket monitor error injection[0:31]			
14	Cortex-M7_1 AHBM	Word1[27:0] – Cortex-M7_1 AHBM gasket monitor error injection[59:32] Word2[31:0] – Cortex-M7_1 AHBM gasket monitor error injection[0:31]		60	0
15	Cortex-M7_2 AHBP	Word1[27:0] – Cortex-M7_2 AHBP gasket monitor error injection[59:32] Word2[31:0] – Cortex-M7_2 AHBP gasket monitor error injection[0:31]		60	0
16	Cortex-M7_2 AHBM	Word1[27:0] – Cortex-M7_2 AHBM gasket monitor error injection[59:32] Word2[31:0] – Cortex-M7_2 AHBM gasket monitor error injection[0:31]		60	0
17	HSE gasket	Word1[27:0] – HSE gasket monitor error injection[59:32] Word2[31:0] – HSE gasket monitor error injection[0:31]		60	0
18	QuadSPI gasket	Word1[27:0] – QuadSPI gasket monitor error injection[59:32]		60	0

Table continues on the next page...

Table 279. EIM_2 channel mapping - S32K388/S32K389 (continued)

Channel Number	Target	Data bits	Check bits	Num of data bits	Num of check bits
		Word2[31:0] – QuadSPI gasket monitor error injection[0:31]			
19	AIPS1	Word1[27:0] – AIPS1 gasket monitor error injection[59:32] Word2[31:0] – AIPS1 gasket monitor error injection[0:31]		60	0
20	AIPS2	Word1[27:0] – AIPS2 gasket monitor error injection[59:32] Word2[31:0] – AIPS2 gasket monitor error injection[0:31]		60	0
21	AIPS0	Word1[27:0] – AIPS0 gasket monitor error injection[59:32] Word2[31:0] – AIPS0 gasket monitor error injection[0:31]		60	0
22	Cortex-M7_0_ahbs gasket	Word1[27:0] – Cortex-M7_0 AHBS write data[63:36] Word2[31:28] – Cortex-M7_0 AHBS write data[35:32] Word2[27:0] – Cortex-M7_0 AHBS write data[31:4] Word3[31:28] – Cortex-		188	0

Table continues on the next page...

Table 279. EIM_2 channel mapping - S32K388/S32K389 (continued)

Channel Number	Target	Data bits	Check bits	Num of data bits	Num of check bits
		M7_0 AHBS write data[3:0] Word3[27:0] – Cortex-M7_0 AHBS read data[63:36] Word4[31:28] – Cortex-M7_0 AHBS read data[35:32] Word4[27:0] – Cortex-M7_0 AHBS read data[31:4] Word5[31:28] – Cortex-M7_0 AHBS read data[3:0] Word5[27:0] – Cortex-M7_0 AHBS gasket monitor error injection[59:32] Word6[31:0] – Cortex-M7_0 AHBS gasket monitor error injection[31:0]			
23	Cortex-M7_1_ahbs gasket	Word1[27:0] – Cortex-M7_1 AHBS write data[63:36] Word2[31:28] – Cortex-M7_1 AHBS write data[35:32] Word2[27:0] – Cortex-M7_1 AHBS write data[31:4] Word3[31:28] – Cortex-M7_1 AHBS write data[3:0]		188	0

Table continues on the next page...

Table 279. EIM_2 channel mapping - S32K388/S32K389 (continued)

Channel Number	Target	Data bits	Check bits	Num of data bits	Num of check bits
		Word3[27:0] – Cortex-M7_1 AHBS read data[63:36] Word4[31:28] – Cortex-M7_1 AHBS read data[35:32] Word4[27:0] – Cortex-M7_1 AHBS read data[31:4] Word5[31:28] – Cortex-M7_1 AHBS read data[3:0] Word5[27:0] – Cortex-M7_1 AHBS gasket monitor error injection[59:32] Word6[31:0] – Cortex-M7_1 AHBS gasket monitor error injection[31:0]			
24	Cortex-M7_2_ahbs gasket	Word1[27:0] – Cortex-M7_2 AHBS write data[63:36] Word2[31:28] – Cortex-M7_2 AHBS write data[35:32] Word2[27:0] – Cortex-M7_2 AHBS write data[31:4] Word3[31:28] – Cortex-M7_2 AHBS write data[3:0] Word3[27:0] – Cortex-		188	0

Table continues on the next page...

Table 279. EIM_2 channel mapping - S32K388/S32K389 (continued)

Channel Number	Target	Data bits	Check bits	Num of data bits	Num of check bits
		M7_2 AHBS read data[63:36] Word4[31:28] – Cortex-M7_2 AHBS read data[35:32] Word4[27:0] – Cortex-M7_2 AHBS read data[31:4] Word5[31:28] – Cortex-M7_2 AHBS read data[3:0] Word5[27:0] – Cortex-M7_2 AHBS gasket monitor error injection[59:32] Word6[31:0] – Cortex-M7_2 AHBS gasket monitor error injection[31:0]			
25	Cortex-M7_3_ahbs gasket	Word1[27:0] – Cortex-M7_3 AHBS write data[63:36] Word2[31:28] – Cortex-M7_3 AHBS write data[35:32] Word2[27:0] – Cortex-M7_3 AHBS write data[31:4] Word3[31:28] – Cortex-M7_3 AHBS write data[3:0] Word3[27:0] – Cortex-M7_3 AHBS read data[63:36]		188	0

Table continues on the next page...

Table 279. EIM_2 channel mapping - S32K388/S32K389 (continued)

Channel Number	Target	Data bits	Check bits	Num of data bits	Num of check bits
		Word4[31:28] – Cortex-M7_3 AHBS read data[35:32] Word4[27:0] – Cortex-M7_3 AHBS read data[31:4] Word5[31:28] – Cortex-M7_3 AHBS read data[3:0] Word5[27:0] – Cortex-M7_3 AHBS gasket monitor error injection[59:32] Word6[31:0] – Cortex-M7_3 AHBS gasket monitor error injection[31:0]			
26	edc1 gaskets addr	Word1 [1:0] -- Inject error on flash controller 3 port address checker Word1 [3:2] -- Inject error on 32 bit CM7_2 TCM path address checker Word1 [5:4] -- Inject error on PRAM2 address checker Word1 [7:6] Inject error on 32 bit CM7_3 TCM path address checker		8	0
27	edc1 gaskets wdata	Word1 [1:0] -- Inject error on CM7_2 TCM path write data checker Word1 [3:2] -- Inject error on PRAM2 write data checker		6	0

Table continues on the next page...

Table 279. EIM_2 channel mapping - S32K388/S32K389 (continued)

Channel Number	Target	Data bits	Check bits	Num of data bits	Num of check bits
		Word1 [5:4] Inject error on CM7_3 TCM path write data checker			
28	edc1 gaskets rdata	Word1[1:0] -- Inject error on CM7_2 AHBM read data checker Word1[3:2] -- Inject error on CM7_2 AHBP read data checker Word1[5:4] -- Inject error on ENET1 read data checker Word1[7:6] Inject error on CM7_3 AHBM read data checker Word1[9:8] Inject error on CM7_3 AHBP read data checker Word1[11:10] Inject error on ace mstr result read data checker Word1[13:12] Inject error on ace master feed read data checker		14	0
29	EDC gaskets addr	Word1[1:0] – Inject error on flash memory controller port 0 address checker Word1[3:2] – Inject error on flash memory controller port 1 address checker Word1[4:4] – Inject error on flash memory		28	0

Table continues on the next page...

Table 279. EIM_2 channel mapping - S32K388/S32K389 (continued)

Channel Number	Target	Data bits	Check bits	Num of data bits	Num of check bits
		controller port 2 address checker Word1[7:6] – Inject error on PRAM0 controller address checker Word1[9:8] – Inject error on PRAM1 controller address checker Word1[11:10] – Inject error on 64-bit (TCM bus address checker for S32K388/ PRAM_3 bus address checker for S32K389) Word1[13:12] – Inject error on QuadSPI path address checker Word1[15:14] – Inject error on AIPS0 address checker Word1[17:16] – Inject error on AIPS1 address checker Word1[19:18] – Inject error on AIPS2 address checker Word1[21:20] – Inject error on 32-bit TCM Cortex-M7_0 path address checker Word1[23:22] – Inject error on 32-bit TCM Cortex-M7_1 path address checker			

Table continues on the next page...

Table 279. EIM_2 channel mapping - S32K388/S32K389 (continued)

Channel Number	Target	Data bits	Check bits	Num of data bits	Num of check bits
		Word1[25:24] – Inject error on DMA AXBS S0 address parity checker Word1[27:26] – Inject error on DMA AXBS S1 address parity checker			
30	EDC gaskets wdata	Word1[1:0] – Inject error on PRAM0 controller write data checker Word1[3:2] – Inject error on PRAM1 controller write data checker Word1[5:4] – Inject error on 64-bit (TCM bus write data checker for S32K388/PRAM_3 bus write data checker for S32K389) Word1[7:6] – Inject error on QSPI write data checker Word1[9:8] – Inject error on AIPS0 write data checker Word1[11:10] – Inject error on AIPS1 write data checker Word1[13:12] – Inject error on AIPS2 write data checker Word1[15:14] – Inject error on 32-bit TCM Cortex-M7_0 path write data checker		18	0

Table continues on the next page...

Table 279. EIM_2 channel mapping - S32K388/S32K389 (continued)

Channel Number	Target	Data bits	Check bits	Num of data bits	Num of check bits
		Word1[17:16] – Inject error on 32-bit TCM Cortex-M7_1 path write data checker			
31	EDC gaskets rdata	Word1[1:0] – Inject error on Cortex-M7_0 AHBM read data checker Word1[3:2] – Inject error on Cortex-M7_0 AHBP read data checker Word1[5:4] – Inject error on DMA read data checker Word1[7:6] – Inject error on STAM read data checker Word1[9:8] – Inject error on HSE read data checker Word1[11:10] – Inject error on EMAC read data checker Word1[13:12] – Inject error on Cortex-M7_1 AHBM read data checker Word1[15:14] – Inject error on Cortex-M7_1 AHBP read data checker		16	0

Table 280. EIM_3 channel mapping - S32K388

Channel Number	Target	Data bits	Check bits	Num of data bits	Num of check bits
0	ACE gasket slave	Word1[27:0] – ACE gasket slave monitor error injection[59:32]		60	0

Table continues on the next page...

Table 280. EIM_3 channel mapping - S32K388 (continued)

Channel Number	Target	Data bits	Check bits	Num of data bits	Num of check bits
		Word2[31:0] – ACE gasket slave monitor error injection[0:31]			
1	ACE gasket master	Word1[27:0] – ACE gasket master monitor error injection[59:32] Word2[31:0] – ACE gasket master monitor error injection[0:31]		60	0
2	ACE DMA 32 channel	Word1[31:0] – ACE FEED_DMA read data[63:32] Word2[31:0] – ACE FEED_DMA read data[31:0]	Word0[31:24] – ACE FEED_DMA read data ECC[7:0]	64	8
3	ACE DMA 24 channel	Word1[31:0] – ACE RESULT_DMA read data[63:32] Word2[31:0] – ACE RESULT_DMA read data[31:0]	Word0[31:24] – ACE RESULT_DMA read data ECC[7:0]	64	8
4	TCM gasket	Word1[27:0] – TCM gasket monitor error injection[59:32] Word2[31:0] – TCM gasket monitor error injection[0:31]		60	0
5-31	Unused				

Table 281. EIM_3 channel mapping - S32K389

Channel Number	Target	Data bits	Check bits	Num of data bits	Num of check bits
0	ACE gasket slave	Word1[27:0] – ACE gasket slave monitor error injection[59:32] Word2[31:0] – ACE gasket slave monitor error injection[0:31]		60	0
1	ACE gasket master	Word1[27:0] – ACE gasket master monitor error injection[59:32] Word2[31:0] – ACE gasket master monitor error injection[0:31]		60	0
2	ACE DMA 32 channel	Word1[31:0] – ACE FEED_DMA read data[63:32] Word2[31:0] – ACE FEED_DMA read data[31:0]	Word0[31:24] – ACE FEED_DMA read data ECC[7:0]	64	8

Table continues on the next page...

Table 281. EIM_3 channel mapping - S32K389 (continued)

Channel Number	Target	Data bits	Check bits	Num of data bits	Num of check bits
3	ACE DMA 24 channel	Word1[31:0] – ACE RESULT_DMA read data[63:32] Word2[31:0] – ACE RESULT_DMA read data[31:0]	Word0[31:24] – ACE RESULT_DMA read data ECC[7:0]	64	8
4	TCM gasket	Word1[27:0] – TCM gasket monitor error injection[59:32] Word2[31:0] – TCM gasket monitor error injection[0:31]		60	0
5	Reserved	This channel is reserved but it uses 60 bits of error injection bus		60	0
6	SRAM3	Word1[31:0] – SRAM3 read data[63:32] Word2[31:0] – SRAM3 read data[31:0]	Word0[31:24] – SRAM3 read data ECC[7:0]	64	8
7	PFC0_0	Word1[31:0] – PFC0_0 error injection[31:0]		32	0
8	PFC0_1	Word1[31:0] – PFC0_1 error injection[31:0]		32	0
9	PFC1_0	Word1[31:0] – PFC1_0 error injection[31:0]		32	0
10	PFC1_1	Word1[31:0] – PFC1_1 error injection[31:0]		32	0
11-31	Unused				

50.1.4 Behavior of EIM error injection on gaskets

The error injection on gaskets don't impact the actual data flow. The gasket read data, write data and the monitor error don't get changed when EIM error injection is done. The gasket compares the actual data with the modified data (with error injection) and flags the gasket alarm. In case of single-bit error also, the alarm is flagged.

NOTE

The channels which depict such behavior are the gasket channels. See section "EIM channel mapping" in this chapter for details.

50.1.5 Constraints to use EIM on CM7 cache memories

The Cortex-M7 processor has Error Correcting Code (ECC) functionality for error detection and correction that is included in the data and instruction caches when implemented. If hard, or permanent errors occur on the cache RAM, the clean, invalidate and retry scheme might cause a deadlock, and the access is continuously replayed. To prevent this, error bank registers are provided to mask the faulty locations as unusable and invalid. To ensure such a deadlock scenario does not occur while testing the cache memory using EIM module, you must take care of the following points:

- Inject one single-bit error on a cache line at a time.
- Ensure that at least one error bank register of the tested cache is free before injecting any faults. This is to make sure that the fault recovery mechanism of the Cortex-M7 core works.

See ‘Cache RAM protection’ section in the Arm Cortex-M7 Processor Technical Reference Manual for more details.

NOTE

- For S32K338, S32K348, S32K328, S32K358, S32K356, S32K388, and S32K389: Error injection on 0th and 1st bit for Icache and Dcache is invalid since the Icache and Dcache size is 16 KB.
- For remaining S32K3xx variant: Error injection on 0th bit for Icache and Dcache is invalid since the Icache and Dcache size is 8 KB.

50.2 Overview

The Error Injection Module (EIM) is mainly used for diagnostic purposes. It provides a method to test the diagnostics (memory ECC, interconnect parity) by error injection in the field. See the chip-specific EIM information to determine which functional safety features are supported by this method.

EIM enables you to inject artificial errors on error-checking mechanisms of a system, such as ECC for RAM read data and parity bits. For each such mechanism that EIM supports on the chip, EIM can inject single-bit and multi-bit inversions on data in the applicable target bus. Injecting faults on memory accesses can be used to exercise the SEC-DED ECC function of the related system.

NOTE

Terminology in this chapter has been updated as follows:

Table 282. Updated terms

Updated term	Deprecated term
Controller	Master

50.2.1 Features

The EIM includes these features:

- Supports upto 32 error injection channel. See the chip-specific EIM information for channel assignment details.
- Protection against accidental enable and reconfiguration error injection function via two-stage enable mechanism

50.2.2 Block diagram

The following diagram shows an example of EIM implementation with a 64-bit read data bus and an 8-bit checkbit bus.

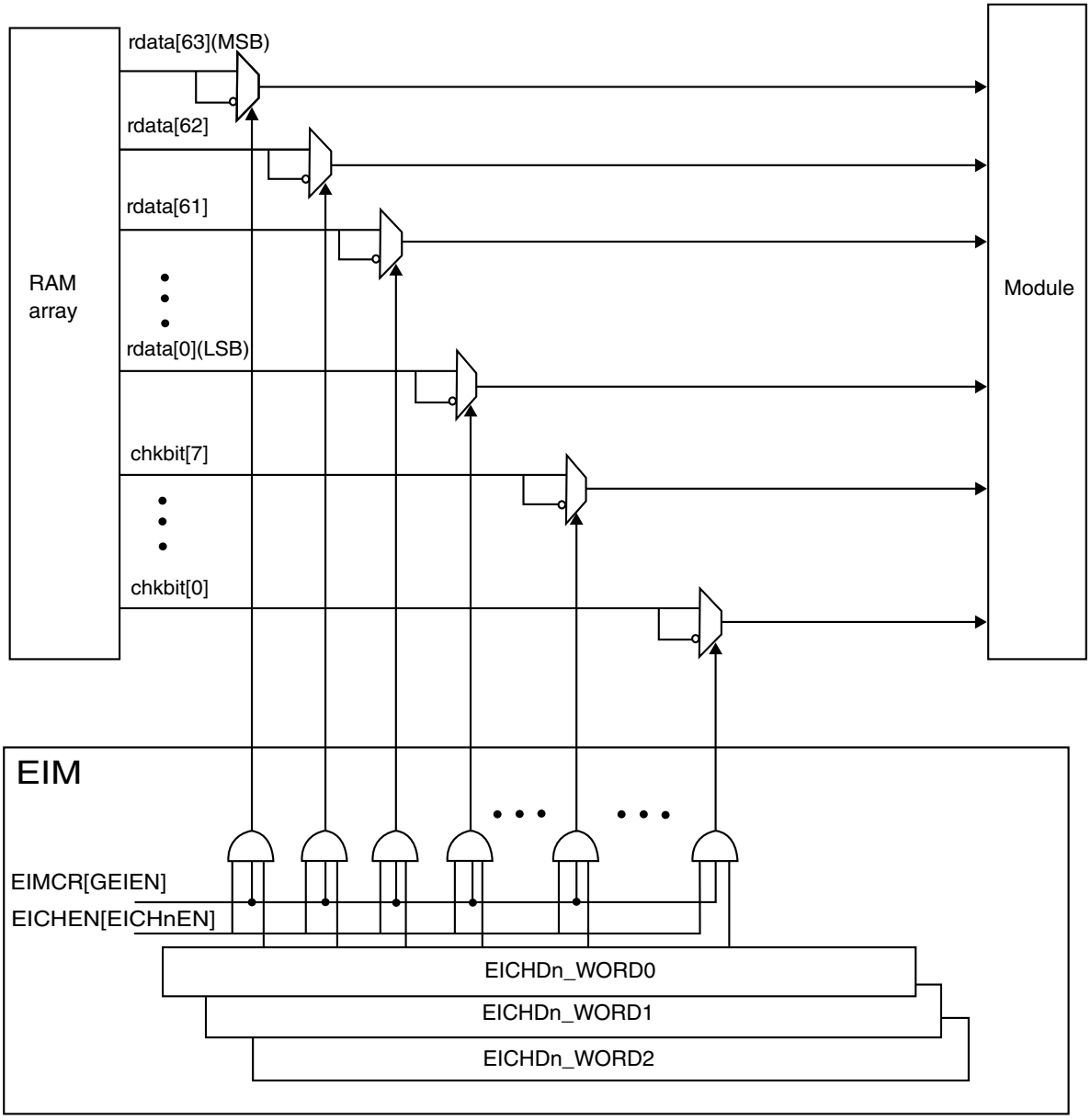


Figure 205. EIM functional block diagram (64-bit read data bus and 8-bit check bit bus)

Several memory elements are implemented within a device, which may not only be the large memory blocks (Flash and SRAM) but also smaller memories like caches, the TCD blocks, and the embedded peripheral memories. Some larger memories may actually be built from multiple memory elements, dependent on their size or function. Each of these memory elements implements its own control logic, the memory controller, that performs the accesses to the actual memory, the memory array. An EIM channel is associated with a memory controller and provides the capability to alter one or multiple signals in the read access path from the corresponding memory array(s). Only memory controllers controlling a safety related memory may be associated with an EIM channel.

50.3 Functional description

The EIM provides protection against accidental enabling and reconfiguration of the error injection function by enforcing a two-stage enablement mechanism. To properly enable the error injection mechanism for a channel:

- Write 1 to the EICHEN[EICH*n*EN] field, where *n* denotes the channel number.

- Write 1 to EIMCR[GEIEN].

NOTE

When the use case for a channel requires writing any EICHD*n*_WORD register, write the EICHD*n*_WORD register before executing the two-stage enablement mechanism. A successful write to any EICHD*n*_WORD register clears the corresponding EICHEN[EICH*n*EN] field.

See the chip-specific EIM information for channel assignment details. Each channel:

- Can be assigned to a single memory array interface by intercepting the assigned memory read data bus and checkbit bus, and injects errors by inverting the value transmitted for selected bits on each bus line.
- Can be assigned to a redundant comparison unit by intercepting the signals being compared, and injecting errors by inverting the value transmitted for selected bits on each bus line.

On a memory read access, the applicable EICHD*n*_WORD registers define which bits of the read data and/or checkbit bus to invert.

Figure 205 depicts the interception and override of a 64-bit read data bus and an 8-bit checkbit data bus for an example memory array.

Error injection scenarios

The EIM supports these cases of error injection:

- To generate a single-bit error, invert only 1 bit of the CHKBIT_MASK or DATA_MASK in the EICHD*n*_WORD registers.
- To generate a multi-bit error, invert only 2 bits of the CHKBIT_MASK or DATA_MASK in the EICHD*n*_WORD registers.

NOTE

An attempt to invert more than 2 bits in one operation might result in undefined behavior.

To enable error injection:

1. Set the EICHD*n*_WORD*m*[CHKBIT_MASK] and EICHD*n*_WORD*m*[B*a*_*b*DATA_MASK] fields for each channel that will be driving an injection.
2. Program the EICHEN register to enable the channels that will be injecting errors.
3. Set the EIMCR[GEIEN] field to globally allow all enabled channels to actively inject errors.

To disable error injection, either disable the EIMCR[GEIEN] field or disable the individual channel enable fields of the EICHEN register.

50.4 Initialization

This module does not require initialization.

50.5 EIM_0 register descriptions

The EIM provides a programming model mapped to an on-platform peripheral slot.

Programming model access

All system bus controllers can access the programming model:

- Only in supervisor mode
- Using only 32-bit (word) accesses

Any of the following attempted references to the programming model generates an error termination:

- In user mode
- Using non-32-bit access sizes

- To undefined (reserved) addresses

Attempted updates to the programming model while the EIM is in the midst of an operation result in non-deterministic behavior.

Error injection channel descriptor: function and structure

Each error injection channel descriptor:

- Specifies a mask that defines which bits of the read data and/or checkbit bus from target RAM are inverted on a read access.
- Consists of a 288-bit (36-byte) structure, composed of nine 32-bit words, in the EIM programming model. Unused words are not documented.
  - Word0 (EICHD*n*_WORD0), if present, defines the checkbit mask.
  - Word1 (EICHD*n*_WORD1) and additional words, if present, define the data mask. Word registers subsequent to Word1 are present only when required by the total width of the channel's data mask. Error injection channel descriptor: DATA_MASK details.

The multiple channel descriptors are organized sequentially.

Error injection channel descriptor: DATA_MASK details

For each channel: The following tables show the distribution of DATA_MASK's bits across the WORD registers. The first table shows the total width of DATA_MASK and the distribution of its bits across WORD1, WORD2, and WORD3. The second table shows the distribution of DATA_MASK's bits across WORD4 and subsequent registers.

Table 283. Error injection channel descriptor: DATA_MASK details

Channel	DATA_MASK total width (bits)	Specific bits of DATA_MASK in		
		WORD1	WORD2	WORD3
0	44	43-32	31-0	—
1	128	127-96	95-64	63-32
2	104	103-96	95-64	63-32
3	128	127-96	95-64	63-32
4	128	127-96	95-64	63-32
5	44	43-32	31-0	—
6	128	127-96	95-64	63-32
7	104	103-96	95-64	63-32
8	128	127-96	95-64	63-32
9	128	127-96	95-64	63-32
10	64	63-32	31-0	—
11	32	31-0	—	—
12	32	31-0	—	—
13	64	63-32	31-0	—

Table continues on the next page...

Table 283. Error injection channel descriptor: DATA_MASK details (continued)

Channel	DATA_MASK total width (bits)	Specific bits of DATA_MASK in		
		WORD1	WORD2	WORD3
14	32	31-0	—	—
15	32	31-0	—	—
16	30	29-0	—	—

Table 284. DATA_MASK bit: Channel-word mapping

Channel	Specific bits of DATA_MASK in				
	WORD4	WORD5	WORD6	WORD7	WORD8
1	31-0	—	—	—	—
2	31-0	—	—	—	—
3	31-0	—	—	—	—
4	31-0	—	—	—	—
6	31-0	—	—	—	—
7	31-0	—	—	—	—
8	31-0	—	—	—	—
9	31-0	—	—	—	—

50.5.1 EIM_0 memory map

EIM_0 base address: 4050_C000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Error Injection Module Configuration Register (EIMCR)	32	RW	0000_0000h
4h	Error Injection Channel Enable register (EICHEN)	32	RW	0000_0000h
100h	Error Injection Channel Descriptor 0, Word0 (EICHD0_WORD0)	32	RW	0000_0000h
104h	Error Injection Channel Descriptor 0, Word1 (EICHD0_WORD1)	32	RW	0000_0000h
108h	Error Injection Channel Descriptor 0, Word2 (EICHD0_WORD2)	32	RW	0000_0000h
140h	Error Injection Channel Descriptor 1, Word0 (EICHD1_WORD0)	32	RW	0000_0000h
144h	Error Injection Channel Descriptor 1, Word1 (EICHD1_WORD1)	32	RW	0000_0000h
148h	Error Injection Channel Descriptor 1, Word2 (EICHD1_WORD2)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
14Ch	Error Injection Channel Descriptor 1, Word3 (EICHD1_WORD3)	32	RW	0000_0000h
150h	Error Injection Channel Descriptor 1, Word4 (EICHD1_WORD4)	32	RW	0000_0000h
180h	Error Injection Channel Descriptor 2, Word0 (EICHD2_WORD0)	32	RW	0000_0000h
184h	Error Injection Channel Descriptor 2, Word1 (EICHD2_WORD1)	32	RW	0000_0000h
188h	Error Injection Channel Descriptor 2, Word2 (EICHD2_WORD2)	32	RW	0000_0000h
18Ch	Error Injection Channel Descriptor 2, Word3 (EICHD2_WORD3)	32	RW	0000_0000h
190h	Error Injection Channel Descriptor 2, Word4 (EICHD2_WORD4)	32	RW	0000_0000h
1C0h	Error Injection Channel Descriptor 3, Word0 (EICHD3_WORD0)	32	RW	0000_0000h
1C4h	Error Injection Channel Descriptor 3, Word1 (EICHD3_WORD1)	32	RW	0000_0000h
1C8h	Error Injection Channel Descriptor 3, Word2 (EICHD3_WORD2)	32	RW	0000_0000h
1CCh	Error Injection Channel Descriptor 3, Word3 (EICHD3_WORD3)	32	RW	0000_0000h
1D0h	Error Injection Channel Descriptor 3, Word4 (EICHD3_WORD4)	32	RW	0000_0000h
200h	Error Injection Channel Descriptor 4, Word0 (EICHD4_WORD0)	32	RW	0000_0000h
204h	Error Injection Channel Descriptor 4, Word1 (EICHD4_WORD1)	32	RW	0000_0000h
208h	Error Injection Channel Descriptor 4, Word2 (EICHD4_WORD2)	32	RW	0000_0000h
20Ch	Error Injection Channel Descriptor 4, Word3 (EICHD4_WORD3)	32	RW	0000_0000h
210h	Error Injection Channel Descriptor 4, Word4 (EICHD4_WORD4)	32	RW	0000_0000h
240h	Error Injection Channel Descriptor 5, Word0 (EICHD5_WORD0)	32	RW	0000_0000h
244h	Error Injection Channel Descriptor 5, Word1 (EICHD5_WORD1)	32	RW	0000_0000h
248h	Error Injection Channel Descriptor 5, Word2 (EICHD5_WORD2)	32	RW	0000_0000h
280h	Error Injection Channel Descriptor 6, Word0 (EICHD6_WORD0)	32	RW	0000_0000h
284h	Error Injection Channel Descriptor 6, Word1 (EICHD6_WORD1)	32	RW	0000_0000h
288h	Error Injection Channel Descriptor 6, Word2 (EICHD6_WORD2)	32	RW	0000_0000h
28Ch	Error Injection Channel Descriptor 6, Word3 (EICHD6_WORD3)	32	RW	0000_0000h
290h	Error Injection Channel Descriptor 6, Word4 (EICHD6_WORD4)	32	RW	0000_0000h
2C0h	Error Injection Channel Descriptor 7, Word0 (EICHD7_WORD0)	32	RW	0000_0000h
2C4h	Error Injection Channel Descriptor 7, Word1 (EICHD7_WORD1)	32	RW	0000_0000h
2C8h	Error Injection Channel Descriptor 7, Word2 (EICHD7_WORD2)	32	RW	0000_0000h
2CCh	Error Injection Channel Descriptor 7, Word3 (EICHD7_WORD3)	32	RW	0000_0000h
2D0h	Error Injection Channel Descriptor 7, Word4 (EICHD7_WORD4)	32	RW	0000_0000h
300h	Error Injection Channel Descriptor 8, Word0 (EICHD8_WORD0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
304h	Error Injection Channel Descriptor 8, Word1 (EICHD8_WORD1)	32	RW	0000_0000h
308h	Error Injection Channel Descriptor 8, Word2 (EICHD8_WORD2)	32	RW	0000_0000h
30Ch	Error Injection Channel Descriptor 8, Word3 (EICHD8_WORD3)	32	RW	0000_0000h
310h	Error Injection Channel Descriptor 8, Word4 (EICHD8_WORD4)	32	RW	0000_0000h
340h	Error Injection Channel Descriptor 9, Word0 (EICHD9_WORD0)	32	RW	0000_0000h
344h	Error Injection Channel Descriptor 9, Word1 (EICHD9_WORD1)	32	RW	0000_0000h
348h	Error Injection Channel Descriptor 9, Word2 (EICHD9_WORD2)	32	RW	0000_0000h
34Ch	Error Injection Channel Descriptor 9, Word3 (EICHD9_WORD3)	32	RW	0000_0000h
350h	Error Injection Channel Descriptor 9, Word4 (EICHD9_WORD4)	32	RW	0000_0000h
380h	Error Injection Channel Descriptor 10, Word0 (EICHD10_WORD0)	32	RW	0000_0000h
384h	Error Injection Channel Descriptor 10, Word1 (EICHD10_WORD1)	32	RW	0000_0000h
388h	Error Injection Channel Descriptor 10, Word2 (EICHD10_WORD2)	32	RW	0000_0000h
3C0h	Error Injection Channel Descriptor 11, Word0 (EICHD11_WORD0)	32	RW	0000_0000h
3C4h	Error Injection Channel Descriptor 11, Word1 (EICHD11_WORD1)	32	RW	0000_0000h
400h	Error Injection Channel Descriptor 12, Word0 (EICHD12_WORD0)	32	RW	0000_0000h
404h	Error Injection Channel Descriptor 12, Word1 (EICHD12_WORD1)	32	RW	0000_0000h
440h	Error Injection Channel Descriptor 13, Word0 (EICHD13_WORD0)	32	RW	0000_0000h
444h	Error Injection Channel Descriptor 13, Word1 (EICHD13_WORD1)	32	RW	0000_0000h
448h	Error Injection Channel Descriptor 13, Word2 (EICHD13_WORD2)	32	RW	0000_0000h
480h	Error Injection Channel Descriptor 14, Word0 (EICHD14_WORD0)	32	RW	0000_0000h
484h	Error Injection Channel Descriptor 14, Word1 (EICHD14_WORD1)	32	RW	0000_0000h
4C0h	Error Injection Channel Descriptor 15, Word0 (EICHD15_WORD0)	32	RW	0000_0000h
4C4h	Error Injection Channel Descriptor 15, Word1 (EICHD15_WORD1)	32	RW	0000_0000h
504h	Error Injection Channel Descriptor 16, Word1 (EICHD16_WORD1)	32	RW	0000_0000h

50.5.2 Error Injection Module Configuration Register (EIMCR)

Offset

Register	Offset
EIMCR	0h

Function

The EIM Configuration Register is used to globally enable/disable the error injection function.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															GEIEN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-1 —	Reserved
0 GEIEN	Global Error Injection Enable This bit globally enables or disables the error injection function of the EIM. This field is initialized by hardware reset. 0b - Disabled 1b - Enabled

50.5.3 Error Injection Channel Enable register (EICHEN)

Offset

Register	Offset
EICHEN	4h

Function

Each field of the Error Injection Channel Enable register (EICHEN) is used to enable or disable the corresponding error injection channel.

NOTE

To enable an error injection channel, the Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	EICH0EN	EICH1EN	EICH2EN	EICH3EN	EICH4EN	EICH5EN	EICH6EN	EICH7EN	EICH8EN	EICH9EN	EICH10EN	EICH11EN	EICH12EN	EICH13EN	EICH14EN	EICH15EN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EICH16EN	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 EICH0EN	Error Injection Channel 0 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 0 1b - Error injection is enabled on Error Injection Channel 0
30 EICH1EN	Error Injection Channel 1 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 1 1b - Error injection is enabled on Error Injection Channel 1
29 EICH2EN	Error Injection Channel 2 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 2 1b - Error injection is enabled on Error Injection Channel 2
28 EICH3EN	Error Injection Channel 3 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 3 1b - Error injection is enabled on Error Injection Channel 3
27 EICH4EN	Error Injection Channel 4 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 4 1b - Error injection is enabled on Error Injection Channel 4
26 EICH5EN	Error Injection Channel 5 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 5 1b - Error injection is enabled on Error Injection Channel 5
25 EICH6EN	Error Injection Channel 6 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 6 1b - Error injection is enabled on Error Injection Channel 6
24 EICH7EN	Error Injection Channel 7 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 7 1b - Error injection is enabled on Error Injection Channel 7
23 EICH8EN	Error Injection Channel 8 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 8 1b - Error injection is enabled on Error Injection Channel 8
22 EICH9EN	Error Injection Channel 9 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 9 1b - Error injection is enabled on Error Injection Channel 9

Table continues on the next page...

Table continued from the previous page...

Field	Function
21 EICH10EN	Error Injection Channel 10 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 10 1b - Error injection is enabled on Error Injection Channel 10
20 EICH11EN	Error Injection Channel 11 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 11 1b - Error injection is enabled on Error Injection Channel 11
19 EICH12EN	Error Injection Channel 12 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 12 1b - Error injection is enabled on Error Injection Channel 12
18 EICH13EN	Error Injection Channel 13 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Error injection is disabled on Error Injection Channel 13 1b - Error injection is enabled on Error Injection Channel 13
17 EICH14EN	Error Injection Channel 14 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 14 1b - Error injection is enabled on Error Injection Channel 14
16 EICH15EN	Error Injection Channel 15 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 15 1b - Error injection is enabled on Error Injection Channel 15
15 EICH16EN	Error Injection Channel 16 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 16 1b - Error injection is enabled on Error Injection Channel 16
14 —	Reserved
13 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
12 —	Reserved
11 —	Reserved
10 —	Reserved
9 —	Reserved
8 —	Reserved
7 —	Reserved
6 —	Reserved
5 —	Reserved
4 —	Reserved
3 —	Reserved
2 —	Reserved
1 —	Reserved
0 —	Reserved

50.5.4 Error Injection Channel Descriptor 0, Word0 (EICHD0_WORD0)

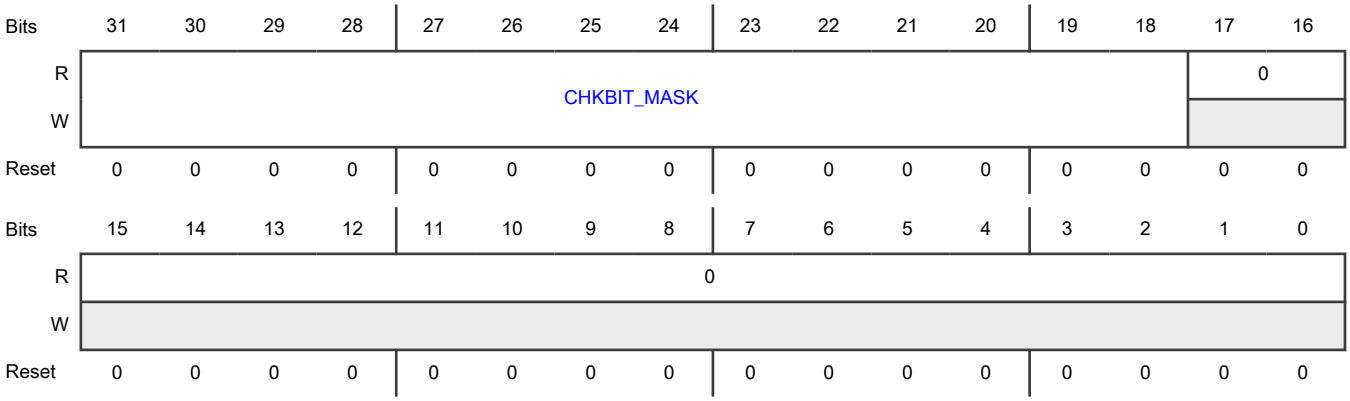
Offset

Register	Offset
EICHD0_WORD0	100h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-18 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[13:0] (14 bits wide), CHKBIT_MASK[13] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
17-0 —	Reserved

50.5.5 Error Injection Channel Descriptor 0, Word1 (EICHD0_WORD1)

Offset

Register	Offset
EICHD0_WORD1	104h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD0_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				B0_3DATA_MASK											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-12 —	Reserved
11-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD0_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD0_WORD1 on the read data bus is inverted.

50.5.6 Error Injection Channel Descriptor 0, Word2 (EICHD0_WORD2)

Offset

Register	Offset
EICHD0_WORD2	108h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to EICHD0_WORD2 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	B4_7DATA_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	B4_7DATA_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD0_WORD2 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD0_WORD2 on the read data bus is inverted.

50.5.7 Error Injection Channel Descriptor 1, Word0 (EICHD1_WORD0)

Offset

Register	Offset
EICHD1_WORD0	140h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	CHKBIT_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[15:0] (16 bits wide), CHKBIT_MASK[15] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
15-0 —	Reserved

50.5.8 Error Injection Channel Descriptor 1, Word1 (EICHD1_WORD1)

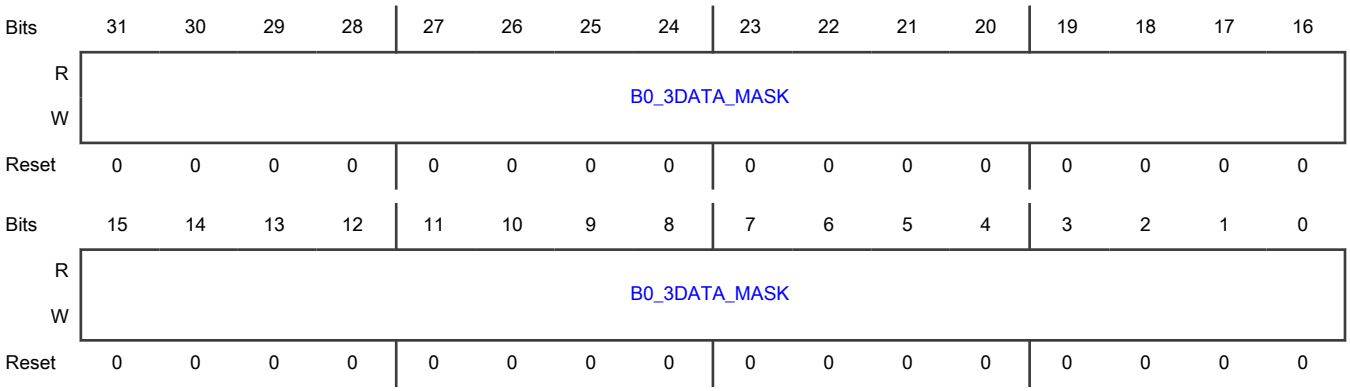
Offset

Register	Offset
EICHD1_WORD1	144h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD1_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD1_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD1_WORD1 on the read data bus is inverted.

50.5.9 Error Injection Channel Descriptor 1, Word2 (EICHD1_WORD2)

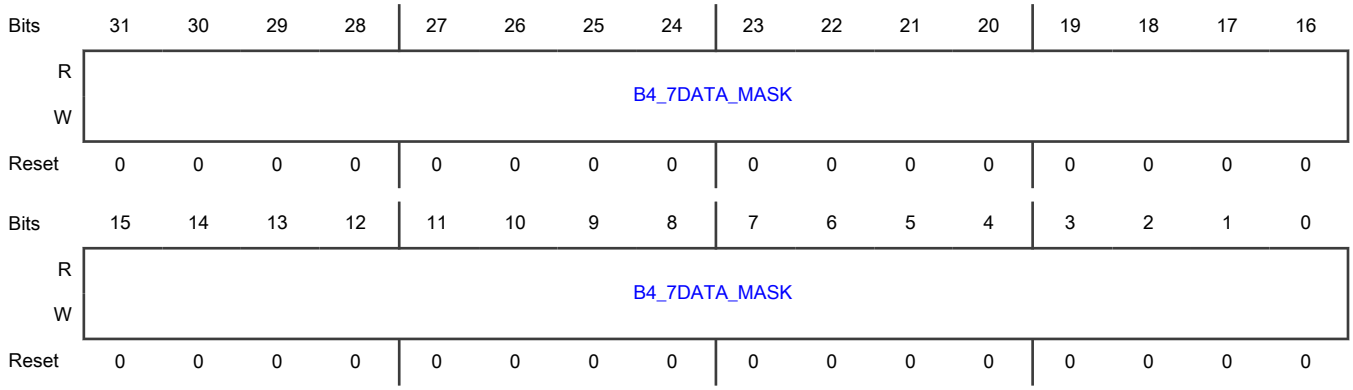
Offset

Register	Offset
EICHD1_WORD2	148h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to EICHD1_WORD2 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD1_WORD2 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD1_WORD2 on the read data bus is inverted.

50.5.10 Error Injection Channel Descriptor 1, Word3 (EICHD1_WORD3)

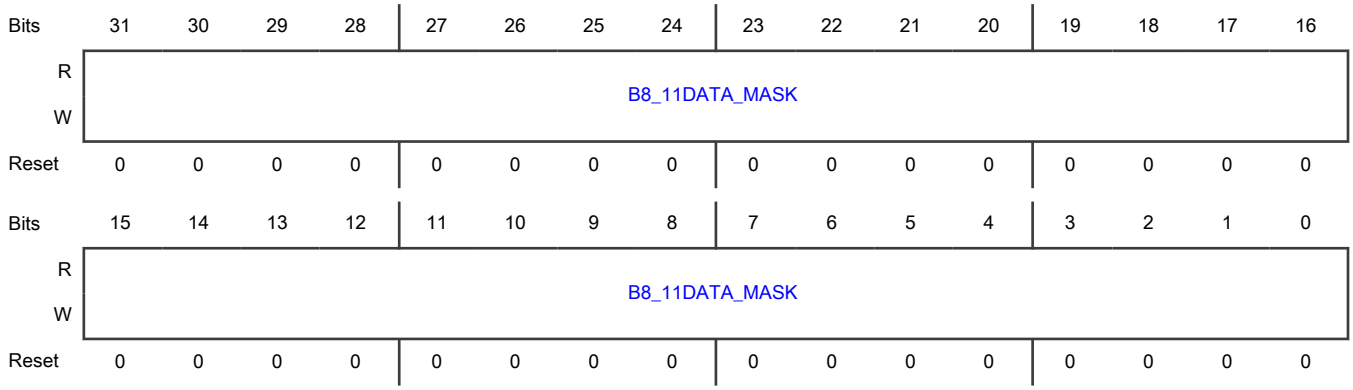
Offset

Register	Offset
EICHD1_WORD3	14Ch

Function

The fourth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B8_11DATA_MASK correspond to EICHD1_WORD3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B8_11DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B8_11DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD1_WORD3 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD1_WORD3 on the read data bus is inverted.

50.5.11 Error Injection Channel Descriptor 1, Word4 (EICHD1_WORD4)

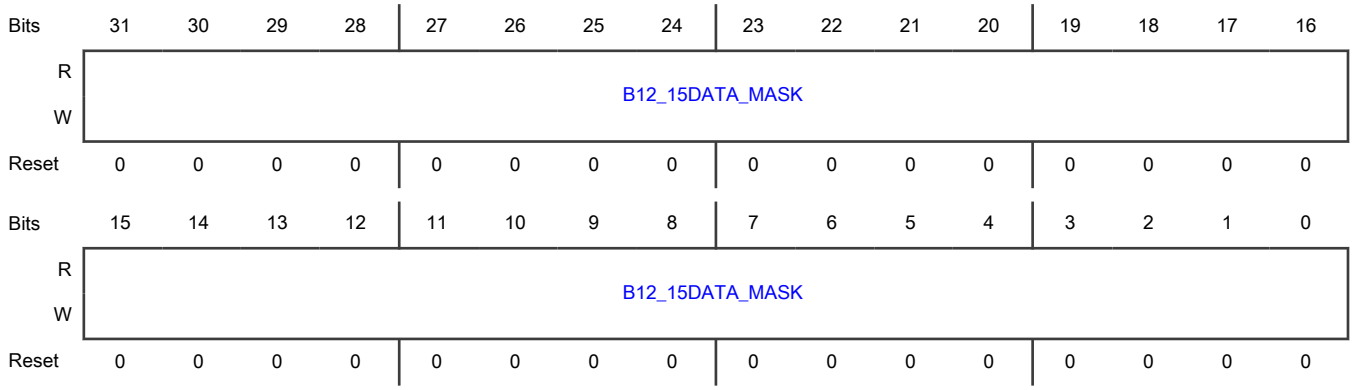
Offset

Register	Offset
EICHD1_WORD4	150h

Function

The fifth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B12_15DATA_MASK correspond to EICHD1_WORD4 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B12_15DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B12_15DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD1_WORD4 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD1_WORD4 on the read data bus is inverted.

50.5.12 Error Injection Channel Descriptor n, Word0 (EICHD2_WORD0 - EICHD4_WORD0)

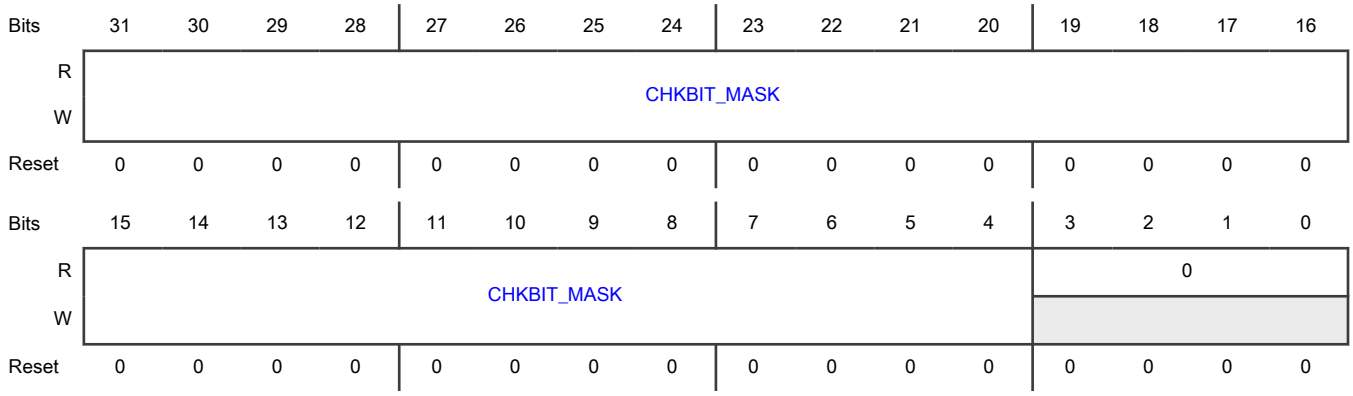
Offset

Register	Offset
EICHD2_WORD0	180h
EICHD3_WORD0	1C0h
EICHD4_WORD0	200h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-4 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[27:0] (28 bits wide), CHKBIT_MASK[27] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
3-0 —	Reserved

50.5.13 Error Injection Channel Descriptor 2, Word1 (EICHD2_WORD1)

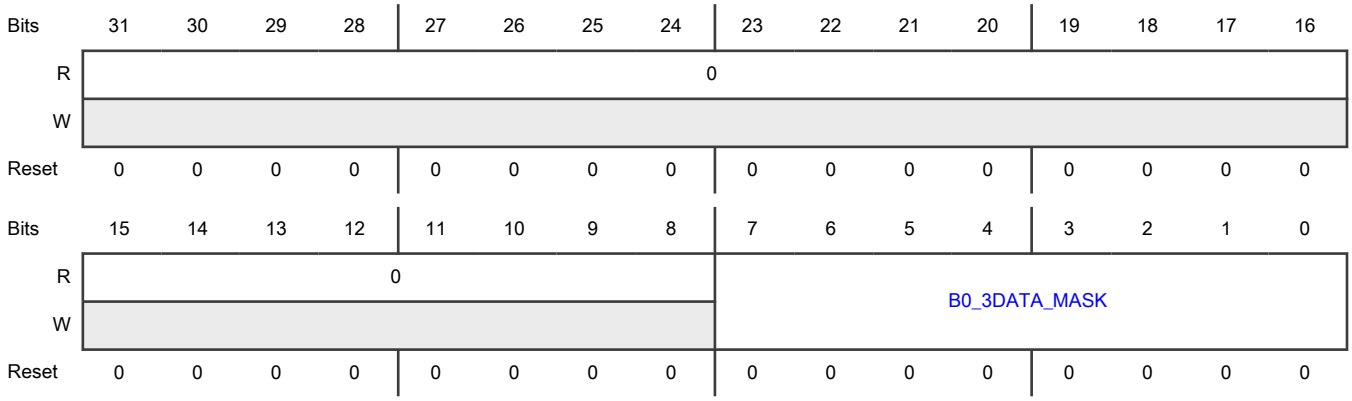
Offset

Register	Offset
EICHD2_WORD1	184h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD2_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD2_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD2_WORD1 on the read data bus is inverted.

50.5.14 Error Injection Channel Descriptor 2, Word2 (EICHD2_WORD2)

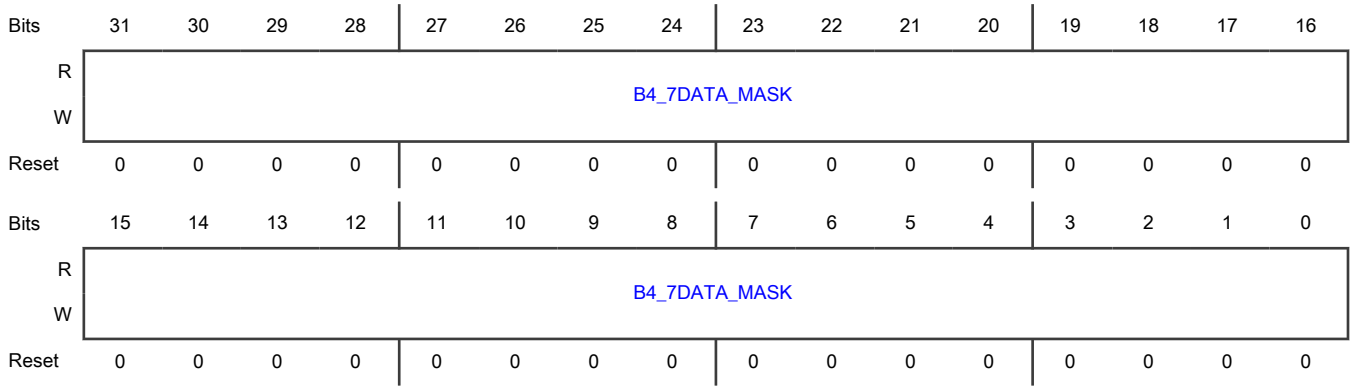
Offset

Register	Offset
EICHD2_WORD2	188h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to EICHD2_WORD2 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD2_WORD2 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD2_WORD2 on the read data bus is inverted.

50.5.15 Error Injection Channel Descriptor 2, Word3 (EICHD2_WORD3)

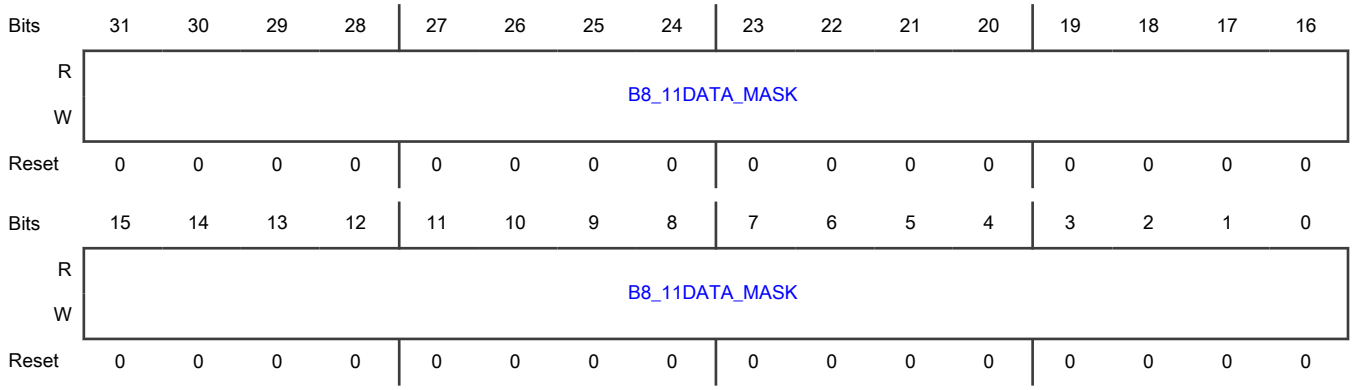
Offset

Register	Offset
EICHD2_WORD3	18Ch

Function

The fourth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B8_11DATA_MASK correspond to EICHD2_WORD3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B8_11DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B8_11DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD2_WORD3 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD2_WORD3 on the read data bus is inverted.

50.5.16 Error Injection Channel Descriptor 2, Word4 (EICHD2_WORD4)

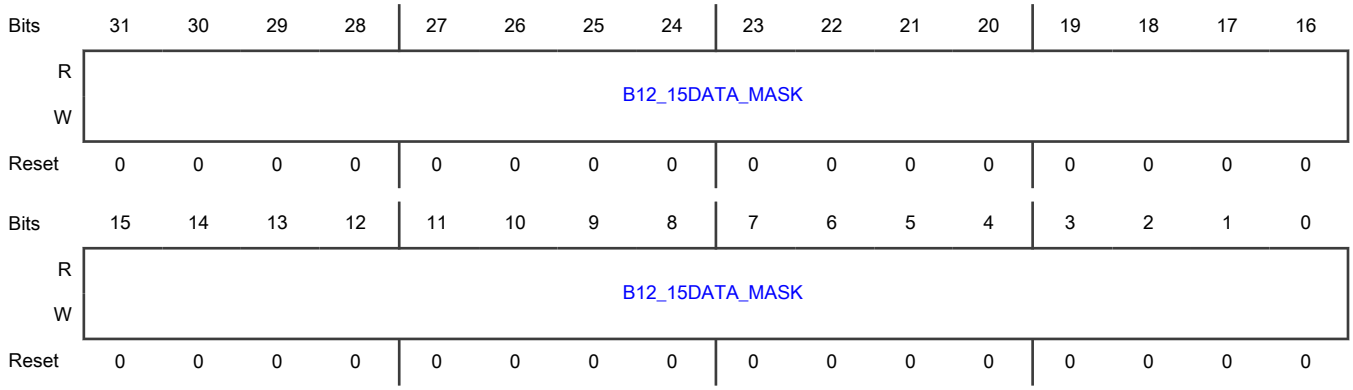
Offset

Register	Offset
EICHD2_WORD4	190h

Function

The fifth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B12_15DATA_MASK correspond to EICHD2_WORD4 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B12_15DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B12_15DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD2_WORD4 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD2_WORD4 on the read data bus is inverted.

50.5.17 Error Injection Channel Descriptor 3, Word1 (EICHD3_WORD1)

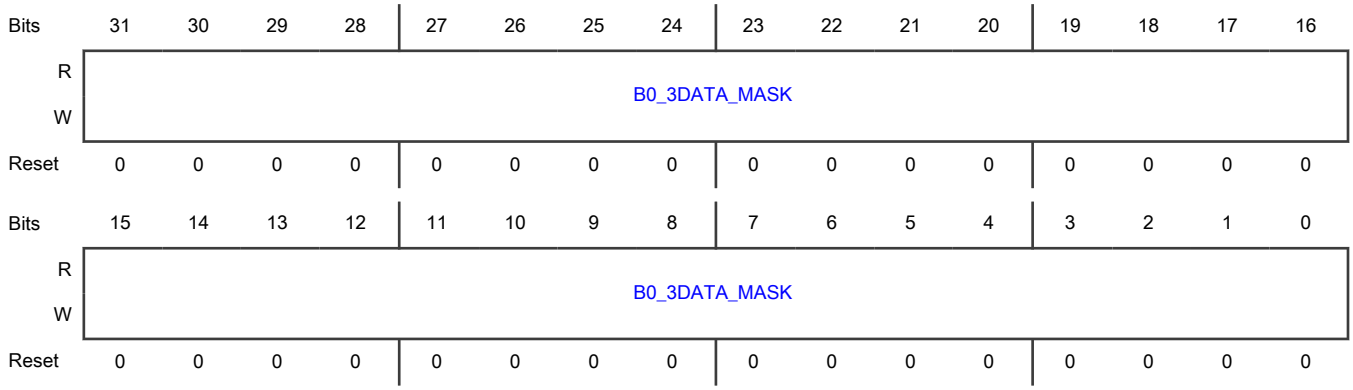
Offset

Register	Offset
EICHD3_WORD1	1C4h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD3_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD3_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD3_WORD1 on the read data bus is inverted.

50.5.18 Error Injection Channel Descriptor 3, Word2 (EICHD3_WORD2)

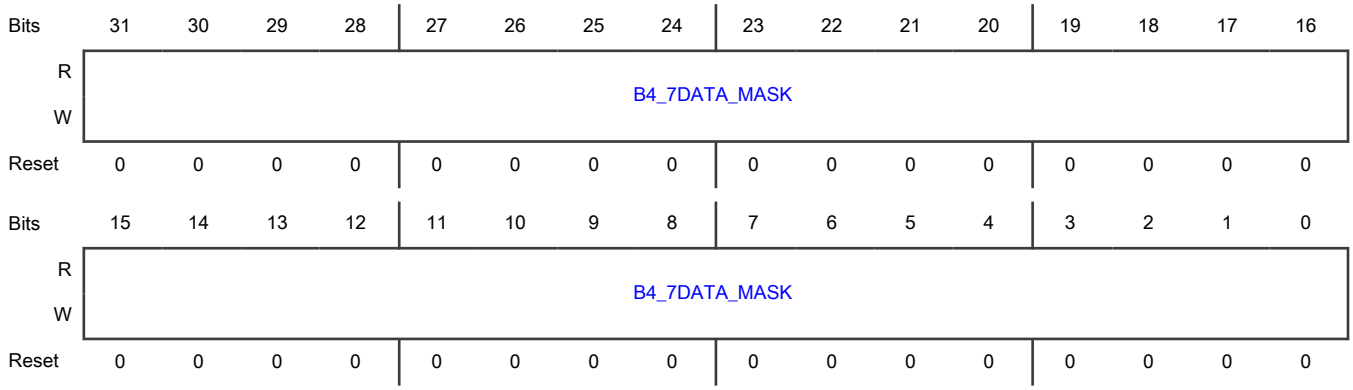
Offset

Register	Offset
EICHD3_WORD2	1C8h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to EICHD3_WORD2 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD3_WORD2 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD3_WORD2 on the read data bus is inverted.

50.5.19 Error Injection Channel Descriptor 3, Word3 (EICHD3_WORD3)

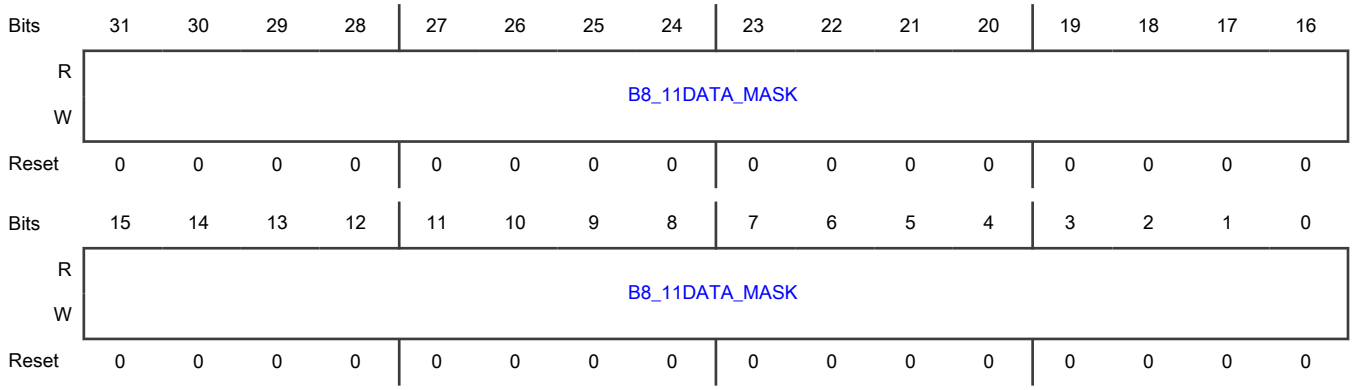
Offset

Register	Offset
EICHD3_WORD3	1CCh

Function

The fourth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B8_11DATA_MASK correspond to EICHD3_WORD3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B8_11DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B8_11DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD3_WORD3 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD3_WORD3 on the read data bus is inverted.

50.5.20 Error Injection Channel Descriptor 3, Word4 (EICHD3_WORD4)

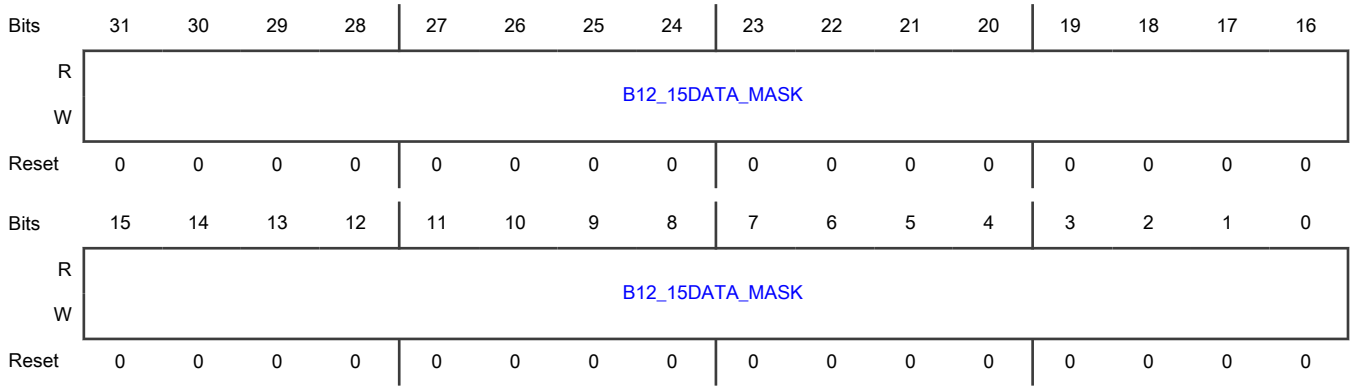
Offset

Register	Offset
EICHD3_WORD4	1D0h

Function

The fifth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B12_15DATA_MASK correspond to EICHD3_WORD4 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B12_15DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B12_15DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD3_WORD4 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD3_WORD4 on the read data bus is inverted.

50.5.21 Error Injection Channel Descriptor 4, Word1 (EICHD4_WORD1)

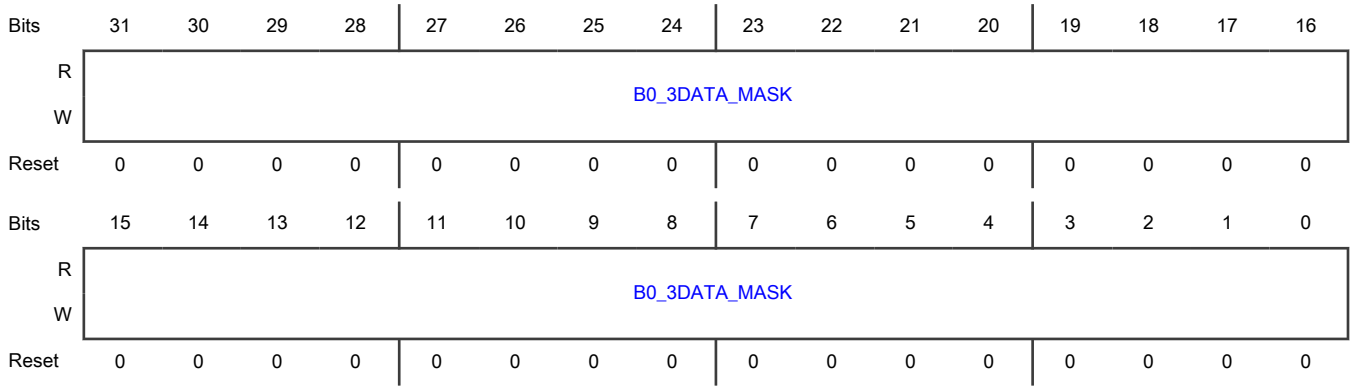
Offset

Register	Offset
EICHD4_WORD1	204h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD4_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD4_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD4_WORD1 on the read data bus is inverted.

50.5.22 Error Injection Channel Descriptor 4, Word2 (EICHD4_WORD2)

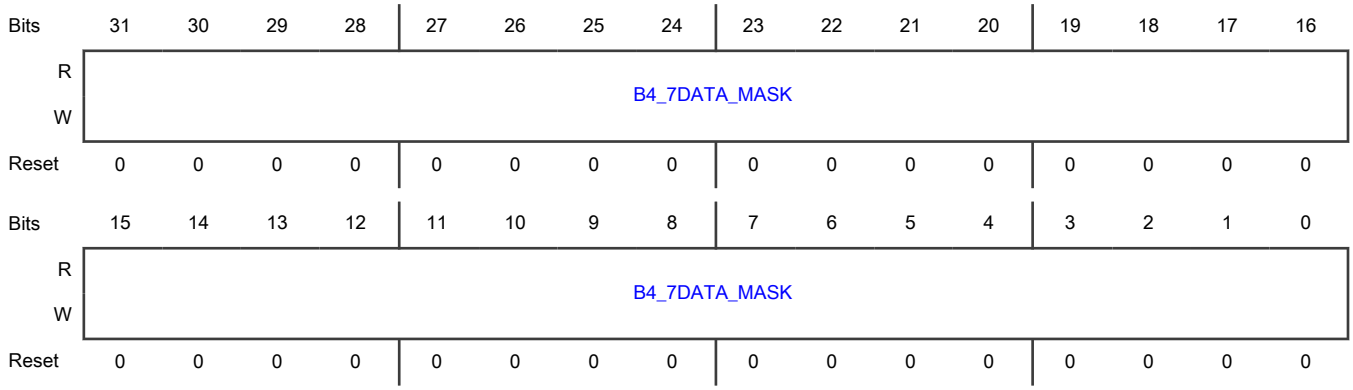
Offset

Register	Offset
EICHD4_WORD2	208h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to EICHD4_WORD2 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD4_WORD2 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD4_WORD2 on the read data bus is inverted.

50.5.23 Error Injection Channel Descriptor 4, Word3 (EICHD4_WORD3)

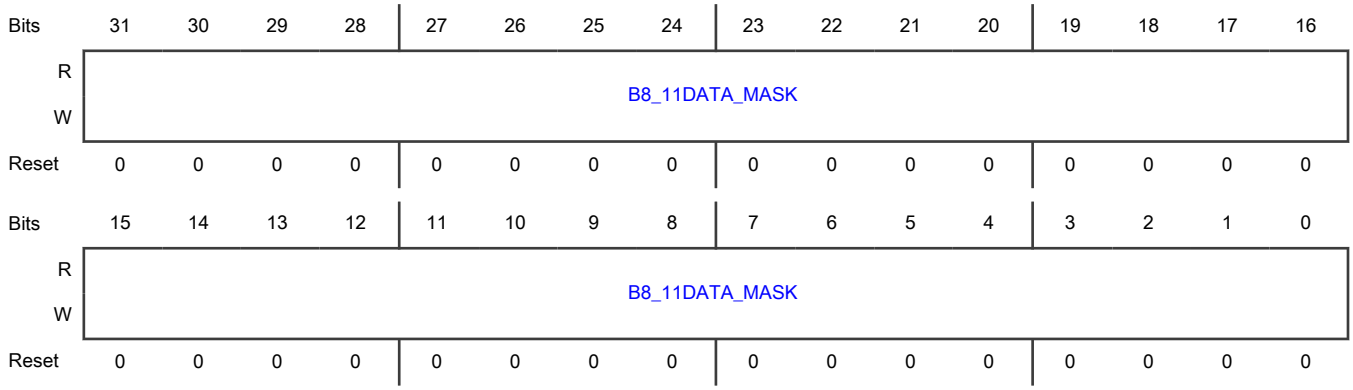
Offset

Register	Offset
EICHD4_WORD3	20Ch

Function

The fourth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B8_11DATA_MASK correspond to EICHD4_WORD3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B8_11DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B8_11DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD4_WORD3 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD4_WORD3 on the read data bus is inverted.

50.5.24 Error Injection Channel Descriptor 4, Word4 (EICHD4_WORD4)

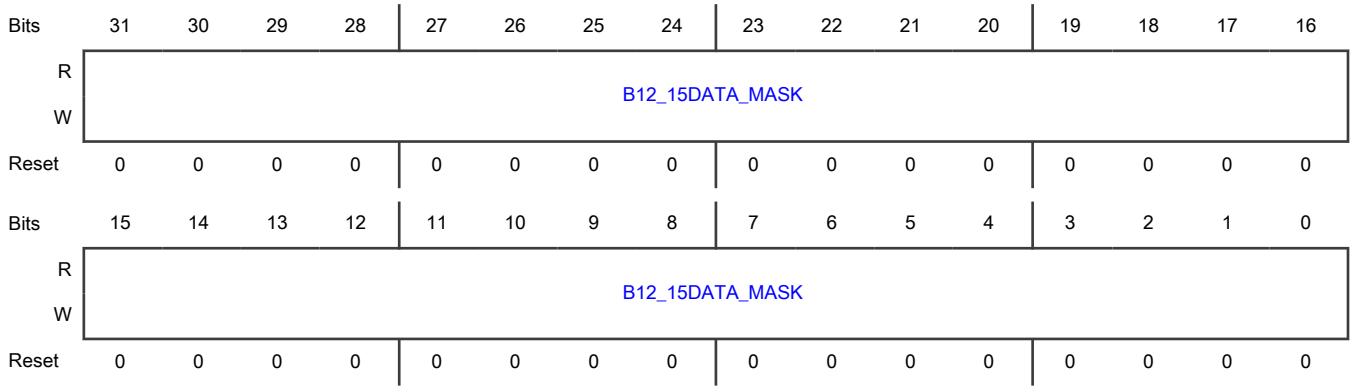
Offset

Register	Offset
EICHD4_WORD4	210h

Function

The fifth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B12_15DATA_MASK correspond to EICHD4_WORD4 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B12_15DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B12_15DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD4_WORD4 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD4_WORD4 on the read data bus is inverted.

50.5.25 Error Injection Channel Descriptor 5, Word0 (EICHD5_WORD0)

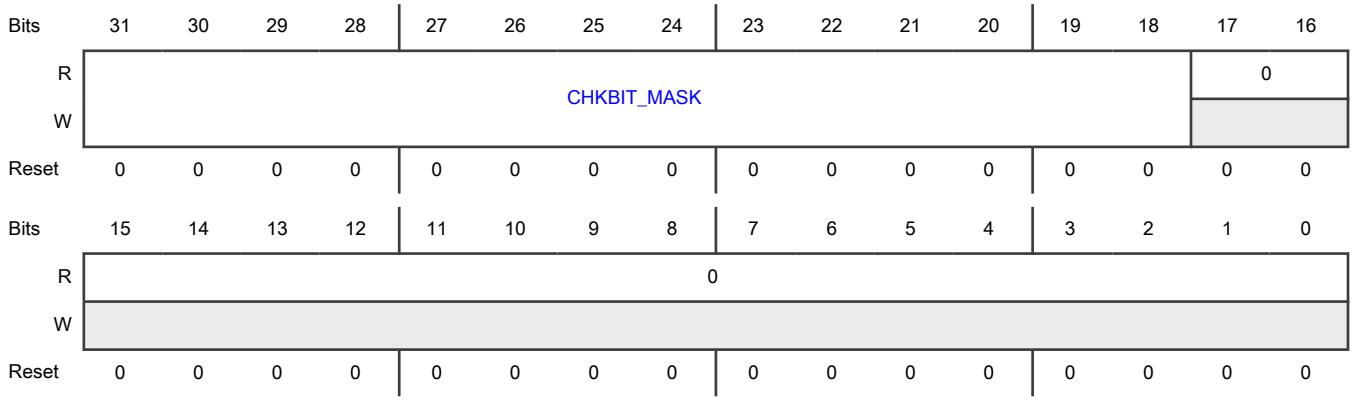
Offset

Register	Offset
EICHD5_WORD0	240h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-18 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[13:0] (14 bits wide), CHKBIT_MASK[13] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
17-0 —	Reserved

50.5.26 Error Injection Channel Descriptor 5, Word1 (EICHD5_WORD1)

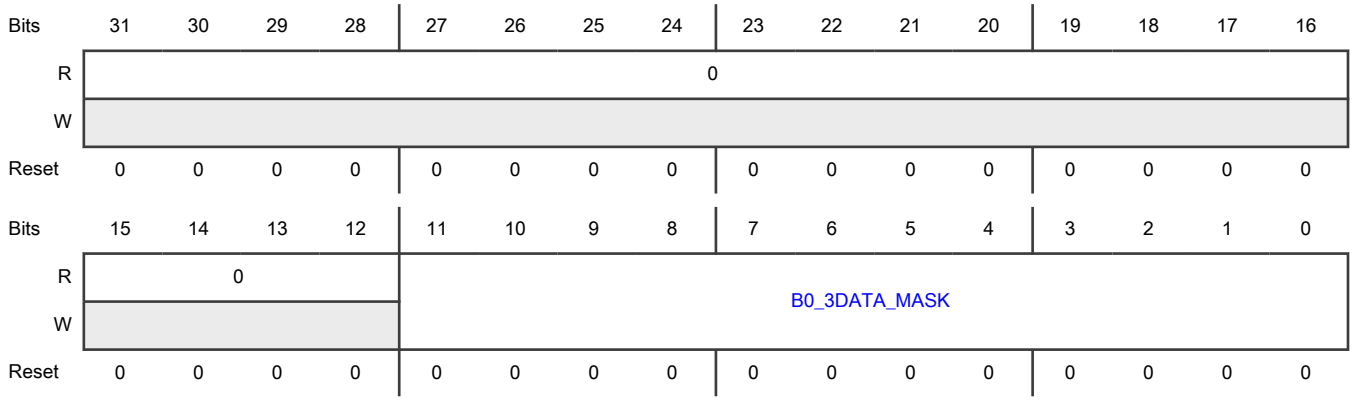
Offset

Register	Offset
EICHD5_WORD1	244h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD5_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-12 —	Reserved
11-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD5_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD5_WORD1 on the read data bus is inverted.

50.5.27 Error Injection Channel Descriptor 5, Word2 (EICHD5_WORD2)

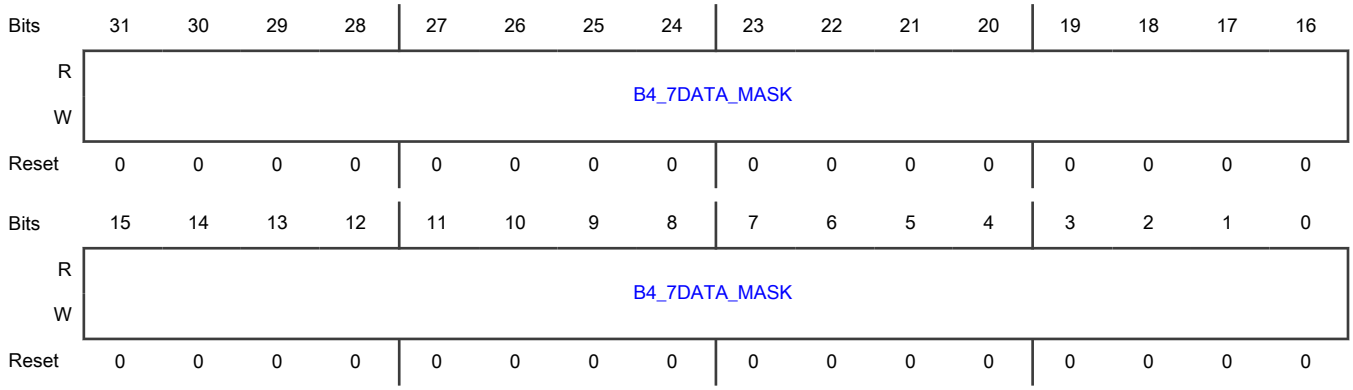
Offset

Register	Offset
EICHD5_WORD2	248h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to EICHD5_WORD2 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD5_WORD2 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD5_WORD2 on the read data bus is inverted.

50.5.28 Error Injection Channel Descriptor 6, Word0 (EICHD6_WORD0)

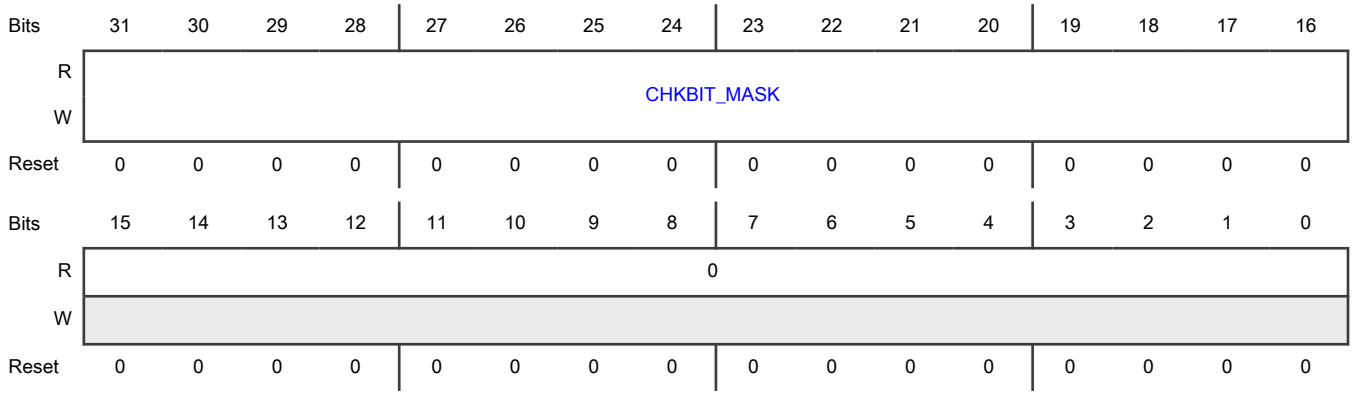
Offset

Register	Offset
EICHD6_WORD0	280h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-16 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[15:0] (16 bits wide), CHKBIT_MASK[15] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
15-0 —	Reserved

50.5.29 Error Injection Channel Descriptor 6, Word1 (EICHD6_WORD1)

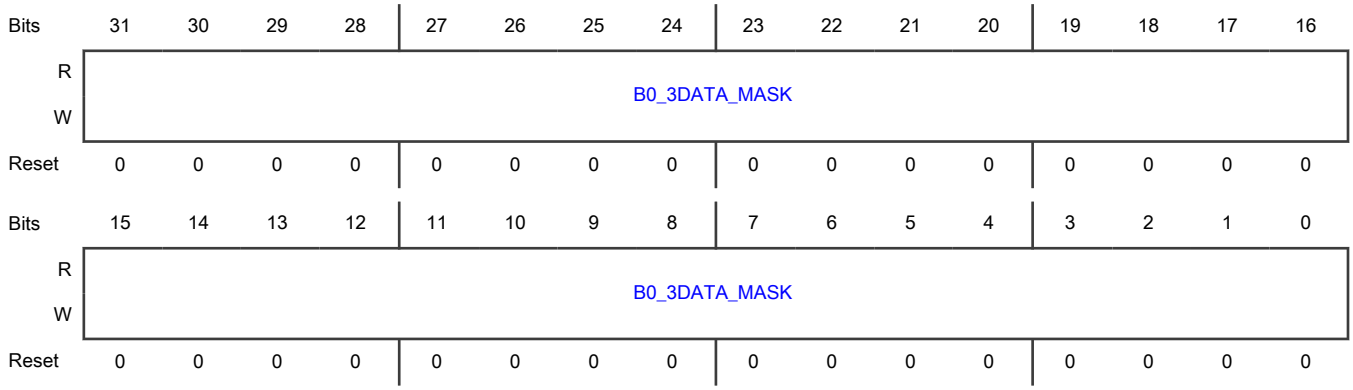
Offset

Register	Offset
EICHD6_WORD1	284h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD6_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD6_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD6_WORD1 on the read data bus is inverted.

50.5.30 Error Injection Channel Descriptor 6, Word2 (EICHD6_WORD2)

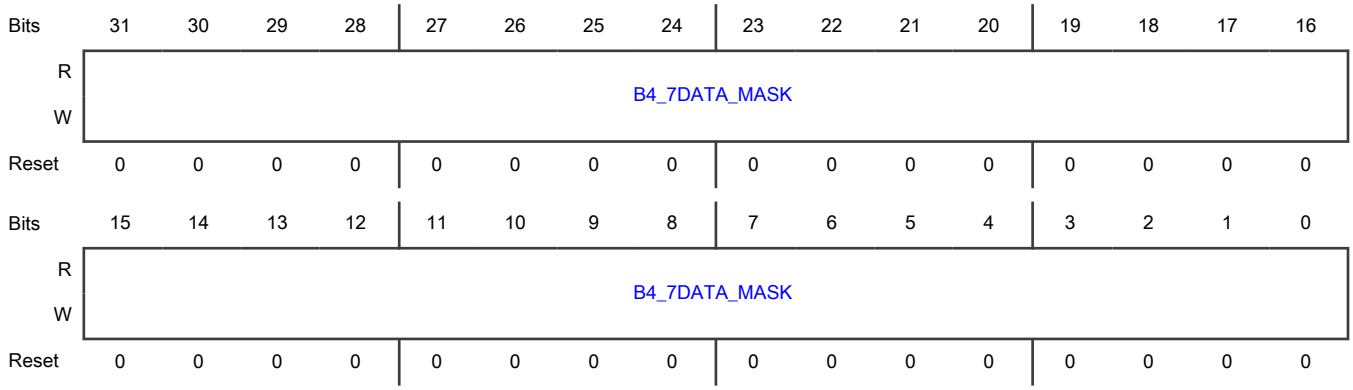
Offset

Register	Offset
EICHD6_WORD2	288h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to EICHD6_WORD2 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD6_WORD2 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD6_WORD2 on the read data bus is inverted.

50.5.31 Error Injection Channel Descriptor 6, Word3 (EICHD6_WORD3)

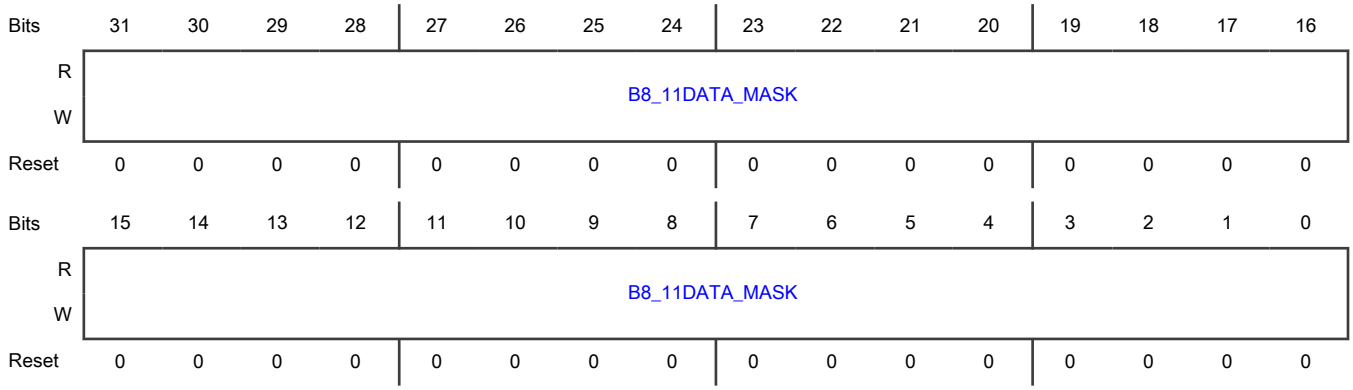
Offset

Register	Offset
EICHD6_WORD3	28Ch

Function

The fourth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B8_11DATA_MASK correspond to EICHD6_WORD3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B8_11DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B8_11DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD6_WORD3 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD6_WORD3 on the read data bus is inverted.

50.5.32 Error Injection Channel Descriptor 6, Word4 (EICHD6_WORD4)

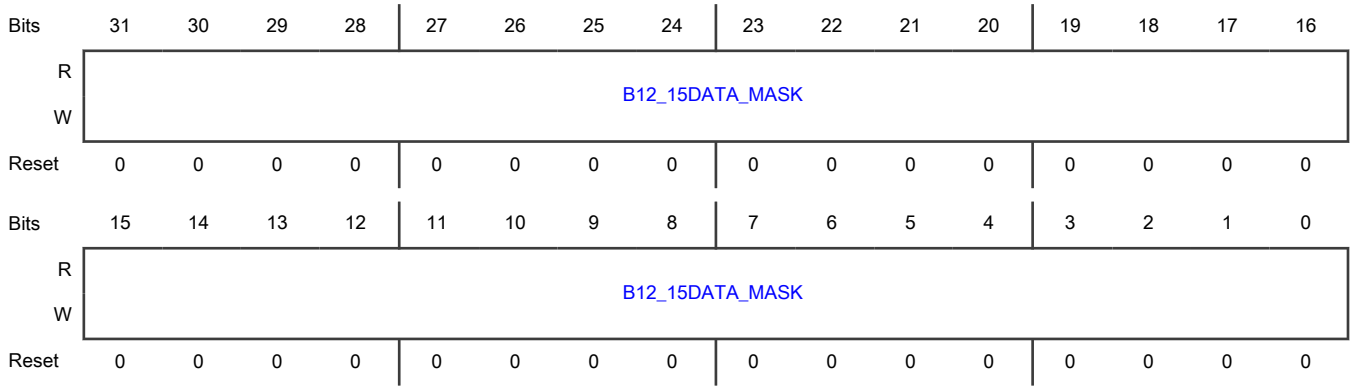
Offset

Register	Offset
EICHD6_WORD4	290h

Function

The fifth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B12_15DATA_MASK correspond to EICHD6_WORD4 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B12_15DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B12_15DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD6_WORD4 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD6_WORD4 on the read data bus is inverted.

50.5.33 Error Injection Channel Descriptor n, Word0 (EICHD7_WORD0 - EICHD9_WORD0)

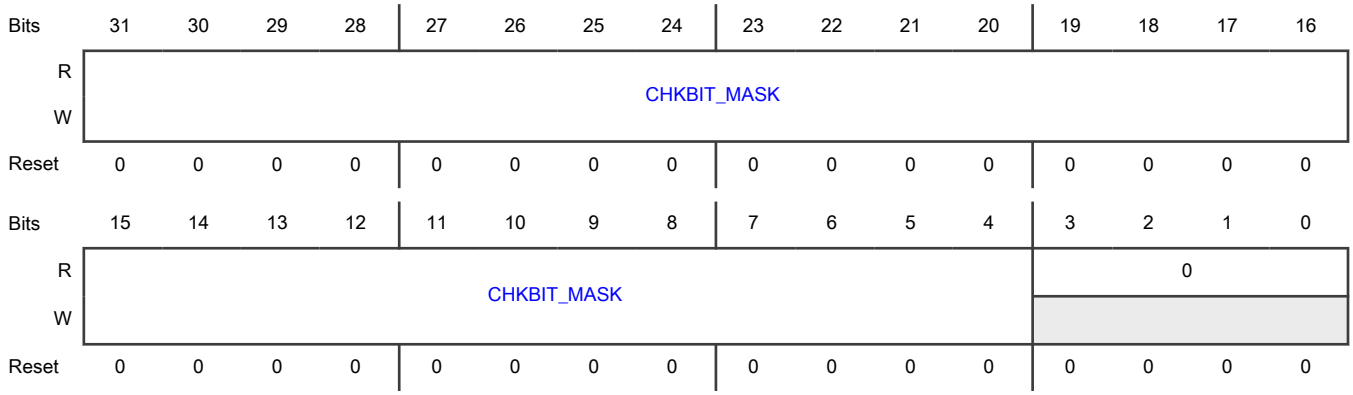
Offset

Register	Offset
EICHD7_WORD0	2C0h
EICHD8_WORD0	300h
EICHD9_WORD0	340h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-4 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[27:0] (28 bits wide), CHKBIT_MASK[27] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
3-0 —	Reserved

50.5.34 Error Injection Channel Descriptor 7, Word1 (EICHD7_WORD1)

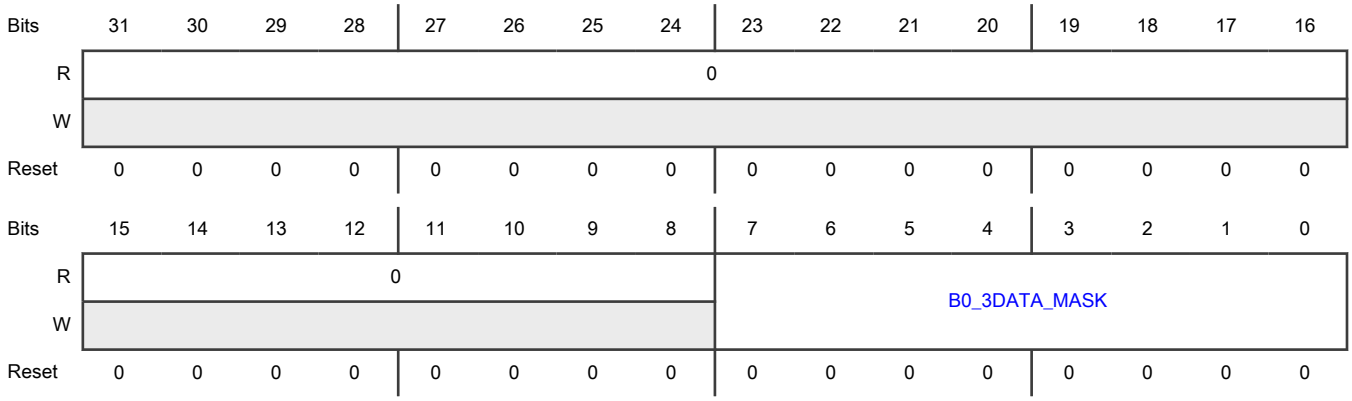
Offset

Register	Offset
EICHD7_WORD1	2C4h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD7_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD7_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD7_WORD1 on the read data bus is inverted.

50.5.35 Error Injection Channel Descriptor 7, Word2 (EICHD7_WORD2)

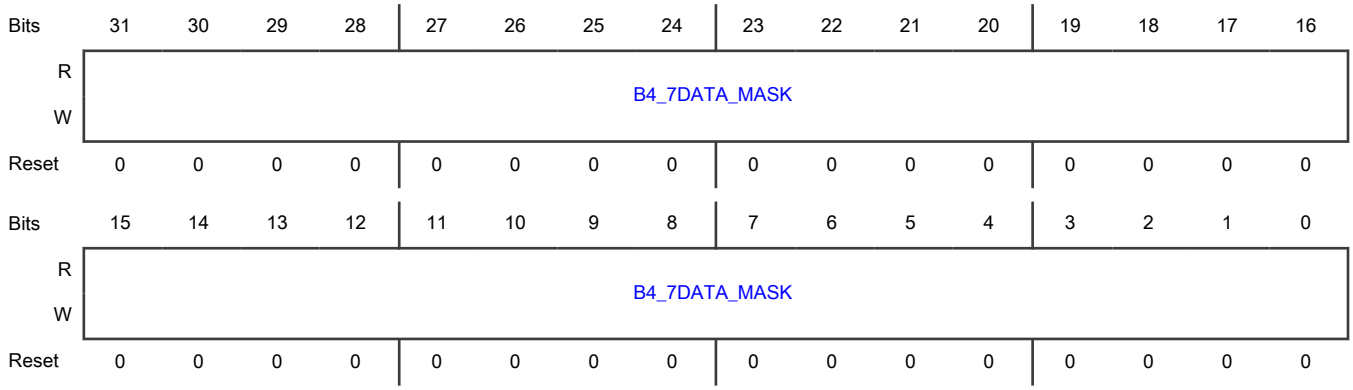
Offset

Register	Offset
EICHD7_WORD2	2C8h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to EICHD7_WORD2 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD7_WORD2 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD7_WORD2 on the read data bus is inverted.

50.5.36 Error Injection Channel Descriptor 7, Word3 (EICHD7_WORD3)

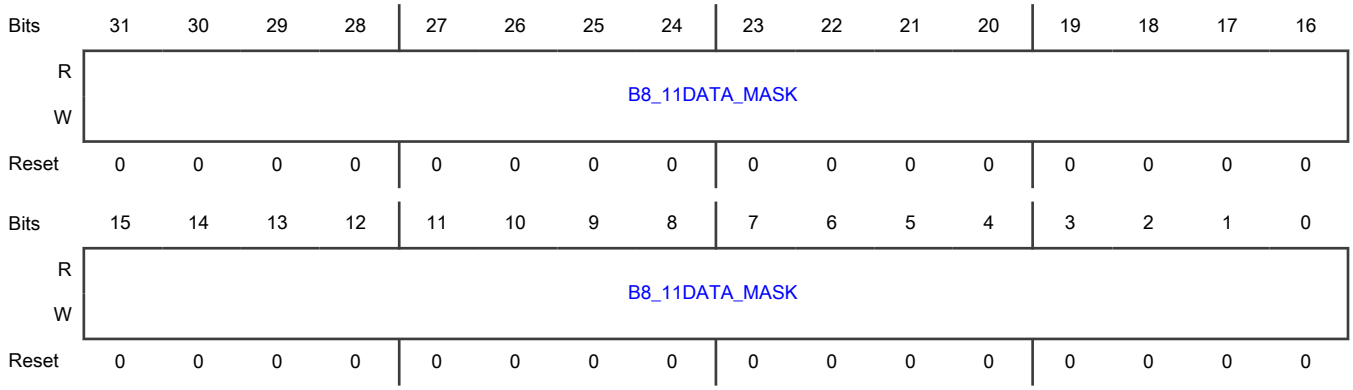
Offset

Register	Offset
EICHD7_WORD3	2CCh

Function

The fourth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B8_11DATA_MASK correspond to EICHD7_WORD3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B8_11DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B8_11DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD7_WORD3 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD7_WORD3 on the read data bus is inverted.

50.5.37 Error Injection Channel Descriptor 7, Word4 (EICHD7_WORD4)

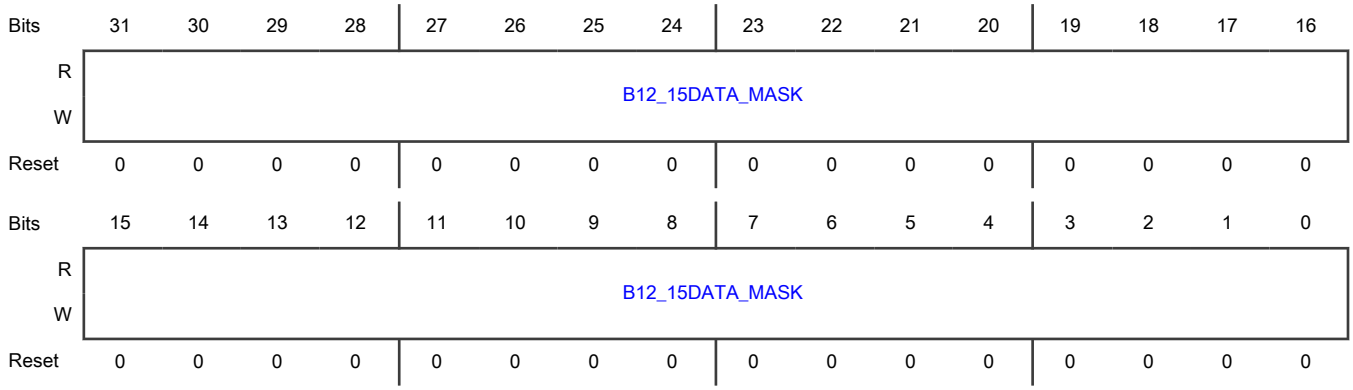
Offset

Register	Offset
EICHD7_WORD4	2D0h

Function

The fifth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B12_15DATA_MASK correspond to EICHD7_WORD4 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B12_15DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B12_15DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD7_WORD4 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD7_WORD4 on the read data bus is inverted.

50.5.38 Error Injection Channel Descriptor 8, Word1 (EICHD8_WORD1)

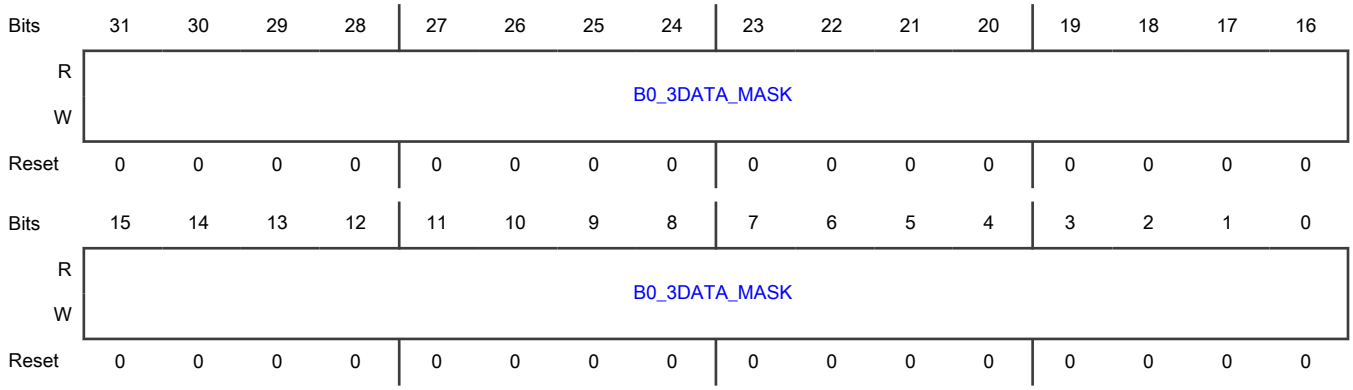
Offset

Register	Offset
EICHD8_WORD1	304h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD8_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD8_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD8_WORD1 on the read data bus is inverted.

50.5.39 Error Injection Channel Descriptor 8, Word2 (EICHD8_WORD2)

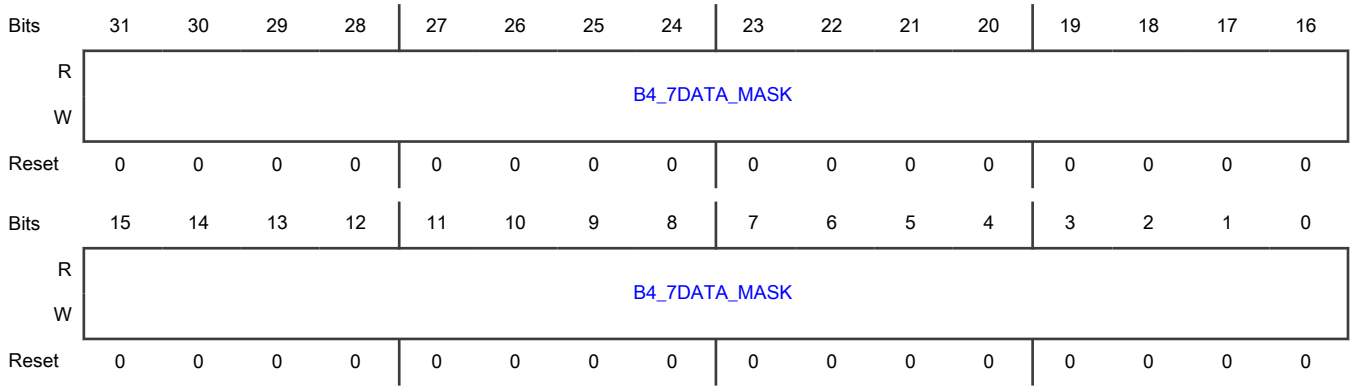
Offset

Register	Offset
EICHD8_WORD2	308h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to EICHD8_WORD2 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD8_WORD2 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD8_WORD2 on the read data bus is inverted.

50.5.40 Error Injection Channel Descriptor 8, Word3 (EICHD8_WORD3)

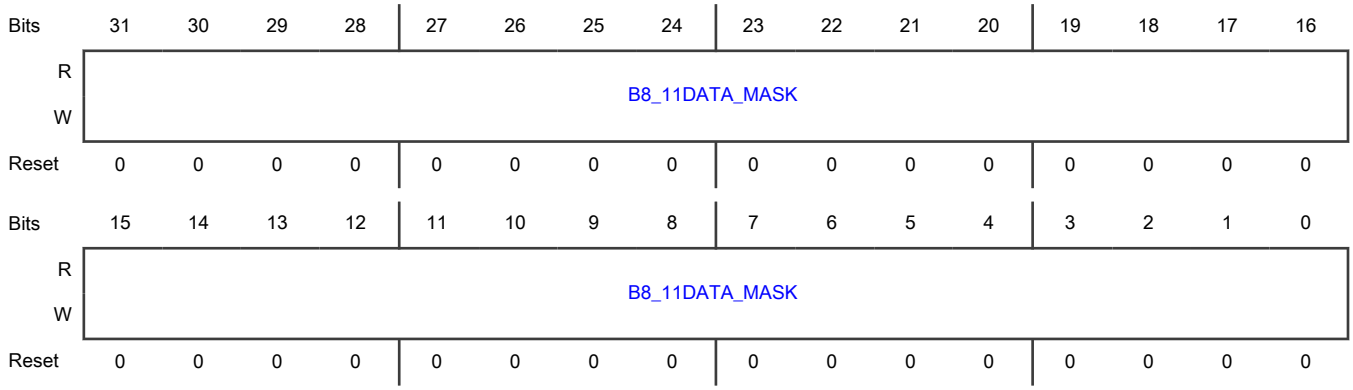
Offset

Register	Offset
EICHD8_WORD3	30Ch

Function

The fourth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B8_11DATA_MASK correspond to EICHD8_WORD3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B8_11DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B8_11DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD8_WORD3 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD8_WORD3 on the read data bus is inverted.

50.5.41 Error Injection Channel Descriptor 8, Word4 (EICHD8_WORD4)

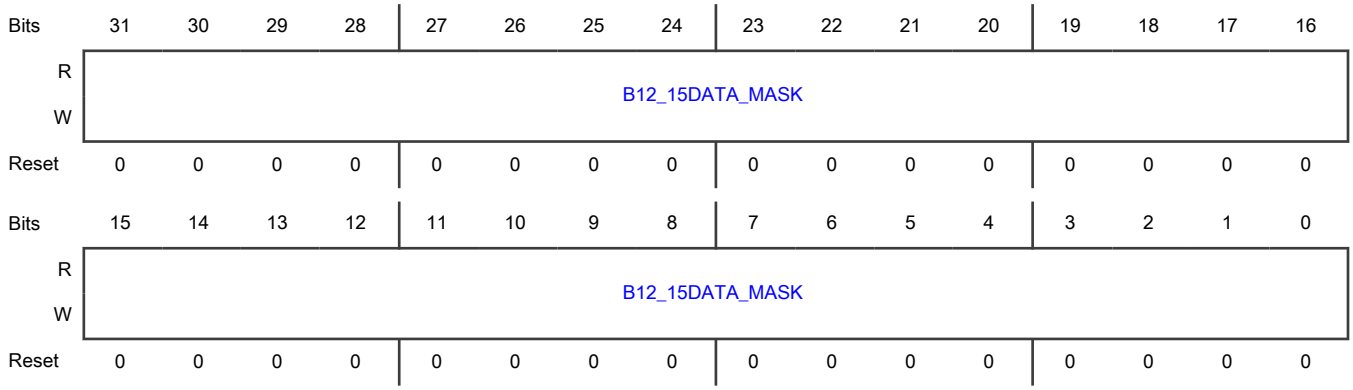
Offset

Register	Offset
EICHD8_WORD4	310h

Function

The fifth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B12_15DATA_MASK correspond to EICHD8_WORD4 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B12_15DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B12_15DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD8_WORD4 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD8_WORD4 on the read data bus is inverted.

50.5.42 Error Injection Channel Descriptor 9, Word1 (EICHD9_WORD1)

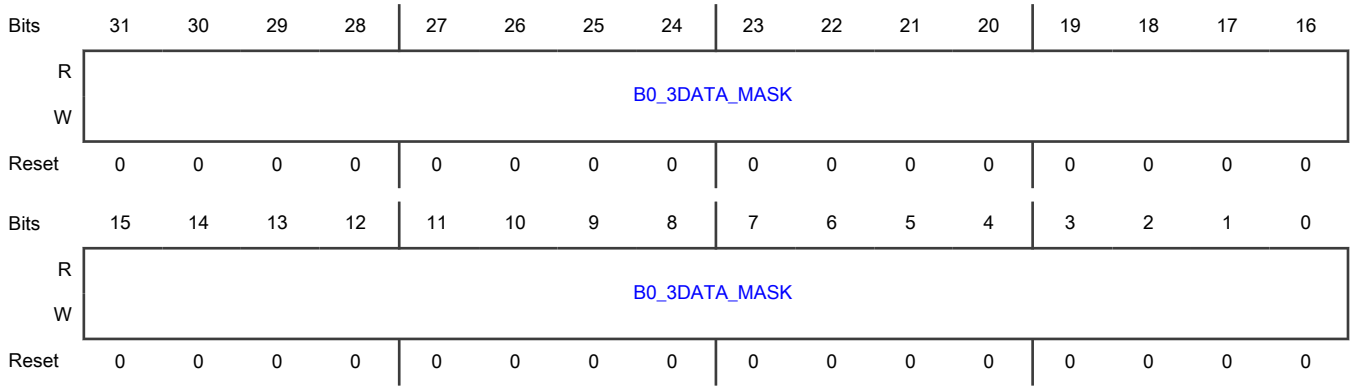
Offset

Register	Offset
EICHD9_WORD1	344h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD9_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD9_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD9_WORD1 on the read data bus is inverted.

50.5.43 Error Injection Channel Descriptor 9, Word2 (EICHD9_WORD2)

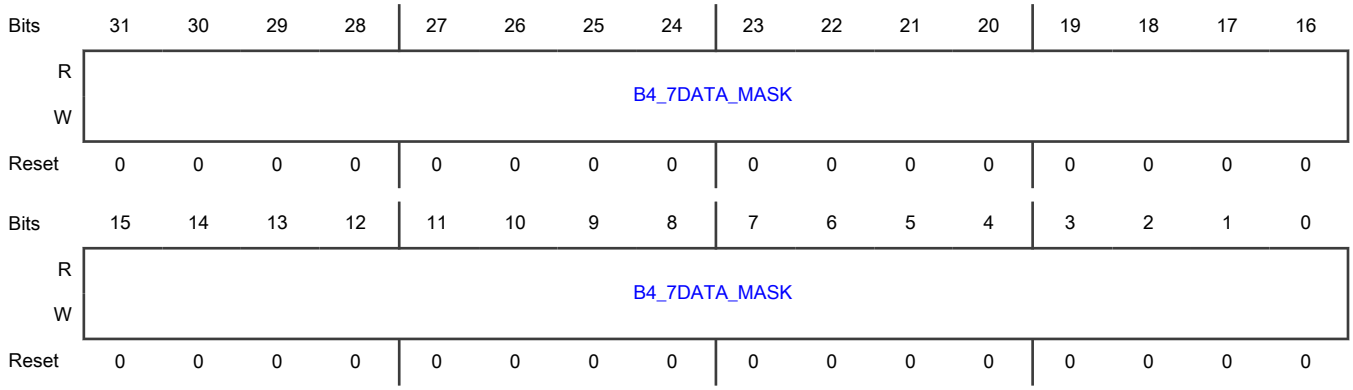
Offset

Register	Offset
EICHD9_WORD2	348h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to EICHD9_WORD2 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD9_WORD2 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD9_WORD2 on the read data bus is inverted.

50.5.44 Error Injection Channel Descriptor 9, Word3 (EICHD9_WORD3)

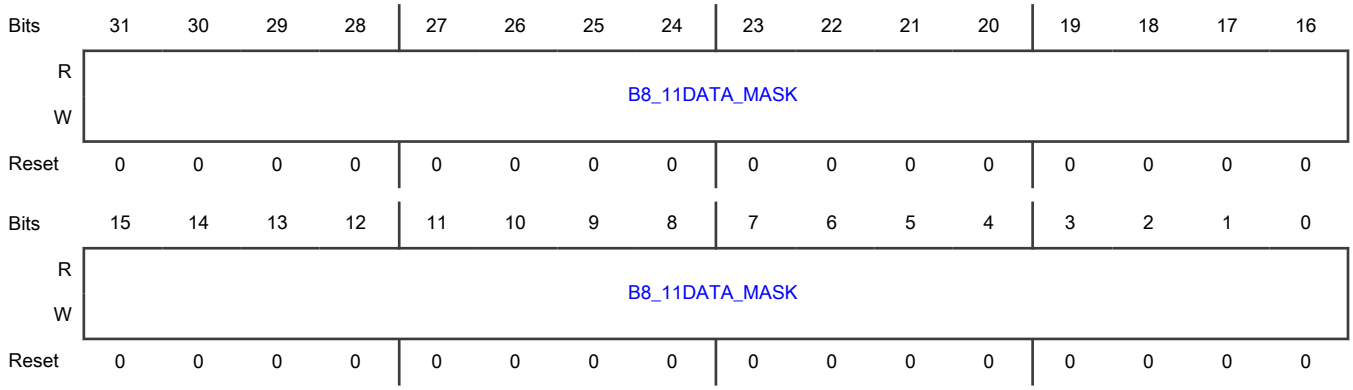
Offset

Register	Offset
EICHD9_WORD3	34Ch

Function

The fourth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B8_11DATA_MASK correspond to EICHD9_WORD3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B8_11DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B8_11DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD9_WORD3 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD9_WORD3 on the read data bus is inverted.

50.5.45 Error Injection Channel Descriptor 9, Word4 (EICHD9_WORD4)

Offset

Register	Offset
EICHD9_WORD4	350h

Function

The fifth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B12_15DATA_MASK correspond to EICHD9_WORD4 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	B12_15DATA_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	B12_15DATA_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 B12_15DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B12_15DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD9_WORD4 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD9_WORD4 on the read data bus is inverted.

50.5.46 Error Injection Channel Descriptor n, Word0 (EICHD10_WORD0 - EICHD15_WORD0)

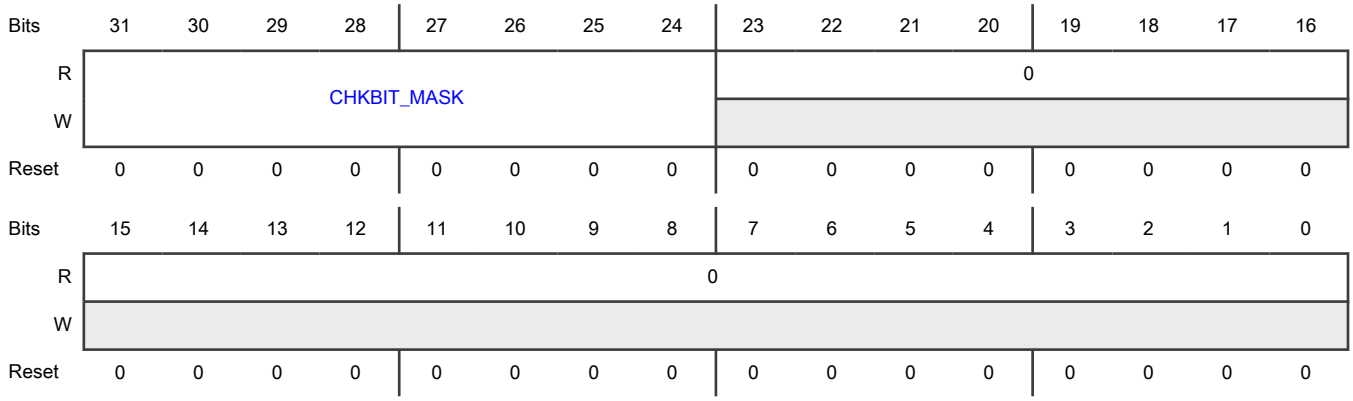
Offset

Register	Offset
EICHD10_WORD0	380h
EICHD11_WORD0	3C0h
EICHD12_WORD0	400h
EICHD13_WORD0	440h
EICHD14_WORD0	480h
EICHD15_WORD0	4C0h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-24 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[7:0] (8 bits wide), CHKBIT_MASK[7] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
23-0 —	Reserved

50.5.47 Error Injection Channel Descriptor 10, Word1 (EICHD10_WORD1)

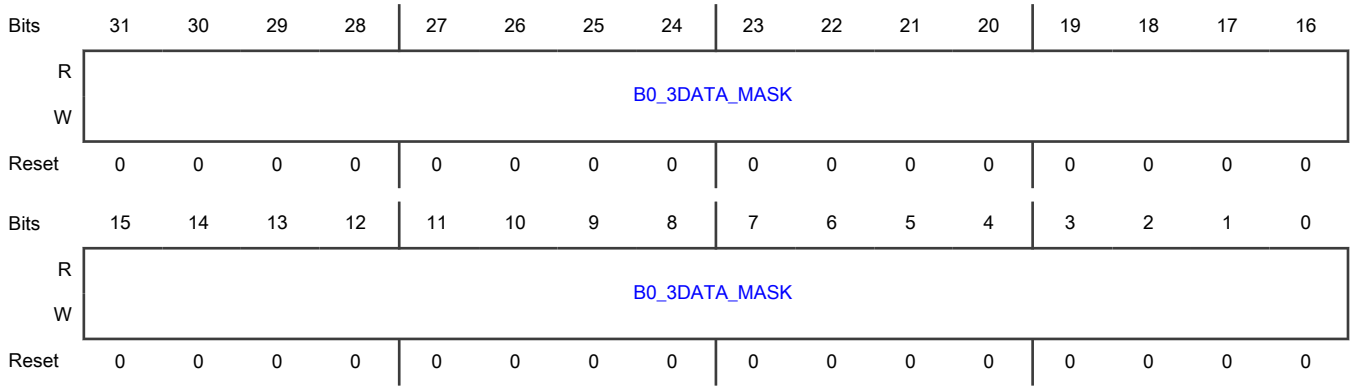
Offset

Register	Offset
EICHD10_WORD1	384h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD10_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD10_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD10_WORD1 on the read data bus is inverted.

50.5.48 Error Injection Channel Descriptor 10, Word2 (EICHD10_WORD2)

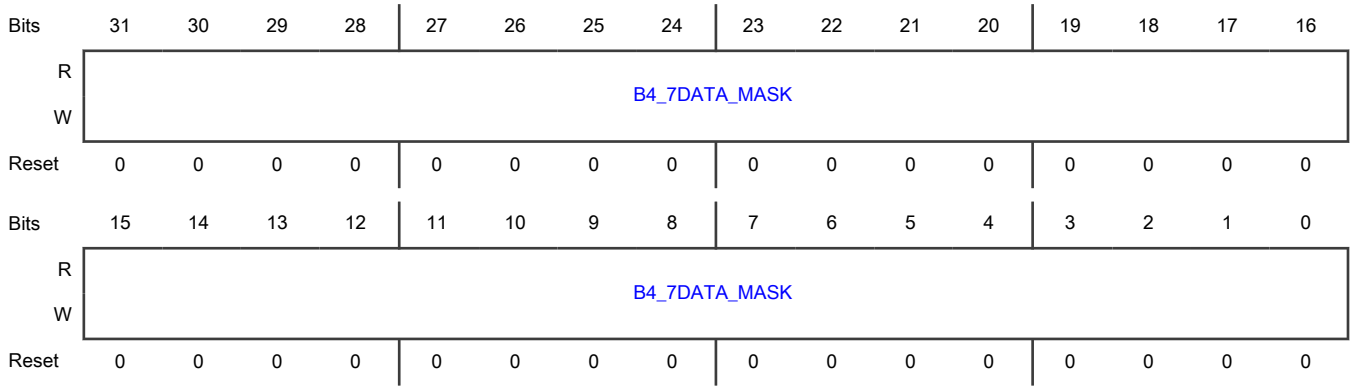
Offset

Register	Offset
EICHD10_WORD2	388h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to EICHD10_WORD2 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD10_WORD2 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD10_WORD2 on the read data bus is inverted.

50.5.49 Error Injection Channel Descriptor 11, Word1 (EICHD11_WORD1)

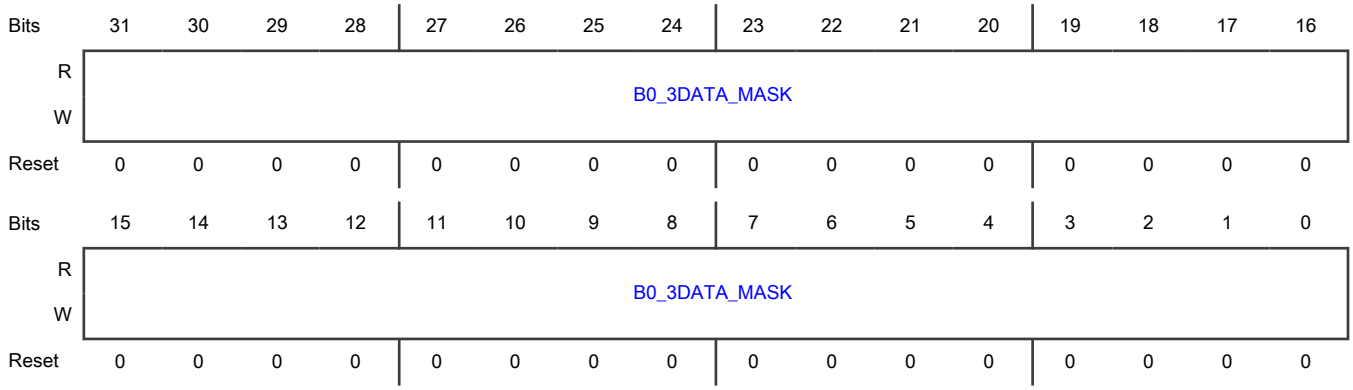
Offset

Register	Offset
EICHD11_WORD1	3C4h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD11_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD11_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD11_WORD1 on the read data bus is inverted.

50.5.50 Error Injection Channel Descriptor 12, Word1 (EICHD12_WORD1)

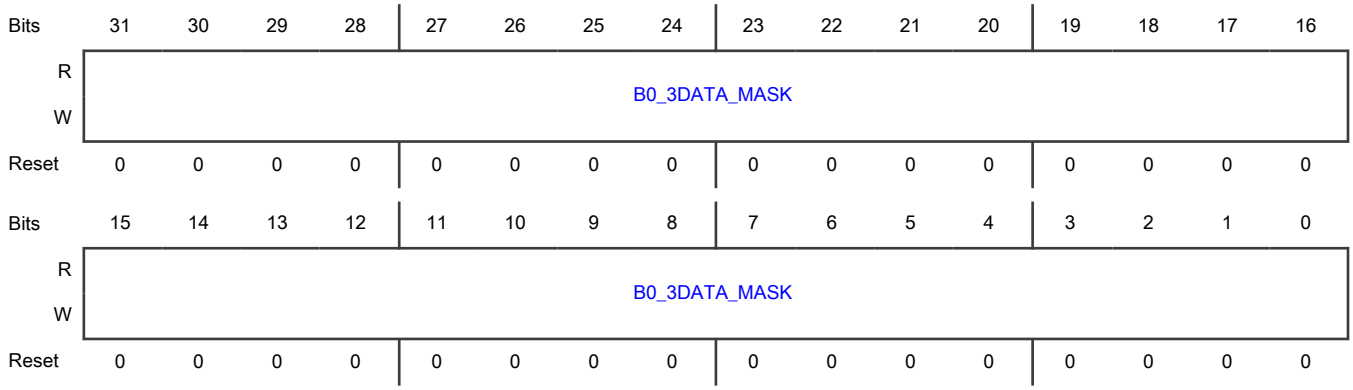
Offset

Register	Offset
EICHD12_WORD1	404h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD12_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD12_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD12_WORD1 on the read data bus is inverted.

50.5.51 Error Injection Channel Descriptor 13, Word1 (EICHD13_WORD1)

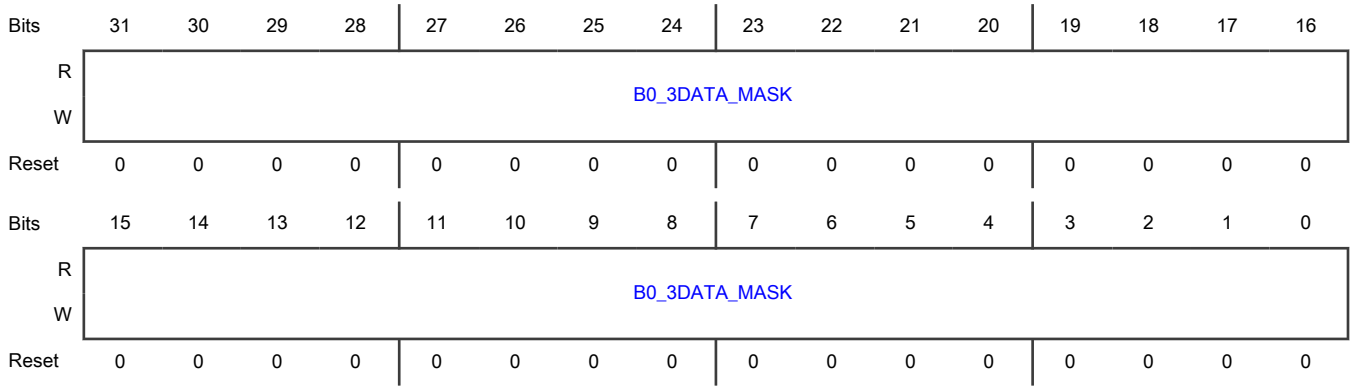
Offset

Register	Offset
EICHD13_WORD1	444h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD13_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD13_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD13_WORD1 on the read data bus is inverted.

50.5.52 Error Injection Channel Descriptor 13, Word2 (EICHD13_WORD2)

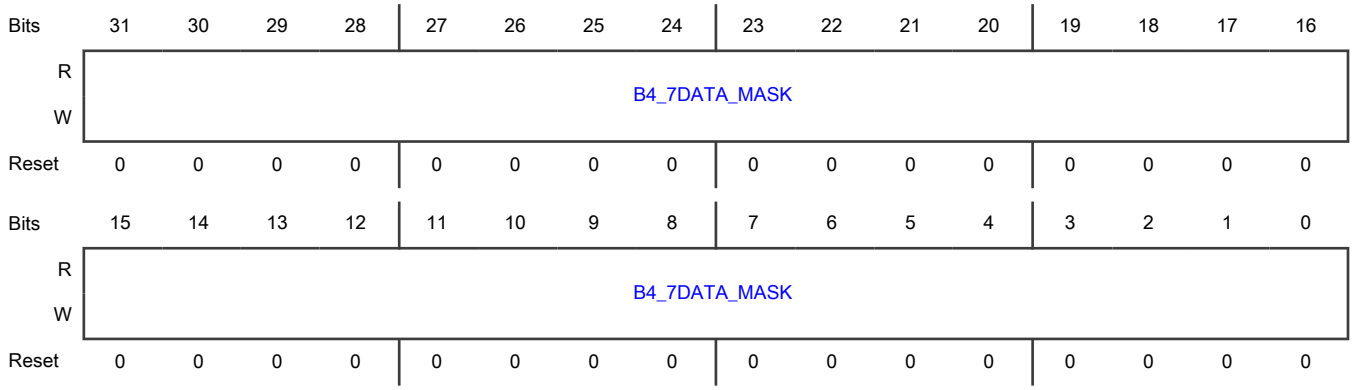
Offset

Register	Offset
EICHD13_WORD2	448h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to EICHD13_WORD2 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD13_WORD2 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD13_WORD2 on the read data bus is inverted.

50.5.53 Error Injection Channel Descriptor 14, Word1 (EICHD14_WORD1)

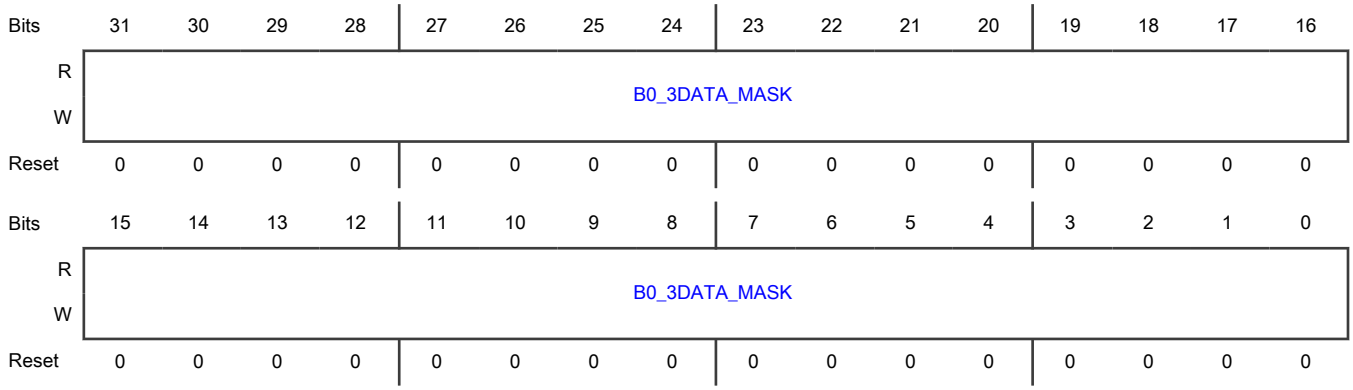
Offset

Register	Offset
EICHD14_WORD1	484h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD14_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD14_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD14_WORD1 on the read data bus is inverted.

50.5.54 Error Injection Channel Descriptor 15, Word1 (EICHD15_WORD1)

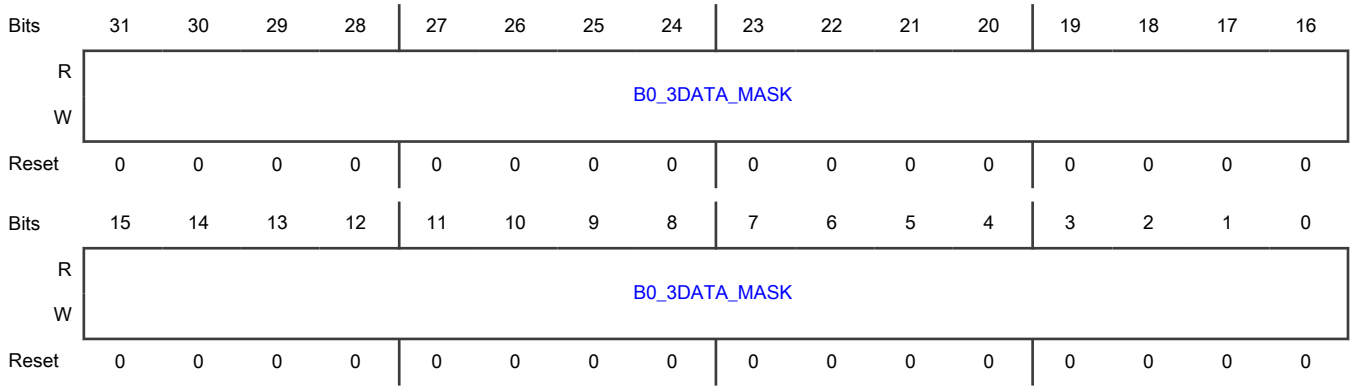
Offset

Register	Offset
EICHD15_WORD1	4C4h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD15_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD15_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD15_WORD1 on the read data bus is inverted.

50.5.55 Error Injection Channel Descriptor 16, Word1 (EICHD16_WORD1)

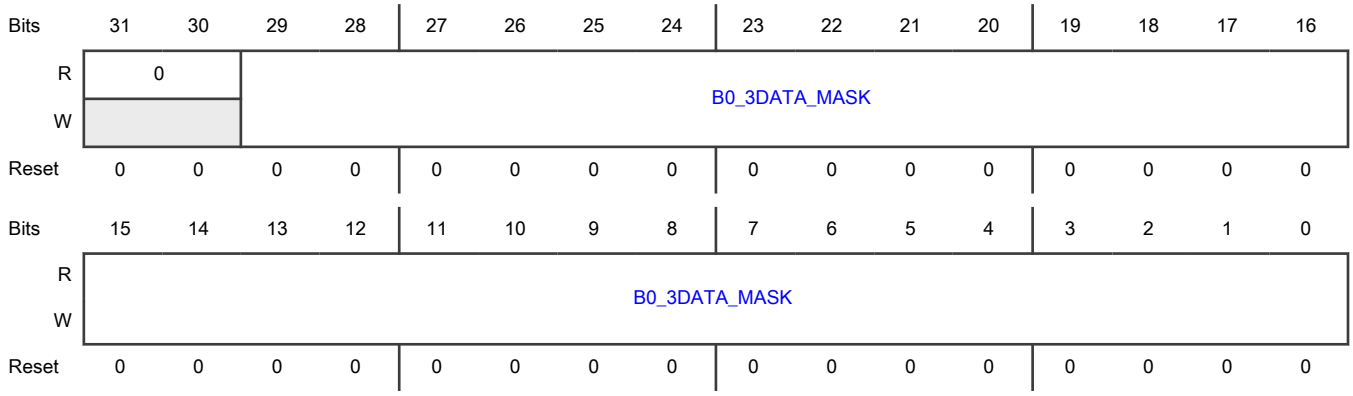
Offset

Register	Offset
EICHD16_WORD1	504h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD16_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-30 —	Reserved
29-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD16_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD16_WORD1 on the read data bus is inverted.

50.6 EIM_1 register descriptions

The EIM provides a programming model mapped to an on-platform peripheral slot.

Programming model access

All system bus controllers can access the programming model:

- Only in supervisor mode
- Using only 32-bit (word) accesses

Any of the following attempted references to the programming model generates an error termination:

- In user mode
- Using non-32-bit access sizes
- To undefined (reserved) addresses

Attempted updates to the programming model while the EIM is in the midst of an operation result in non-deterministic behavior.

Error injection channel descriptor: function and structure

Each error injection channel descriptor:

- Specifies a mask that defines which bits of the read data and/or checkbit bus from target RAM are inverted on a read access.
- Consists of a 288-bit (36-byte) structure, composed of nine 32-bit words, in the EIM programming model. Unused words are not documented.
  - Word0 (EICHD*n*_WORD0), if present, defines the checkbit mask.
  - Word1 (EICHD*n*_WORD1) and additional words, if present, define the data mask. Word registers subsequent to Word1 are present only when required by the total width of the channel's data mask. Error injection channel descriptor: DATA_MASK details.

The multiple channel descriptors are organized sequentially.

Error injection channel descriptor: DATA_MASK details

For each channel: The following tables show the distribution of DATA_MASK's bits across the WORD registers. The first table shows the total width of DATA_MASK and the distribution of its bits across WORD1, WORD2, and WORD3. The second table shows the distribution of DATA_MASK's bits across WORD4 and subsequent registers.

Table 285. Error injection channel descriptor: DATA_MASK details

Channel	DATA_MASK total width (bits)	Specific bits of DATA_MASK in		
		WORD1	WORD2	WORD3
0	44	43-32	31-0	—
1	128	127-96	95-64	63-32
2	104	103-96	95-64	63-32
3	128	127-96	95-64	63-32
4	128	127-96	95-64	63-32
5	44	43-32	31-0	—
6	128	127-96	95-64	63-32
7	104	103-96	95-64	63-32
8	128	127-96	95-64	63-32
9	128	127-96	95-64	63-32
10	64	63-32	31-0	—
11	32	31-0	—	—
12	32	31-0	—	—
13	64	63-32	31-0	—
14	32	31-0	—	—
15	32	31-0	—	—
16	30	29-0	—	—

Table 286. DATA_MASK bit: Channel-word mapping

Channel	Specific bits of DATA_MASK in				
	WORD4	WORD5	WORD6	WORD7	WORD8
1	31-0	—	—	—	—
2	31-0	—	—	—	—
3	31-0	—	—	—	—
4	31-0	—	—	—	—
6	31-0	—	—	—	—
7	31-0	—	—	—	—
8	31-0	—	—	—	—
9	31-0	—	—	—	—

50.6.1 EIM_1 memory map

EIM_1 base address: 4051_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Error Injection Module Configuration Register (EIMCR)	32	RW	0000_0000h
4h	Error Injection Channel Enable register (EICHEN)	32	RW	0000_0000h
100h	Error Injection Channel Descriptor 0, Word0 (EICHD0_WORD0)	32	RW	0000_0000h
104h	Error Injection Channel Descriptor 0, Word1 (EICHD0_WORD1)	32	RW	0000_0000h
108h	Error Injection Channel Descriptor 0, Word2 (EICHD0_WORD2)	32	RW	0000_0000h
140h	Error Injection Channel Descriptor 1, Word0 (EICHD1_WORD0)	32	RW	0000_0000h
144h	Error Injection Channel Descriptor 1, Word1 (EICHD1_WORD1)	32	RW	0000_0000h
148h	Error Injection Channel Descriptor 1, Word2 (EICHD1_WORD2)	32	RW	0000_0000h
14Ch	Error Injection Channel Descriptor 1, Word3 (EICHD1_WORD3)	32	RW	0000_0000h
150h	Error Injection Channel Descriptor 1, Word4 (EICHD1_WORD4)	32	RW	0000_0000h
180h	Error Injection Channel Descriptor 2, Word0 (EICHD2_WORD0)	32	RW	0000_0000h
184h	Error Injection Channel Descriptor 2, Word1 (EICHD2_WORD1)	32	RW	0000_0000h
188h	Error Injection Channel Descriptor 2, Word2 (EICHD2_WORD2)	32	RW	0000_0000h
18Ch	Error Injection Channel Descriptor 2, Word3 (EICHD2_WORD3)	32	RW	0000_0000h
190h	Error Injection Channel Descriptor 2, Word4 (EICHD2_WORD4)	32	RW	0000_0000h
1C0h	Error Injection Channel Descriptor 3, Word0 (EICHD3_WORD0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1C4h	Error Injection Channel Descriptor 3, Word1 (EICHD3_WORD1)	32	RW	0000_0000h
1C8h	Error Injection Channel Descriptor 3, Word2 (EICHD3_WORD2)	32	RW	0000_0000h
1CCh	Error Injection Channel Descriptor 3, Word3 (EICHD3_WORD3)	32	RW	0000_0000h
1D0h	Error Injection Channel Descriptor 3, Word4 (EICHD3_WORD4)	32	RW	0000_0000h
200h	Error Injection Channel Descriptor 4, Word0 (EICHD4_WORD0)	32	RW	0000_0000h
204h	Error Injection Channel Descriptor 4, Word1 (EICHD4_WORD1)	32	RW	0000_0000h
208h	Error Injection Channel Descriptor 4, Word2 (EICHD4_WORD2)	32	RW	0000_0000h
20Ch	Error Injection Channel Descriptor 4, Word3 (EICHD4_WORD3)	32	RW	0000_0000h
210h	Error Injection Channel Descriptor 4, Word4 (EICHD4_WORD4)	32	RW	0000_0000h
240h	Error Injection Channel Descriptor 5, Word0 (EICHD5_WORD0)	32	RW	0000_0000h
244h	Error Injection Channel Descriptor 5, Word1 (EICHD5_WORD1)	32	RW	0000_0000h
248h	Error Injection Channel Descriptor 5, Word2 (EICHD5_WORD2)	32	RW	0000_0000h
280h	Error Injection Channel Descriptor 6, Word0 (EICHD6_WORD0)	32	RW	0000_0000h
284h	Error Injection Channel Descriptor 6, Word1 (EICHD6_WORD1)	32	RW	0000_0000h
288h	Error Injection Channel Descriptor 6, Word2 (EICHD6_WORD2)	32	RW	0000_0000h
28Ch	Error Injection Channel Descriptor 6, Word3 (EICHD6_WORD3)	32	RW	0000_0000h
290h	Error Injection Channel Descriptor 6, Word4 (EICHD6_WORD4)	32	RW	0000_0000h
2C0h	Error Injection Channel Descriptor 7, Word0 (EICHD7_WORD0)	32	RW	0000_0000h
2C4h	Error Injection Channel Descriptor 7, Word1 (EICHD7_WORD1)	32	RW	0000_0000h
2C8h	Error Injection Channel Descriptor 7, Word2 (EICHD7_WORD2)	32	RW	0000_0000h
2CCh	Error Injection Channel Descriptor 7, Word3 (EICHD7_WORD3)	32	RW	0000_0000h
2D0h	Error Injection Channel Descriptor 7, Word4 (EICHD7_WORD4)	32	RW	0000_0000h
300h	Error Injection Channel Descriptor 8, Word0 (EICHD8_WORD0)	32	RW	0000_0000h
304h	Error Injection Channel Descriptor 8, Word1 (EICHD8_WORD1)	32	RW	0000_0000h
308h	Error Injection Channel Descriptor 8, Word2 (EICHD8_WORD2)	32	RW	0000_0000h
30Ch	Error Injection Channel Descriptor 8, Word3 (EICHD8_WORD3)	32	RW	0000_0000h
310h	Error Injection Channel Descriptor 8, Word4 (EICHD8_WORD4)	32	RW	0000_0000h
340h	Error Injection Channel Descriptor 9, Word0 (EICHD9_WORD0)	32	RW	0000_0000h
344h	Error Injection Channel Descriptor 9, Word1 (EICHD9_WORD1)	32	RW	0000_0000h
348h	Error Injection Channel Descriptor 9, Word2 (EICHD9_WORD2)	32	RW	0000_0000h
34Ch	Error Injection Channel Descriptor 9, Word3 (EICHD9_WORD3)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
350h	Error Injection Channel Descriptor 9, Word4 (EICHD9_WORD4)	32	RW	0000_0000h
380h	Error Injection Channel Descriptor 10, Word0 (EICHD10_WORD0)	32	RW	0000_0000h
384h	Error Injection Channel Descriptor 10, Word1 (EICHD10_WORD1)	32	RW	0000_0000h
388h	Error Injection Channel Descriptor 10, Word2 (EICHD10_WORD2)	32	RW	0000_0000h
3C0h	Error Injection Channel Descriptor 11, Word0 (EICHD11_WORD0)	32	RW	0000_0000h
3C4h	Error Injection Channel Descriptor 11, Word1 (EICHD11_WORD1)	32	RW	0000_0000h
400h	Error Injection Channel Descriptor 12, Word0 (EICHD12_WORD0)	32	RW	0000_0000h
404h	Error Injection Channel Descriptor 12, Word1 (EICHD12_WORD1)	32	RW	0000_0000h
440h	Error Injection Channel Descriptor 13, Word0 (EICHD13_WORD0)	32	RW	0000_0000h
444h	Error Injection Channel Descriptor 13, Word1 (EICHD13_WORD1)	32	RW	0000_0000h
448h	Error Injection Channel Descriptor 13, Word2 (EICHD13_WORD2)	32	RW	0000_0000h
480h	Error Injection Channel Descriptor 14, Word0 (EICHD14_WORD0)	32	RW	0000_0000h
484h	Error Injection Channel Descriptor 14, Word1 (EICHD14_WORD1)	32	RW	0000_0000h
4C0h	Error Injection Channel Descriptor 15, Word0 (EICHD15_WORD0)	32	RW	0000_0000h
4C4h	Error Injection Channel Descriptor 15, Word1 (EICHD15_WORD1)	32	RW	0000_0000h
504h	Error Injection Channel Descriptor 16, Word1 (EICHD16_WORD1)	32	RW	0000_0000h

50.6.2 Error Injection Module Configuration Register (EIMCR)

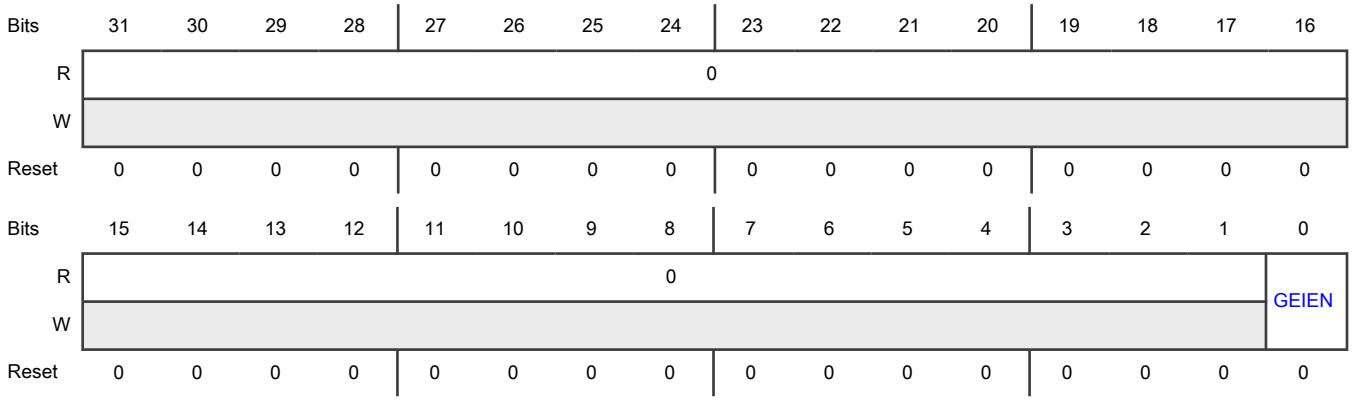
Offset

Register	Offset
EIMCR	0h

Function

The EIM Configuration Register is used to globally enable/disable the error injection function.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 GEIEN	Global Error Injection Enable This bit globally enables or disables the error injection function of the EIM. This field is initialized by hardware reset. 0b - Disabled 1b - Enabled

50.6.3 Error Injection Channel Enable register (EICHEN)

Offset

Register	Offset
EICHEN	4h

Function

Each field of the Error Injection Channel Enable register (EICHEN) is used to enable or disable the corresponding error injection channel.

NOTE

To enable an error injection channel, the Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	EICH0EN	EICH1EN	EICH2EN	EICH3EN	EICH4EN	EICH5EN	EICH6EN	EICH7EN	EICH8EN	EICH9EN	EICH10EN	EICH11EN	EICH12EN	EICH13EN	EICH14EN	EICH15EN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EICH16EN	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 EICH0EN	Error Injection Channel 0 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 0 1b - Error injection is enabled on Error Injection Channel 0
30 EICH1EN	Error Injection Channel 1 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 1 1b - Error injection is enabled on Error Injection Channel 1
29 EICH2EN	Error Injection Channel 2 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 2 1b - Error injection is enabled on Error Injection Channel 2
28 EICH3EN	Error Injection Channel 3 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 3 1b - Error injection is enabled on Error Injection Channel 3
27 EICH4EN	Error Injection Channel 4 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 4 1b - Error injection is enabled on Error Injection Channel 4
26 EICH5EN	Error Injection Channel 5 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 5 1b - Error injection is enabled on Error Injection Channel 5
25 EICH6EN	Error Injection Channel 6 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 6 1b - Error injection is enabled on Error Injection Channel 6
24 EICH7EN	Error Injection Channel 7 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 7 1b - Error injection is enabled on Error Injection Channel 7
23 EICH8EN	Error Injection Channel 8 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 8 1b - Error injection is enabled on Error Injection Channel 8
22 EICH9EN	Error Injection Channel 9 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 9 1b - Error injection is enabled on Error Injection Channel 9

Table continues on the next page...

Table continued from the previous page...

Field	Function
21 EICH10EN	Error Injection Channel 10 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 10 1b - Error injection is enabled on Error Injection Channel 10
20 EICH11EN	Error Injection Channel 11 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 11 1b - Error injection is enabled on Error Injection Channel 11
19 EICH12EN	Error Injection Channel 12 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 12 1b - Error injection is enabled on Error Injection Channel 12
18 EICH13EN	Error Injection Channel 13 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Error injection is disabled on Error Injection Channel 13 1b - Error injection is enabled on Error Injection Channel 13
17 EICH14EN	Error Injection Channel 14 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 14 1b - Error injection is enabled on Error Injection Channel 14
16 EICH15EN	Error Injection Channel 15 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 15 1b - Error injection is enabled on Error Injection Channel 15
15 EICH16EN	Error Injection Channel 16 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 16 1b - Error injection is enabled on Error Injection Channel 16
14 —	Reserved
13 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
12 —	Reserved
11 —	Reserved
10 —	Reserved
9 —	Reserved
8 —	Reserved
7 —	Reserved
6 —	Reserved
5 —	Reserved
4 —	Reserved
3 —	Reserved
2 —	Reserved
1 —	Reserved
0 —	Reserved

50.6.4 Error Injection Channel Descriptor 0, Word0 (EICHD0_WORD0)

Offset

Register	Offset
EICHD0_WORD0	100h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CHKBIT_MASK														0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-18 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[13:0] (14 bits wide), CHKBIT_MASK[13] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
17-0 —	Reserved

50.6.5 Error Injection Channel Descriptor 0, Word1 (EICHD0_WORD1)

Offset

Register	Offset
EICHD0_WORD1	104h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD0_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				B0_3DATA_MASK											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-12 —	Reserved
11-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD0_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD0_WORD1 on the read data bus is inverted.

50.6.6 Error Injection Channel Descriptor 0, Word2 (EICHD0_WORD2)

Offset

Register	Offset
EICHD0_WORD2	108h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to EICHD0_WORD2 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	B4_7DATA_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	B4_7DATA_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD0_WORD2 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD0_WORD2 on the read data bus is inverted.

50.6.7 Error Injection Channel Descriptor 1, Word0 (EICHD1_WORD0)

Offset

Register	Offset
EICHD1_WORD0	140h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CHKBIT_MASK															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[15:0] (16 bits wide), CHKBIT_MASK[15] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
15-0 —	Reserved

50.6.8 Error Injection Channel Descriptor 1, Word1 (EICHD1_WORD1)

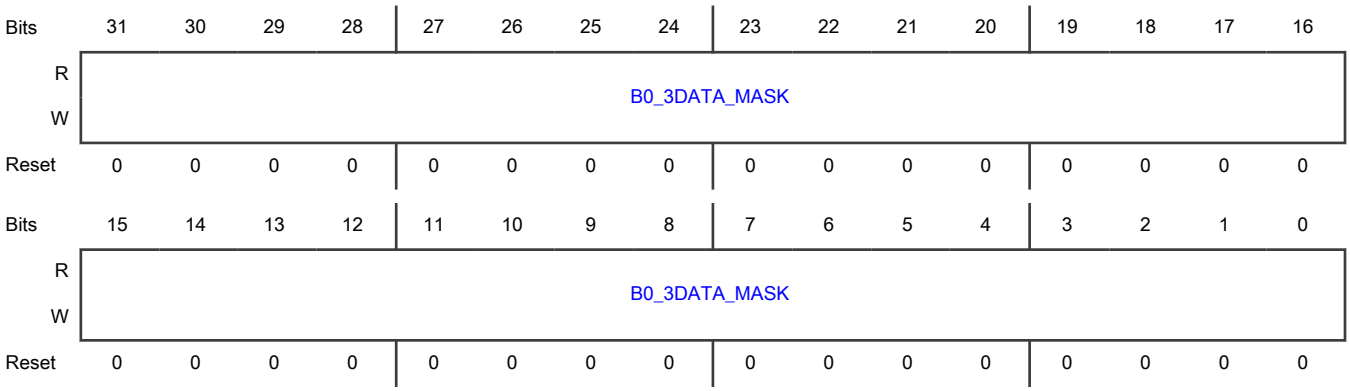
Offset

Register	Offset
EICHD1_WORD1	144h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD1_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD1_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD1_WORD1 on the read data bus is inverted.

50.6.9 Error Injection Channel Descriptor 1, Word2 (EICHD1_WORD2)

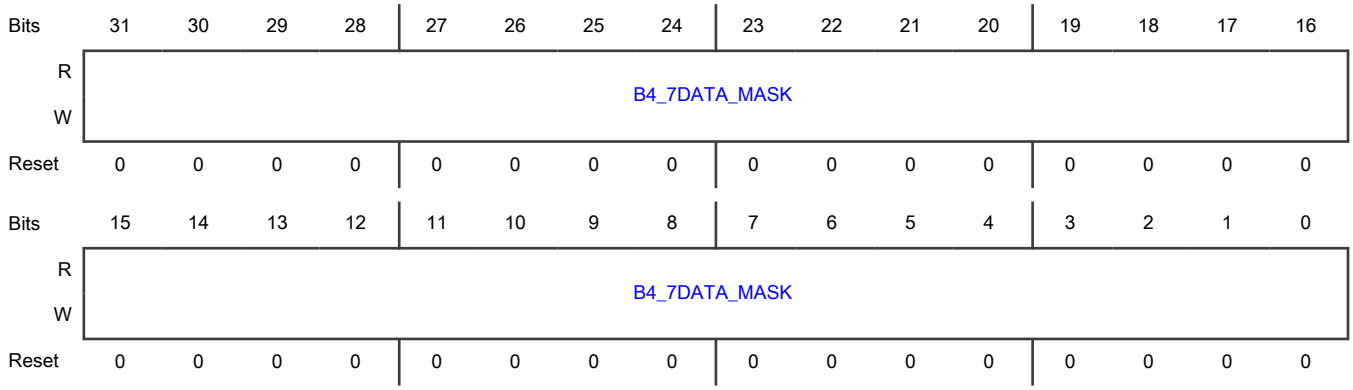
Offset

Register	Offset
EICHD1_WORD2	148h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to EICHD1_WORD2 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD1_WORD2 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD1_WORD2 on the read data bus is inverted.

50.6.10 Error Injection Channel Descriptor 1, Word3 (EICHD1_WORD3)

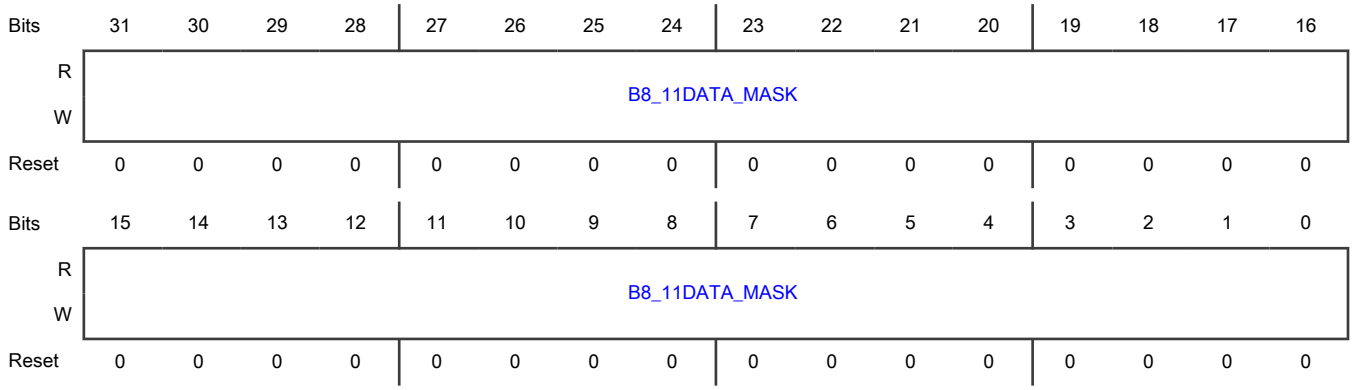
Offset

Register	Offset
EICHD1_WORD3	14Ch

Function

The fourth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B8_11DATA_MASK correspond to EICHD1_WORD3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B8_11DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B8_11DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD1_WORD3 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD1_WORD3 on the read data bus is inverted.

50.6.11 Error Injection Channel Descriptor 1, Word4 (EICHD1_WORD4)

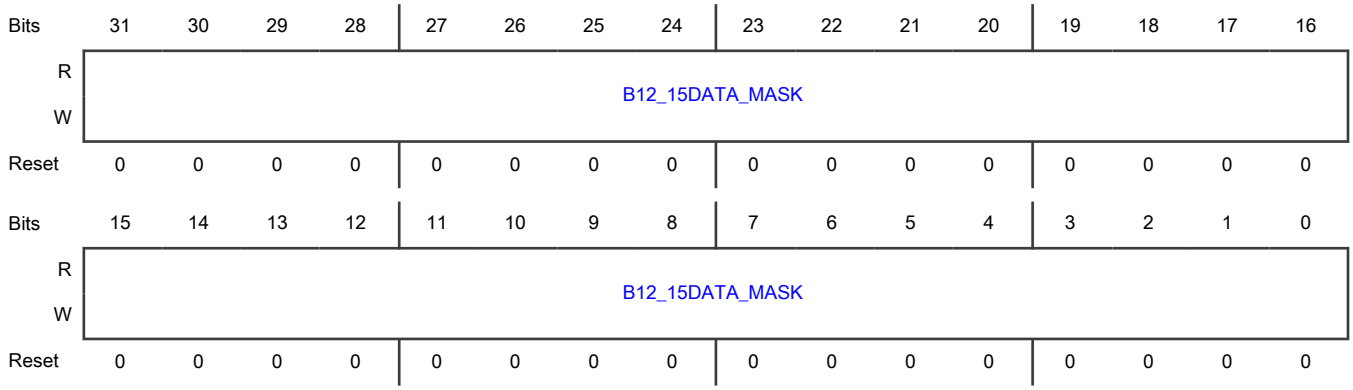
Offset

Register	Offset
EICHD1_WORD4	150h

Function

The fifth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B12_15DATA_MASK correspond to EICHD1_WORD4 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B12_15DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B12_15DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD1_WORD4 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD1_WORD4 on the read data bus is inverted.

50.6.12 Error Injection Channel Descriptor n, Word0 (EICHD2_WORD0 - EICHD4_WORD0)

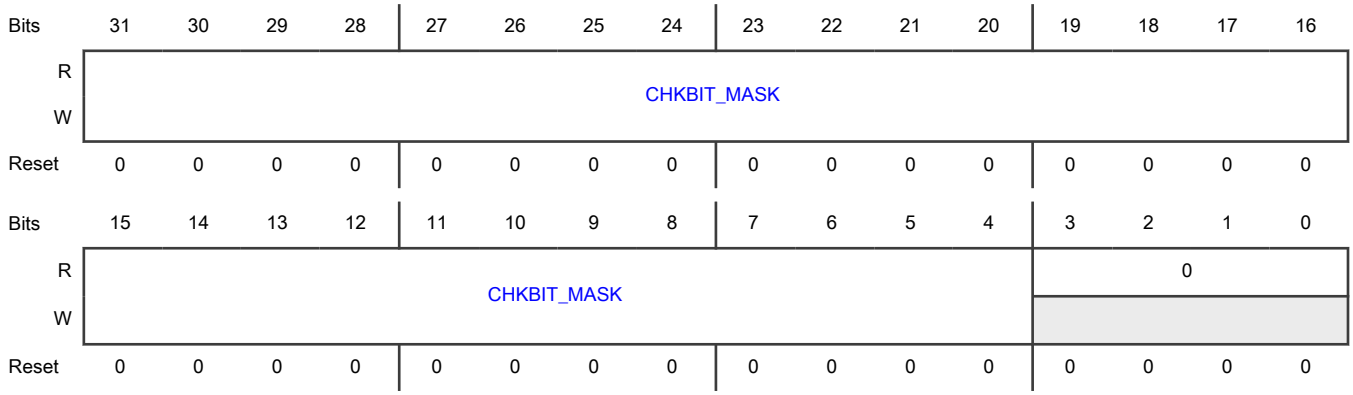
Offset

Register	Offset
EICHD2_WORD0	180h
EICHD3_WORD0	1C0h
EICHD4_WORD0	200h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-4 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[27:0] (28 bits wide), CHKBIT_MASK[27] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
3-0 —	Reserved

50.6.13 Error Injection Channel Descriptor 2, Word1 (EICHD2_WORD1)

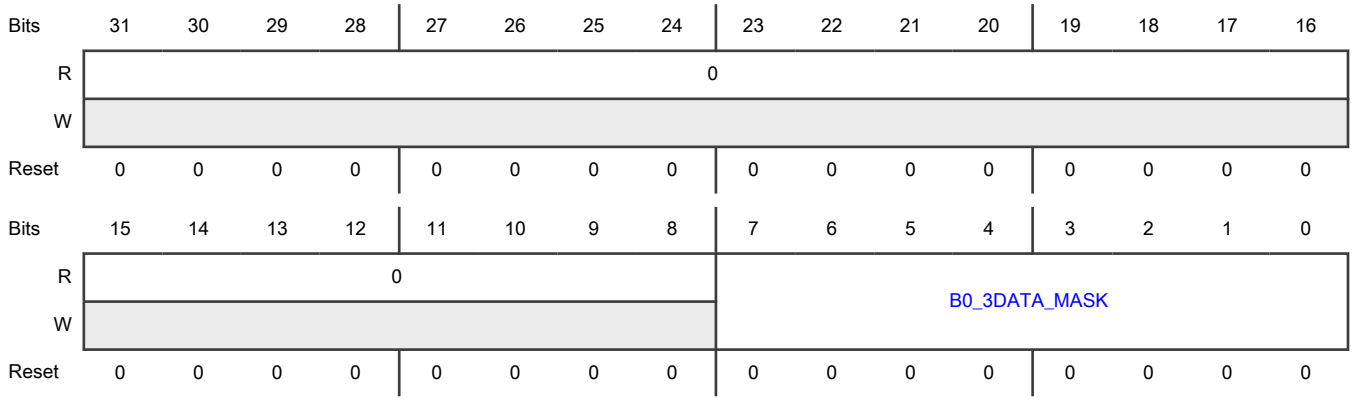
Offset

Register	Offset
EICHD2_WORD1	184h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD2_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD2_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD2_WORD1 on the read data bus is inverted.

50.6.14 Error Injection Channel Descriptor 2, Word2 (EICHD2_WORD2)

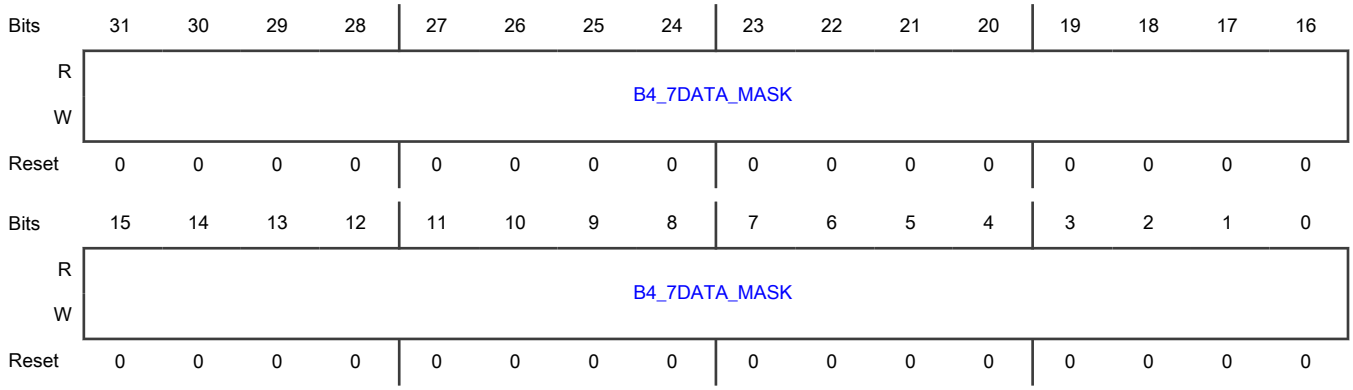
Offset

Register	Offset
EICHD2_WORD2	188h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to EICHD2_WORD2 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD2_WORD2 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD2_WORD2 on the read data bus is inverted.

50.6.15 Error Injection Channel Descriptor 2, Word3 (EICHD2_WORD3)

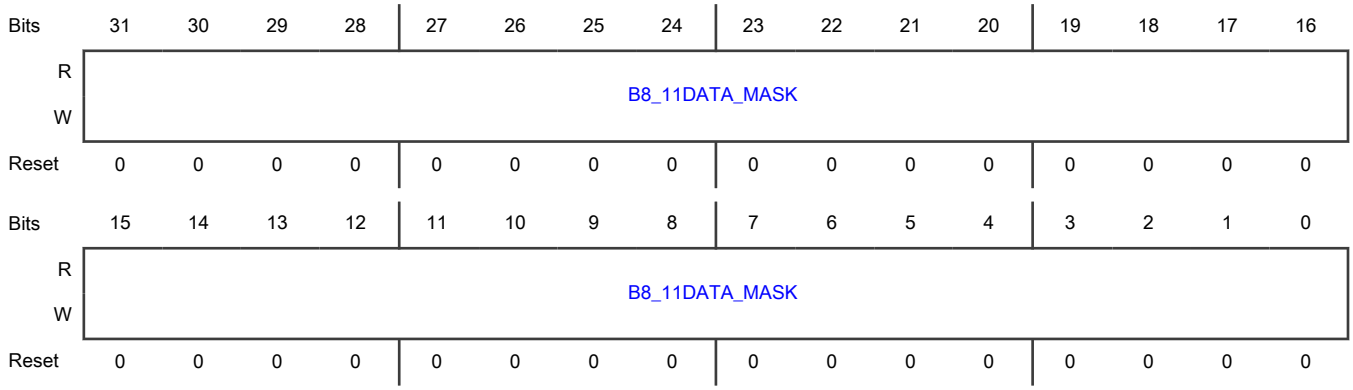
Offset

Register	Offset
EICHD2_WORD3	18Ch

Function

The fourth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B8_11DATA_MASK correspond to EICHD2_WORD3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B8_11DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B8_11DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD2_WORD3 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD2_WORD3 on the read data bus is inverted.

50.6.16 Error Injection Channel Descriptor 2, Word4 (EICHD2_WORD4)

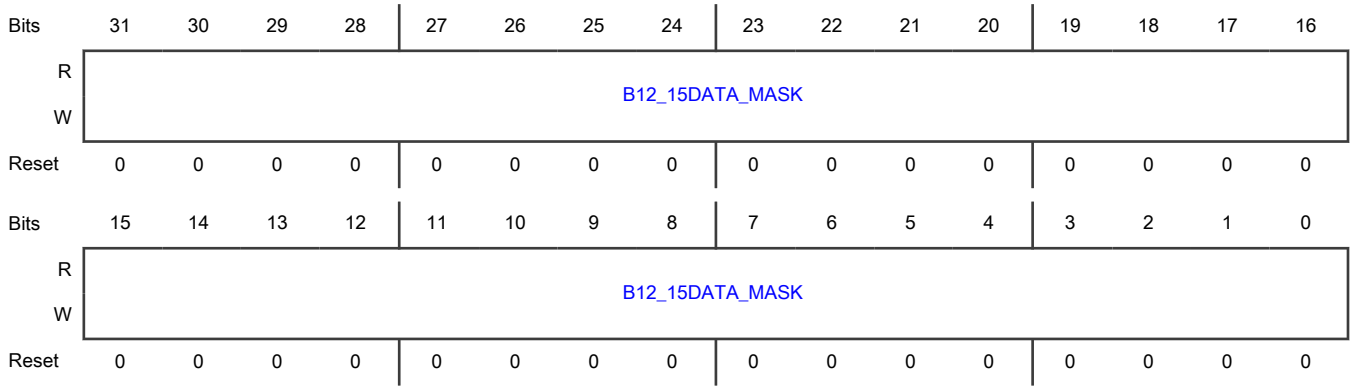
Offset

Register	Offset
EICHD2_WORD4	190h

Function

The fifth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B12_15DATA_MASK correspond to EICHD2_WORD4 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B12_15DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B12_15DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD2_WORD4 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD2_WORD4 on the read data bus is inverted.

50.6.17 Error Injection Channel Descriptor 3, Word1 (EICHD3_WORD1)

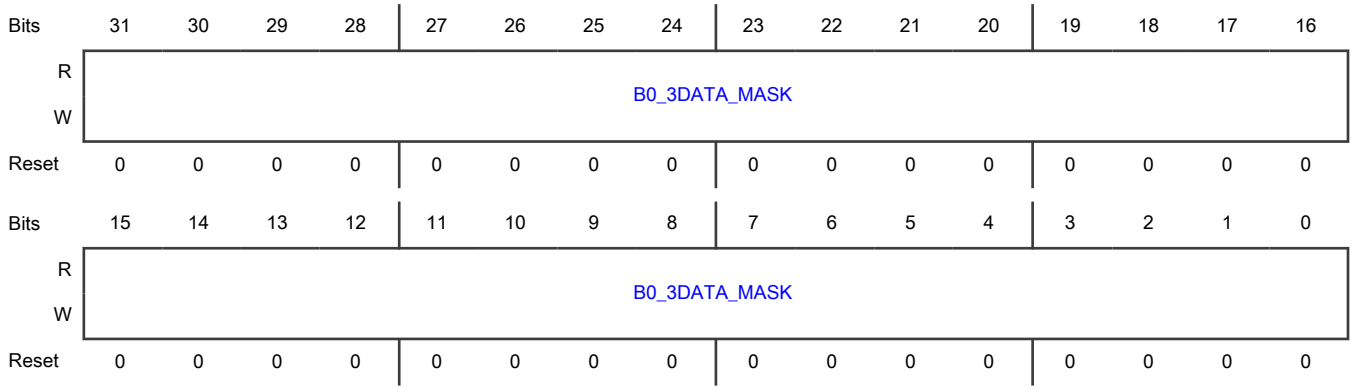
Offset

Register	Offset
EICHD3_WORD1	1C4h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD3_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD3_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD3_WORD1 on the read data bus is inverted.

50.6.18 Error Injection Channel Descriptor 3, Word2 (EICHD3_WORD2)

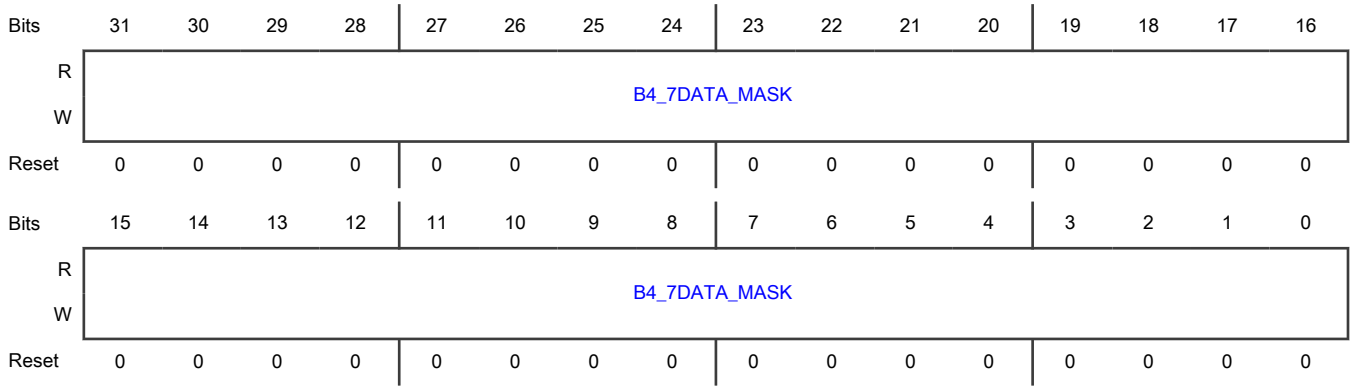
Offset

Register	Offset
EICHD3_WORD2	1C8h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to EICHD3_WORD2 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD3_WORD2 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD3_WORD2 on the read data bus is inverted.

50.6.19 Error Injection Channel Descriptor 3, Word3 (EICHD3_WORD3)

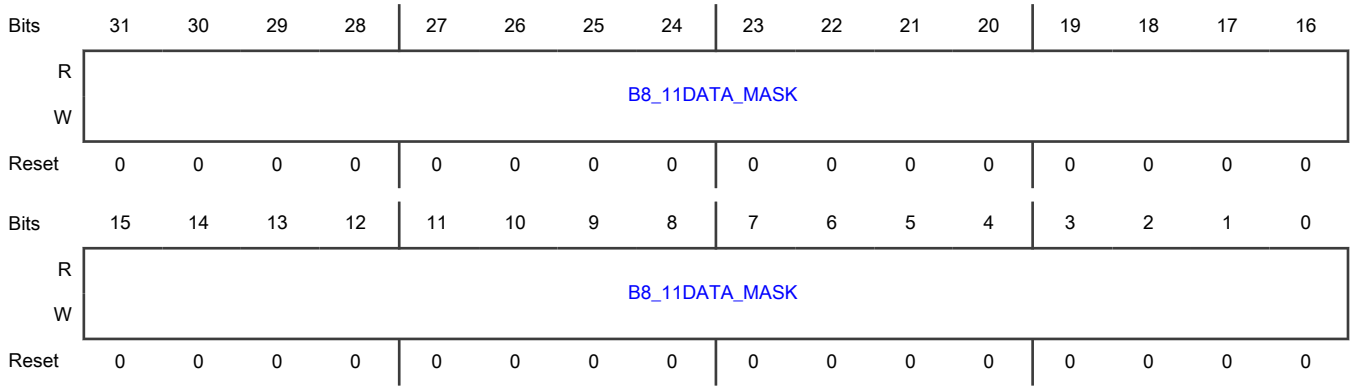
Offset

Register	Offset
EICHD3_WORD3	1CCh

Function

The fourth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B8_11DATA_MASK correspond to EICHD3_WORD3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B8_11DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B8_11DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD3_WORD3 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD3_WORD3 on the read data bus is inverted.

50.6.20 Error Injection Channel Descriptor 3, Word4 (EICHD3_WORD4)

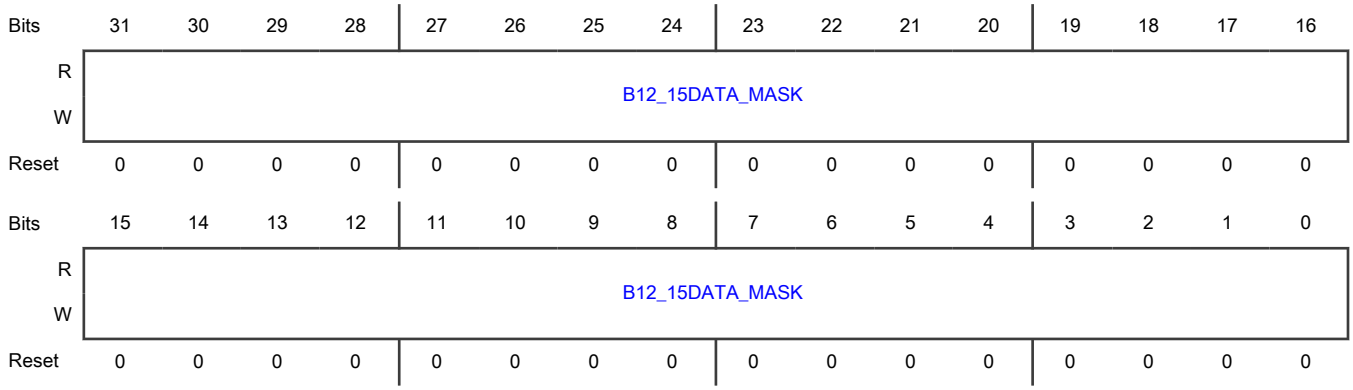
Offset

Register	Offset
EICHD3_WORD4	1D0h

Function

The fifth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B12_15DATA_MASK correspond to EICHD3_WORD4 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B12_15DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B12_15DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD3_WORD4 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD3_WORD4 on the read data bus is inverted.

50.6.21 Error Injection Channel Descriptor 4, Word1 (EICHD4_WORD1)

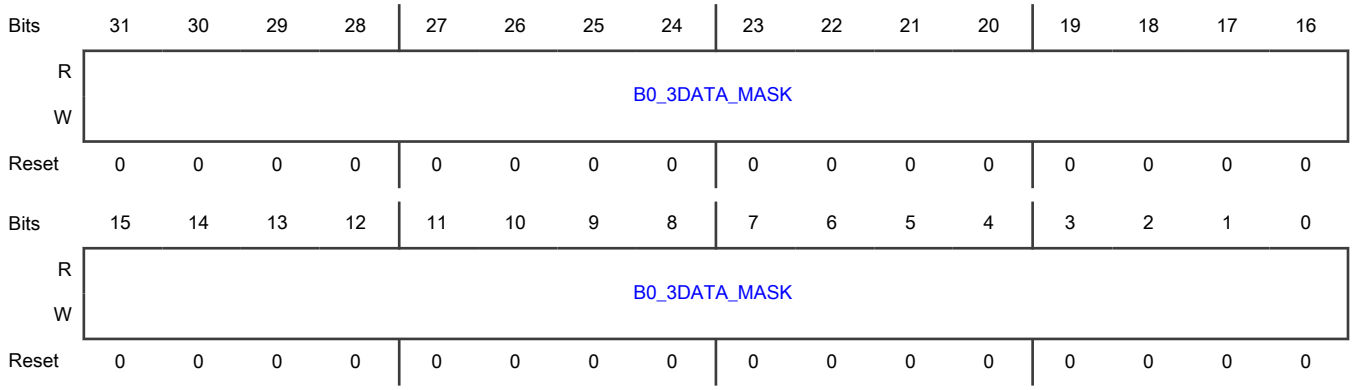
Offset

Register	Offset
EICHD4_WORD1	204h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD4_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD4_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD4_WORD1 on the read data bus is inverted.

50.6.22 Error Injection Channel Descriptor 4, Word2 (EICHD4_WORD2)

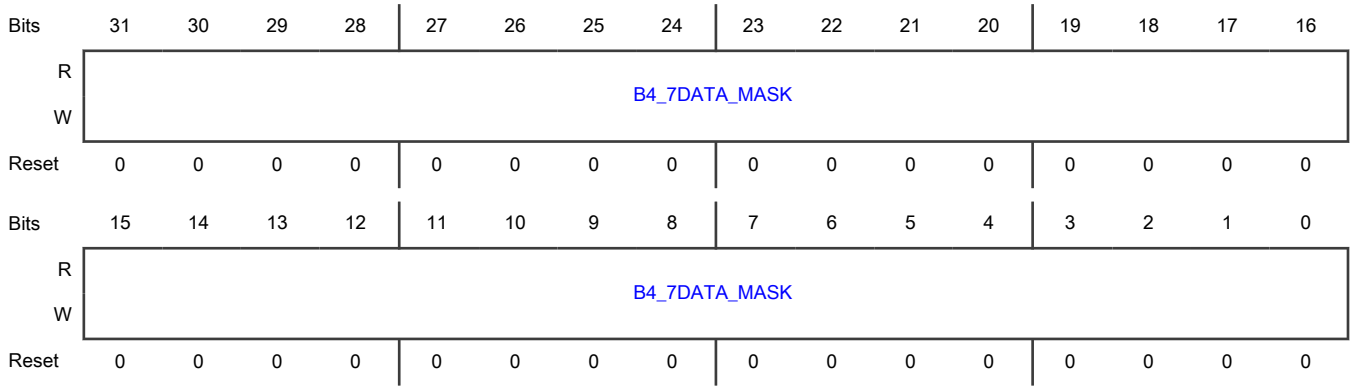
Offset

Register	Offset
EICHD4_WORD2	208h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to EICHD4_WORD2 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD4_WORD2 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD4_WORD2 on the read data bus is inverted.

50.6.23 Error Injection Channel Descriptor 4, Word3 (EICHD4_WORD3)

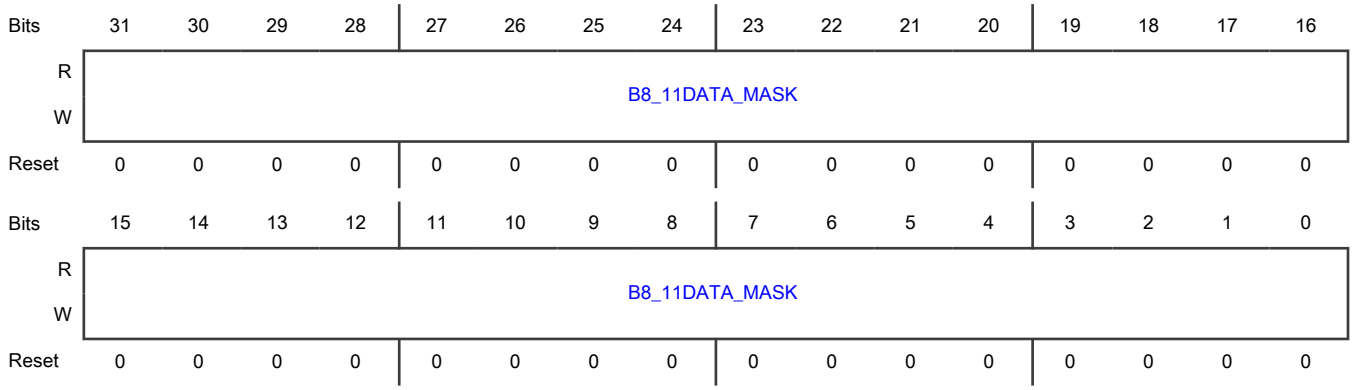
Offset

Register	Offset
EICHD4_WORD3	20Ch

Function

The fourth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B8_11DATA_MASK correspond to EICHD4_WORD3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B8_11DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B8_11DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD4_WORD3 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD4_WORD3 on the read data bus is inverted.

50.6.24 Error Injection Channel Descriptor 4, Word4 (EICHD4_WORD4)

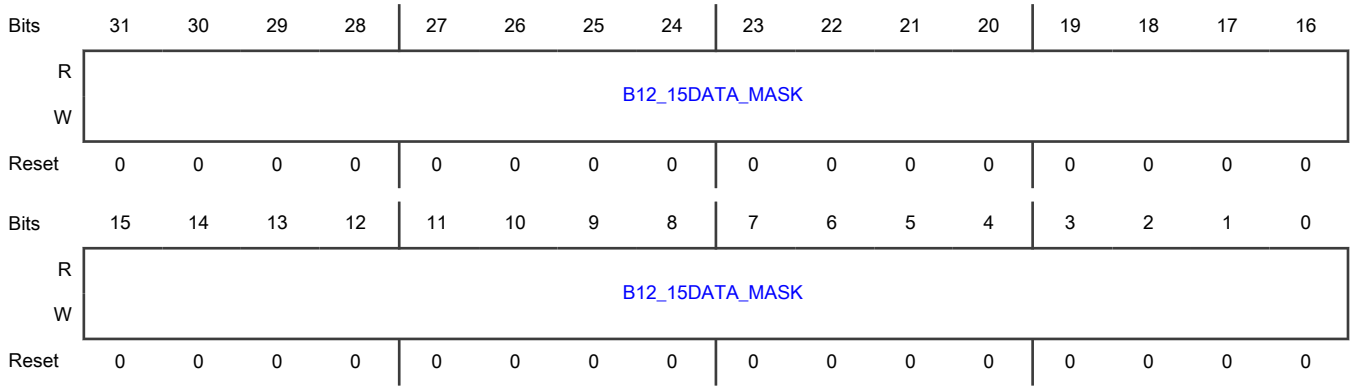
Offset

Register	Offset
EICHD4_WORD4	210h

Function

The fifth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B12_15DATA_MASK correspond to EICHD4_WORD4 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B12_15DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B12_15DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD4_WORD4 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD4_WORD4 on the read data bus is inverted.

50.6.25 Error Injection Channel Descriptor 5, Word0 (EICHD5_WORD0)

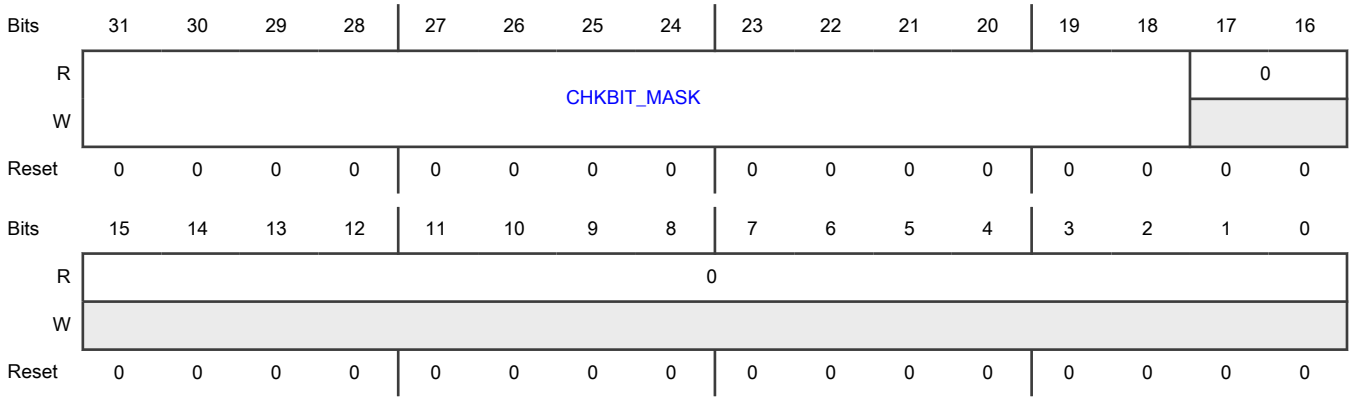
Offset

Register	Offset
EICHD5_WORD0	240h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-18 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[13:0] (14 bits wide), CHKBIT_MASK[13] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
17-0 —	Reserved

50.6.26 Error Injection Channel Descriptor 5, Word1 (EICHD5_WORD1)

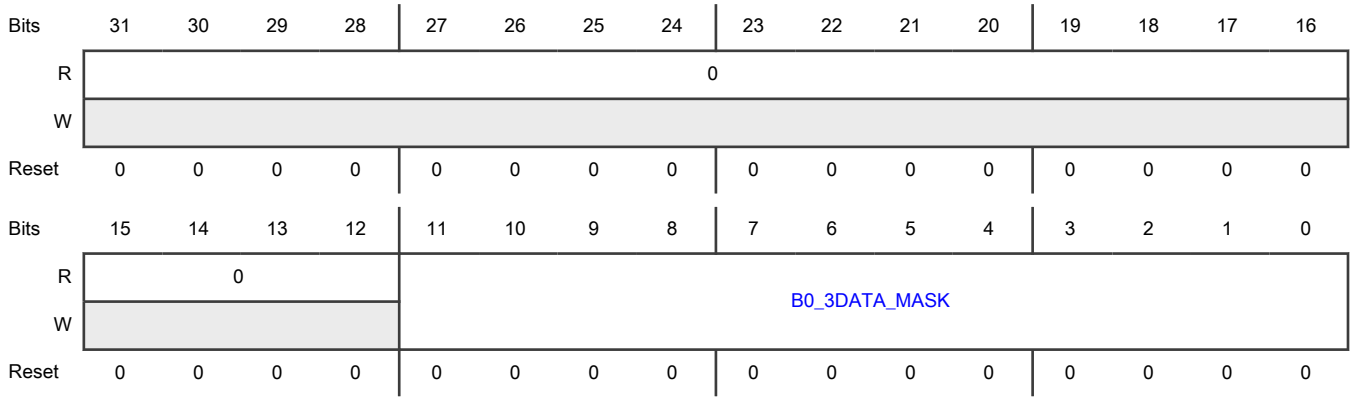
Offset

Register	Offset
EICHD5_WORD1	244h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD5_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-12 —	Reserved
11-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD5_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD5_WORD1 on the read data bus is inverted.

50.6.27 Error Injection Channel Descriptor 5, Word2 (EICHD5_WORD2)

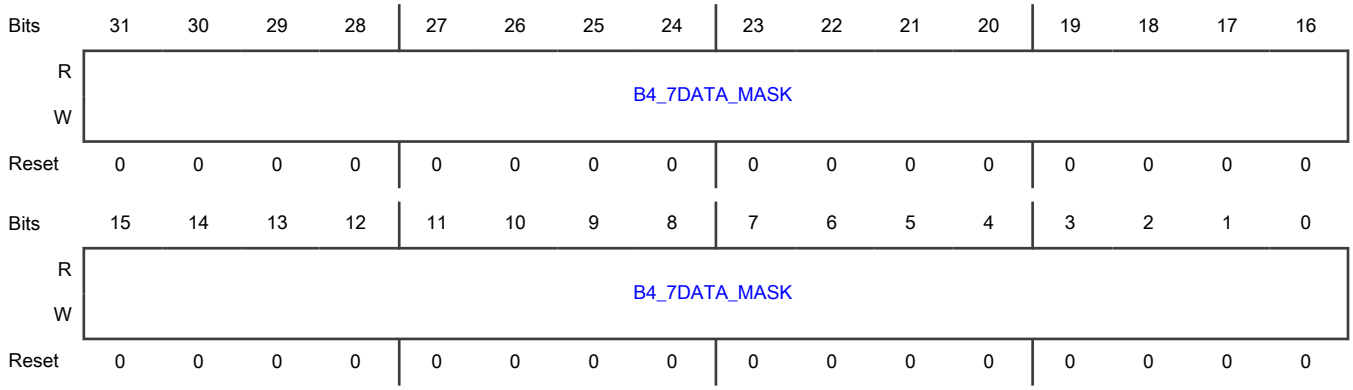
Offset

Register	Offset
EICHD5_WORD2	248h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to EICHD5_WORD2 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD5_WORD2 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD5_WORD2 on the read data bus is inverted.

50.6.28 Error Injection Channel Descriptor 6, Word0 (EICHD6_WORD0)

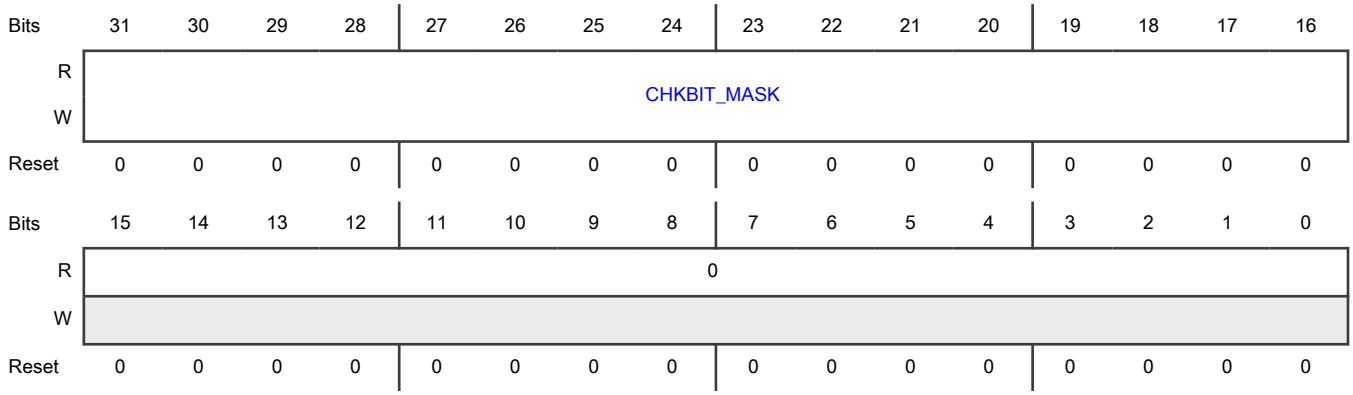
Offset

Register	Offset
EICHD6_WORD0	280h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-16 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[15:0] (16 bits wide), CHKBIT_MASK[15] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
15-0 —	Reserved

50.6.29 Error Injection Channel Descriptor 6, Word1 (EICHD6_WORD1)

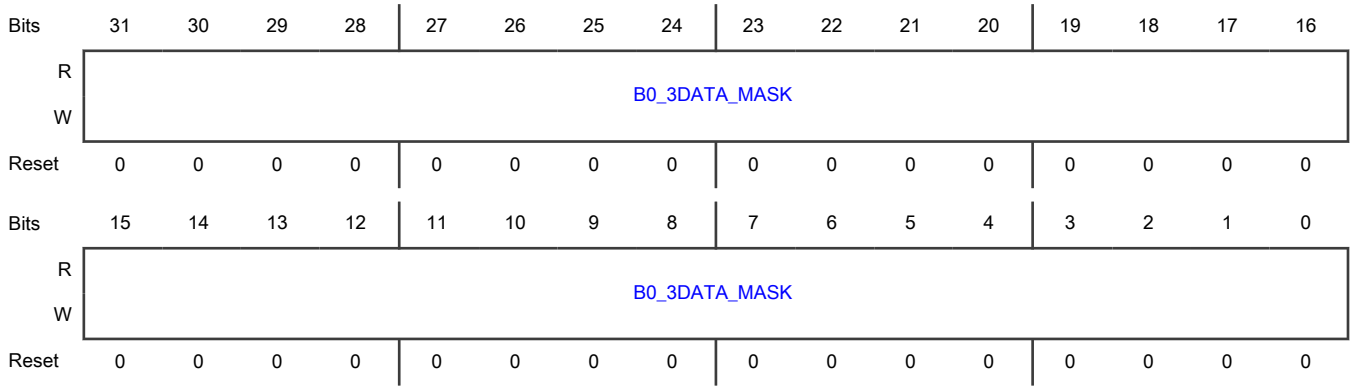
Offset

Register	Offset
EICHD6_WORD1	284h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD6_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD6_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD6_WORD1 on the read data bus is inverted.

50.6.30 Error Injection Channel Descriptor 6, Word2 (EICHD6_WORD2)

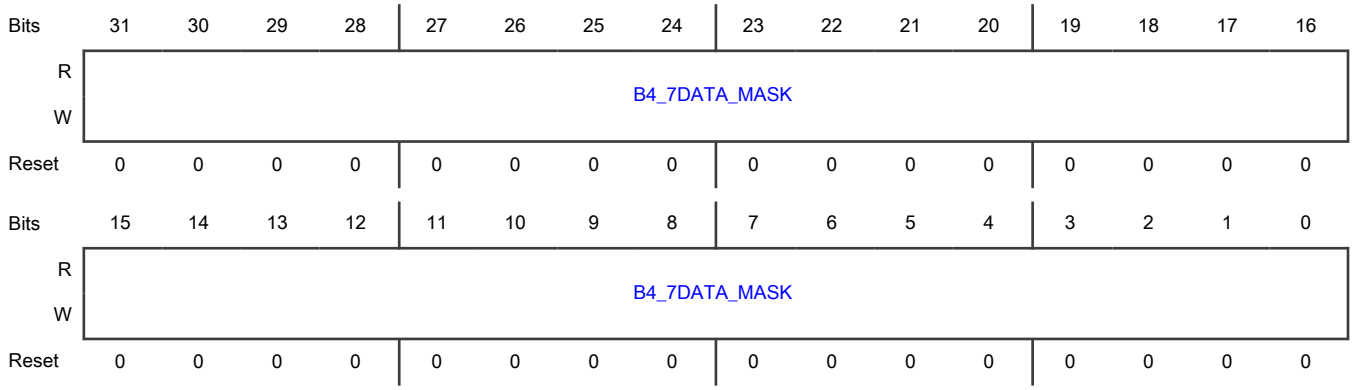
Offset

Register	Offset
EICHD6_WORD2	288h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to EICHD6_WORD2 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD6_WORD2 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD6_WORD2 on the read data bus is inverted.

50.6.31 Error Injection Channel Descriptor 6, Word3 (EICHD6_WORD3)

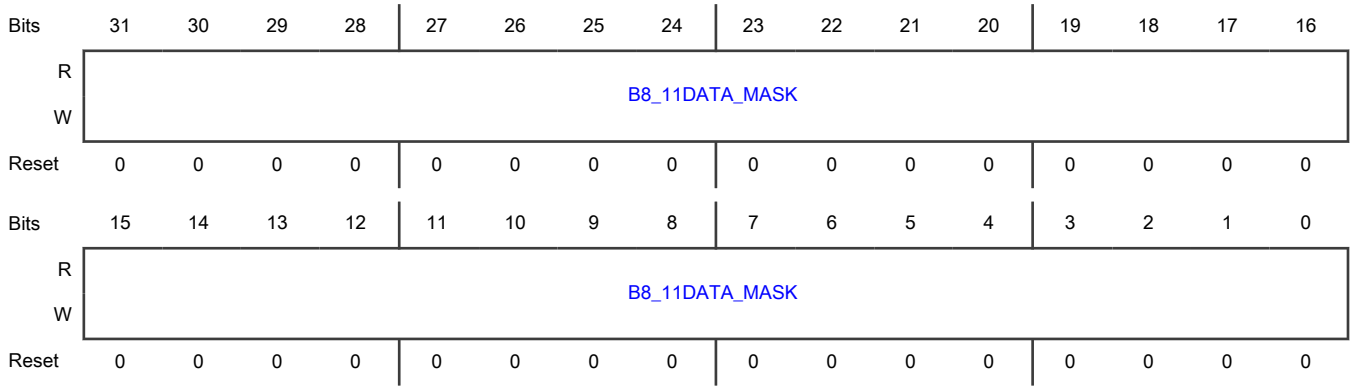
Offset

Register	Offset
EICHD6_WORD3	28Ch

Function

The fourth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B8_11DATA_MASK correspond to EICHD6_WORD3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B8_11DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B8_11DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD6_WORD3 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD6_WORD3 on the read data bus is inverted.

50.6.32 Error Injection Channel Descriptor 6, Word4 (EICHD6_WORD4)

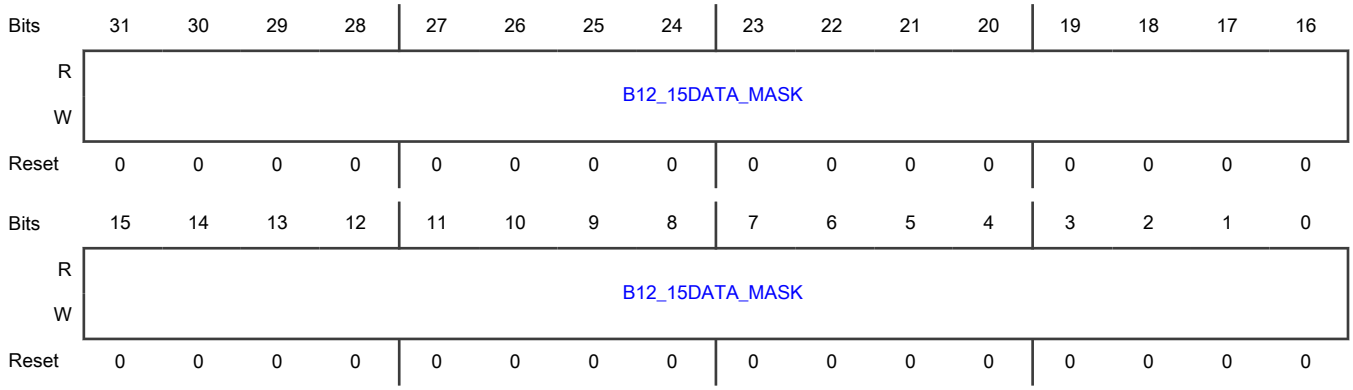
Offset

Register	Offset
EICHD6_WORD4	290h

Function

The fifth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B12_15DATA_MASK correspond to EICHD6_WORD4 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B12_15DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B12_15DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD6_WORD4 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD6_WORD4 on the read data bus is inverted.

50.6.33 Error Injection Channel Descriptor n, Word0 (EICHD7_WORD0 - EICHD9_WORD0)

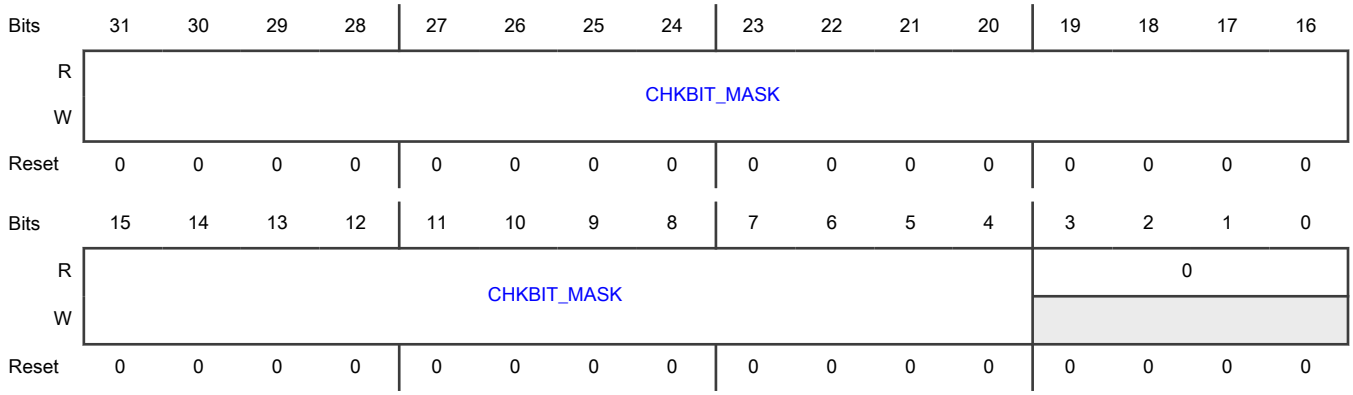
Offset

Register	Offset
EICHD7_WORD0	2C0h
EICHD8_WORD0	300h
EICHD9_WORD0	340h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-4 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[27:0] (28 bits wide), CHKBIT_MASK[27] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
3-0 —	Reserved

50.6.34 Error Injection Channel Descriptor 7, Word1 (EICHD7_WORD1)

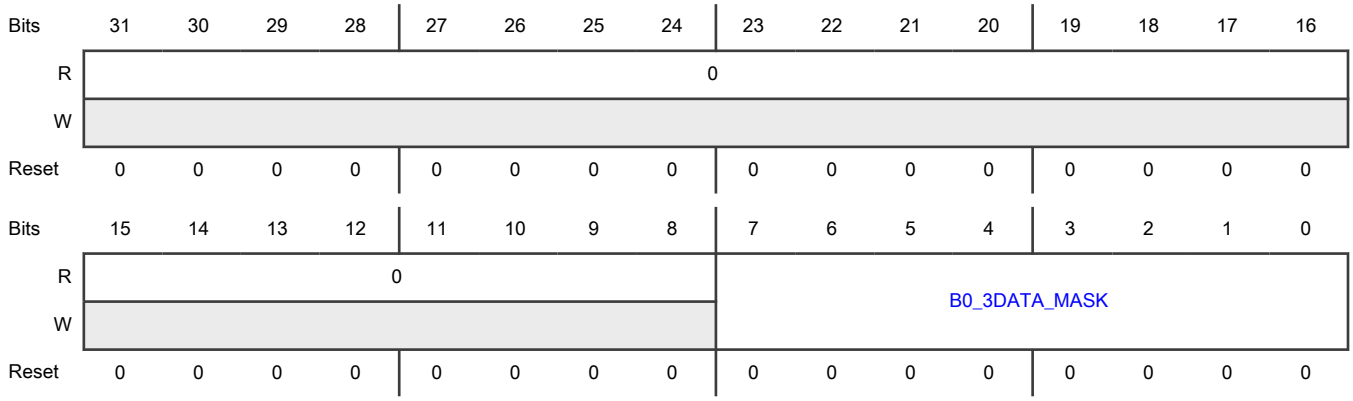
Offset

Register	Offset
EICHD7_WORD1	2C4h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD7_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD7_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD7_WORD1 on the read data bus is inverted.

50.6.35 Error Injection Channel Descriptor 7, Word2 (EICHD7_WORD2)

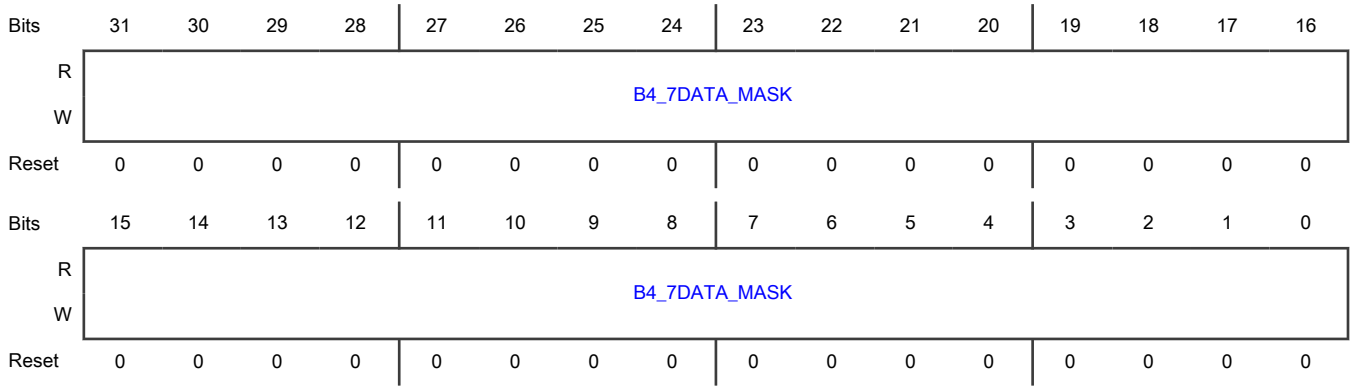
Offset

Register	Offset
EICHD7_WORD2	2C8h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to EICHD7_WORD2 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD7_WORD2 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD7_WORD2 on the read data bus is inverted.

50.6.36 Error Injection Channel Descriptor 7, Word3 (EICHD7_WORD3)

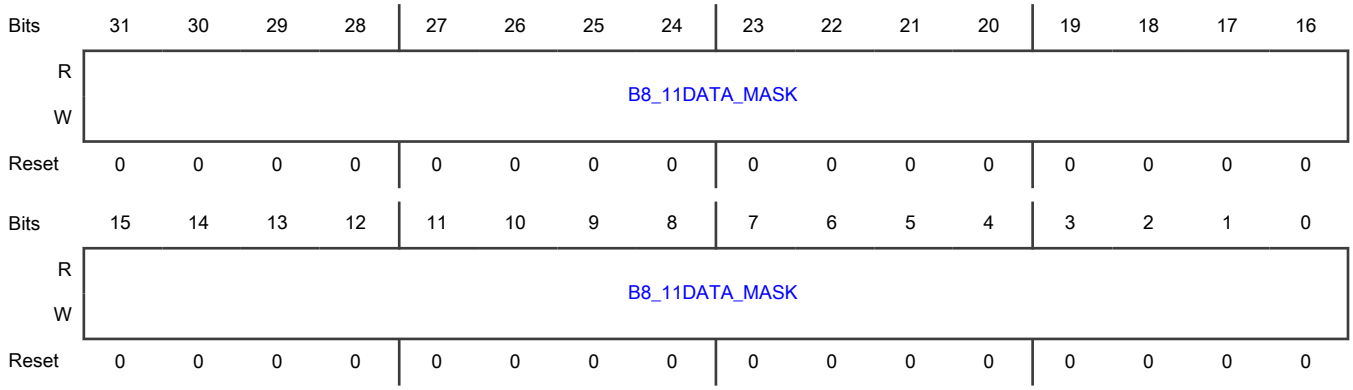
Offset

Register	Offset
EICHD7_WORD3	2CCh

Function

The fourth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B8_11DATA_MASK correspond to EICHD7_WORD3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B8_11DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B8_11DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD7_WORD3 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD7_WORD3 on the read data bus is inverted.

50.6.37 Error Injection Channel Descriptor 7, Word4 (EICHD7_WORD4)

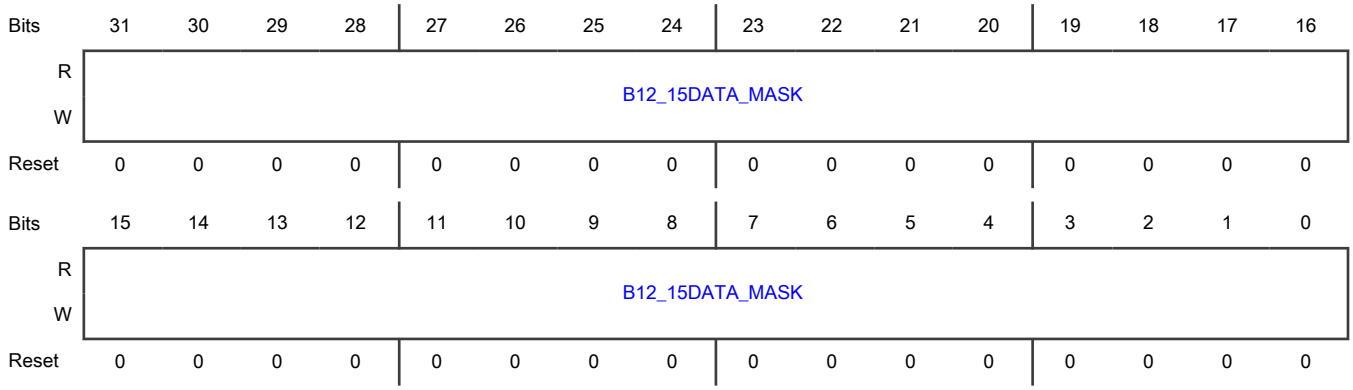
Offset

Register	Offset
EICHD7_WORD4	2D0h

Function

The fifth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B12_15DATA_MASK correspond to EICHD7_WORD4 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B12_15DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B12_15DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD7_WORD4 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD7_WORD4 on the read data bus is inverted.

50.6.38 Error Injection Channel Descriptor 8, Word1 (EICHD8_WORD1)

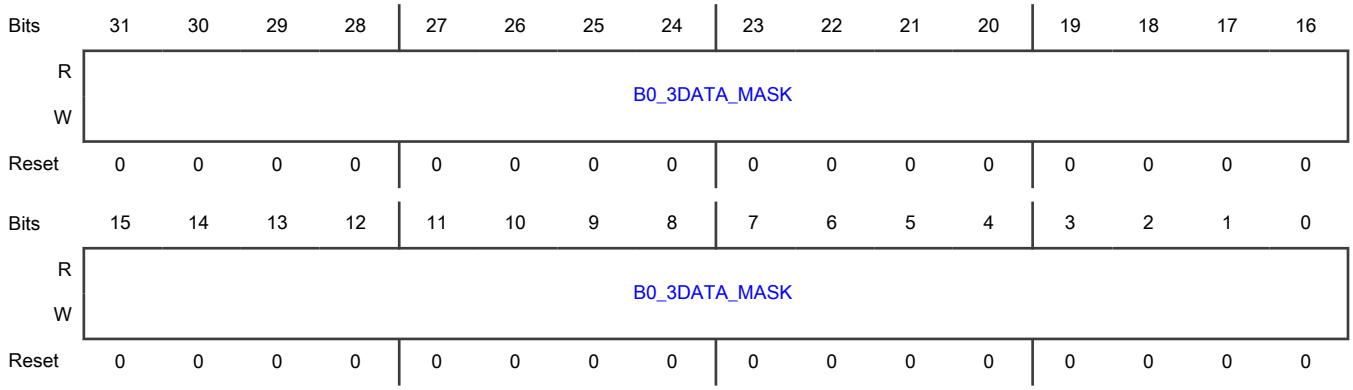
Offset

Register	Offset
EICHD8_WORD1	304h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD8_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD8_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD8_WORD1 on the read data bus is inverted.

50.6.39 Error Injection Channel Descriptor 8, Word2 (EICHD8_WORD2)

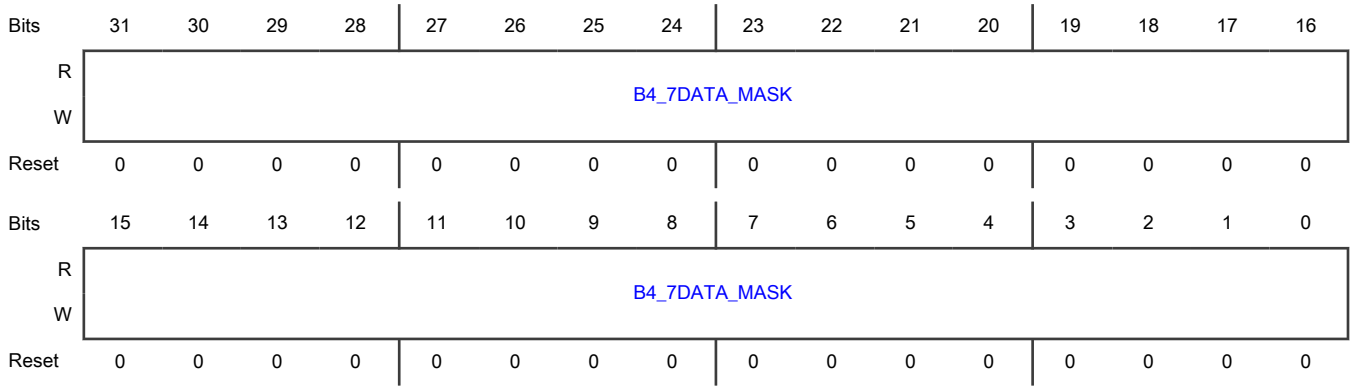
Offset

Register	Offset
EICHD8_WORD2	308h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to EICHD8_WORD2 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD8_WORD2 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD8_WORD2 on the read data bus is inverted.

50.6.40 Error Injection Channel Descriptor 8, Word3 (EICHD8_WORD3)

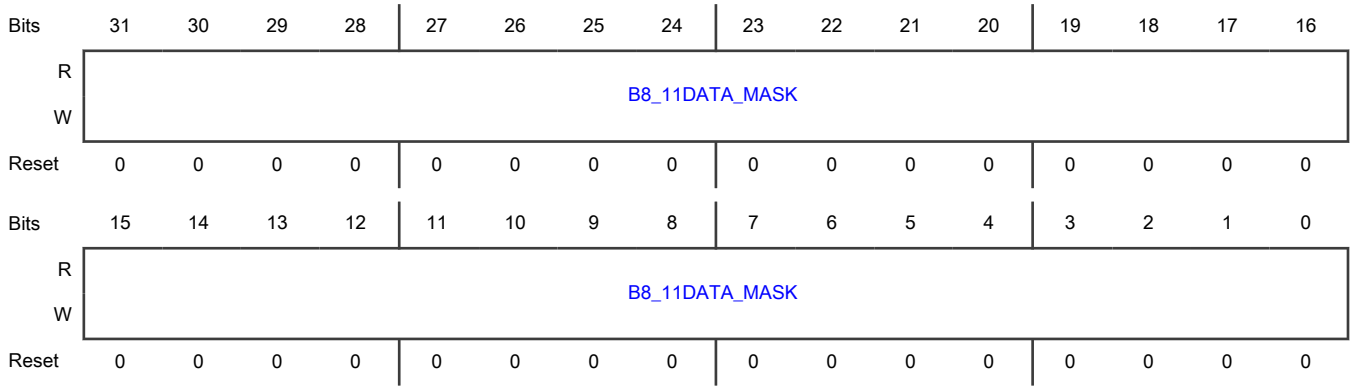
Offset

Register	Offset
EICHD8_WORD3	30Ch

Function

The fourth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B8_11DATA_MASK correspond to EICHD8_WORD3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B8_11DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B8_11DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD8_WORD3 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD8_WORD3 on the read data bus is inverted.

50.6.41 Error Injection Channel Descriptor 8, Word4 (EICHD8_WORD4)

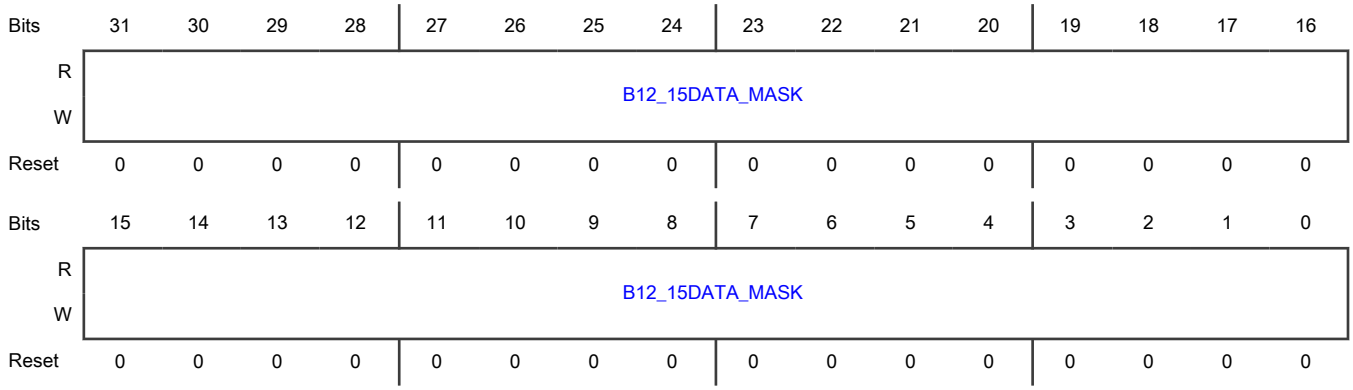
Offset

Register	Offset
EICHD8_WORD4	310h

Function

The fifth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B12_15DATA_MASK correspond to EICHD8_WORD4 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B12_15DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B12_15DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD8_WORD4 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD8_WORD4 on the read data bus is inverted.

50.6.42 Error Injection Channel Descriptor 9, Word1 (EICHD9_WORD1)

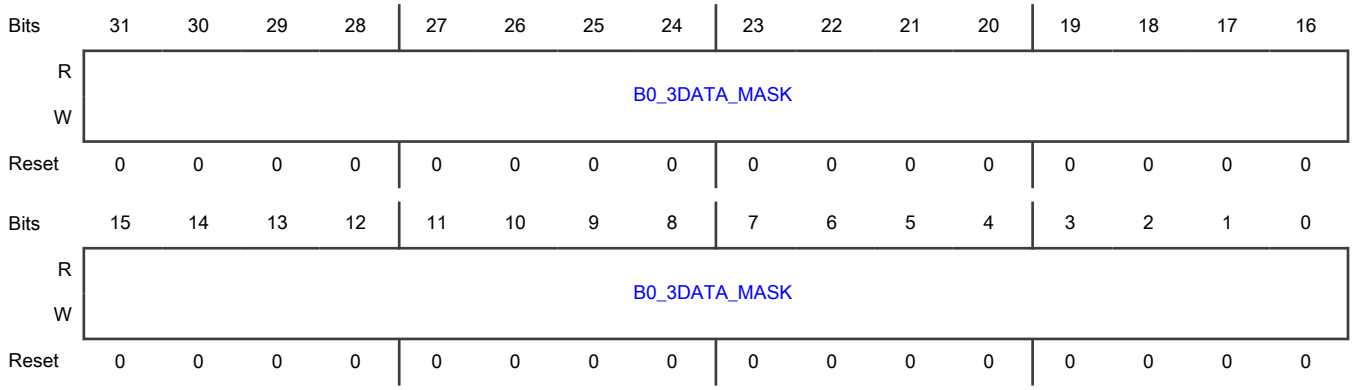
Offset

Register	Offset
EICHD9_WORD1	344h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD9_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD9_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD9_WORD1 on the read data bus is inverted.

50.6.43 Error Injection Channel Descriptor 9, Word2 (EICHD9_WORD2)

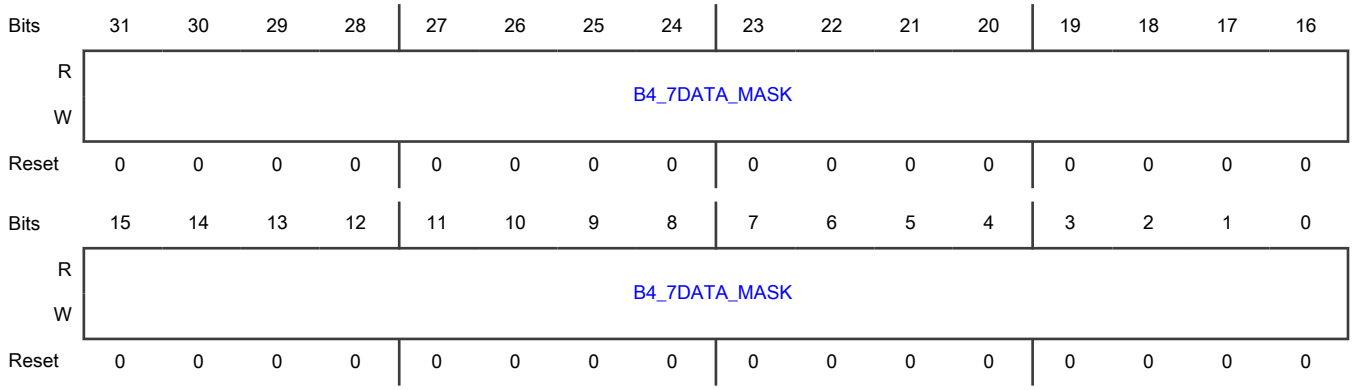
Offset

Register	Offset
EICHD9_WORD2	348h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to EICHD9_WORD2 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD9_WORD2 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD9_WORD2 on the read data bus is inverted.

50.6.44 Error Injection Channel Descriptor 9, Word3 (EICHD9_WORD3)

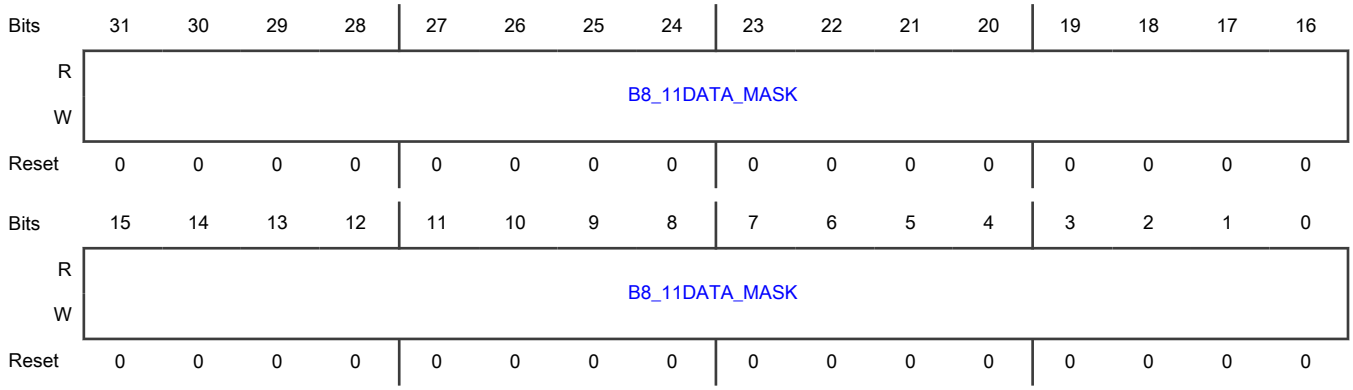
Offset

Register	Offset
EICHD9_WORD3	34Ch

Function

The fourth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B8_11DATA_MASK correspond to EICHD9_WORD3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B8_11DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B8_11DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD9_WORD3 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD9_WORD3 on the read data bus is inverted.

50.6.45 Error Injection Channel Descriptor 9, Word4 (EICHD9_WORD4)

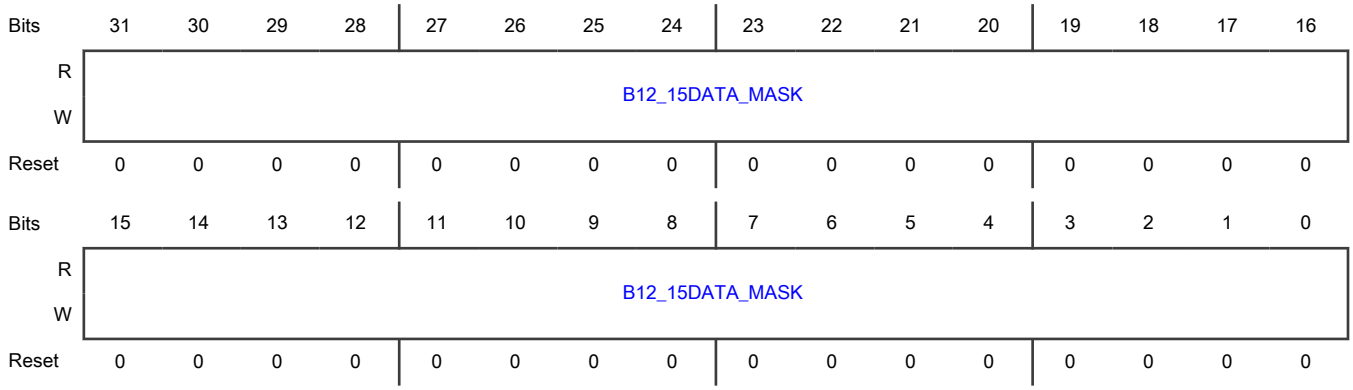
Offset

Register	Offset
EICHD9_WORD4	350h

Function

The fifth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B12_15DATA_MASK correspond to EICHD9_WORD4 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B12_15DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B12_15DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD9_WORD4 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD9_WORD4 on the read data bus is inverted.

50.6.46 Error Injection Channel Descriptor n, Word0 (EICHD10_WORD0 - EICHD15_WORD0)

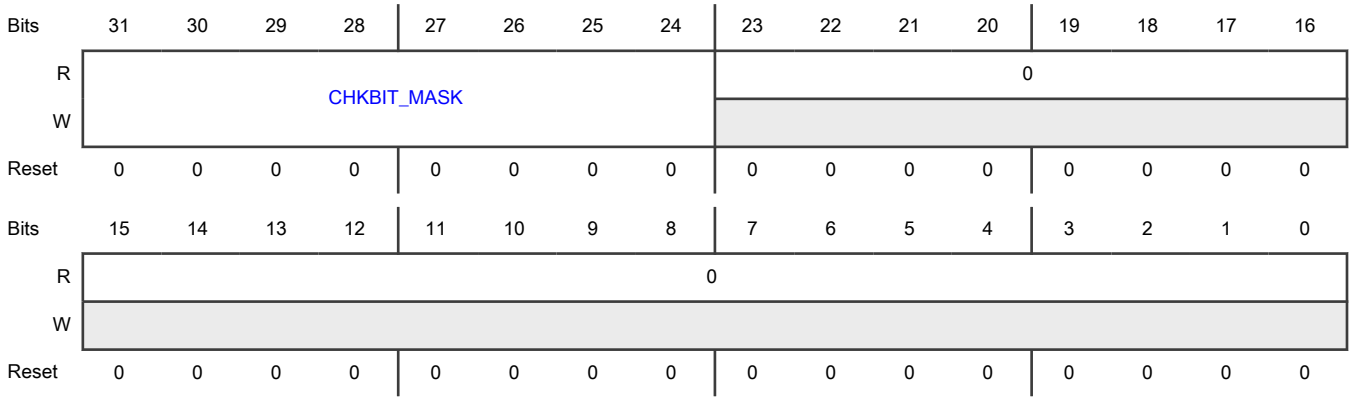
Offset

Register	Offset
EICHD10_WORD0	380h
EICHD11_WORD0	3C0h
EICHD12_WORD0	400h
EICHD13_WORD0	440h
EICHD14_WORD0	480h
EICHD15_WORD0	4C0h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-24 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[7:0] (8 bits wide), CHKBIT_MASK[7] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
23-0 —	Reserved

50.6.47 Error Injection Channel Descriptor 10, Word1 (EICHD10_WORD1)

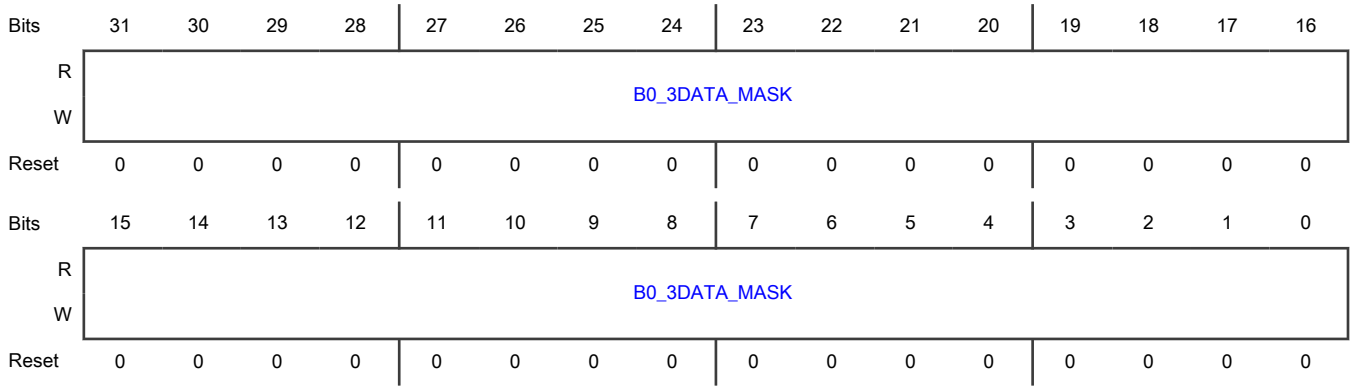
Offset

Register	Offset
EICHD10_WORD1	384h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD10_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD10_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD10_WORD1 on the read data bus is inverted.

50.6.48 Error Injection Channel Descriptor 10, Word2 (EICHD10_WORD2)

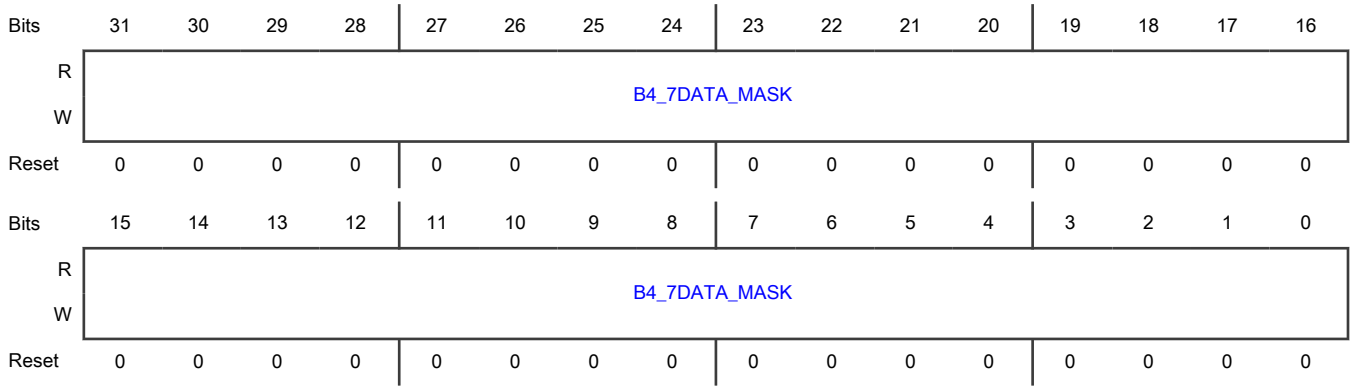
Offset

Register	Offset
EICHD10_WORD2	388h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to EICHD10_WORD2 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD10_WORD2 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD10_WORD2 on the read data bus is inverted.

50.6.49 Error Injection Channel Descriptor 11, Word1 (EICHD11_WORD1)

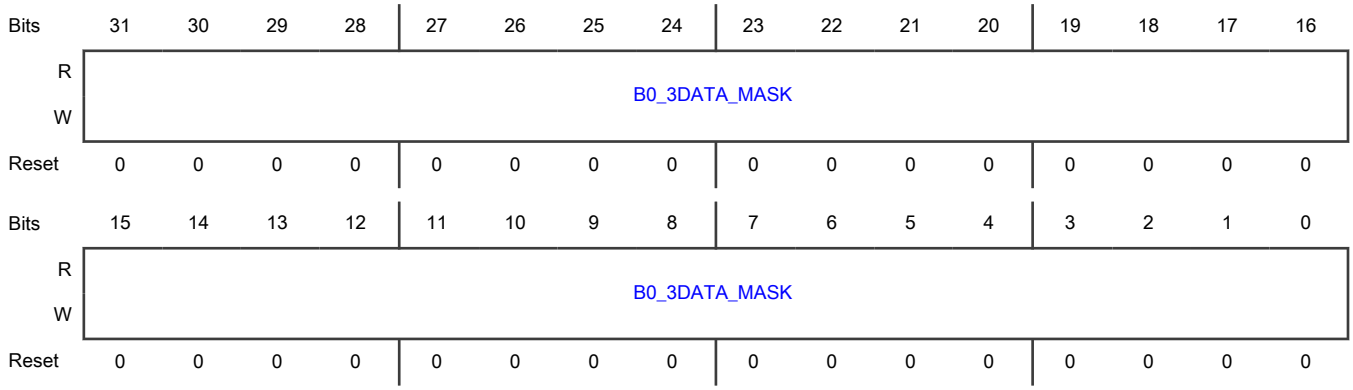
Offset

Register	Offset
EICHD11_WORD1	3C4h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD11_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD11_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD11_WORD1 on the read data bus is inverted.

50.6.50 Error Injection Channel Descriptor 12, Word1 (EICHD12_WORD1)

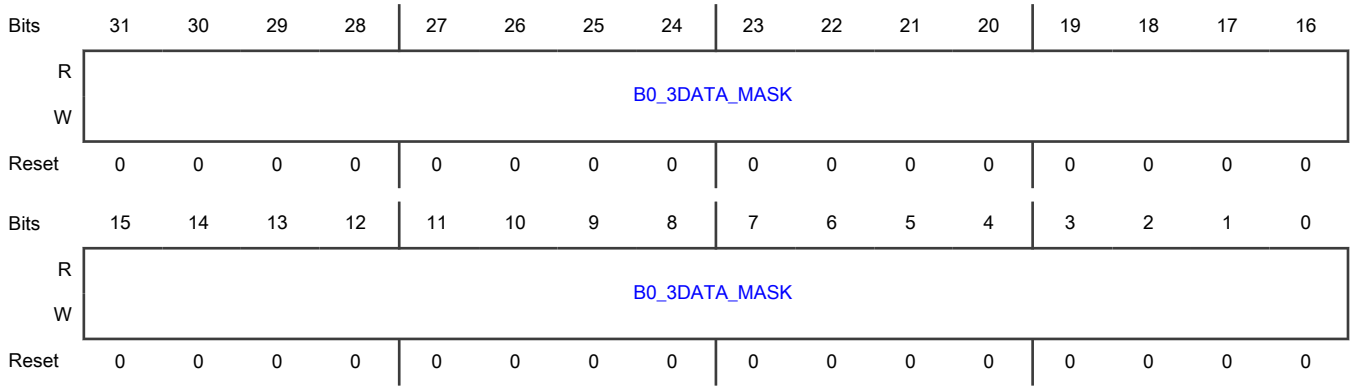
Offset

Register	Offset
EICHD12_WORD1	404h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD12_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD12_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD12_WORD1 on the read data bus is inverted.

50.6.51 Error Injection Channel Descriptor 13, Word1 (EICHD13_WORD1)

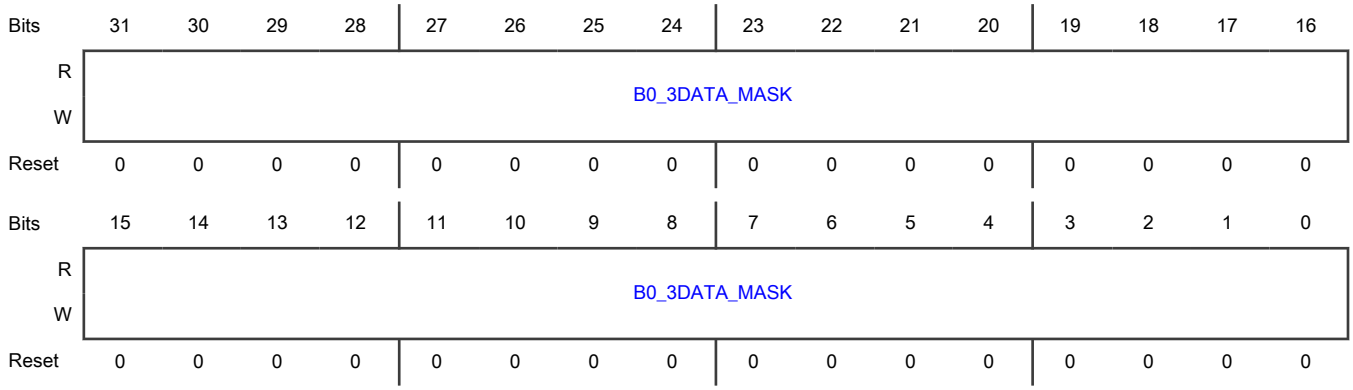
Offset

Register	Offset
EICHD13_WORD1	444h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD13_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD13_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD13_WORD1 on the read data bus is inverted.

50.6.52 Error Injection Channel Descriptor 13, Word2 (EICHD13_WORD2)

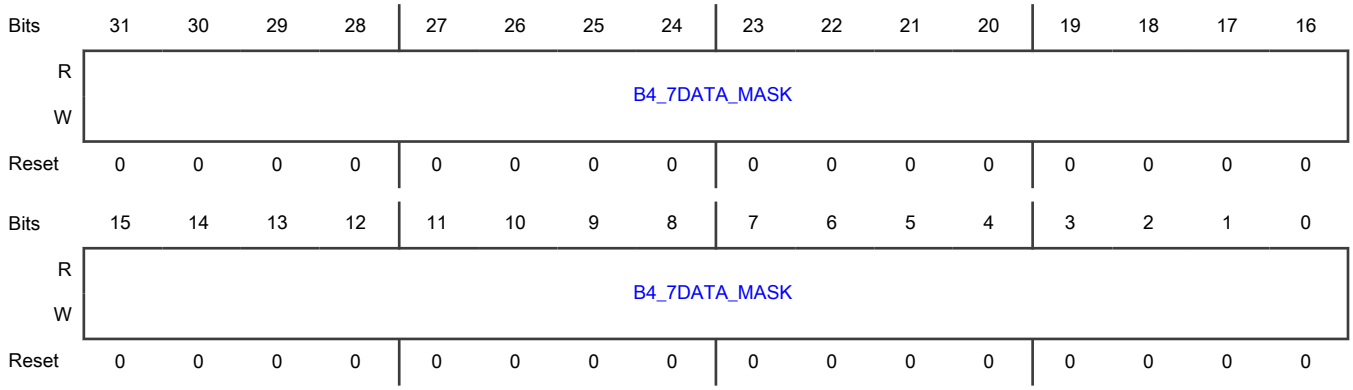
Offset

Register	Offset
EICHD13_WORD2	448h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to EICHD13_WORD2 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD13_WORD2 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD13_WORD2 on the read data bus is inverted.

50.6.53 Error Injection Channel Descriptor 14, Word1 (EICHD14_WORD1)

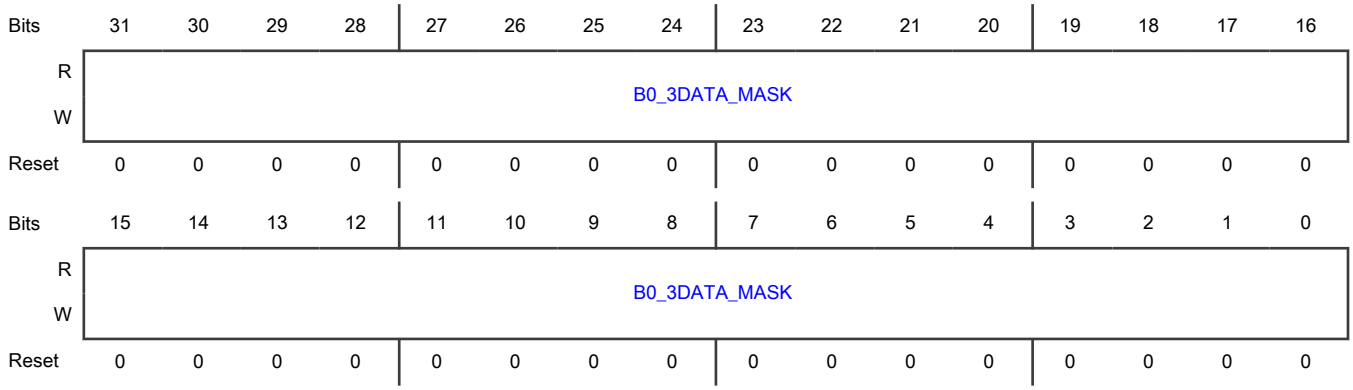
Offset

Register	Offset
EICHD14_WORD1	484h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD14_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD14_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD14_WORD1 on the read data bus is inverted.

50.6.54 Error Injection Channel Descriptor 15, Word1 (EICHD15_WORD1)

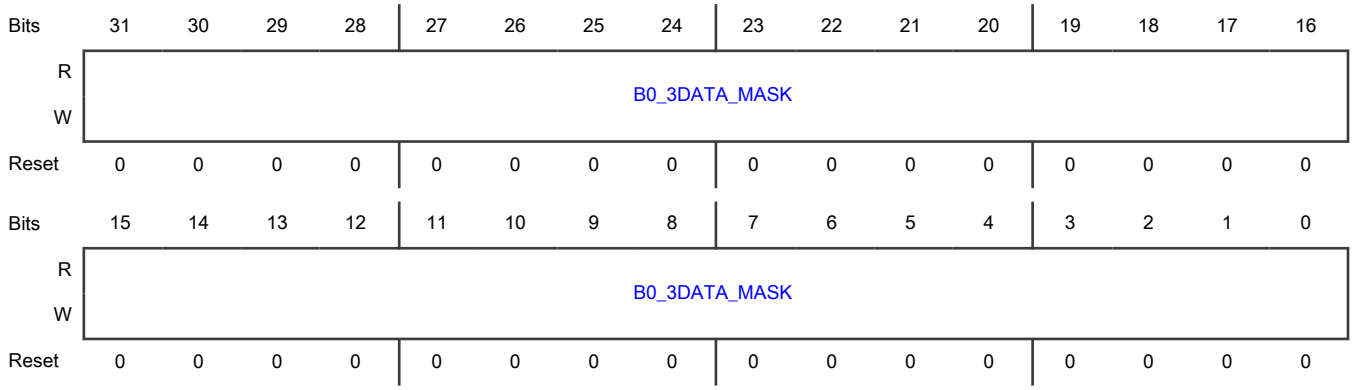
Offset

Register	Offset
EICHD15_WORD1	4C4h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD15_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD15_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD15_WORD1 on the read data bus is inverted.

50.6.55 Error Injection Channel Descriptor 16, Word1 (EICHD16_WORD1)

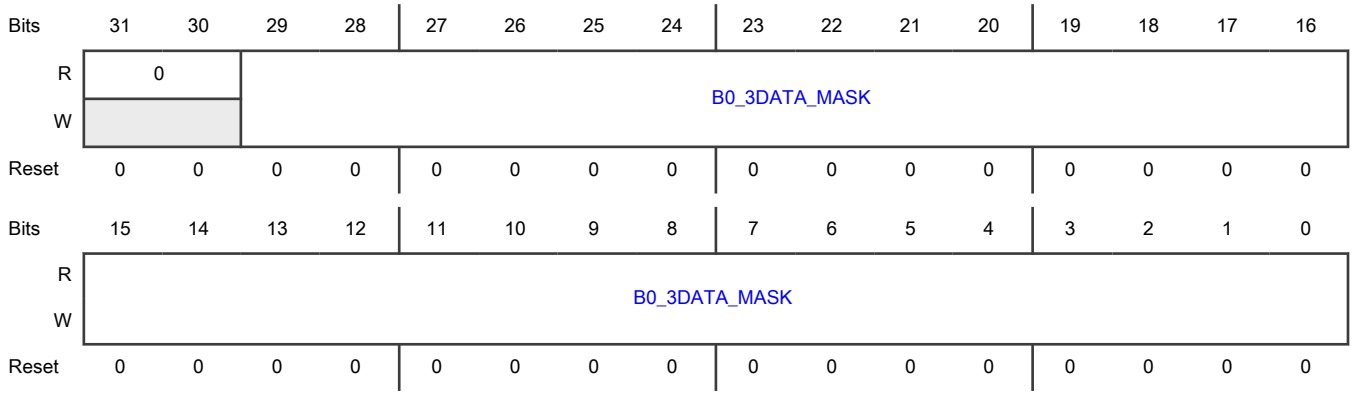
Offset

Register	Offset
EICHD16_WORD1	504h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD16_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-30 —	Reserved
29-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD16_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD16_WORD1 on the read data bus is inverted.

50.7 EIM_2 register descriptions

The EIM provides a programming model mapped to an on-platform peripheral slot.

Programming model access

All system bus controllers can access the programming model:

- Only in supervisor mode
- Using only 32-bit (word) accesses

Any of the following attempted references to the programming model generates an error termination:

- In user mode
- Using non-32-bit access sizes
- To undefined (reserved) addresses

Attempted updates to the programming model while the EIM is in the midst of an operation result in non-deterministic behavior.

Error injection channel descriptor: function and structure

Each error injection channel descriptor:

- Specifies a mask that defines which bits of the read data and/or checkbit bus from target RAM are inverted on a read access.
- Consists of a 288-bit (36-byte) structure, composed of nine 32-bit words, in the EIM programming model. Unused words are not documented.
  - Word0 (EICHD*n*_WORD0), if present, defines the checkbit mask.
  - Word1 (EICHD*n*_WORD1) and additional words, if present, define the data mask. Word registers subsequent to Word1 are present only when required by the total width of the channel's data mask. Error injection channel descriptor: DATA_MASK details.

The multiple channel descriptors are organized sequentially.

Error injection channel descriptor: DATA_MASK details

For each channel: The following tables show the distribution of DATA_MASK's bits across the WORD registers. The first table shows the total width of DATA_MASK and the distribution of its bits across WORD1, WORD2, and WORD3. The second table shows the distribution of DATA_MASK's bits across WORD4 and subsequent registers.

Table 287. Error injection channel descriptor: DATA_MASK details

Channel	DATA_MASK total width (bits)	Specific bits of DATA_MASK in		
		WORD1	WORD2	WORD3
0	64	63-32	31-0	—
1	64	63-32	31-0	—
2	64	63-32	31-0	—
3	64	63-32	31-0	—
4	60	59-32	31-0	—
5	60	59-32	31-0	—
7	60	59-32	31-0	—
8	60	59-32	31-0	—
9	60	59-32	31-0	—
10	60	59-32	31-0	—
11	60	59-32	31-0	—
12	60	59-32	31-0	—
13	60	59-32	31-0	—
14	60	59-32	31-0	—
15	60	59-32	31-0	—
16	60	59-32	31-0	—
17	60	59-32	31-0	—

Table continues on the next page...

Table 287. Error injection channel descriptor: DATA_MASK details (continued)

Channel	DATA_MASK total width (bits)	Specific bits of DATA_MASK in		
		WORD1	WORD2	WORD3
18	60	59-32	31-0	—
19	60	59-32	31-0	—
20	60	59-32	31-0	—
21	60	59-32	31-0	—
22	188	187-160	159-128	127-96
23	188	187-160	159-128	127-96
24	188	187-160	159-128	127-96
25	188	187-160	159-128	127-96
26	14	13-0	—	—
27	8	7-0	—	—
28	14	13-0	—	—
29	28	27-0	—	—
30	18	17-0	—	—
31	16	15-0	—	—

Table 288. DATA_MASK bit: Channel-word mapping

Channel	Specific bits of DATA_MASK in				
	WORD4	WORD5	WORD6	WORD7	WORD8
22	95-64	63-32	31-0	—	—
23	95-64	63-32	31-0	—	—
24	95-64	63-32	31-0	—	—
25	95-64	63-32	31-0	—	—

50.7.1 EIM_2 memory map

EIM_2 base address: 4051_4000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Error Injection Module Configuration Register (EIMCR)	32	RW	0000_0000h
4h	Error Injection Channel Enable register (EICHEN)	32	RW	0000_0000h
100h	Error Injection Channel Descriptor 0, Word0 (EICHD0_WORD0)	32	RW	0000_0000h
104h	Error Injection Channel Descriptor 0, Word1 (EICHD0_WORD1)	32	RW	0000_0000h
108h	Error Injection Channel Descriptor 0, Word2 (EICHD0_WORD2)	32	RW	0000_0000h
140h	Error Injection Channel Descriptor 1, Word0 (EICHD1_WORD0)	32	RW	0000_0000h
144h	Error Injection Channel Descriptor 1, Word1 (EICHD1_WORD1)	32	RW	0000_0000h
148h	Error Injection Channel Descriptor 1, Word2 (EICHD1_WORD2)	32	RW	0000_0000h
180h	Error Injection Channel Descriptor 2, Word0 (EICHD2_WORD0)	32	RW	0000_0000h
184h	Error Injection Channel Descriptor 2, Word1 (EICHD2_WORD1)	32	RW	0000_0000h
188h	Error Injection Channel Descriptor 2, Word2 (EICHD2_WORD2)	32	RW	0000_0000h
1C0h	Error Injection Channel Descriptor 3, Word0 (EICHD3_WORD0)	32	RW	0000_0000h
1C4h	Error Injection Channel Descriptor 3, Word1 (EICHD3_WORD1)	32	RW	0000_0000h
1C8h	Error Injection Channel Descriptor 3, Word2 (EICHD3_WORD2)	32	RW	0000_0000h
204h	Error Injection Channel Descriptor 4, Word1 (EICHD4_WORD1)	32	RW	0000_0000h
208h	Error Injection Channel Descriptor 4, Word2 (EICHD4_WORD2)	32	RW	0000_0000h
244h	Error Injection Channel Descriptor 5, Word1 (EICHD5_WORD1)	32	RW	0000_0000h
248h	Error Injection Channel Descriptor 5, Word2 (EICHD5_WORD2)	32	RW	0000_0000h
2C4h	Error Injection Channel Descriptor 7, Word1 (EICHD7_WORD1)	32	RW	0000_0000h
2C8h	Error Injection Channel Descriptor 7, Word2 (EICHD7_WORD2)	32	RW	0000_0000h
304h	Error Injection Channel Descriptor 8, Word1 (EICHD8_WORD1)	32	RW	0000_0000h
308h	Error Injection Channel Descriptor 8, Word2 (EICHD8_WORD2)	32	RW	0000_0000h
344h	Error Injection Channel Descriptor 9, Word1 (EICHD9_WORD1)	32	RW	0000_0000h
348h	Error Injection Channel Descriptor 9, Word2 (EICHD9_WORD2)	32	RW	0000_0000h
384h	Error Injection Channel Descriptor 10, Word1 (EICHD10_WORD1)	32	RW	0000_0000h
388h	Error Injection Channel Descriptor 10, Word2 (EICHD10_WORD2)	32	RW	0000_0000h
3C4h	Error Injection Channel Descriptor 11, Word1 (EICHD11_WORD1)	32	RW	0000_0000h
3C8h	Error Injection Channel Descriptor 11, Word2 (EICHD11_WORD2)	32	RW	0000_0000h
404h	Error Injection Channel Descriptor 12, Word1 (EICHD12_WORD1)	32	RW	0000_0000h
408h	Error Injection Channel Descriptor 12, Word2 (EICHD12_WORD2)	32	RW	0000_0000h
444h	Error Injection Channel Descriptor 13, Word1 (EICHD13_WORD1)	32	RW	0000_0000h
448h	Error Injection Channel Descriptor 13, Word2 (EICHD13_WORD2)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
484h	Error Injection Channel Descriptor 14, Word1 (EICHD14_WORD1)	32	RW	0000_0000h
488h	Error Injection Channel Descriptor 14, Word2 (EICHD14_WORD2)	32	RW	0000_0000h
4C4h	Error Injection Channel Descriptor 15, Word1 (EICHD15_WORD1)	32	RW	0000_0000h
4C8h	Error Injection Channel Descriptor 15, Word2 (EICHD15_WORD2)	32	RW	0000_0000h
504h	Error Injection Channel Descriptor 16, Word1 (EICHD16_WORD1)	32	RW	0000_0000h
508h	Error Injection Channel Descriptor 16, Word2 (EICHD16_WORD2)	32	RW	0000_0000h
544h	Error Injection Channel Descriptor 17, Word1 (EICHD17_WORD1)	32	RW	0000_0000h
548h	Error Injection Channel Descriptor 17, Word2 (EICHD17_WORD2)	32	RW	0000_0000h
584h	Error Injection Channel Descriptor 18, Word1 (EICHD18_WORD1)	32	RW	0000_0000h
588h	Error Injection Channel Descriptor 18, Word2 (EICHD18_WORD2)	32	RW	0000_0000h
5C4h	Error Injection Channel Descriptor 19, Word1 (EICHD19_WORD1)	32	RW	0000_0000h
5C8h	Error Injection Channel Descriptor 19, Word2 (EICHD19_WORD2)	32	RW	0000_0000h
604h	Error Injection Channel Descriptor 20, Word1 (EICHD20_WORD1)	32	RW	0000_0000h
608h	Error Injection Channel Descriptor 20, Word2 (EICHD20_WORD2)	32	RW	0000_0000h
644h	Error Injection Channel Descriptor 21, Word1 (EICHD21_WORD1)	32	RW	0000_0000h
648h	Error Injection Channel Descriptor 21, Word2 (EICHD21_WORD2)	32	RW	0000_0000h
684h	Error Injection Channel Descriptor 22, Word1 (EICHD22_WORD1)	32	RW	0000_0000h
688h	Error Injection Channel Descriptor 22, Word2 (EICHD22_WORD2)	32	RW	0000_0000h
68Ch	Error Injection Channel Descriptor 22, Word3 (EICHD22_WORD3)	32	RW	0000_0000h
690h	Error Injection Channel Descriptor 22, Word4 (EICHD22_WORD4)	32	RW	0000_0000h
694h	Error Injection Channel Descriptor 22, Word5 (EICHD22_WORD5)	32	RW	0000_0000h
698h	Error Injection Channel Descriptor 22, Word6 (EICHD22_WORD6)	32	RW	0000_0000h
6C4h	Error Injection Channel Descriptor 23, Word1 (EICHD23_WORD1)	32	RW	0000_0000h
6C8h	Error Injection Channel Descriptor 23, Word2 (EICHD23_WORD2)	32	RW	0000_0000h
6CCh	Error Injection Channel Descriptor 23, Word3 (EICHD23_WORD3)	32	RW	0000_0000h
6D0h	Error Injection Channel Descriptor 23, Word4 (EICHD23_WORD4)	32	RW	0000_0000h
6D4h	Error Injection Channel Descriptor 23, Word5 (EICHD23_WORD5)	32	RW	0000_0000h
6D8h	Error Injection Channel Descriptor 23, Word6 (EICHD23_WORD6)	32	RW	0000_0000h
704h	Error Injection Channel Descriptor 24, Word1 (EICHD24_WORD1)	32	RW	0000_0000h
708h	Error Injection Channel Descriptor 24, Word2 (EICHD24_WORD2)	32	RW	0000_0000h
70Ch	Error Injection Channel Descriptor 24, Word3 (EICHD24_WORD3)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
710h	Error Injection Channel Descriptor 24, Word4 (EICHD24_WORD4)	32	RW	0000_0000h
714h	Error Injection Channel Descriptor 24, Word5 (EICHD24_WORD5)	32	RW	0000_0000h
718h	Error Injection Channel Descriptor 24, Word6 (EICHD24_WORD6)	32	RW	0000_0000h
744h	Error Injection Channel Descriptor 25, Word1 (EICHD25_WORD1)	32	RW	0000_0000h
748h	Error Injection Channel Descriptor 25, Word2 (EICHD25_WORD2)	32	RW	0000_0000h
74Ch	Error Injection Channel Descriptor 25, Word3 (EICHD25_WORD3)	32	RW	0000_0000h
750h	Error Injection Channel Descriptor 25, Word4 (EICHD25_WORD4)	32	RW	0000_0000h
754h	Error Injection Channel Descriptor 25, Word5 (EICHD25_WORD5)	32	RW	0000_0000h
758h	Error Injection Channel Descriptor 25, Word6 (EICHD25_WORD6)	32	RW	0000_0000h
784h	Error Injection Channel Descriptor 26, Word1 (EICHD26_WORD1)	32	RW	0000_0000h
7C4h	Error Injection Channel Descriptor 27, Word1 (EICHD27_WORD1)	32	RW	0000_0000h
804h	Error Injection Channel Descriptor 28, Word1 (EICHD28_WORD1)	32	RW	0000_0000h
844h	Error Injection Channel Descriptor 29, Word1 (EICHD29_WORD1)	32	RW	0000_0000h
884h	Error Injection Channel Descriptor 30, Word1 (EICHD30_WORD1)	32	RW	0000_0000h
8C4h	Error Injection Channel Descriptor 31, Word1 (EICHD31_WORD1)	32	RW	0000_0000h

50.7.2 Error Injection Module Configuration Register (EIMCR)

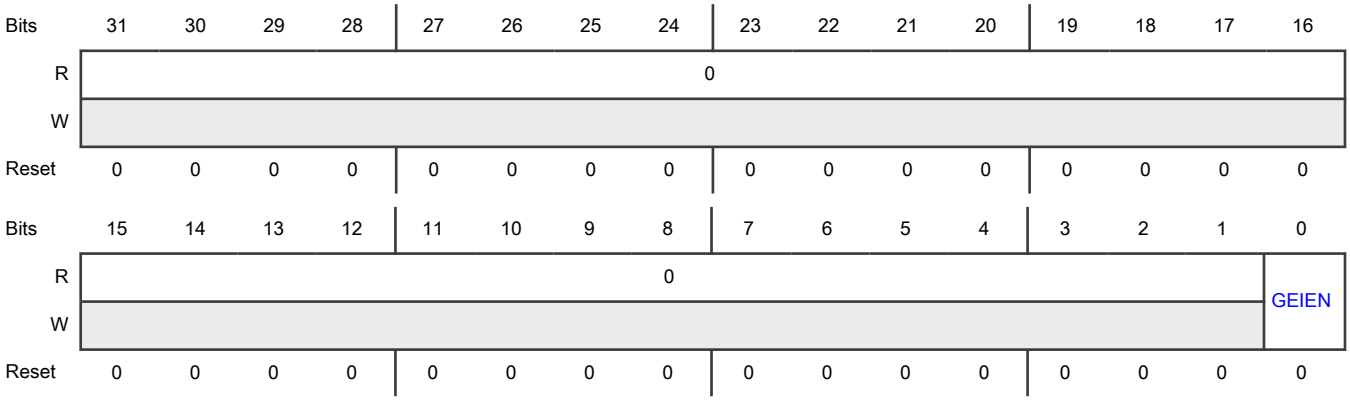
Offset

Register	Offset
EIMCR	0h

Function

The EIM Configuration Register is used to globally enable/disable the error injection function.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 GEIEN	Global Error Injection Enable This bit globally enables or disables the error injection function of the EIM. This field is initialized by hardware reset. 0b - Disabled 1b - Enabled

50.7.3 Error Injection Channel Enable register (EICHEN)

Offset

Register	Offset
EICHEN	4h

Function

Each field of the Error Injection Channel Enable register (EICHEN) is used to enable or disable the corresponding error injection channel.

NOTE

To enable an error injection channel, the Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	EICH0EN	EICH1EN	EICH2EN	EICH3EN	EICH4EN	EICH5EN	0	EICH7EN	EICH8EN	EICH9EN	EICH10EN	EICH11EN	EICH12EN	EICH13EN	EICH14EN	EICH15EN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EICH16EN	EICH17EN	EICH18EN	EICH19EN	EICH20EN	EICH21EN	EICH22EN	EICH23EN	EICH24EN	EICH25EN	EICH26EN	EICH27EN	EICH28EN	EICH29EN	EICH30EN	EICH31EN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 EICH0EN	Error Injection Channel 0 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 0 1b - Error injection is enabled on Error Injection Channel 0
30 EICH1EN	Error Injection Channel 1 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 1 1b - Error injection is enabled on Error Injection Channel 1
29 EICH2EN	Error Injection Channel 2 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 2 1b - Error injection is enabled on Error Injection Channel 2
28 EICH3EN	Error Injection Channel 3 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 3 1b - Error injection is enabled on Error Injection Channel 3
27 EICH4EN	Error Injection Channel 4 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 4 1b - Error injection is enabled on Error Injection Channel 4
26 EICH5EN	Error Injection Channel 5 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 5 1b - Error injection is enabled on Error Injection Channel 5
25 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
24 EICH7EN	Error Injection Channel 7 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 7 1b - Error injection is enabled on Error Injection Channel 7
23 EICH8EN	Error Injection Channel 8 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 8 1b - Error injection is enabled on Error Injection Channel 8
22 EICH9EN	Error Injection Channel 9 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 9 1b - Error injection is enabled on Error Injection Channel 9
21 EICH10EN	Error Injection Channel 10 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Error injection is disabled on Error Injection Channel 10 1b - Error injection is enabled on Error Injection Channel 10
20 EICH11EN	Error Injection Channel 11 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 11 1b - Error injection is enabled on Error Injection Channel 11
19 EICH12EN	Error Injection Channel 12 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 12 1b - Error injection is enabled on Error Injection Channel 12
18 EICH13EN	Error Injection Channel 13 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 13 1b - Error injection is enabled on Error Injection Channel 13
17 EICH14EN	Error Injection Channel 14 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 14 1b - Error injection is enabled on Error Injection Channel 14
16 EICH15EN	Error Injection Channel 15 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 15 1b - Error injection is enabled on Error Injection Channel 15
15 EICH16EN	Error Injection Channel 16 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 16 1b - Error injection is enabled on Error Injection Channel 16
14 EICH17EN	Error Injection Channel 17 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 17 1b - Error injection is enabled on Error Injection Channel 17

Table continues on the next page...

Table continued from the previous page...

Field	Function
13 EICH18EN	Error Injection Channel 18 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 18 1b - Error injection is enabled on Error Injection Channel 18
12 EICH19EN	Error Injection Channel 19 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 19 1b - Error injection is enabled on Error Injection Channel 19
11 EICH20EN	Error Injection Channel 20 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 20 1b - Error injection is enabled on Error Injection Channel 20
10 EICH21EN	Error Injection Channel 21 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Error injection is disabled on Error Injection Channel 21 1b - Error injection is enabled on Error Injection Channel 21
9 EICH22EN	Error Injection Channel 22 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 22 1b - Error injection is enabled on Error Injection Channel 22
8 EICH23EN	Error Injection Channel 23 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 23 1b - Error injection is enabled on Error Injection Channel 23
7 EICH24EN	Error Injection Channel 24 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 24 1b - Error injection is enabled on Error Injection Channel 24
6 EICH25EN	Error Injection Channel 25 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 25 1b - Error injection is enabled on Error Injection Channel 25
5 EICH26EN	Error Injection Channel 26 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 26 1b - Error injection is enabled on Error Injection Channel 26
4 EICH27EN	Error Injection Channel 27 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 27 1b - Error injection is enabled on Error Injection Channel 27
3 EICH28EN	Error Injection Channel 28 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 28 1b - Error injection is enabled on Error Injection Channel 28

Table continues on the next page...

Table continued from the previous page...

Field	Function
2 EICH29EN	Error Injection Channel 29 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Optionally, a hardware error event from the logic being injected by this channel can also disable the injection if that feature is utilized at the instantiation of the EIM. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 29 1b - Error injection is enabled on Error Injection Channel 29
1 EICH30EN	Error Injection Channel 30 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Optionally, a hardware error event from the logic being injected by this channel can also disable the injection if that feature is utilized at the instantiation of the EIM. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 30 1b - Error injection is enabled on Error Injection Channel 30
0 EICH31EN	Error Injection Channel 31 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Optionally, a hardware error event from the logic being injected by this channel can also disable the injection if that feature is utilized at the instantiation of the EIM. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 31 1b - Error injection is enabled on Error Injection Channel 31

50.7.4 Error Injection Channel Descriptor n, Word0 (EICHD0_WORD0 - EICHD3_WORD0)

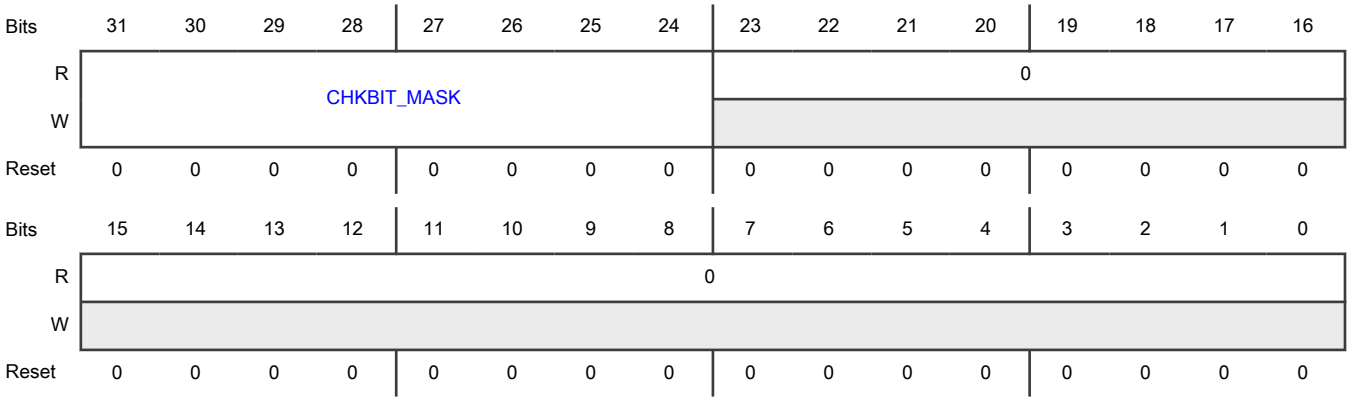
Offset

Register	Offset
EICHD0_WORD0	100h
EICHD1_WORD0	140h
EICHD2_WORD0	180h
EICHD3_WORD0	1C0h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-24 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[7:0] (8 bits wide), CHKBIT_MASK[7] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - The corresponding bit of the checkbit bus is inverted.
23-0 —	Reserved

50.7.5 Error Injection Channel Descriptor 0, Word1 (EICHD0_WORD1)

Offset

Register	Offset
EICHD0_WORD1	104h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD0_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	B0_3DATA_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	B0_3DATA_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details.

Table continues on the next page...

Field	Function
	0b - The corresponding bit of EICHD0_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD0_WORD1 on the read data bus is inverted.

50.7.6 Error Injection Channel Descriptor 0, Word2 (EICHD0_WORD2)

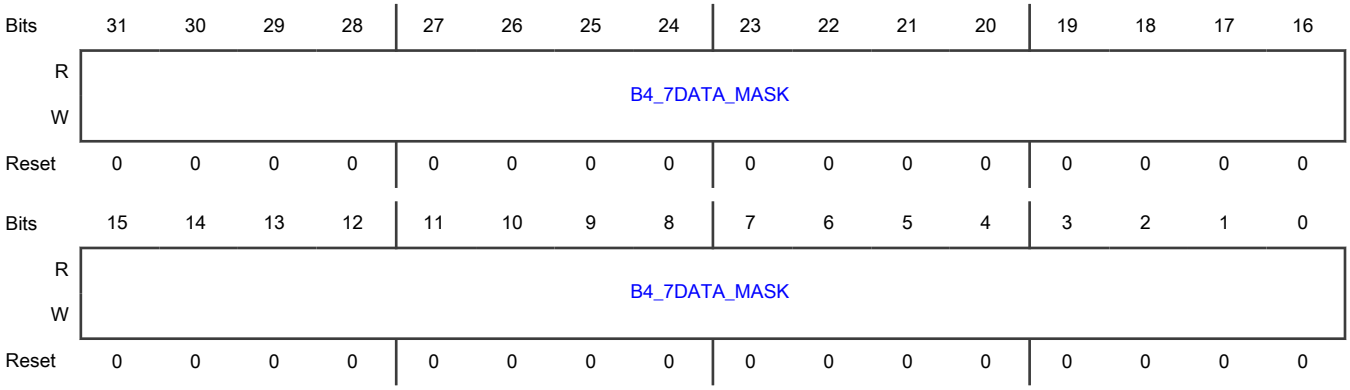
Offset

Register	Offset
EICHD0_WORD2	108h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to EICHD0_WORD2 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD0_WORD2 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD0_WORD2 on the read data bus is inverted.

50.7.7 Error Injection Channel Descriptor 1, Word1 (EICHD1_WORD1)

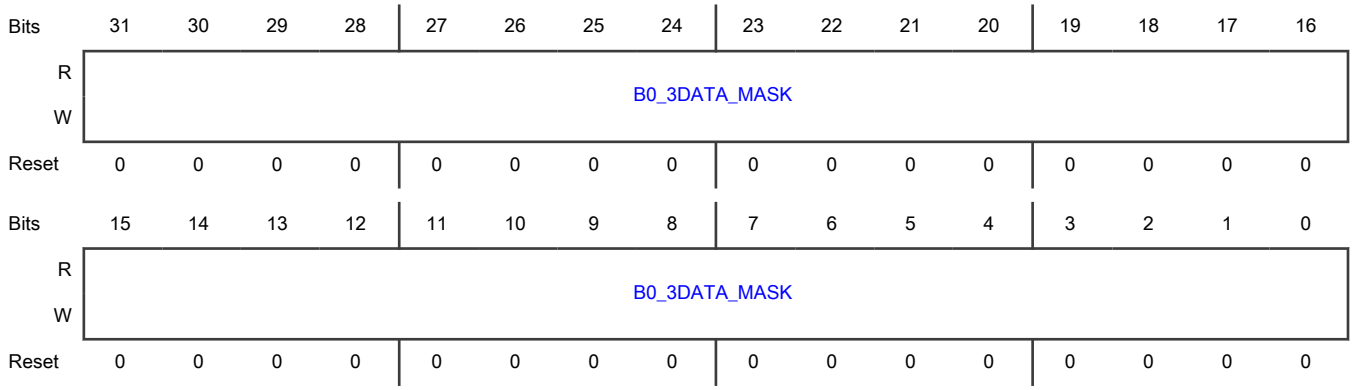
Offset

Register	Offset
EICHD1_WORD1	144h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD1_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD1_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD1_WORD1 on the read data bus is inverted.

50.7.8 Error Injection Channel Descriptor 1, Word2 (EICHD1_WORD2)

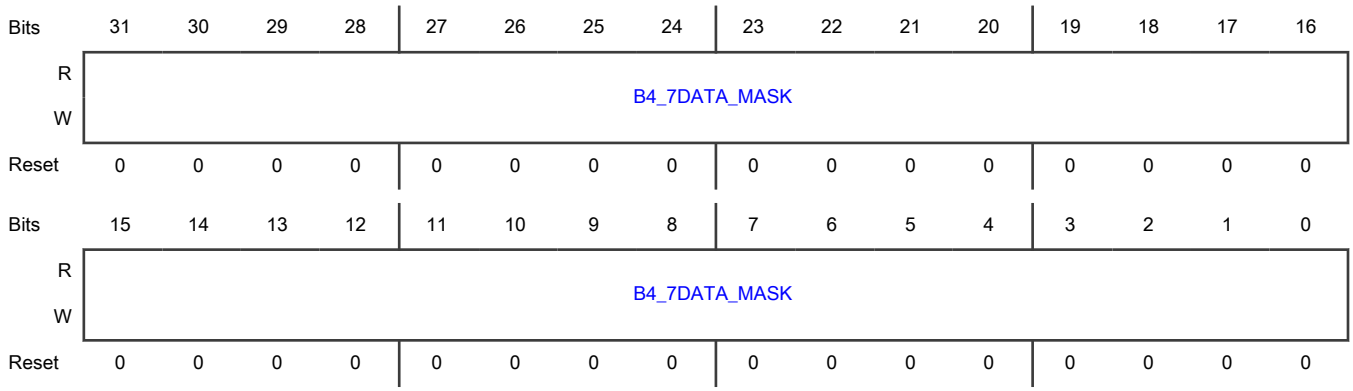
Offset

Register	Offset
EICHD1_WORD2	148h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to EICHD1_WORD2 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD1_WORD2 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD1_WORD2 on the read data bus is inverted.

50.7.9 Error Injection Channel Descriptor 2, Word1 (EICHD2_WORD1)

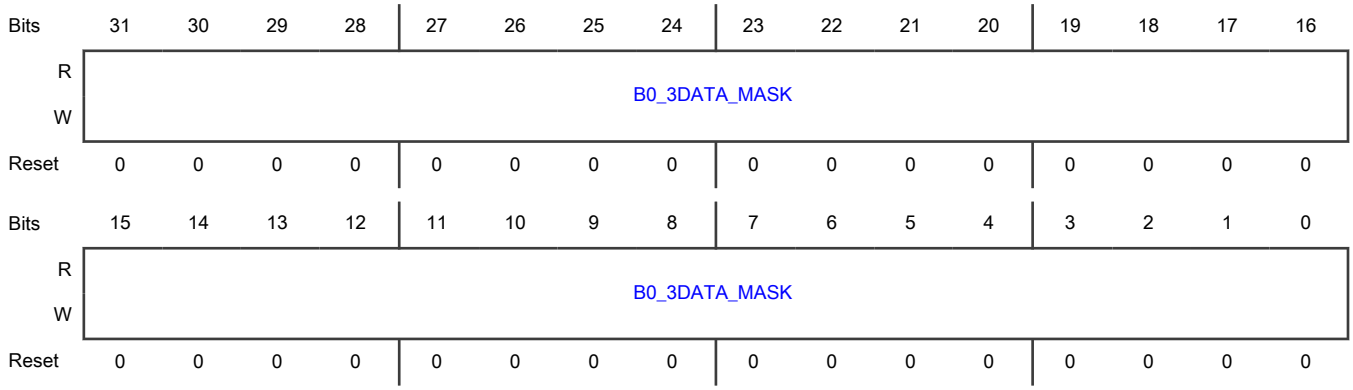
Offset

Register	Offset
EICHD2_WORD1	184h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD2_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD2_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD2_WORD1 on the read data bus is inverted.

50.7.10 Error Injection Channel Descriptor 2, Word2 (EICHD2_WORD2)

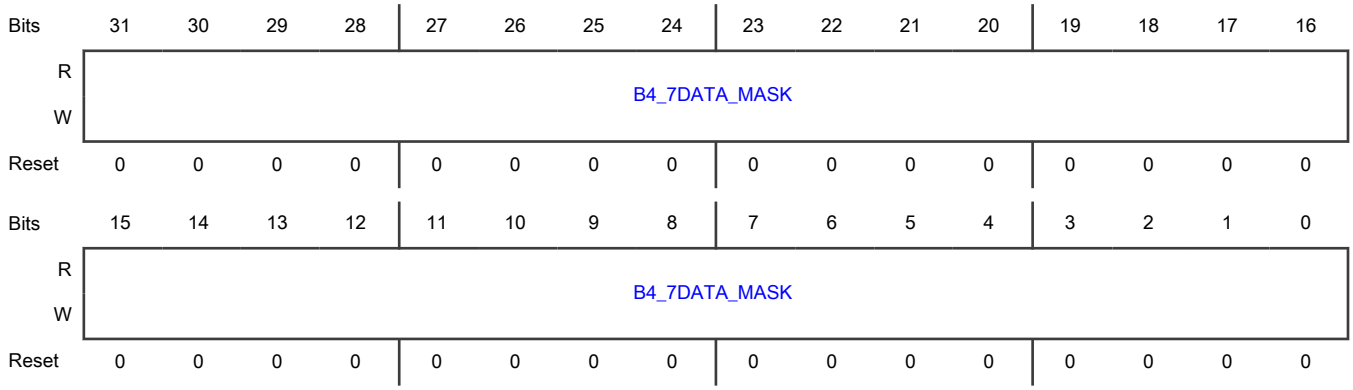
Offset

Register	Offset
EICHD2_WORD2	188h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to EICHD2_WORD2 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD2_WORD2 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD2_WORD2 on the read data bus is inverted.

50.7.11 Error Injection Channel Descriptor 3, Word1 (EICHD3_WORD1)

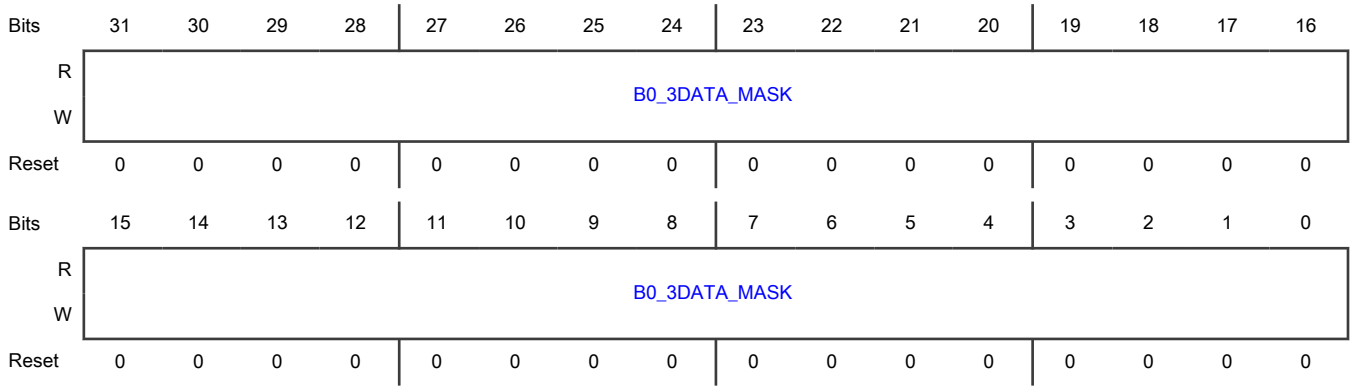
Offset

Register	Offset
EICHD3_WORD1	1C4h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD3_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD3_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD3_WORD1 on the read data bus is inverted.

50.7.12 Error Injection Channel Descriptor 3, Word2 (EICHD3_WORD2)

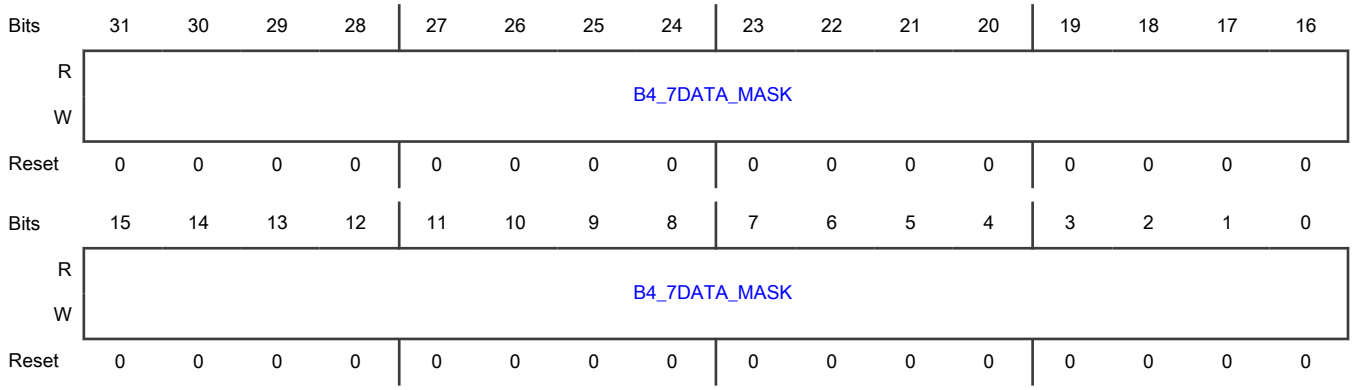
Offset

Register	Offset
EICHD3_WORD2	1C8h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to EICHD3_WORD2 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD3_WORD2 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD3_WORD2 on the read data bus is inverted.

50.7.13 Error Injection Channel Descriptor 4, Word1 (EICHD4_WORD1)

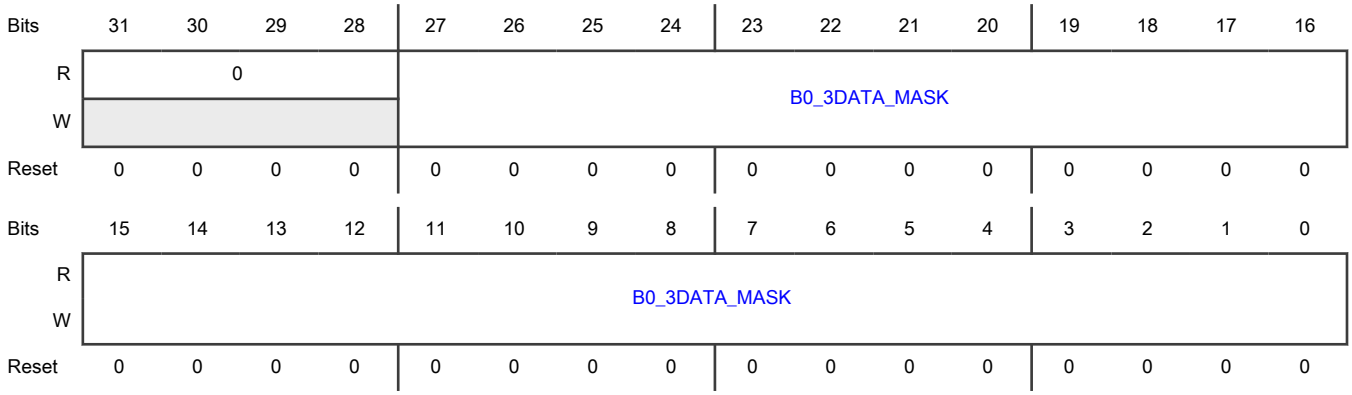
Offset

Register	Offset
EICHD4_WORD1	204h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD4_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-28 —	Reserved
27-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD4_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD4_WORD1 on the read data bus is inverted.

50.7.14 Error Injection Channel Descriptor 4, Word2 (EICHD4_WORD2)

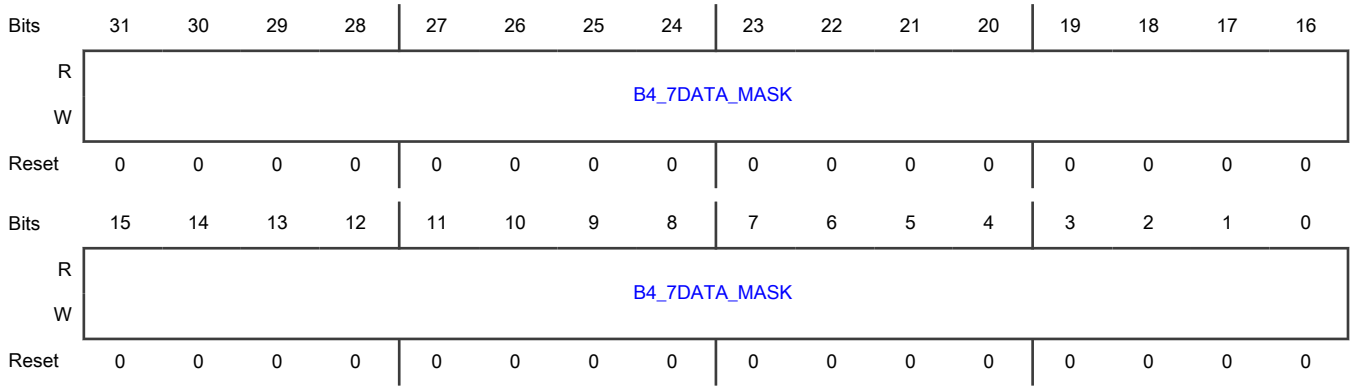
Offset

Register	Offset
EICHD4_WORD2	208h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to EICHD4_WORD2 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD4_WORD2 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD4_WORD2 on the read data bus is inverted.

50.7.15 Error Injection Channel Descriptor 5, Word1 (EICHD5_WORD1)

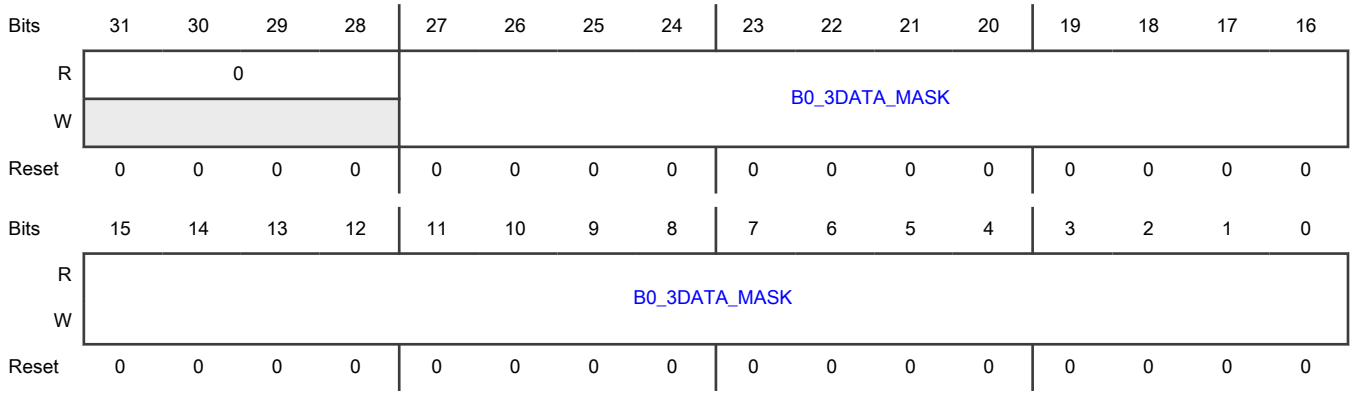
Offset

Register	Offset
EICHD5_WORD1	244h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD5_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-28 —	Reserved
27-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD5_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD5_WORD1 on the read data bus is inverted.

50.7.16 Error Injection Channel Descriptor 5, Word2 (EICHD5_WORD2)

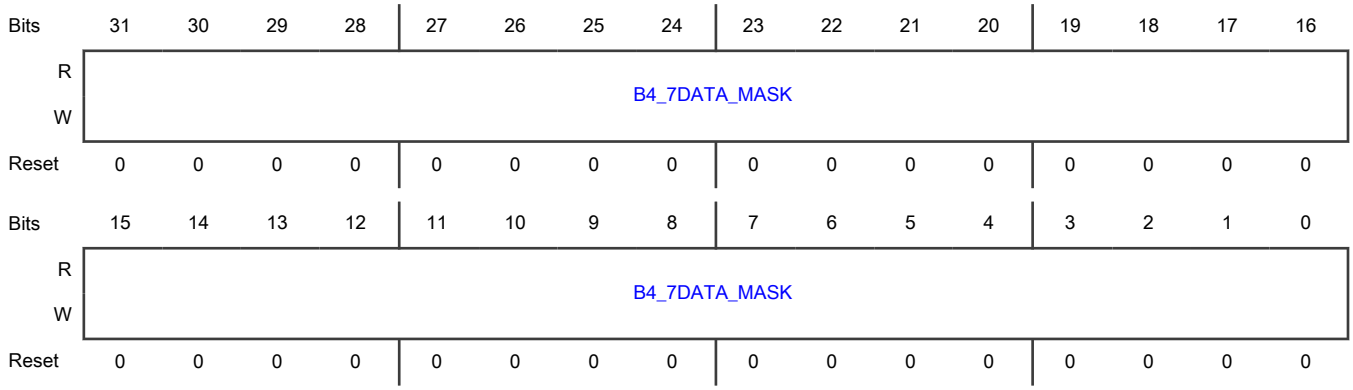
Offset

Register	Offset
EICHD5_WORD2	248h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to EICHD5_WORD2 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD5_WORD2 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD5_WORD2 on the read data bus is inverted.

50.7.17 Error Injection Channel Descriptor 7, Word1 (EICHD7_WORD1)

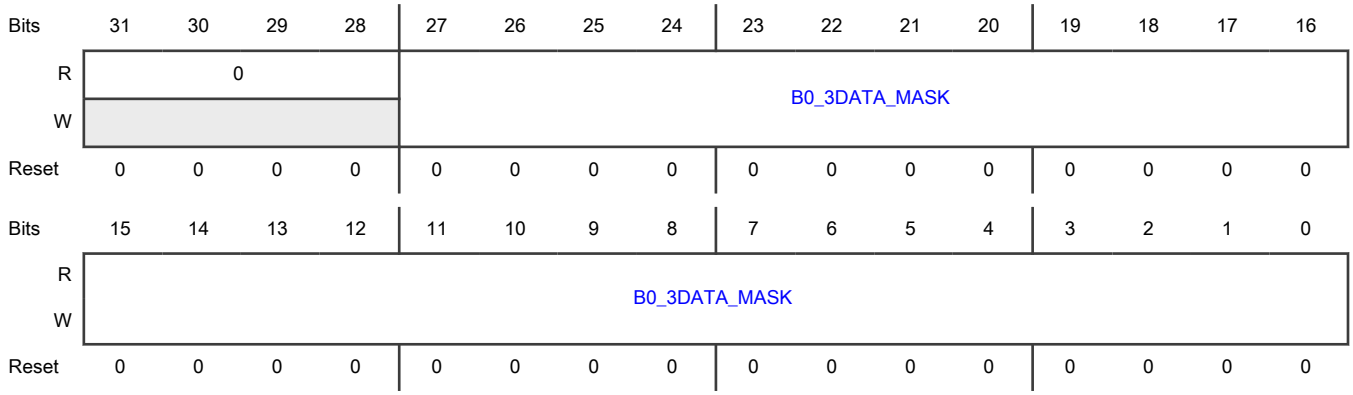
Offset

Register	Offset
EICHD7_WORD1	2C4h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD7_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-28 —	Reserved
27-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD7_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD7_WORD1 on the read data bus is inverted.

50.7.18 Error Injection Channel Descriptor 7, Word2 (EICHD7_WORD2)

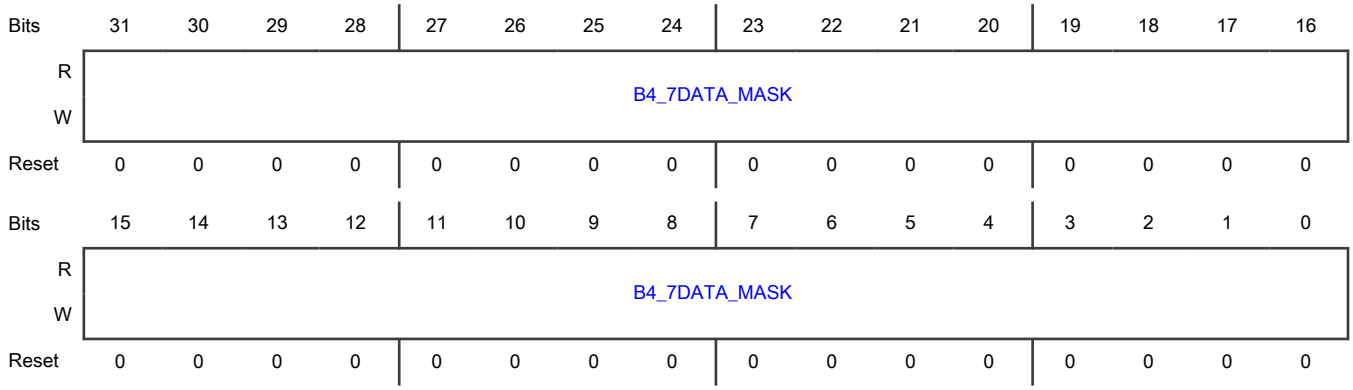
Offset

Register	Offset
EICHD7_WORD2	2C8h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to EICHD7_WORD2 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD7_WORD2 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD7_WORD2 on the read data bus is inverted.

50.7.19 Error Injection Channel Descriptor 8, Word1 (EICHD8_WORD1)

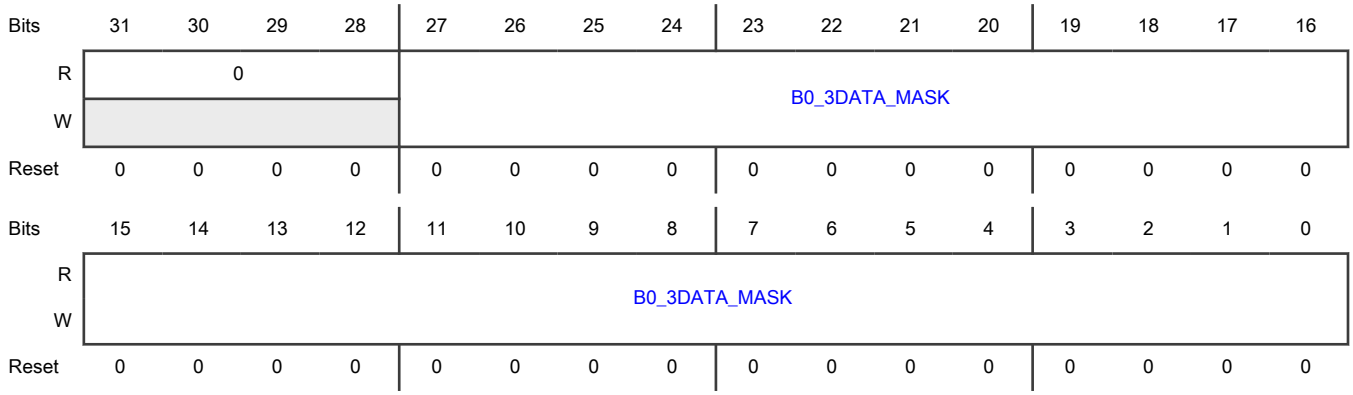
Offset

Register	Offset
EICHD8_WORD1	304h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD8_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-28 —	Reserved
27-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD8_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD8_WORD1 on the read data bus is inverted.

50.7.20 Error Injection Channel Descriptor 8, Word2 (EICHD8_WORD2)

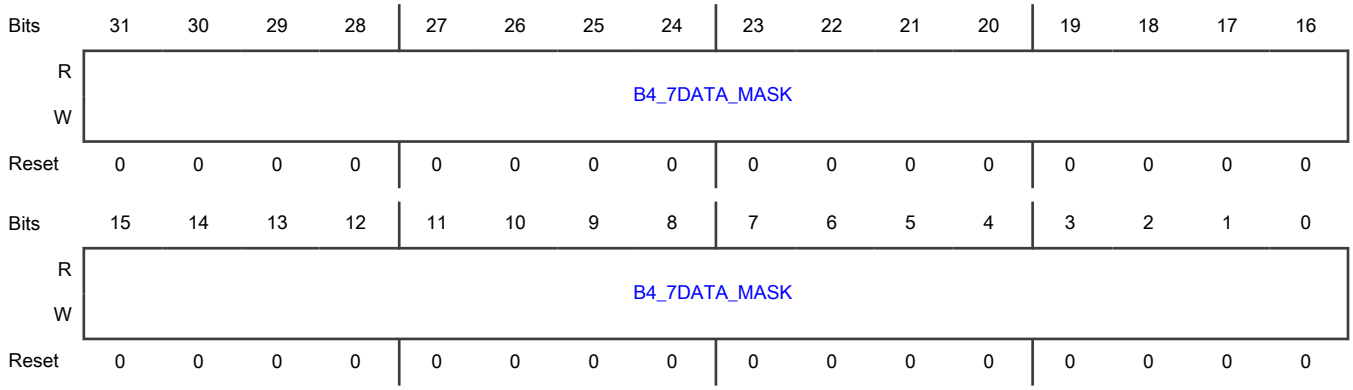
Offset

Register	Offset
EICHD8_WORD2	308h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to EICHD8_WORD2 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD8_WORD2 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD8_WORD2 on the read data bus is inverted.

50.7.21 Error Injection Channel Descriptor 9, Word1 (EICHD9_WORD1)

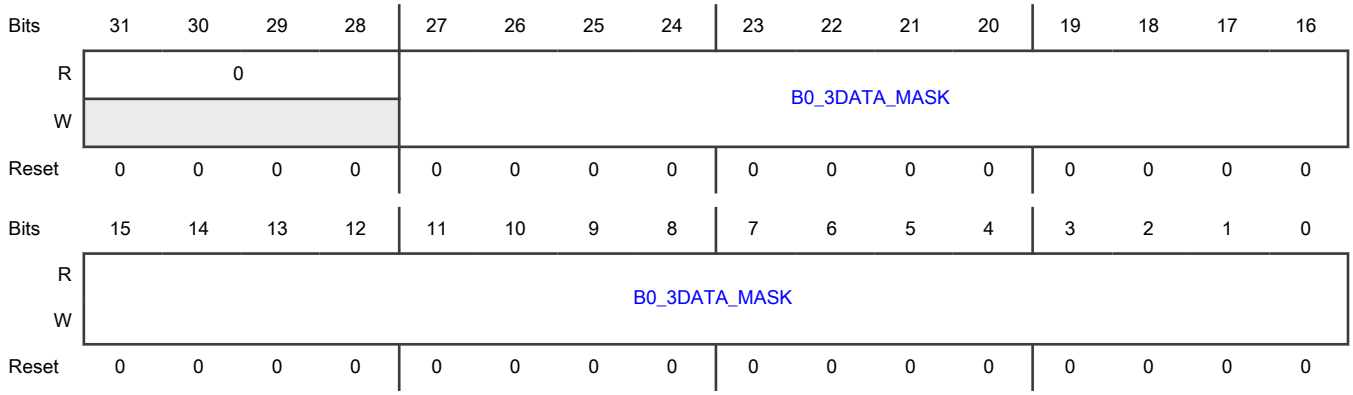
Offset

Register	Offset
EICHD9_WORD1	344h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD9_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-28 —	Reserved
27-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD9_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD9_WORD1 on the read data bus is inverted.

50.7.22 Error Injection Channel Descriptor 9, Word2 (EICHD9_WORD2)

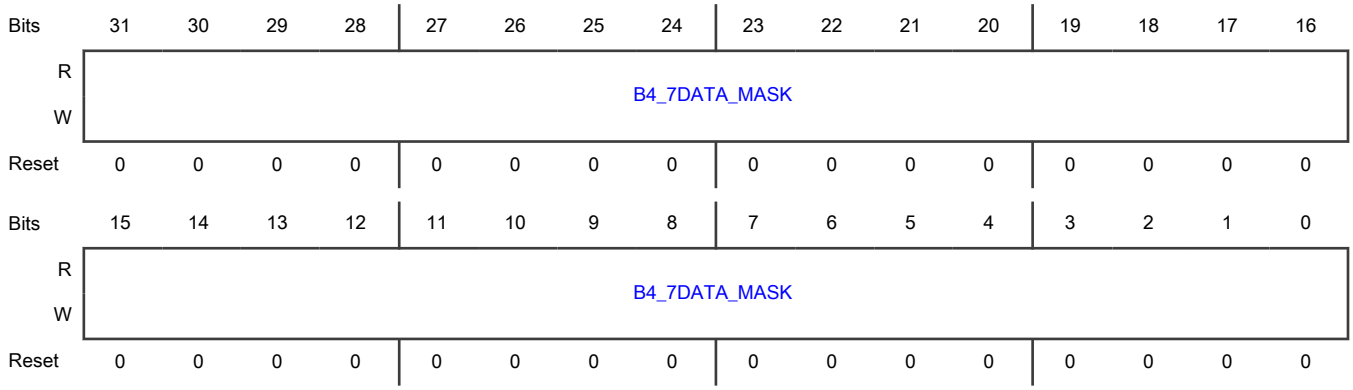
Offset

Register	Offset
EICHD9_WORD2	348h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to EICHD9_WORD2 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD9_WORD2 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD9_WORD2 on the read data bus is inverted.

50.7.23 Error Injection Channel Descriptor 10, Word1 (EICHD10_WORD1)

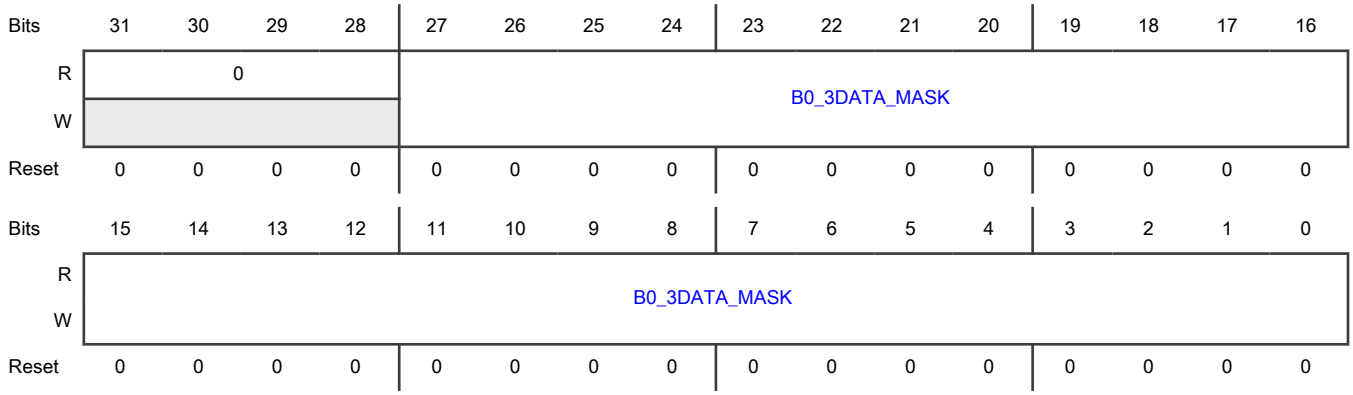
Offset

Register	Offset
EICHD10_WORD1	384h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD10_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-28 —	Reserved
27-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD10_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD10_WORD1 on the read data bus is inverted.

50.7.24 Error Injection Channel Descriptor 10, Word2 (EICHD10_WORD2)

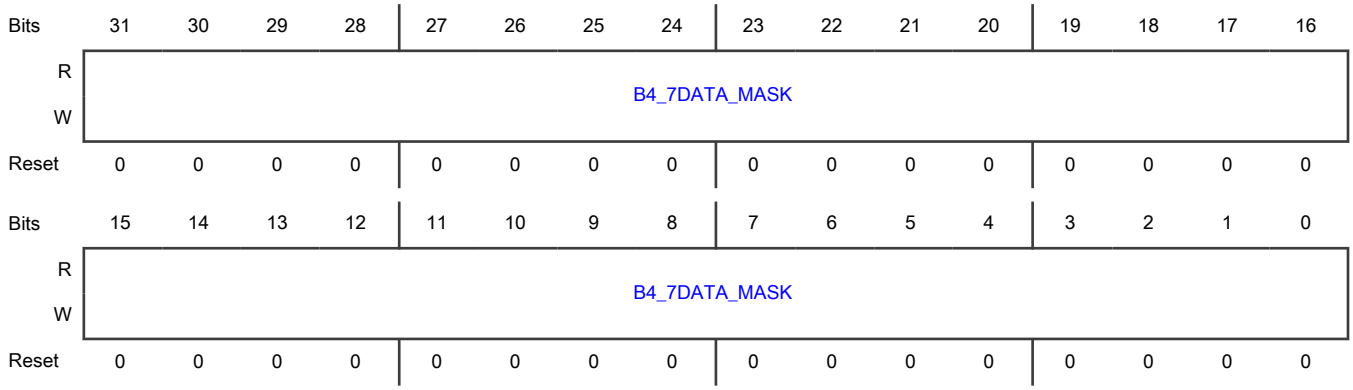
Offset

Register	Offset
EICHD10_WORD2	388h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to EICHD10_WORD2 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD10_WORD2 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD10_WORD2 on the read data bus is inverted.

50.7.25 Error Injection Channel Descriptor 11, Word1 (EICHD11_WORD1)

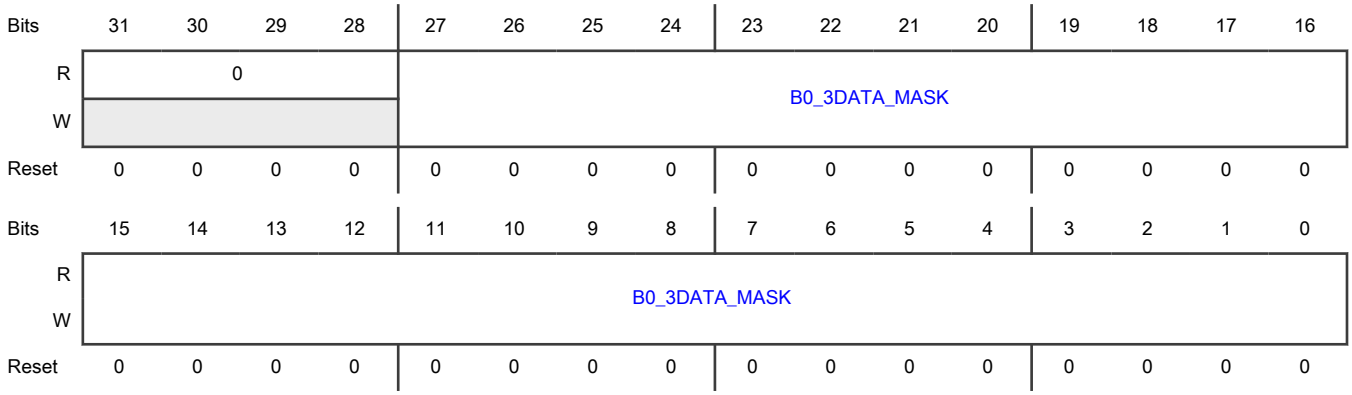
Offset

Register	Offset
EICHD11_WORD1	3C4h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD11_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-28 —	Reserved
27-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD11_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD11_WORD1 on the read data bus is inverted.

50.7.26 Error Injection Channel Descriptor 11, Word2 (EICHD11_WORD2)

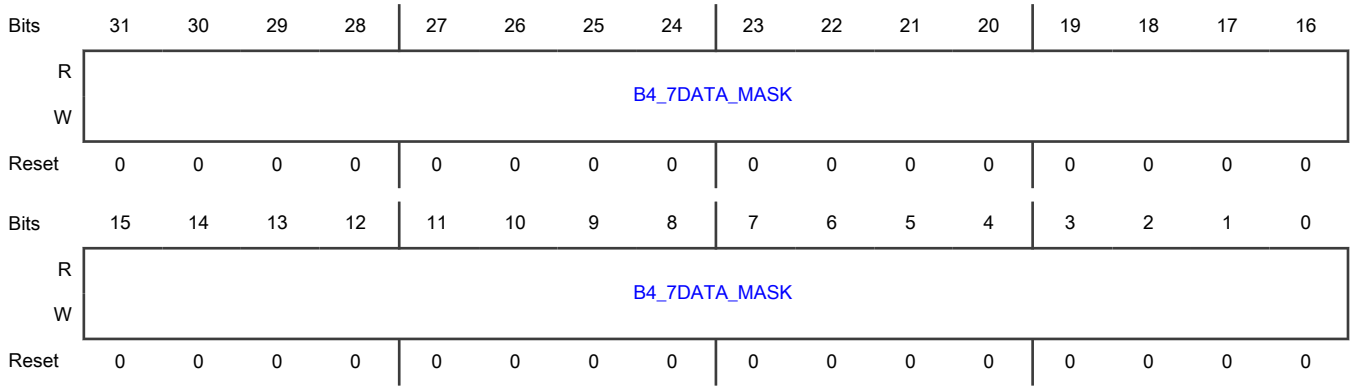
Offset

Register	Offset
EICHD11_WORD2	3C8h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to EICHD11_WORD2 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD11_WORD2 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD11_WORD2 on the read data bus is inverted.

50.7.27 Error Injection Channel Descriptor 12, Word1 (EICHD12_WORD1)

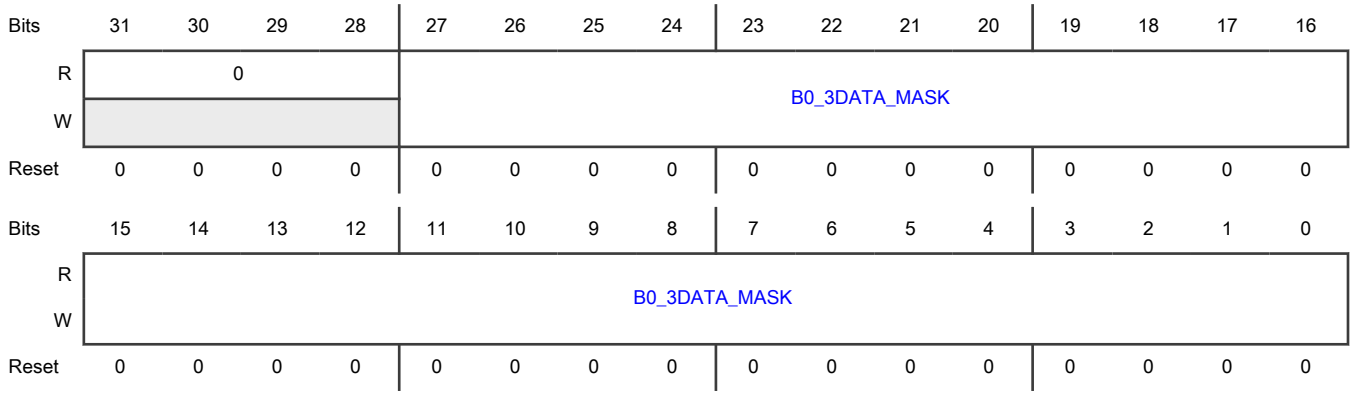
Offset

Register	Offset
EICHD12_WORD1	404h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD12_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-28 —	Reserved
27-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD12_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD12_WORD1 on the read data bus is inverted.

50.7.28 Error Injection Channel Descriptor 12, Word2 (EICHD12_WORD2)

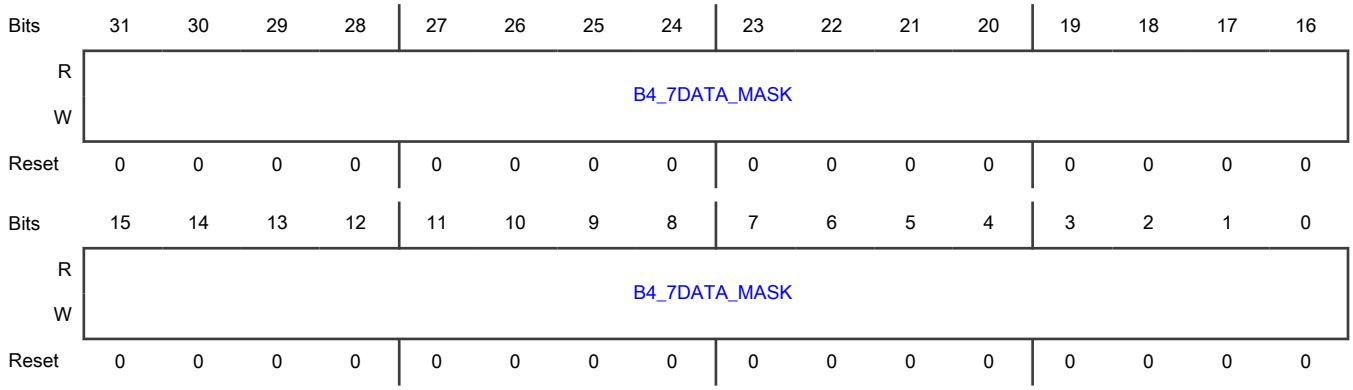
Offset

Register	Offset
EICHD12_WORD2	408h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to EICHD12_WORD2 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD12_WORD2 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD12_WORD2 on the read data bus is inverted.

50.7.29 Error Injection Channel Descriptor 13, Word1 (EICHD13_WORD1)

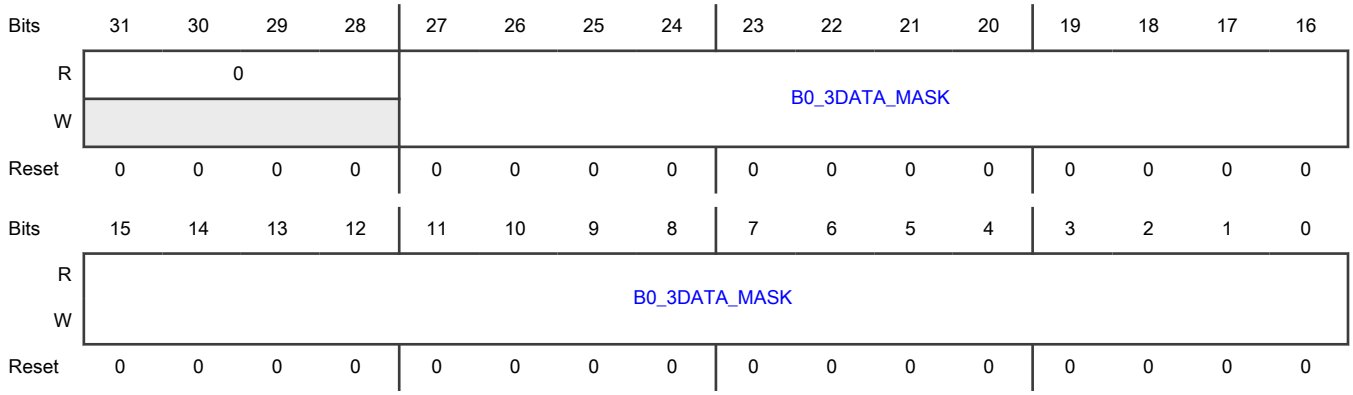
Offset

Register	Offset
EICHD13_WORD1	444h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD13_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-28 —	Reserved
27-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD13_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD13_WORD1 on the read data bus is inverted.

50.7.30 Error Injection Channel Descriptor 13, Word2 (EICHD13_WORD2)

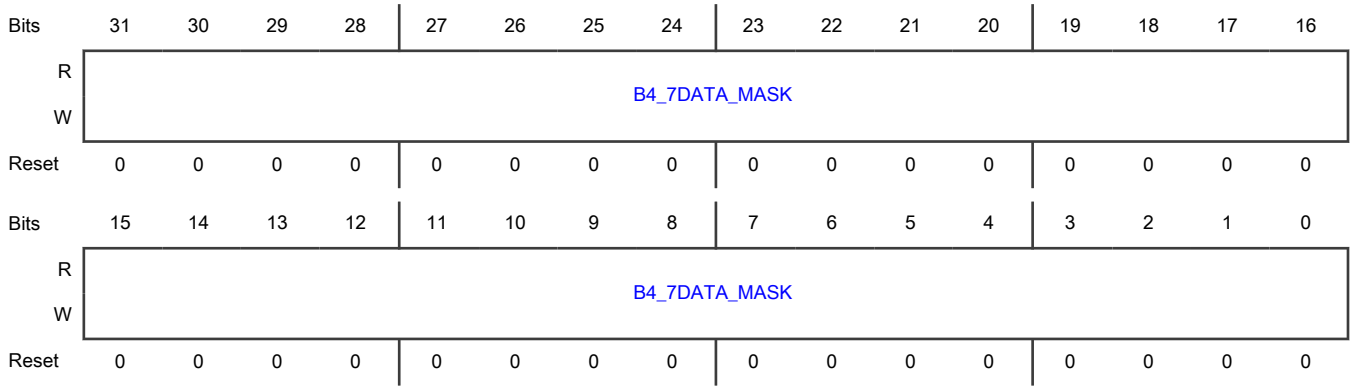
Offset

Register	Offset
EICHD13_WORD2	448h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to EICHD13_WORD2 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD13_WORD2 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD13_WORD2 on the read data bus is inverted.

50.7.31 Error Injection Channel Descriptor 14, Word1 (EICHD14_WORD1)

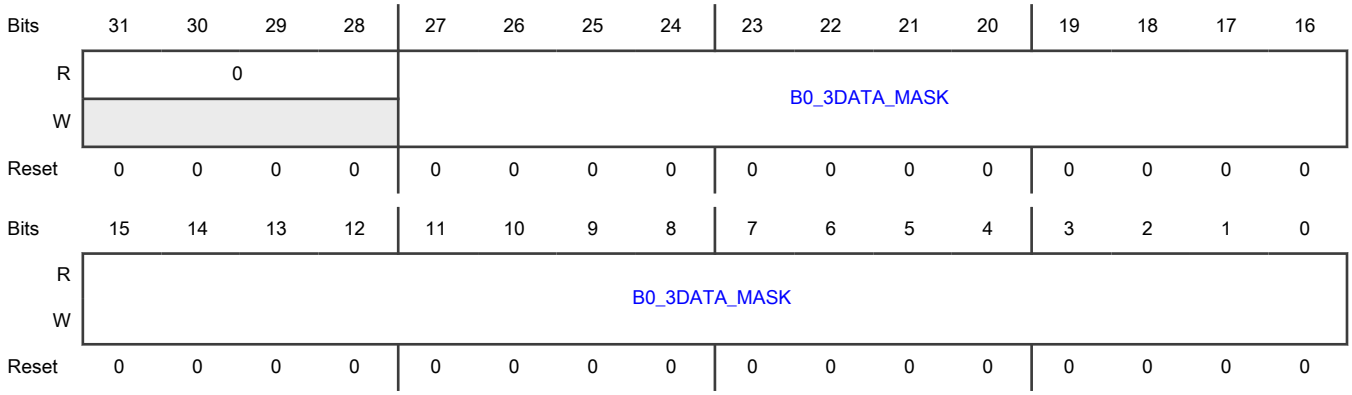
Offset

Register	Offset
EICHD14_WORD1	484h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD14_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-28 —	Reserved
27-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD14_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD14_WORD1 on the read data bus is inverted.

50.7.32 Error Injection Channel Descriptor 14, Word2 (EICHD14_WORD2)

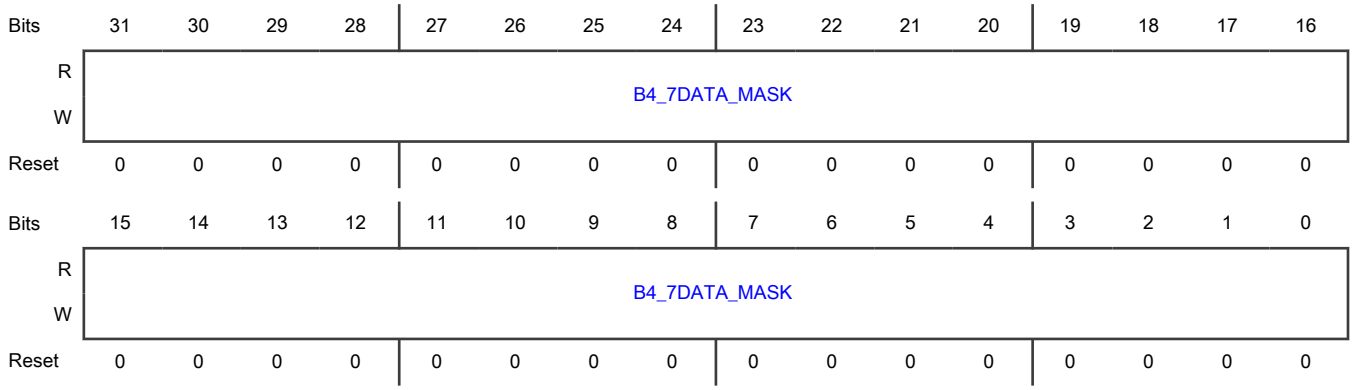
Offset

Register	Offset
EICHD14_WORD2	488h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to EICHD14_WORD2 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD14_WORD2 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD14_WORD2 on the read data bus is inverted.

50.7.33 Error Injection Channel Descriptor 15, Word1 (EICHD15_WORD1)

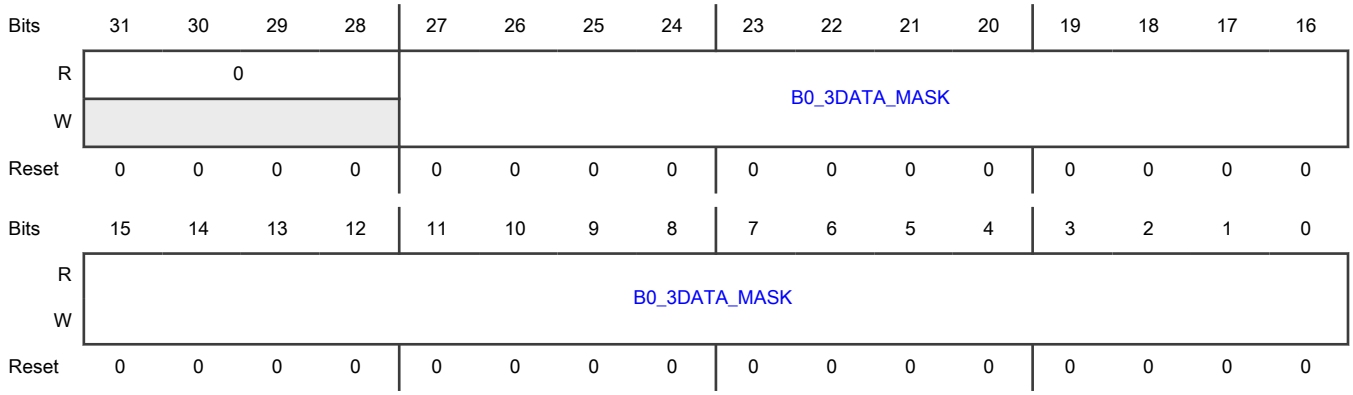
Offset

Register	Offset
EICHD15_WORD1	4C4h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD15_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-28 —	Reserved
27-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD15_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD15_WORD1 on the read data bus is inverted.

50.7.34 Error Injection Channel Descriptor 15, Word2 (EICHD15_WORD2)

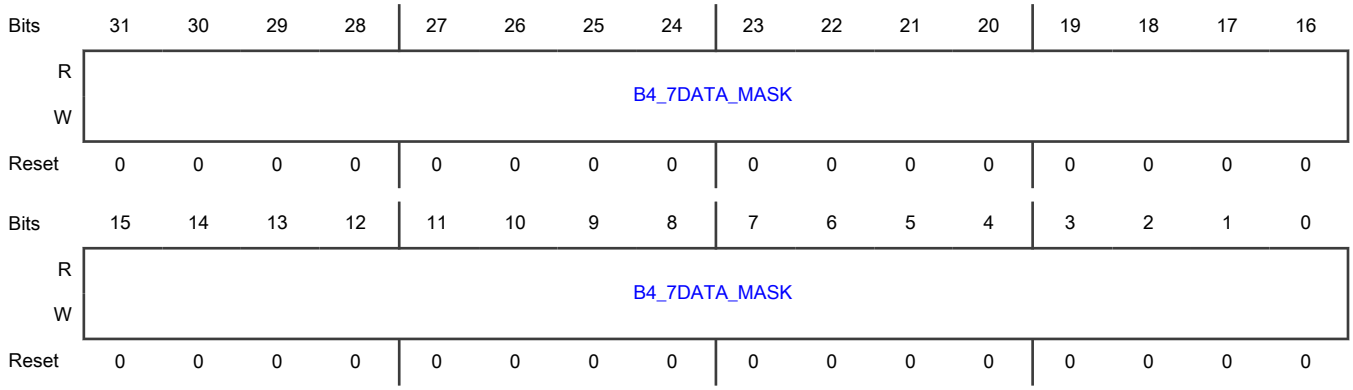
Offset

Register	Offset
EICHD15_WORD2	4C8h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to EICHD15_WORD2 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD15_WORD2 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD15_WORD2 on the read data bus is inverted.

50.7.35 Error Injection Channel Descriptor 16, Word1 (EICHD16_WORD1)

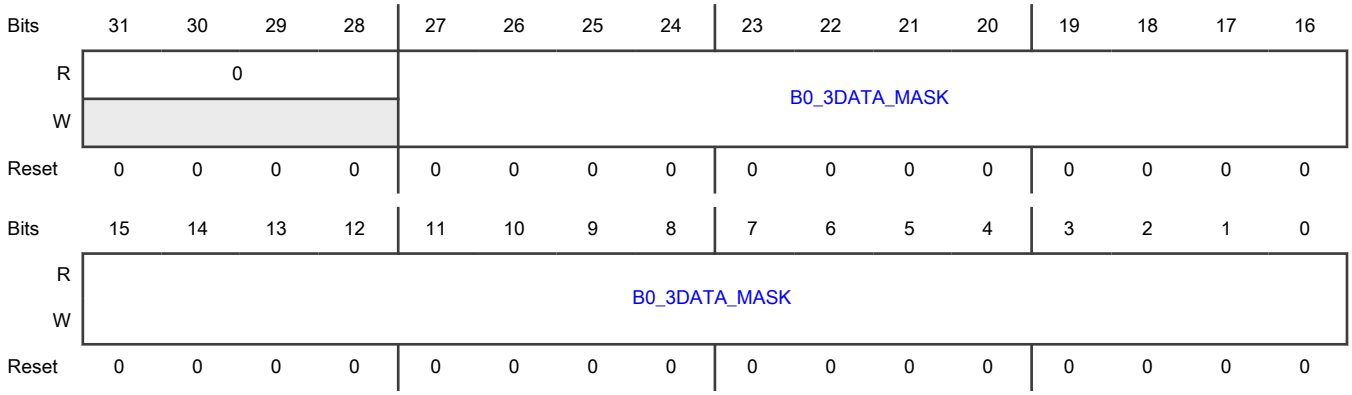
Offset

Register	Offset
EICHD16_WORD1	504h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD16_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-28 —	Reserved
27-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD16_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD16_WORD1 on the read data bus is inverted.

50.7.36 Error Injection Channel Descriptor 16, Word2 (EICHD16_WORD2)

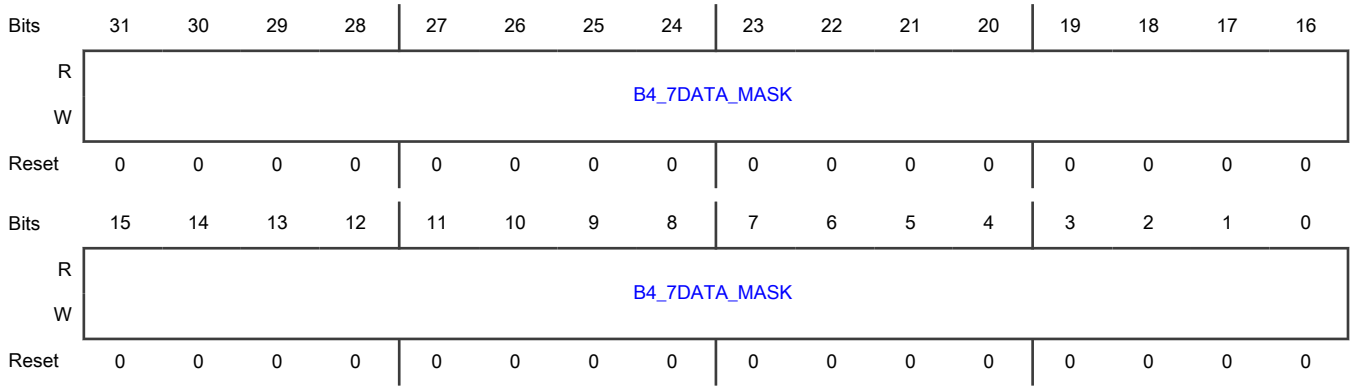
Offset

Register	Offset
EICHD16_WORD2	508h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to EICHD16_WORD2 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD16_WORD2 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD16_WORD2 on the read data bus is inverted.

50.7.37 Error Injection Channel Descriptor 17, Word1 (EICHD17_WORD1)

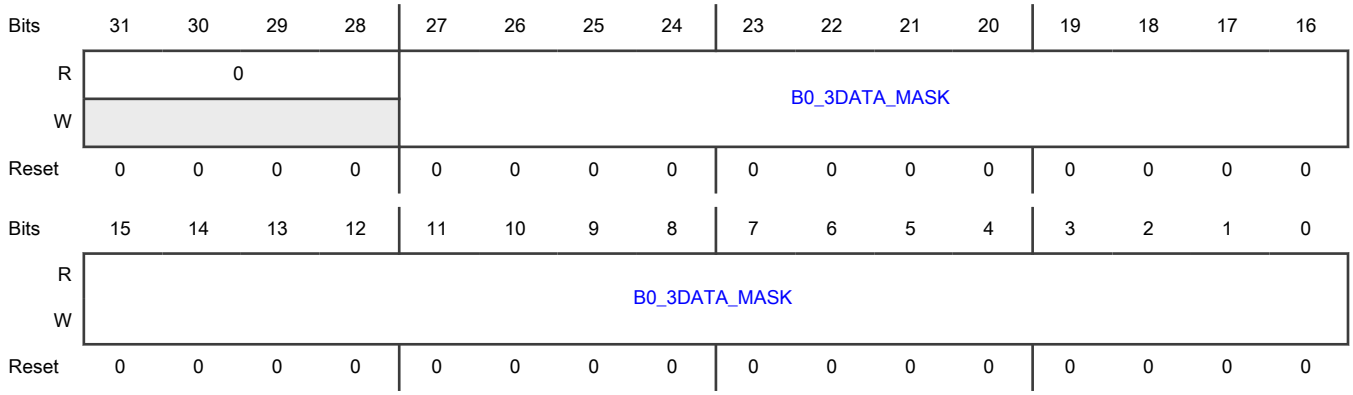
Offset

Register	Offset
EICHD17_WORD1	544h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD17_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-28 —	Reserved
27-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD17_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD17_WORD1 on the read data bus is inverted.

50.7.38 Error Injection Channel Descriptor 17, Word2 (EICHD17_WORD2)

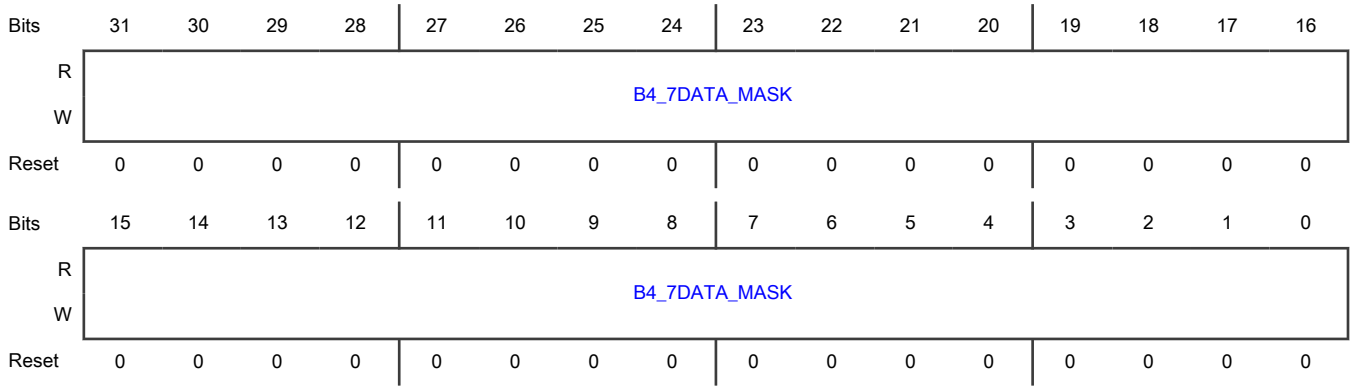
Offset

Register	Offset
EICHD17_WORD2	548h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to EICHD17_WORD2 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD17_WORD2 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD17_WORD2 on the read data bus is inverted.

50.7.39 Error Injection Channel Descriptor 18, Word1 (EICHD18_WORD1)

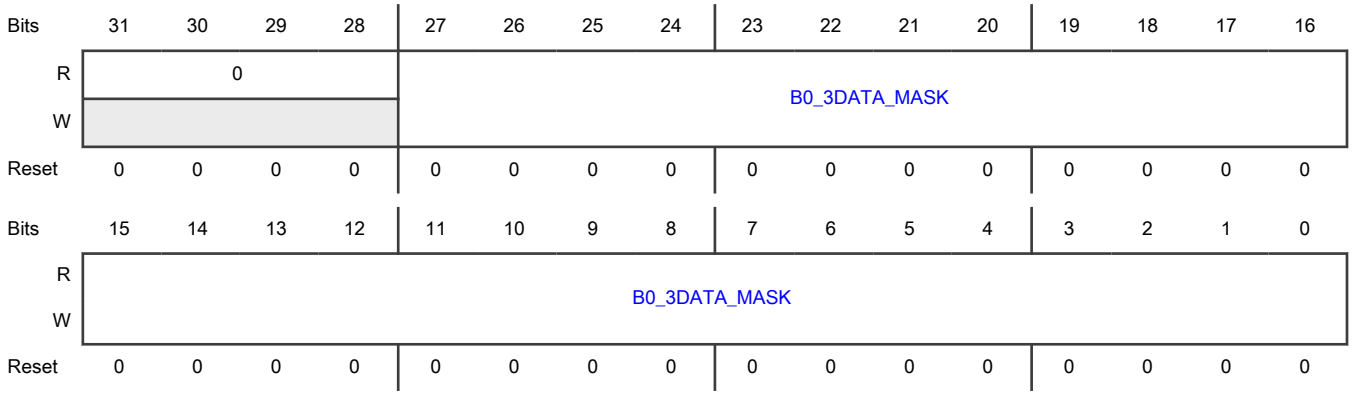
Offset

Register	Offset
EICHD18_WORD1	584h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD18_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-28 —	Reserved
27-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD18_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD18_WORD1 on the read data bus is inverted.

50.7.40 Error Injection Channel Descriptor 18, Word2 (EICHD18_WORD2)

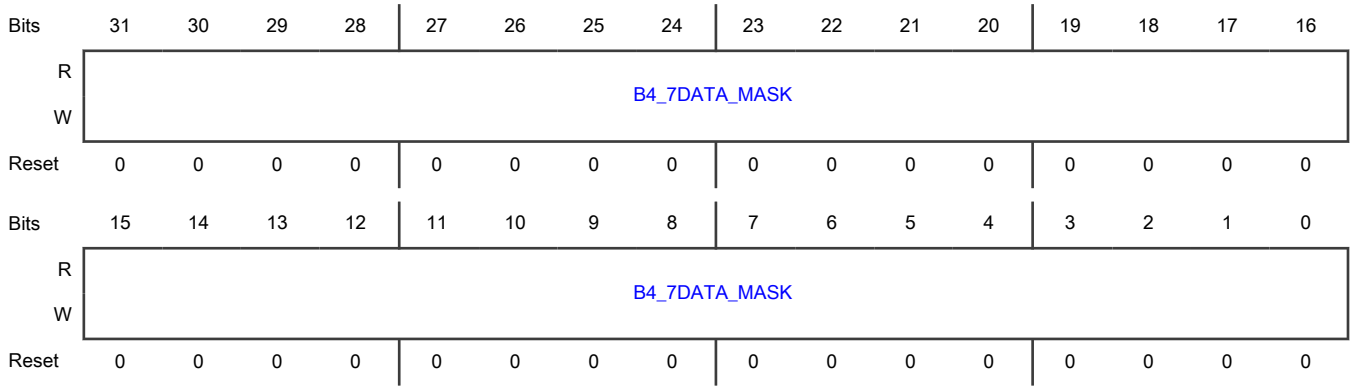
Offset

Register	Offset
EICHD18_WORD2	588h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to EICHD18_WORD2 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD18_WORD2 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD18_WORD2 on the read data bus is inverted.

50.7.41 Error Injection Channel Descriptor 19, Word1 (EICHD19_WORD1)

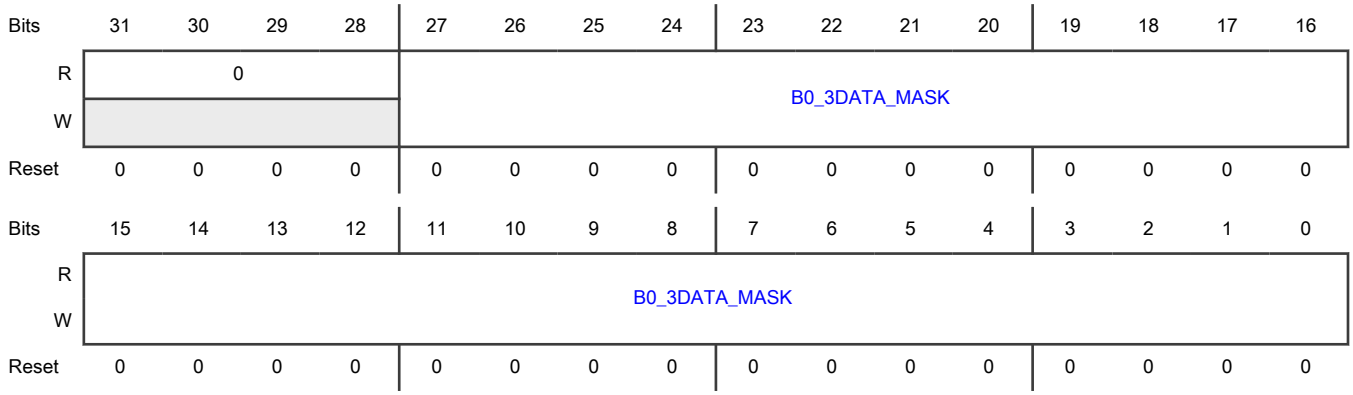
Offset

Register	Offset
EICHD19_WORD1	5C4h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD19_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-28 —	Reserved
27-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD19_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD19_WORD1 on the read data bus is inverted.

50.7.42 Error Injection Channel Descriptor 19, Word2 (EICHD19_WORD2)

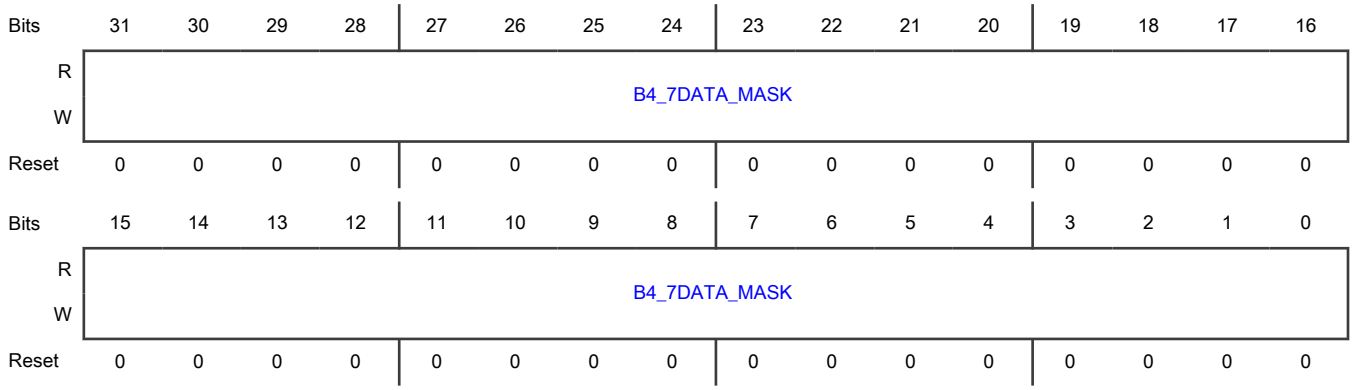
Offset

Register	Offset
EICHD19_WORD2	5C8h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to EICHD19_WORD2 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD19_WORD2 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD19_WORD2 on the read data bus is inverted.

50.7.43 Error Injection Channel Descriptor 20, Word1 (EICHD20_WORD1)

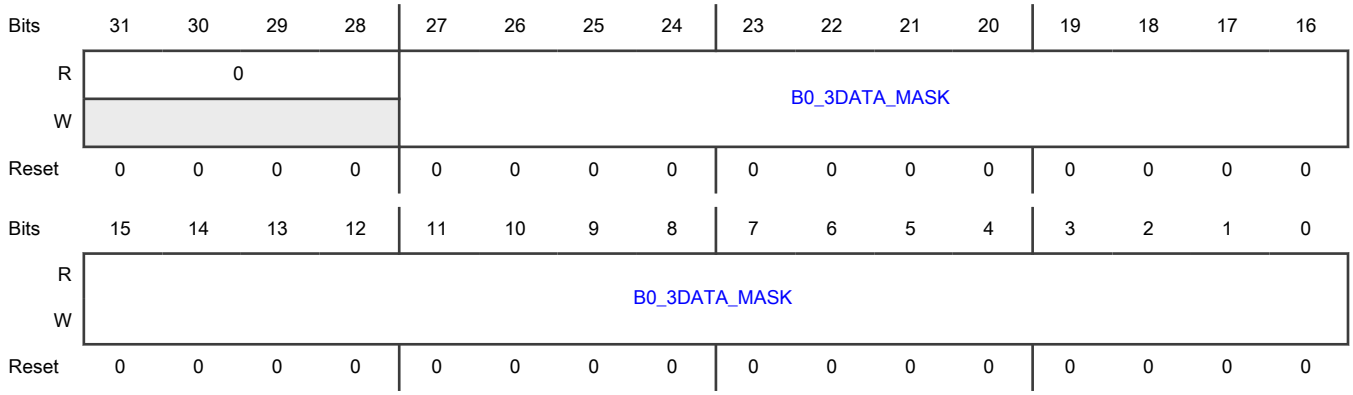
Offset

Register	Offset
EICHD20_WORD1	604h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD20_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-28 —	Reserved
27-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD20_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD20_WORD1 on the read data bus is inverted.

50.7.44 Error Injection Channel Descriptor 20, Word2 (EICHD20_WORD2)

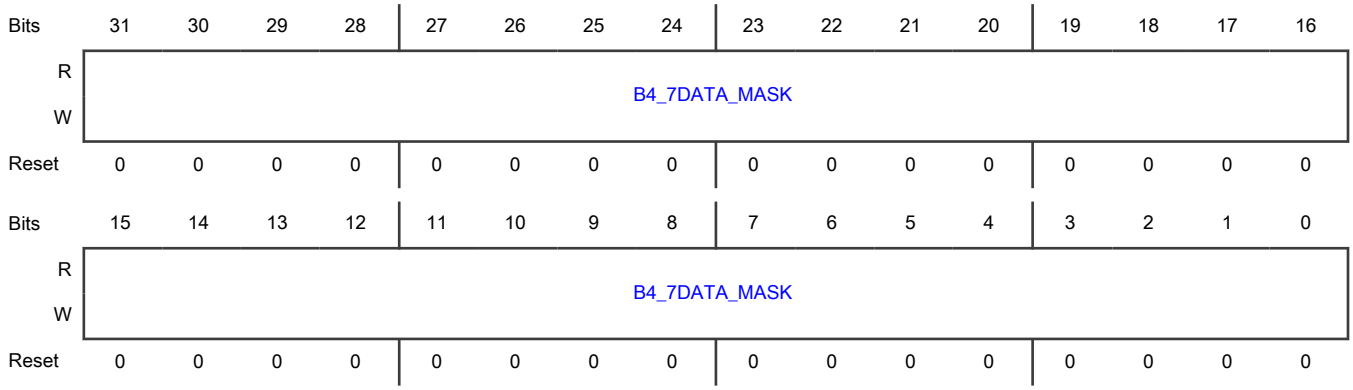
Offset

Register	Offset
EICHD20_WORD2	608h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to EICHD20_WORD2 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD20_WORD2 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD20_WORD2 on the read data bus is inverted.

50.7.45 Error Injection Channel Descriptor 21, Word1 (EICHD21_WORD1)

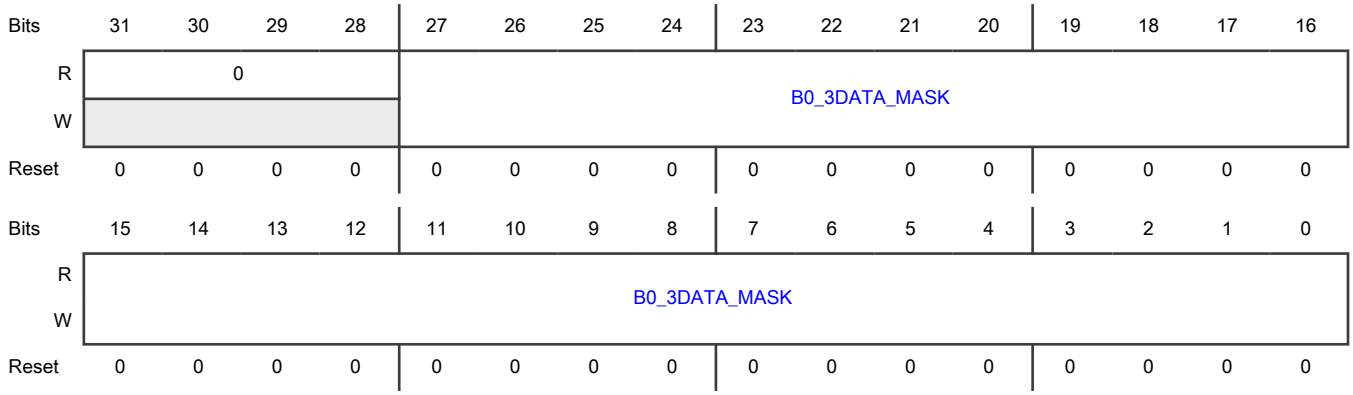
Offset

Register	Offset
EICHD21_WORD1	644h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD21_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-28 —	Reserved
27-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD21_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD21_WORD1 on the read data bus is inverted.

50.7.46 Error Injection Channel Descriptor 21, Word2 (EICHD21_WORD2)

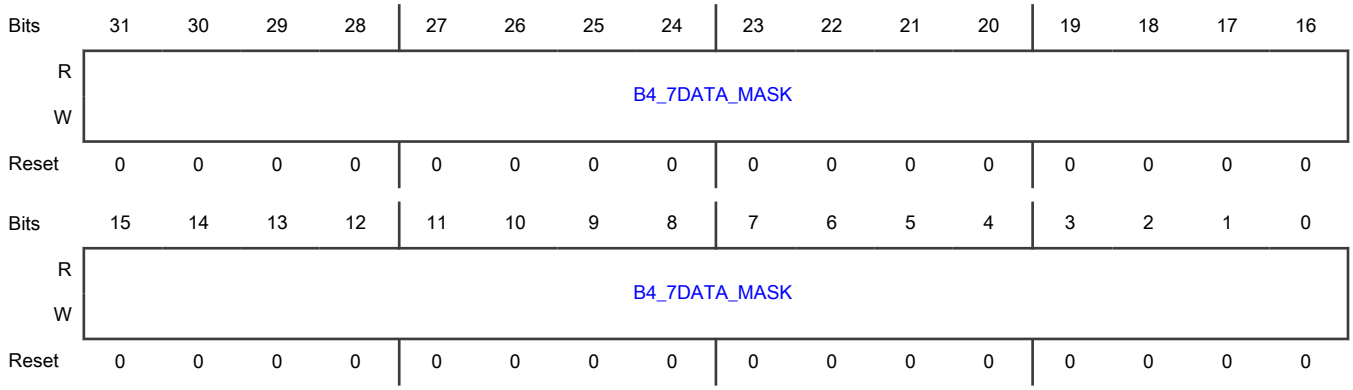
Offset

Register	Offset
EICHD21_WORD2	648h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to EICHD21_WORD2 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD21_WORD2 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD21_WORD2 on the read data bus is inverted.

50.7.47 Error Injection Channel Descriptor 22, Word1 (EICHD22_WORD1)

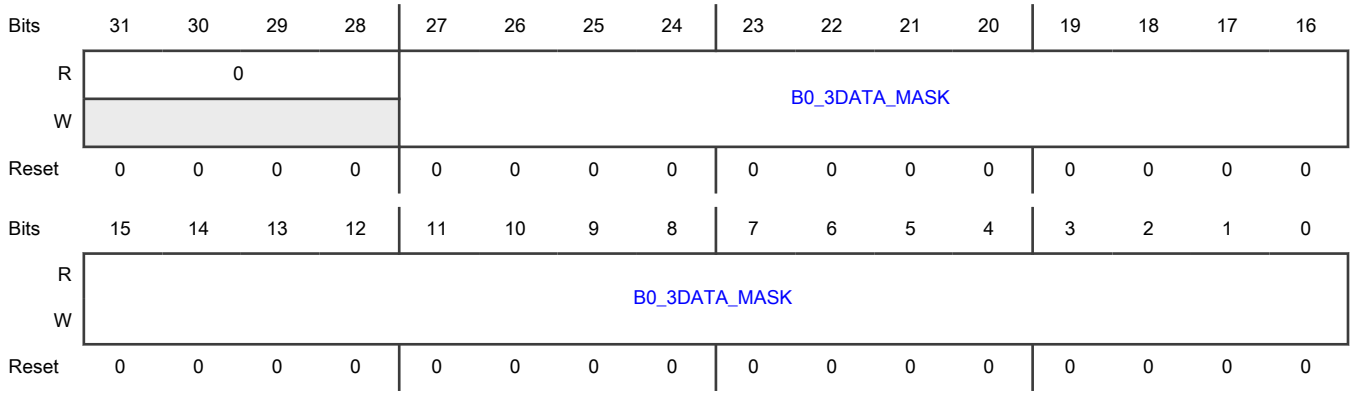
Offset

Register	Offset
EICHD22_WORD1	684h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD22_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-28 —	Reserved
27-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD22_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD22_WORD1 on the read data bus is inverted.

50.7.48 Error Injection Channel Descriptor 22, Word2 (EICHD22_WORD2)

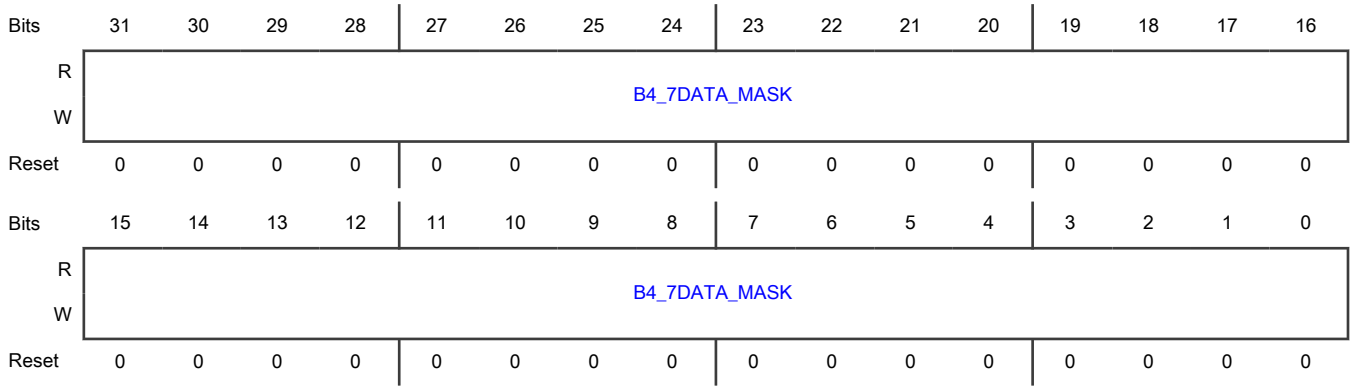
Offset

Register	Offset
EICHD22_WORD2	688h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to EICHD22_WORD2 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD22_WORD2 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD22_WORD2 on the read data bus is inverted.

50.7.49 Error Injection Channel Descriptor 22, Word3 (EICHD22_WORD3)

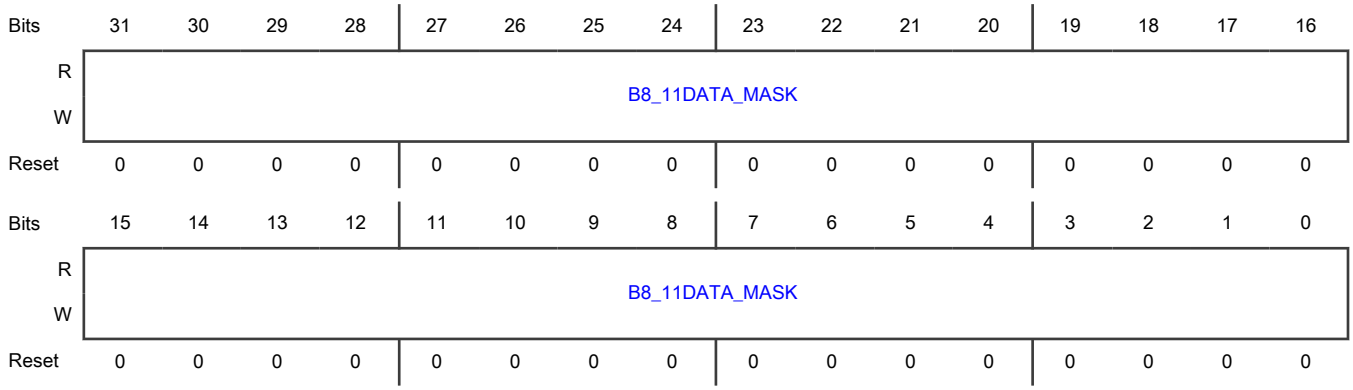
Offset

Register	Offset
EICHD22_WORD3	68Ch

Function

The fourth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B8_11DATA_MASK correspond to EICHD22_WORD3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B8_11DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B8_11DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD22_WORD3 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD22_WORD3 on the read data bus is inverted.

50.7.50 Error Injection Channel Descriptor 22, Word4 (EICHD22_WORD4)

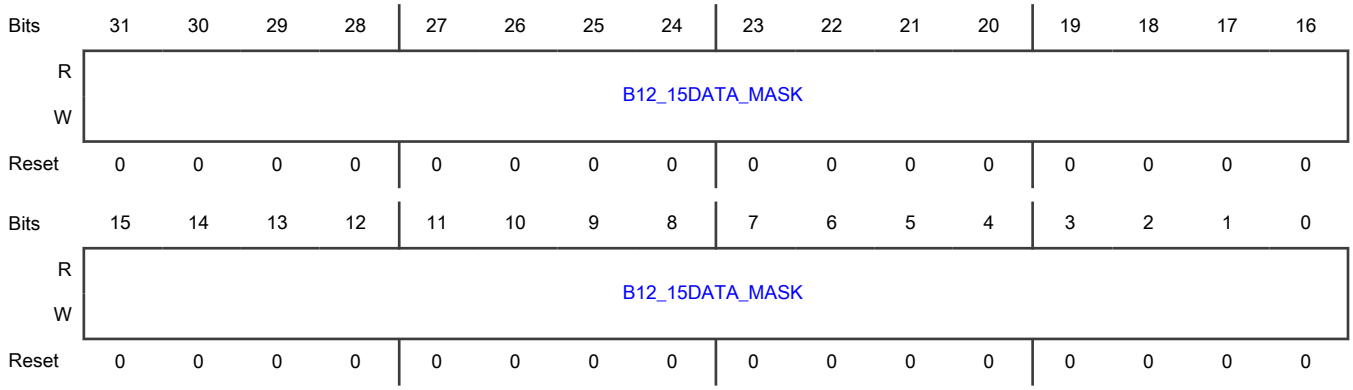
Offset

Register	Offset
EICHD22_WORD4	690h

Function

The fifth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B12_15DATA_MASK correspond to EICHD22_WORD4 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B12_15DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B12_15DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD22_WORD4 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD22_WORD4 on the read data bus is inverted.

50.7.51 Error Injection Channel Descriptor 22, Word5 (EICHD22_WORD5)

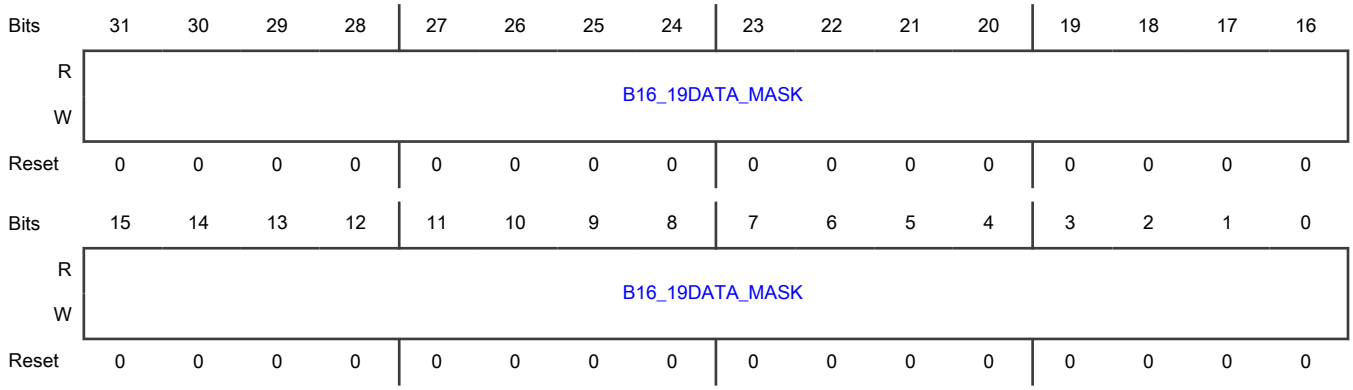
Offset

Register	Offset
EICHD22_WORD5	694h

Function

The sixth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B16_19DATA_MASK correspond to EICHD22_WORD5 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B16_19DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B16_19DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD22_WORD5 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD22_WORD5 on the read data bus is inverted.

50.7.52 Error Injection Channel Descriptor 22, Word6 (EICHD22_WORD6)

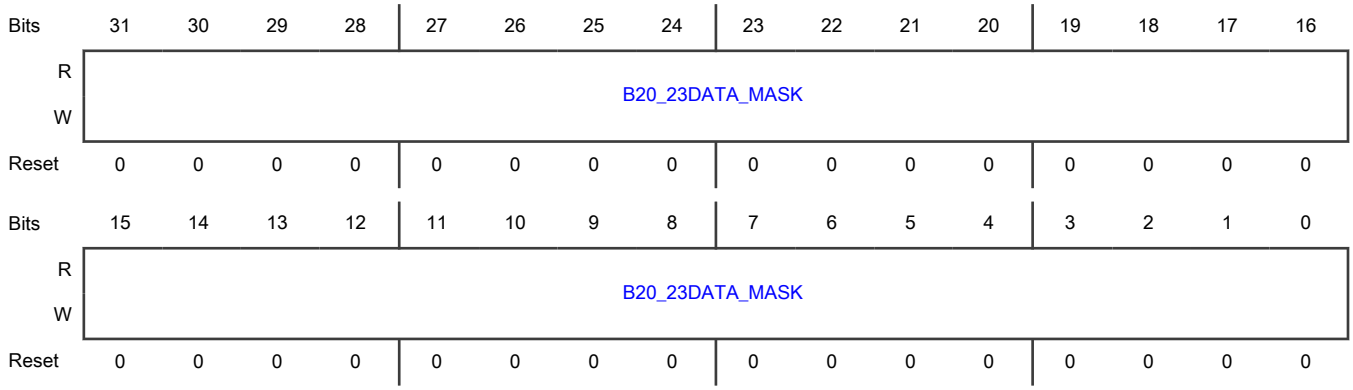
Offset

Register	Offset
EICHD22_WORD6	698h

Function

The seventh word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B20_23DATA_MASK correspond to EICHD22_WORD6 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B20_23DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B20_23DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD22_WORD6 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD22_WORD6 on the read data bus is inverted.

50.7.53 Error Injection Channel Descriptor 23, Word1 (EICHD23_WORD1)

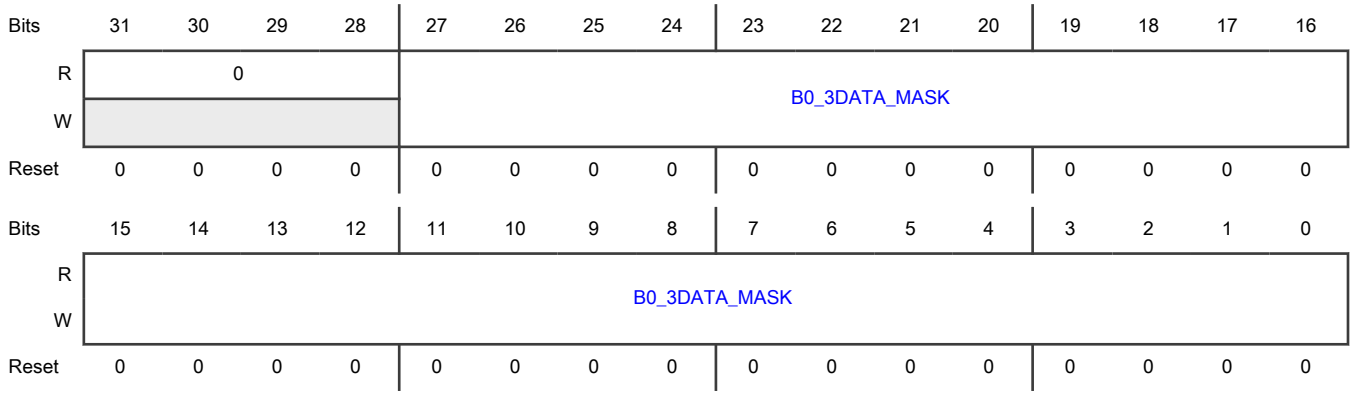
Offset

Register	Offset
EICHD23_WORD1	6C4h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD23_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-28 —	Reserved
27-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD23_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD23_WORD1 on the read data bus is inverted.

50.7.54 Error Injection Channel Descriptor 23, Word2 (EICHD23_WORD2)

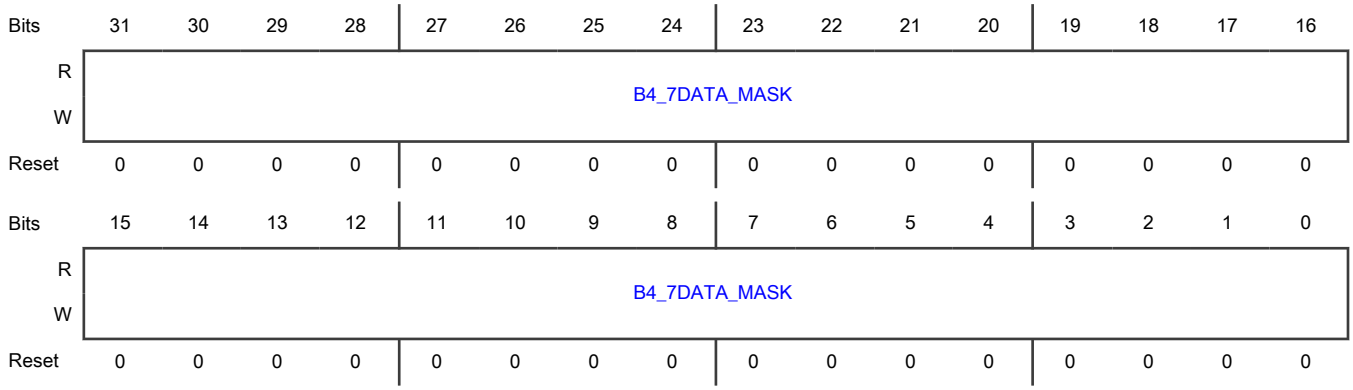
Offset

Register	Offset
EICHD23_WORD2	6C8h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to EICHD23_WORD2 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD23_WORD2 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD23_WORD2 on the read data bus is inverted.

50.7.55 Error Injection Channel Descriptor 23, Word3 (EICHD23_WORD3)

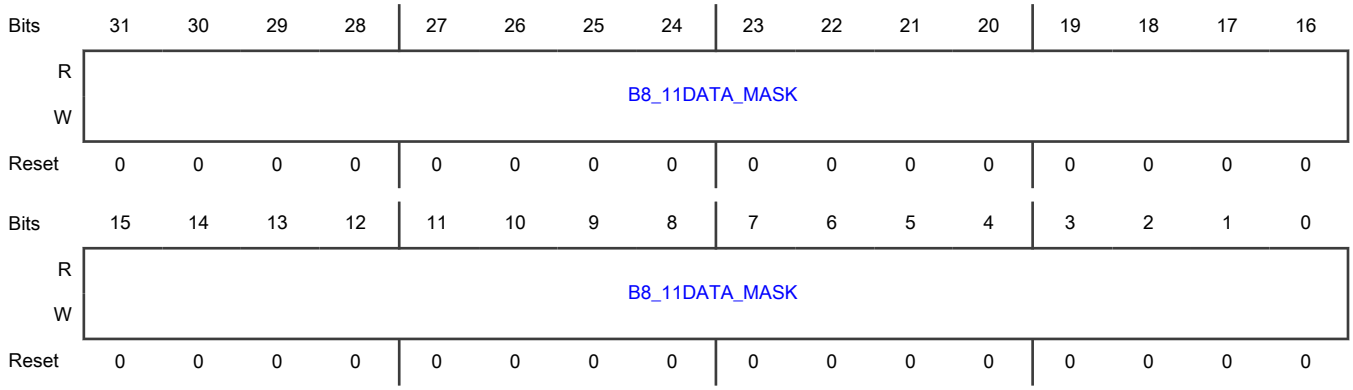
Offset

Register	Offset
EICHD23_WORD3	6CCh

Function

The fourth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B8_11DATA_MASK correspond to EICHD23_WORD3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B8_11DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B8_11DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD23_WORD3 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD23_WORD3 on the read data bus is inverted.

50.7.56 Error Injection Channel Descriptor 23, Word4 (EICHD23_WORD4)

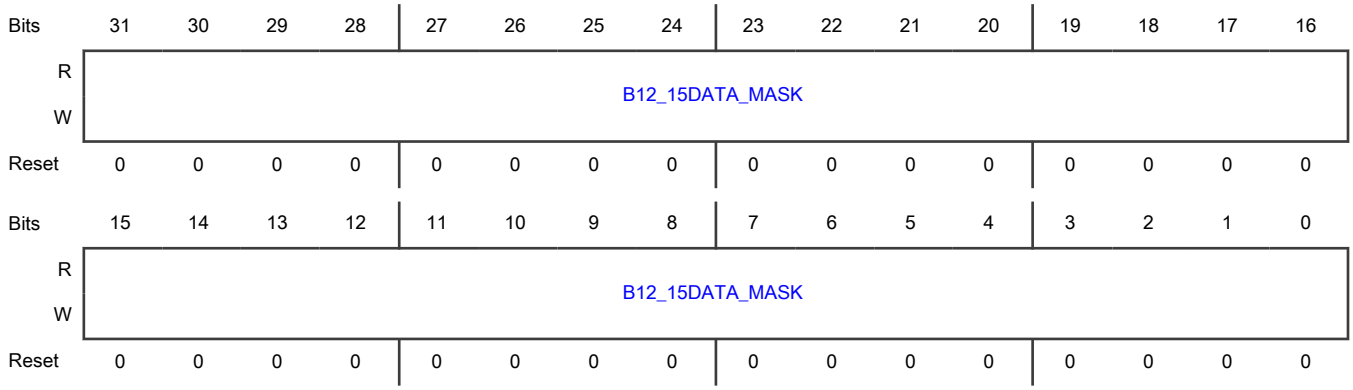
Offset

Register	Offset
EICHD23_WORD4	6D0h

Function

The fifth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B12_15DATA_MASK correspond to EICHD23_WORD4 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B12_15DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B12_15DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD23_WORD4 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD23_WORD4 on the read data bus is inverted.

50.7.57 Error Injection Channel Descriptor 23, Word5 (EICHD23_WORD5)

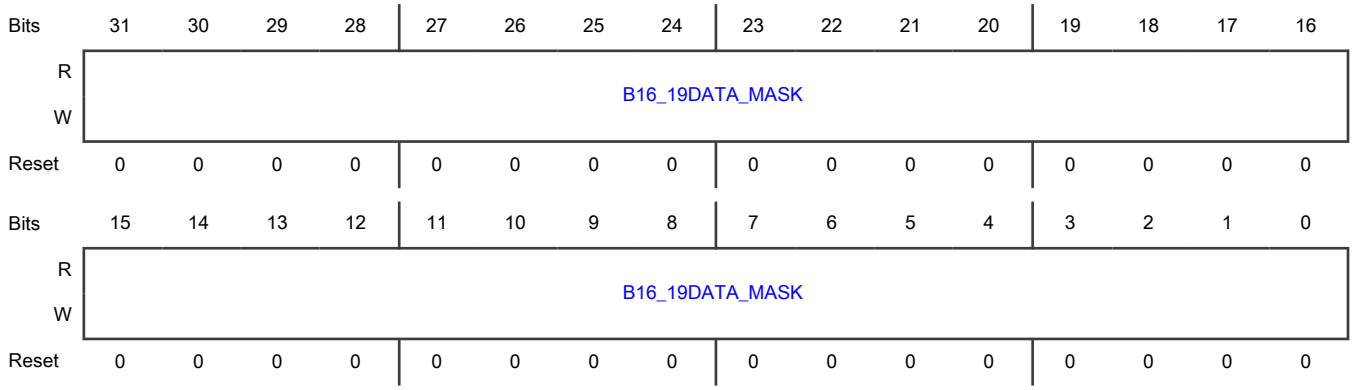
Offset

Register	Offset
EICHD23_WORD5	6D4h

Function

The sixth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B16_19DATA_MASK correspond to EICHD23_WORD5 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B16_19DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B16_19DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD23_WORD5 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD23_WORD5 on the read data bus is inverted.

50.7.58 Error Injection Channel Descriptor 23, Word6 (EICHD23_WORD6)

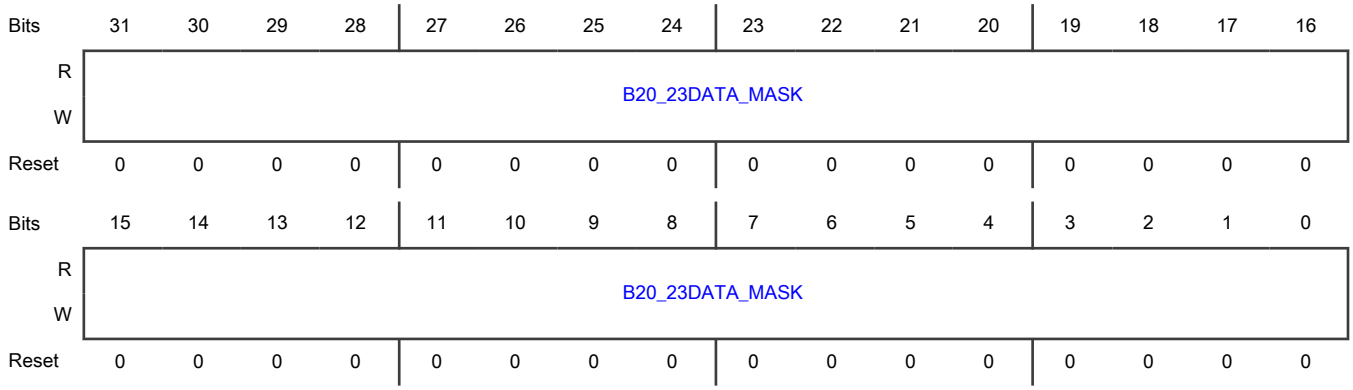
Offset

Register	Offset
EICHD23_WORD6	6D8h

Function

The seventh word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B20_23DATA_MASK correspond to EICHD23_WORD6 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B20_23DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B20_23DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD23_WORD6 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD23_WORD6 on the read data bus is inverted.

50.7.59 Error Injection Channel Descriptor 24, Word1 (EICHD24_WORD1)

Offset

Register	Offset
EICHD24_WORD1	704h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD24_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				B0_3DATA_MASK											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	B0_3DATA_MASK															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved
27-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD24_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD24_WORD1 on the read data bus is inverted.

50.7.60 Error Injection Channel Descriptor 24, Word2 (EICHD24_WORD2)

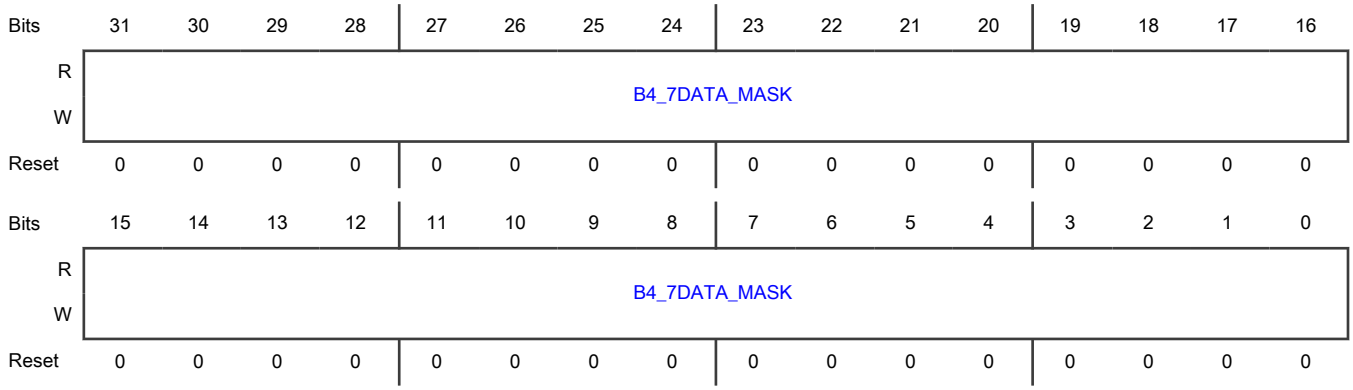
Offset

Register	Offset
EICHD24_WORD2	708h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to EICHD24_WORD2 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD24_WORD2 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD24_WORD2 on the read data bus is inverted.

50.7.61 Error Injection Channel Descriptor 24, Word3 (EICHD24_WORD3)

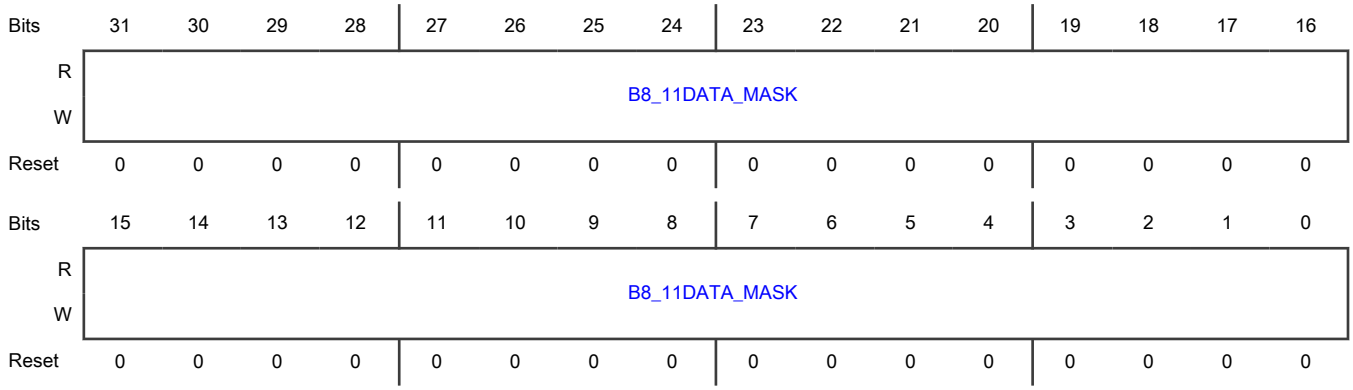
Offset

Register	Offset
EICHD24_WORD3	70Ch

Function

The fourth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B8_11DATA_MASK correspond to EICHD24_WORD3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B8_11DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B8_11DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD24_WORD3 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD24_WORD3 on the read data bus is inverted.

50.7.62 Error Injection Channel Descriptor 24, Word4 (EICHD24_WORD4)

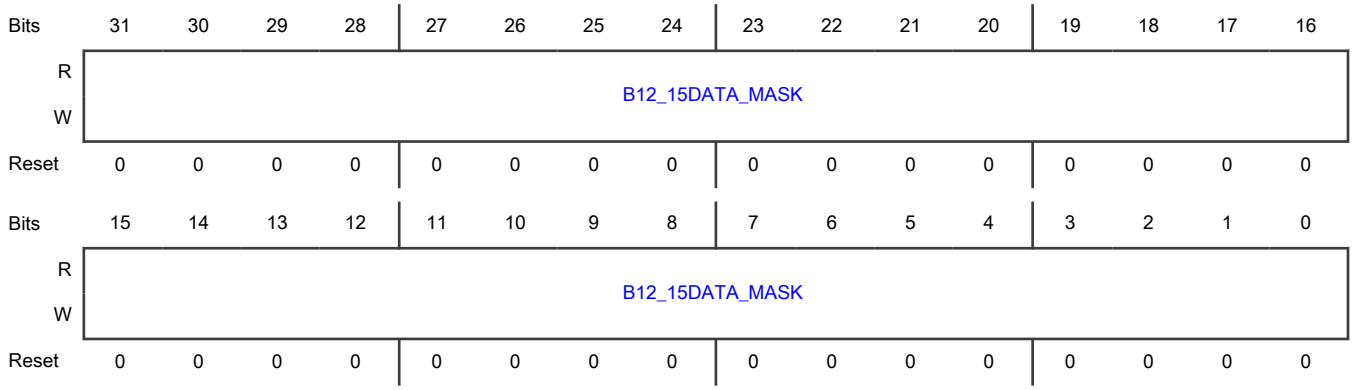
Offset

Register	Offset
EICHD24_WORD4	710h

Function

The fifth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B12_15DATA_MASK correspond to EICHD24_WORD4 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B12_15DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B12_15DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD24_WORD4 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD24_WORD4 on the read data bus is inverted.

50.7.63 Error Injection Channel Descriptor 24, Word5 (EICHD24_WORD5)

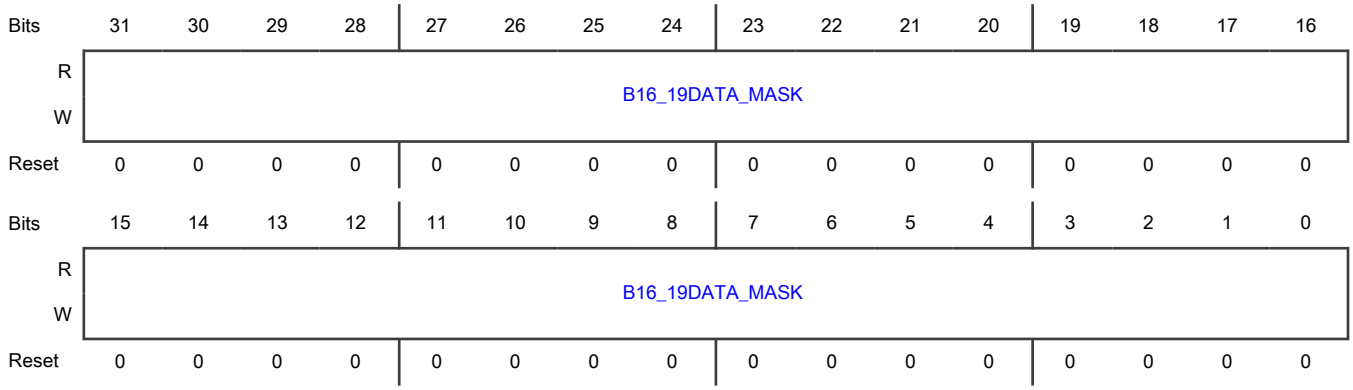
Offset

Register	Offset
EICHD24_WORD5	714h

Function

The sixth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B16_19DATA_MASK correspond to EICHD24_WORD5 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B16_19DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B16_19DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD24_WORD5 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD24_WORD5 on the read data bus is inverted.

50.7.64 Error Injection Channel Descriptor 24, Word6 (EICHD24_WORD6)

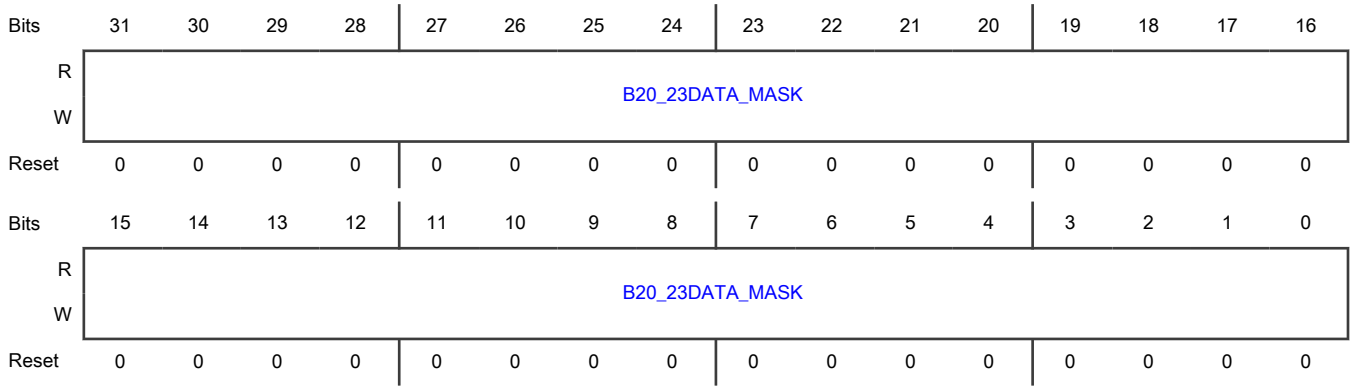
Offset

Register	Offset
EICHD24_WORD6	718h

Function

The seventh word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B20_23DATA_MASK correspond to EICHD24_WORD6 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B20_23DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B20_23DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD24_WORD6 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD24_WORD6 on the read data bus is inverted.

50.7.65 Error Injection Channel Descriptor 25, Word1 (EICHD25_WORD1)

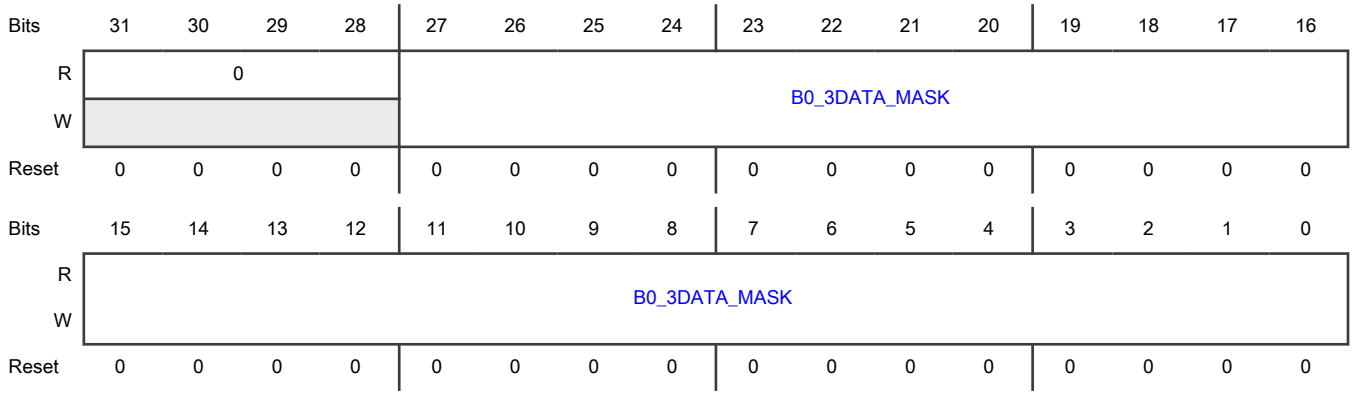
Offset

Register	Offset
EICHD25_WORD1	744h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD25_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-28 —	Reserved
27-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD25_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD25_WORD1 on the read data bus is inverted.

50.7.66 Error Injection Channel Descriptor 25, Word2 (EICHD25_WORD2)

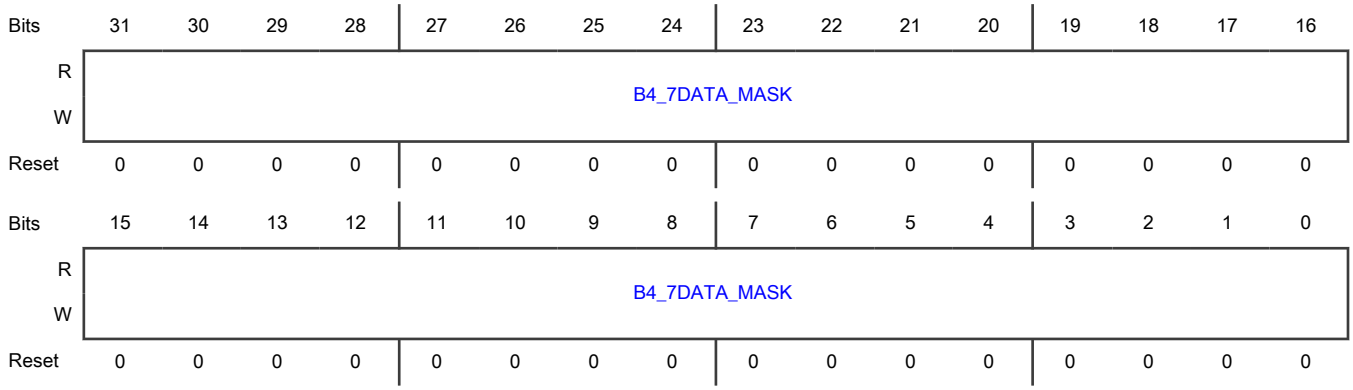
Offset

Register	Offset
EICHD25_WORD2	748h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to EICHD25_WORD2 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD25_WORD2 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD25_WORD2 on the read data bus is inverted.

50.7.67 Error Injection Channel Descriptor 25, Word3 (EICHD25_WORD3)

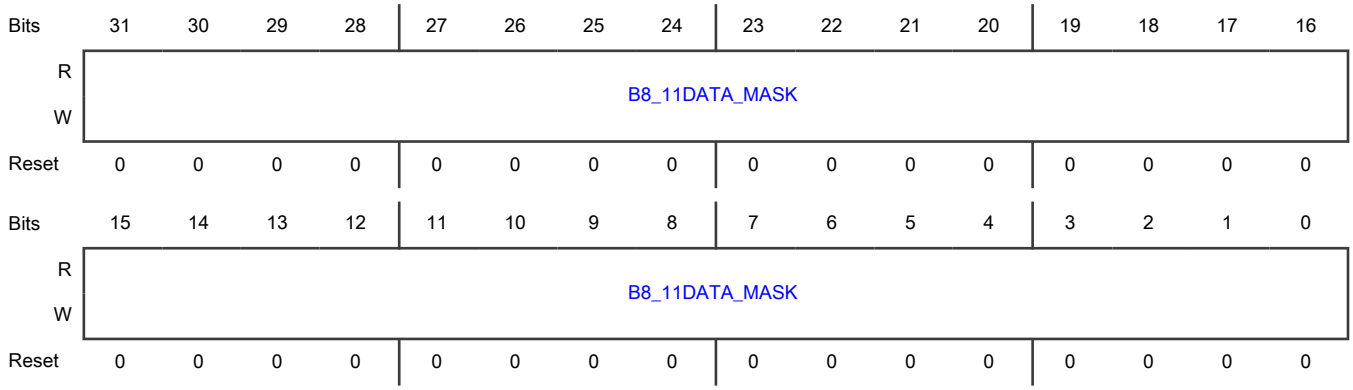
Offset

Register	Offset
EICHD25_WORD3	74Ch

Function

The fourth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B8_11DATA_MASK correspond to EICHD25_WORD3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B8_11DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B8_11DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD25_WORD3 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD25_WORD3 on the read data bus is inverted.

50.7.68 Error Injection Channel Descriptor 25, Word4 (EICHD25_WORD4)

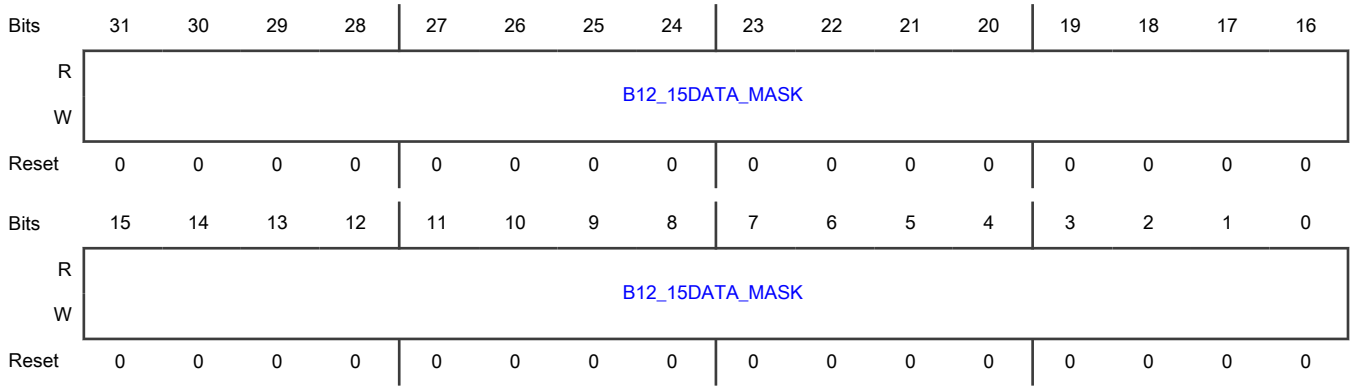
Offset

Register	Offset
EICHD25_WORD4	750h

Function

The fifth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B12_15DATA_MASK correspond to EICHD25_WORD4 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B12_15DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B12_15DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD25_WORD4 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD25_WORD4 on the read data bus is inverted.

50.7.69 Error Injection Channel Descriptor 25, Word5 (EICHD25_WORD5)

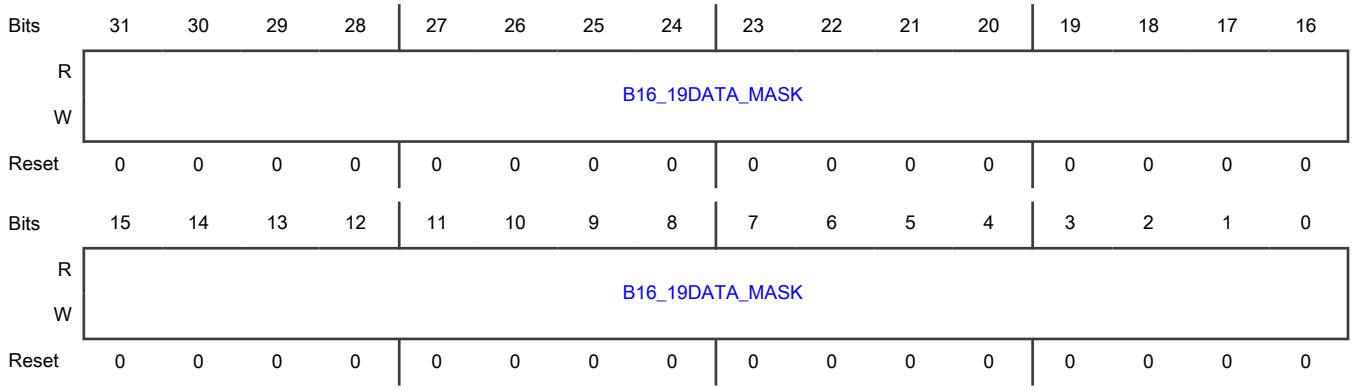
Offset

Register	Offset
EICHD25_WORD5	754h

Function

The sixth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B16_19DATA_MASK correspond to EICHD25_WORD5 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B16_19DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B16_19DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD25_WORD5 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD25_WORD5 on the read data bus is inverted.

50.7.70 Error Injection Channel Descriptor 25, Word6 (EICHD25_WORD6)

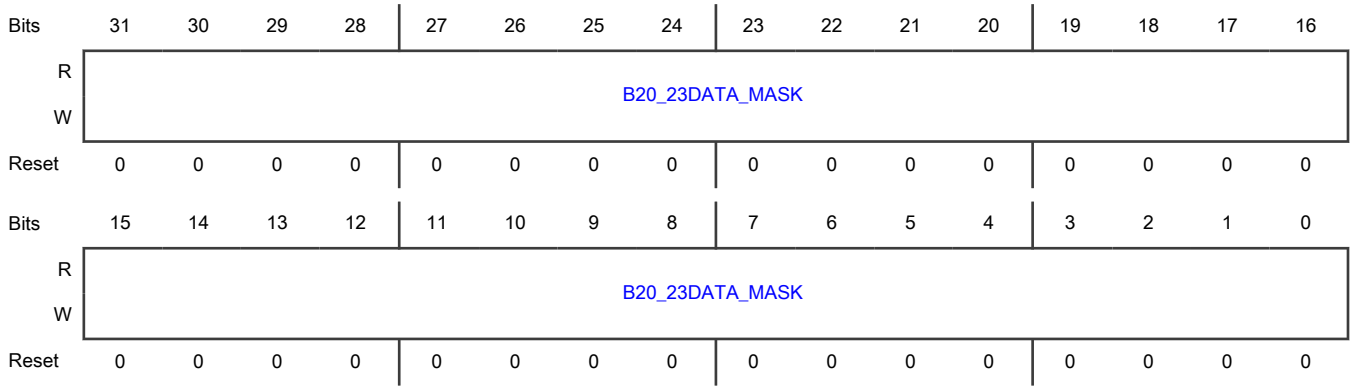
Offset

Register	Offset
EICHD25_WORD6	758h

Function

The seventh word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B20_23DATA_MASK correspond to EICHD25_WORD6 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B20_23DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B20_23DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD25_WORD6 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD25_WORD6 on the read data bus is inverted.

50.7.71 Error Injection Channel Descriptor 26, Word1 (EICHD26_WORD1)

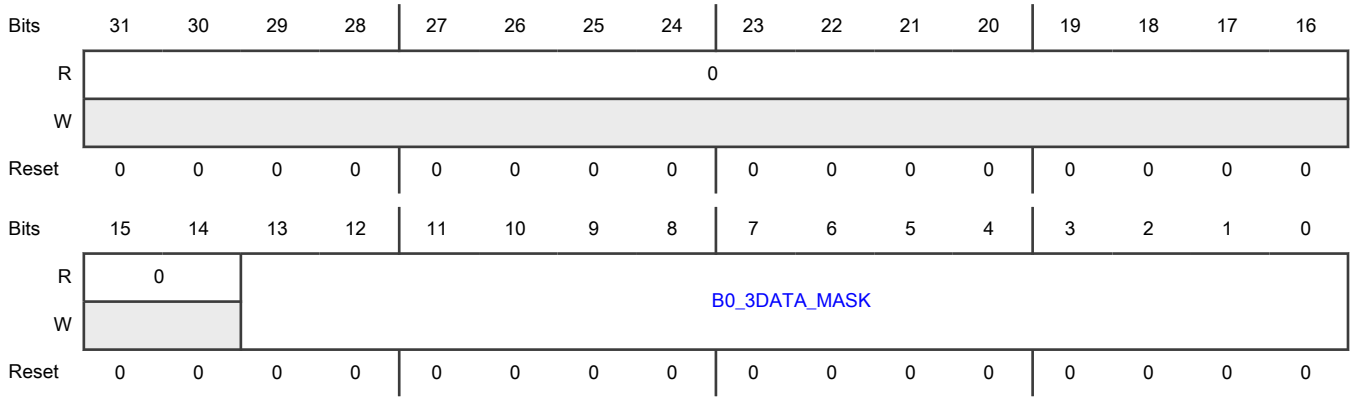
Offset

Register	Offset
EICHD26_WORD1	784h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD26_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-14 —	Reserved
13-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD26_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD26_WORD1 on the read data bus is inverted.

50.7.72 Error Injection Channel Descriptor 27, Word1 (EICHD27_WORD1)

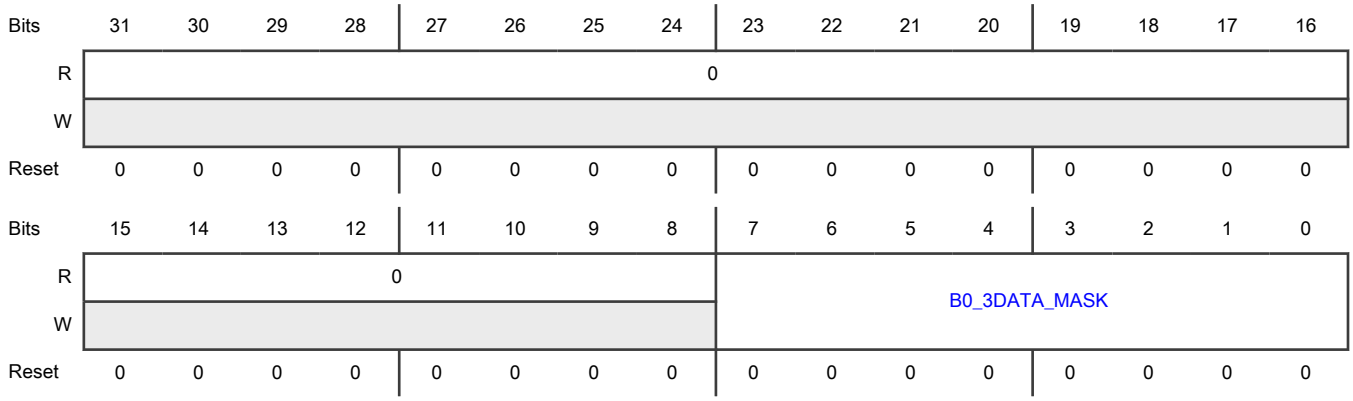
Offset

Register	Offset
EICHD27_WORD1	7C4h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD27_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD27_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD27_WORD1 on the read data bus is inverted.

50.7.73 Error Injection Channel Descriptor 28, Word1 (EICHD28_WORD1)

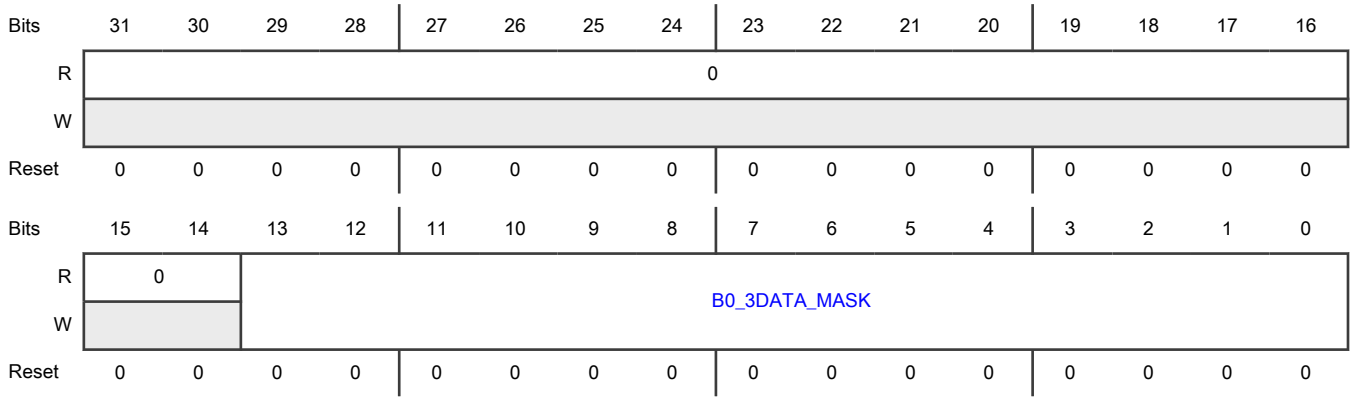
Offset

Register	Offset
EICHD28_WORD1	804h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD28_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-14 —	Reserved
13-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD28_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD28_WORD1 on the read data bus is inverted.

50.7.74 Error Injection Channel Descriptor 29, Word1 (EICHD29_WORD1)

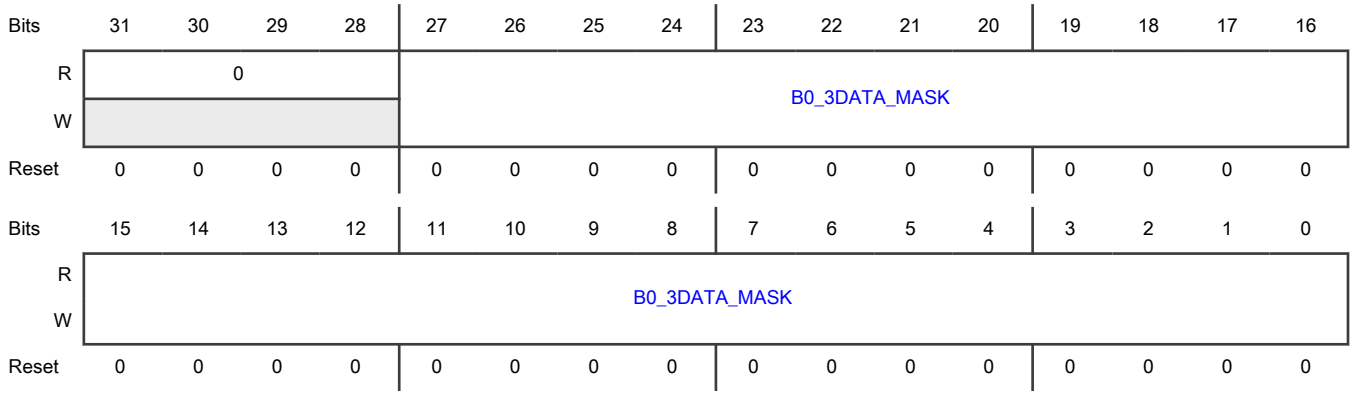
Offset

Register	Offset
EICHD29_WORD1	844h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD29_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-28 —	Reserved
27-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD29_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD29_WORD1 on the read data bus is inverted.

50.7.75 Error Injection Channel Descriptor 30, Word1 (EICHD30_WORD1)

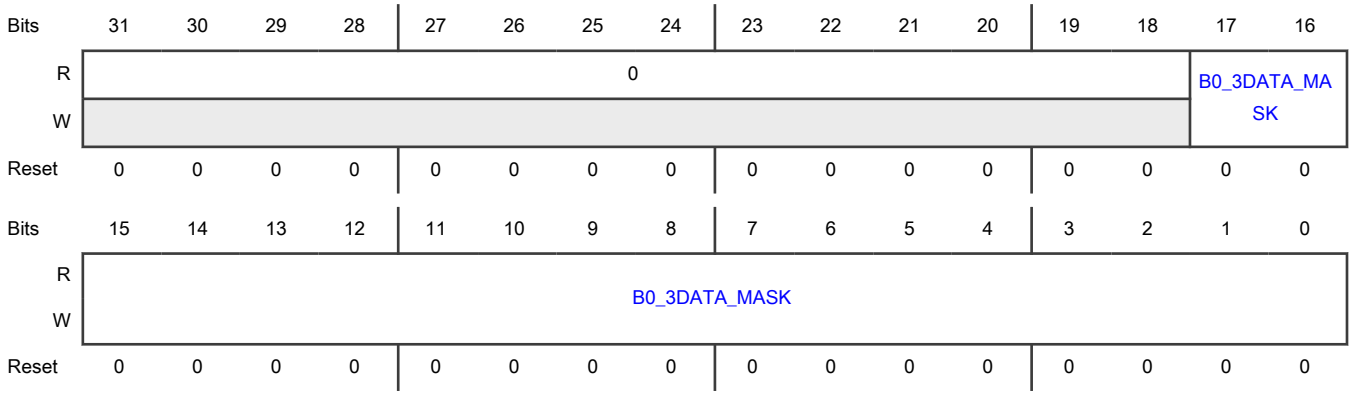
Offset

Register	Offset
EICHD30_WORD1	884h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD30_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-18 —	Reserved
17-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD30_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD30_WORD1 on the read data bus is inverted.

50.7.76 Error Injection Channel Descriptor 31, Word1 (EICHD31_WORD1)

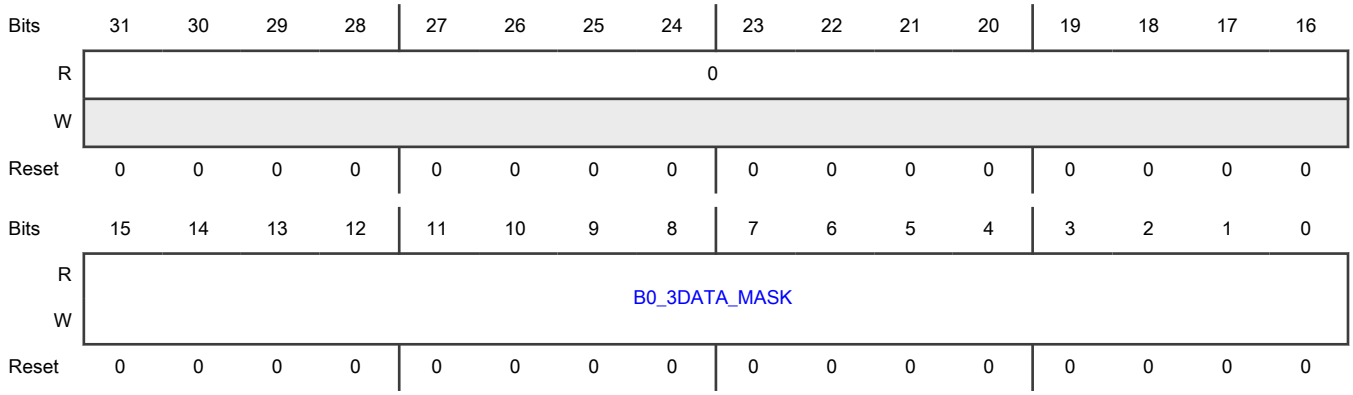
Offset

Register	Offset
EICHD31_WORD1	8C4h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD31_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD31_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD31_WORD1 on the read data bus is inverted.

50.8 EIM_3 register descriptions

The EIM provides a programming model mapped to an on-platform peripheral slot.

Programming model access

All system bus controllers can access the programming model:

- Only in supervisor mode
- Using only 32-bit (word) accesses

Any of the following attempted references to the programming model generates an error termination:

- In user mode
- Using non-32-bit access sizes
- To undefined (reserved) addresses

Attempted updates to the programming model while the EIM is in the midst of an operation result in non-deterministic behavior.

Error injection channel descriptor: function and structure

Each error injection channel descriptor:

- Specifies a mask that defines which bits of the read data and/or checkbit bus from target RAM are inverted on a read access.
- Consists of a 128-bit (16-byte) structure, composed of four 32-bit words, in the EIM programming model. Unused words are not documented.
  - Word0 (EICHD*n*_WORD0), if present, defines the checkbit mask.
  - Word1-3 (EICHD*n*_WORD1-3), if present, define the data mask. Word2 and Word3 are present only when required by the total width of the channel's data mask. See Error injection channel descriptor: DATA_MASK details.

The multiple channel descriptors are organized sequentially.

Error injection channel descriptor: DATA_MASK details

For each channel: The following table shows the total width of DATA_MASK and the distribution of its bits across the WORD registers.

Table 289. Error injection channel descriptor: DATA_MASK details

Channel	DATA_MASK total width (bits)	Specific bits of DATA_MASK in		
		WORD1	WORD2	WORD3
0	60	59-32	31-0	—
1	60	59-32	31-0	—
2	64	63-32	31-0	—
3	64	63-32	31-0	—
4	60	59-32	31-0	—
5	60	59-32	31-0	—
6	64	63-32	31-0	—
7	32	31-0	—	—
8	32	31-0	—	—
9	32	31-0	—	—
10	32	31-0	—	—

50.8.1 EIM_3 memory map

EIM_3 base address: 4051_8000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Error Injection Module Configuration Register (EIMCR)	32	RW	0000_0000h
4h	Error Injection Channel Enable register (EICHEN)	32	RW	0000_0000h
104h	Error Injection Channel Descriptor 0, Word1 (EICHD0_WORD1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
108h	Error Injection Channel Descriptor 0, Word2 (EICHD0_WORD2)	32	RW	0000_0000h
144h	Error Injection Channel Descriptor 1, Word1 (EICHD1_WORD1)	32	RW	0000_0000h
148h	Error Injection Channel Descriptor 1, Word2 (EICHD1_WORD2)	32	RW	0000_0000h
180h	Error Injection Channel Descriptor 2, Word0 (EICHD2_WORD0)	32	RW	0000_0000h
184h	Error Injection Channel Descriptor 2, Word1 (EICHD2_WORD1)	32	RW	0000_0000h
188h	Error Injection Channel Descriptor 2, Word2 (EICHD2_WORD2)	32	RW	0000_0000h
1C0h	Error Injection Channel Descriptor 3, Word0 (EICHD3_WORD0)	32	RW	0000_0000h
1C4h	Error Injection Channel Descriptor 3, Word1 (EICHD3_WORD1)	32	RW	0000_0000h
1C8h	Error Injection Channel Descriptor 3, Word2 (EICHD3_WORD2)	32	RW	0000_0000h
204h	Error Injection Channel Descriptor 4, Word1 (EICHD4_WORD1)	32	RW	0000_0000h
208h	Error Injection Channel Descriptor 4, Word2 (EICHD4_WORD2)	32	RW	0000_0000h
244h	Error Injection Channel Descriptor 5, Word1 (EICHD5_WORD1)	32	RW	0000_0000h
248h	Error Injection Channel Descriptor 5, Word2 (EICHD5_WORD2)	32	RW	0000_0000h
280h	Error Injection Channel Descriptor 6, Word0 (EICHD6_WORD0)	32	RW	0000_0000h
284h	Error Injection Channel Descriptor 6, Word1 (EICHD6_WORD1)	32	RW	0000_0000h
288h	Error Injection Channel Descriptor 6, Word2 (EICHD6_WORD2)	32	RW	0000_0000h
2C4h	Error Injection Channel Descriptor 7, Word1 (EICHD7_WORD1)	32	RW	0000_0000h
304h	Error Injection Channel Descriptor 8, Word1 (EICHD8_WORD1)	32	RW	0000_0000h
344h	Error Injection Channel Descriptor 9, Word1 (EICHD9_WORD1)	32	RW	0000_0000h
384h	Error Injection Channel Descriptor 10, Word1 (EICHD10_WORD1)	32	RW	0000_0000h

50.8.2 Error Injection Module Configuration Register (EIMCR)

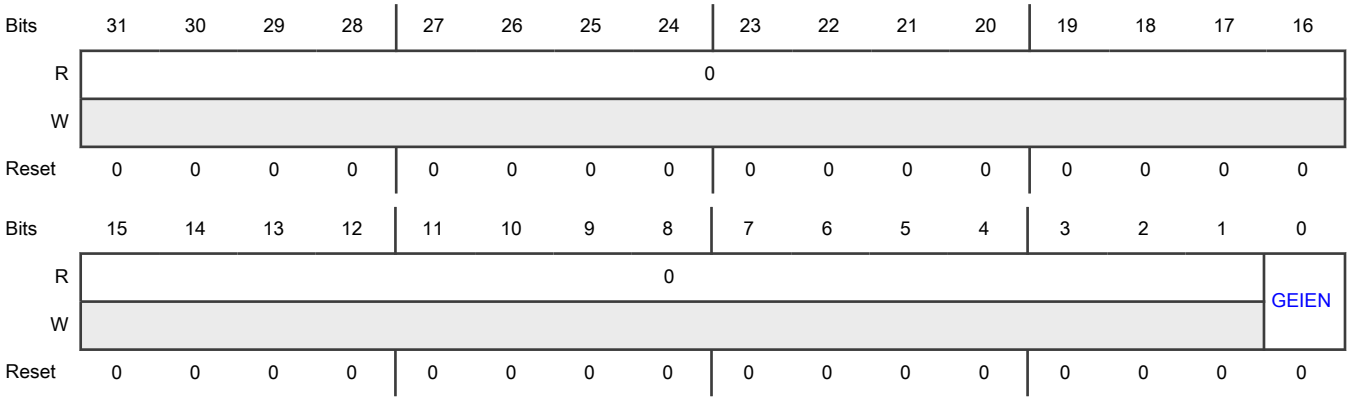
Offset

Register	Offset
EIMCR	0h

Function

The EIM Configuration Register is used to globally enable/disable the error injection function.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 GEIEN	Global Error Injection Enable This bit globally enables or disables the error injection function of the EIM. This field is initialized by hardware reset. 0b - Disabled 1b - Enabled

50.8.3 Error Injection Channel Enable register (EICHEN)

Offset

Register	Offset
EICHEN	4h

Function

Each field of the Error Injection Channel Enable register (EICHEN) is used to enable or disable the corresponding error injection channel.

NOTE

To enable an error injection channel, the Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	EICH0EN	EICH1EN	EICH2EN	EICH3EN	EICH4EN	EICH5EN	EICH6EN	EICH7EN	EICH8EN	EICH9EN	EICH10EN	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 EICH0EN	Error Injection Channel 0 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 0 1b - Error injection is enabled on Error Injection Channel 0
30 EICH1EN	Error Injection Channel 1 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 1 1b - Error injection is enabled on Error Injection Channel 1
29 EICH2EN	Error Injection Channel 2 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 2 1b - Error injection is enabled on Error Injection Channel 2
28 EICH3EN	Error Injection Channel 3 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 3 1b - Error injection is enabled on Error Injection Channel 3
27 EICH4EN	Error Injection Channel 4 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 4 1b - Error injection is enabled on Error Injection Channel 4
26 EICH5EN	Error Injection Channel 5 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 5 1b - Error injection is enabled on Error Injection Channel 5
25 EICH6EN	Error Injection Channel 6 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 6 1b - Error injection is enabled on Error Injection Channel 6
24 EICH7EN	Error Injection Channel 7 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 7 1b - Error injection is enabled on Error Injection Channel 7
23 EICH8EN	Error Injection Channel 8 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 8 1b - Error injection is enabled on Error Injection Channel 8
22 EICH9EN	Error Injection Channel 9 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 9 1b - Error injection is enabled on Error Injection Channel 9

Table continues on the next page...

Table continued from the previous page...

Field	Function
21 EICH10EN	Error Injection Channel 10 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 10 1b - Error injection is enabled on Error Injection Channel 10
20 —	Reserved
19 —	Reserved
18 —	Reserved
17 —	Reserved
16 —	Reserved
15 —	Reserved
14 —	Reserved
13 —	Reserved
12 —	Reserved
11 —	Reserved
10	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
9 —	Reserved
8 —	Reserved
7 —	Reserved
6 —	Reserved
5 —	Reserved
4 —	Reserved
3 —	Reserved
2 —	Reserved
1 —	Reserved
0 —	Reserved

50.8.4 Error Injection Channel Descriptor 0, Word1 (EICHD0_WORD1)

Offset

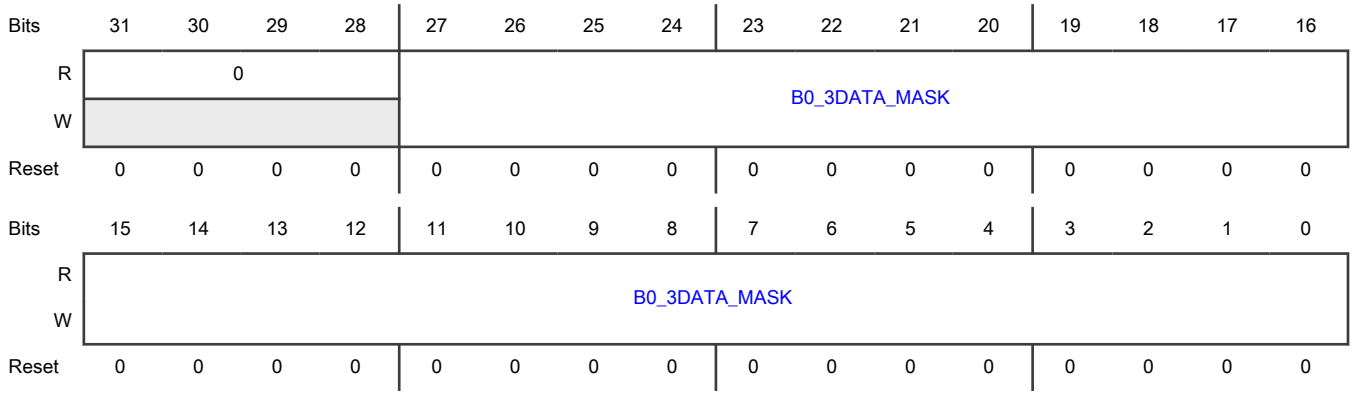
Register	Offset
EICHD0_WORD1	104h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD0_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be

inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-28 —	Reserved
27-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD0_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD0_WORD1 on the read data bus is inverted.

50.8.5 Error Injection Channel Descriptor 0, Word2 (EICHD0_WORD2)

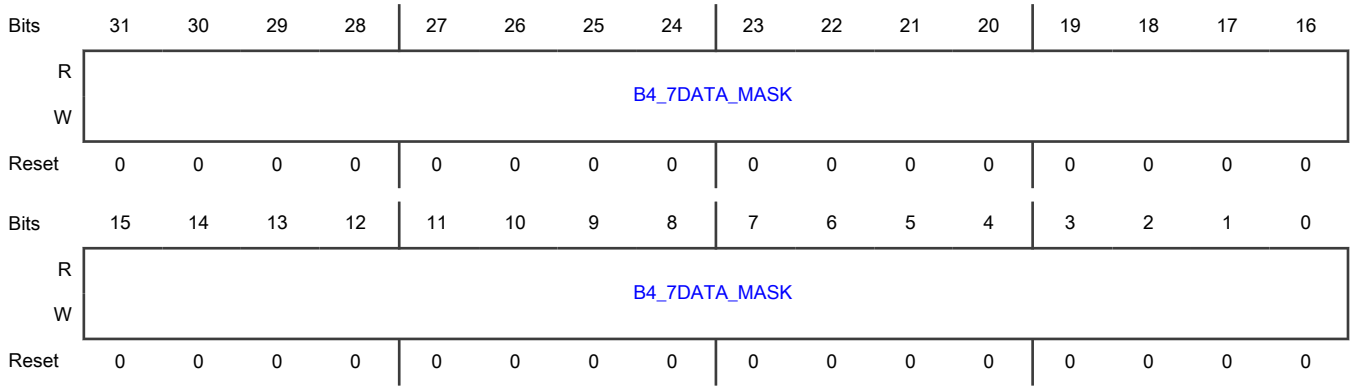
Offset

Register	Offset
EICHD0_WORD2	108h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to EICHD0_WORD2 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD0_WORD2 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD0_WORD2 on the read data bus is inverted.

50.8.6 Error Injection Channel Descriptor 1, Word1 (EICHD1_WORD1)

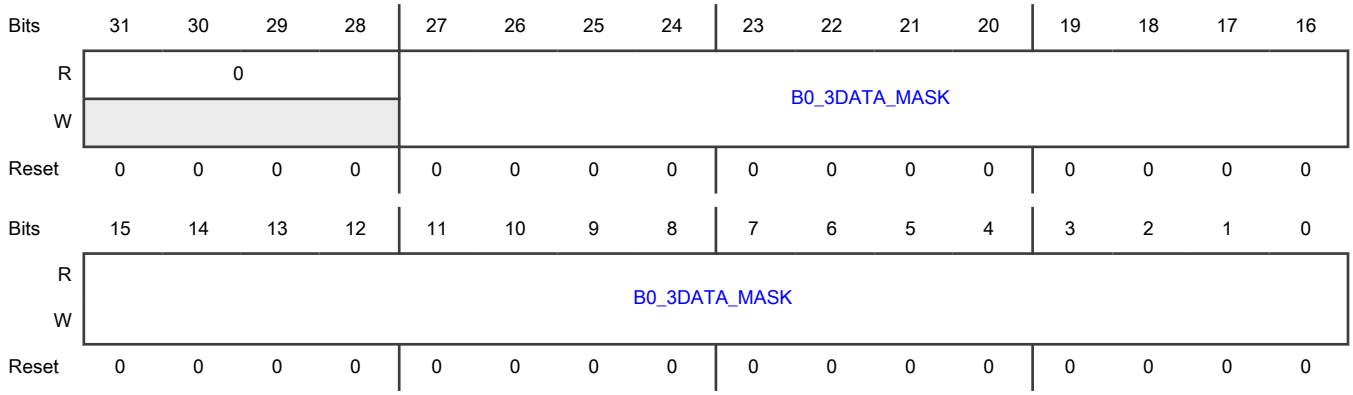
Offset

Register	Offset
EICHD1_WORD1	144h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD1_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-28 —	Reserved
27-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD1_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD1_WORD1 on the read data bus is inverted.

50.8.7 Error Injection Channel Descriptor 1, Word2 (EICHD1_WORD2)

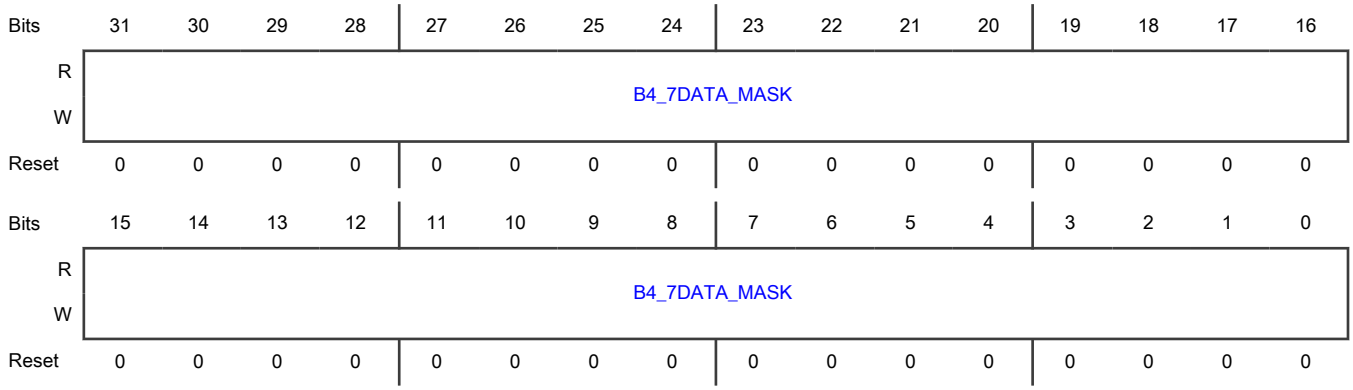
Offset

Register	Offset
EICHD1_WORD2	148h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to EICHD1_WORD2 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD1_WORD2 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD1_WORD2 on the read data bus is inverted.

50.8.8 Error Injection Channel Descriptor n, Word0 (EICHD2_WORD0 - EICHD6_WORD0)

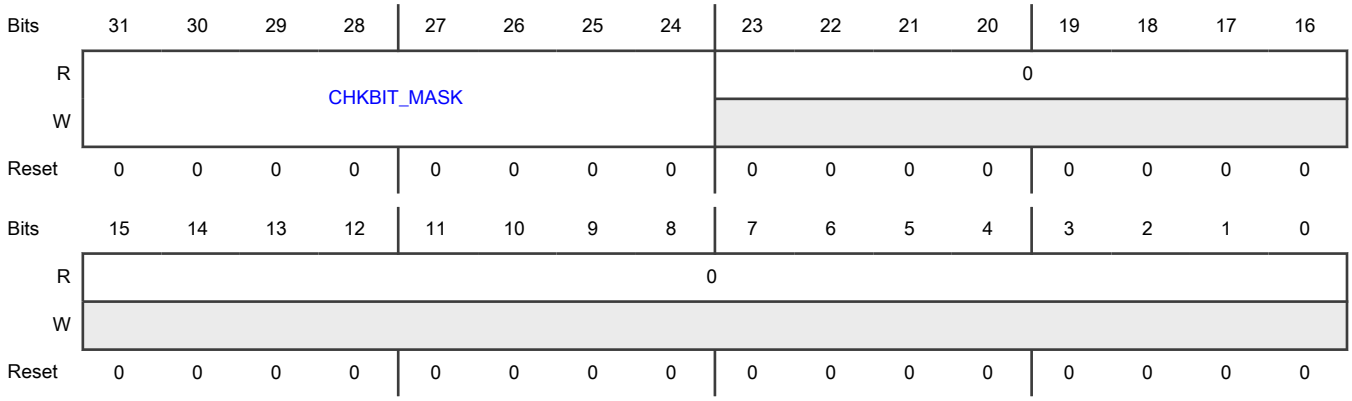
Offset

Register	Offset
EICHD2_WORD0	180h
EICHD3_WORD0	1C0h
EICHD6_WORD0	280h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-24 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[7:0] (8 bits wide), CHKBIT_MASK[7] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
23-0 —	Reserved

50.8.9 Error Injection Channel Descriptor 2, Word1 (EICHD2_WORD1)

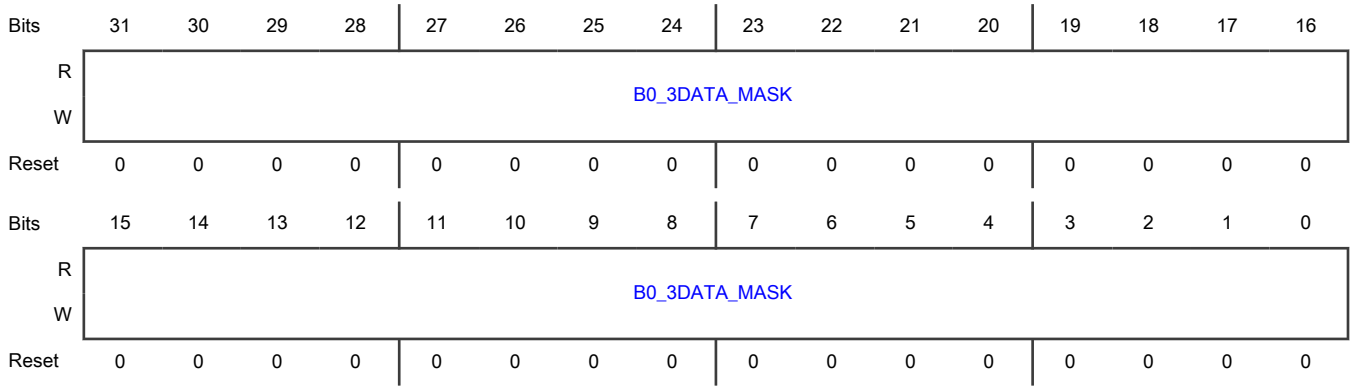
Offset

Register	Offset
EICHD2_WORD1	184h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD2_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD2_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD2_WORD1 on the read data bus is inverted.

50.8.10 Error Injection Channel Descriptor 2, Word2 (EICHD2_WORD2)

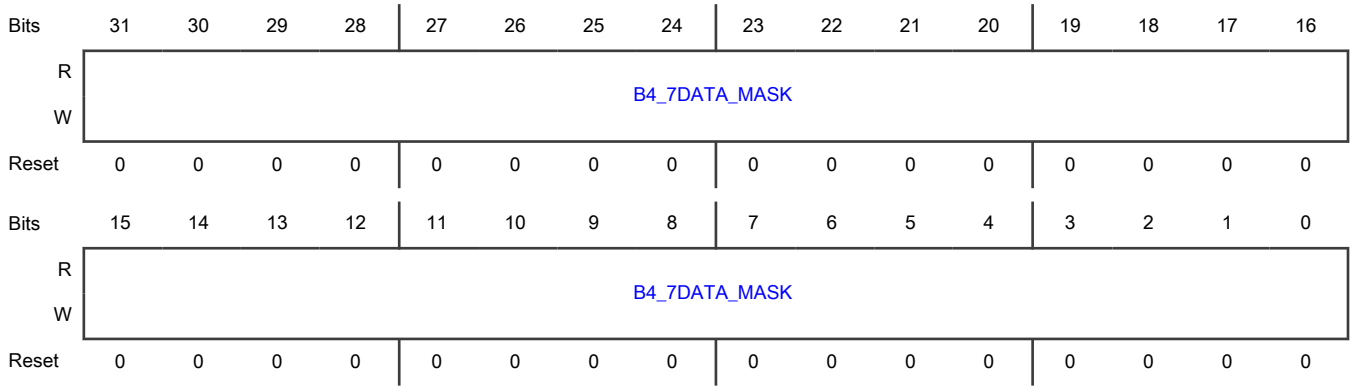
Offset

Register	Offset
EICHD2_WORD2	188h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to EICHD2_WORD2 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD2_WORD2 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD2_WORD2 on the read data bus is inverted.

50.8.11 Error Injection Channel Descriptor 3, Word1 (EICHD3_WORD1)

Offset

Register	Offset
EICHD3_WORD1	1C4h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD3_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	B0_3DATA_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	B0_3DATA_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD3_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD3_WORD1 on the read data bus is inverted.

50.8.12 Error Injection Channel Descriptor 3, Word2 (EICHD3_WORD2)

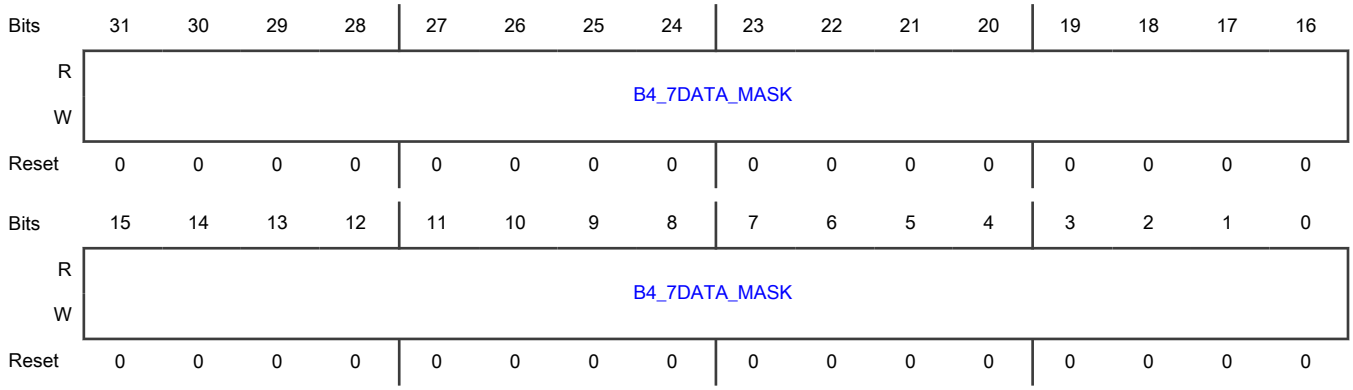
Offset

Register	Offset
EICHD3_WORD2	1C8h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to EICHD3_WORD2 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD3_WORD2 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD3_WORD2 on the read data bus is inverted.

50.8.13 Error Injection Channel Descriptor 4, Word1 (EICHD4_WORD1)

Offset

Register	Offset
EICHD4_WORD1	204h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD4_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				B0_3DATA_MASK											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	B0_3DATA_MASK															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved
27-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD4_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD4_WORD1 on the read data bus is inverted.

50.8.14 Error Injection Channel Descriptor 4, Word2 (EICHD4_WORD2)

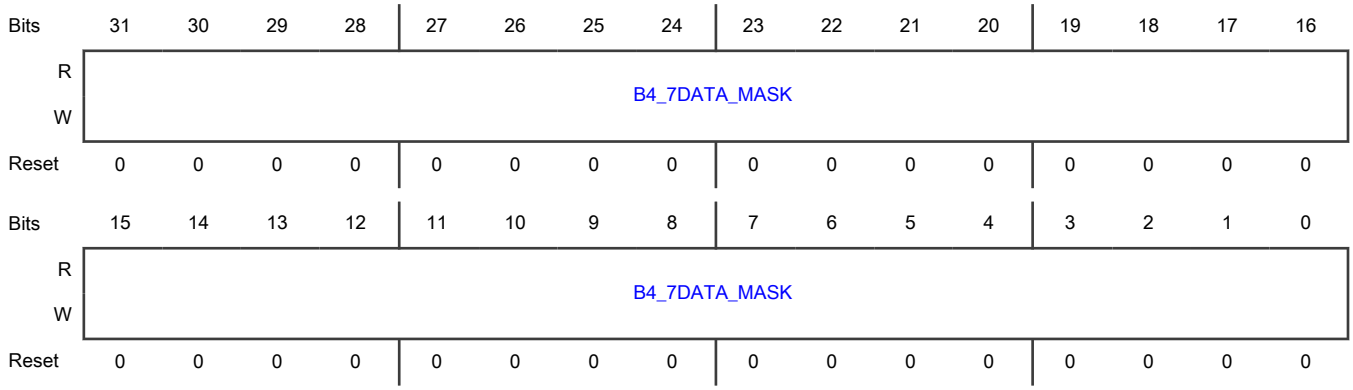
Offset

Register	Offset
EICHD4_WORD2	208h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to EICHD4_WORD2 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD4_WORD2 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD4_WORD2 on the read data bus is inverted.

50.8.15 Error Injection Channel Descriptor 5, Word1 (EICHD5_WORD1)

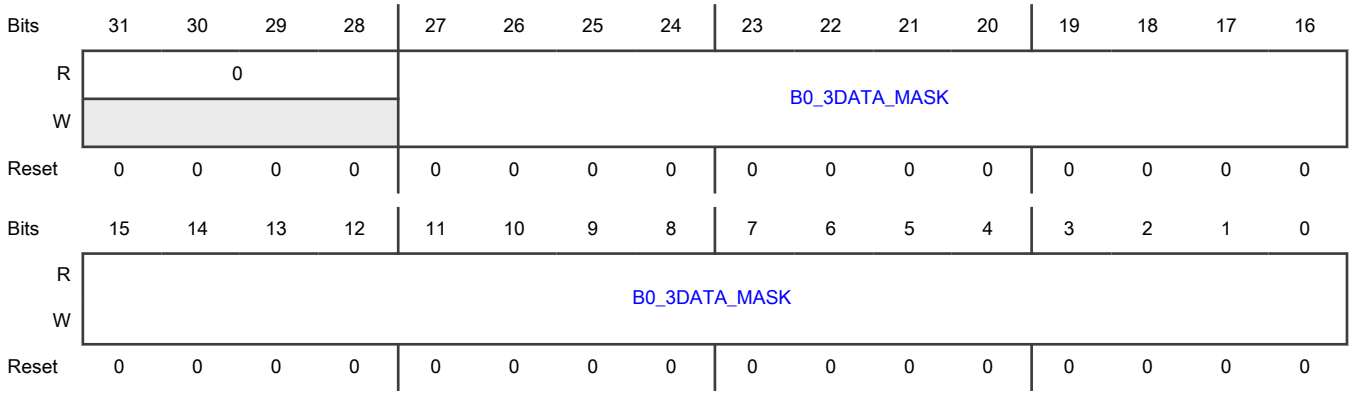
Offset

Register	Offset
EICHD5_WORD1	244h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD5_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-28 —	Reserved
27-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD5_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD5_WORD1 on the read data bus is inverted.

50.8.16 Error Injection Channel Descriptor 5, Word2 (EICHD5_WORD2)

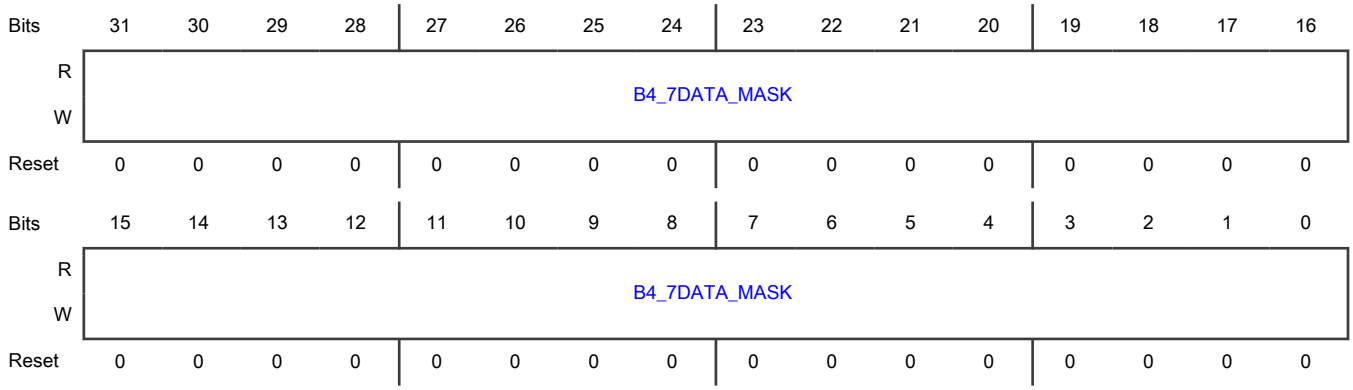
Offset

Register	Offset
EICHD5_WORD2	248h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to EICHD5_WORD2 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD5_WORD2 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD5_WORD2 on the read data bus is inverted.

50.8.17 Error Injection Channel Descriptor 6, Word1 (EICHD6_WORD1)

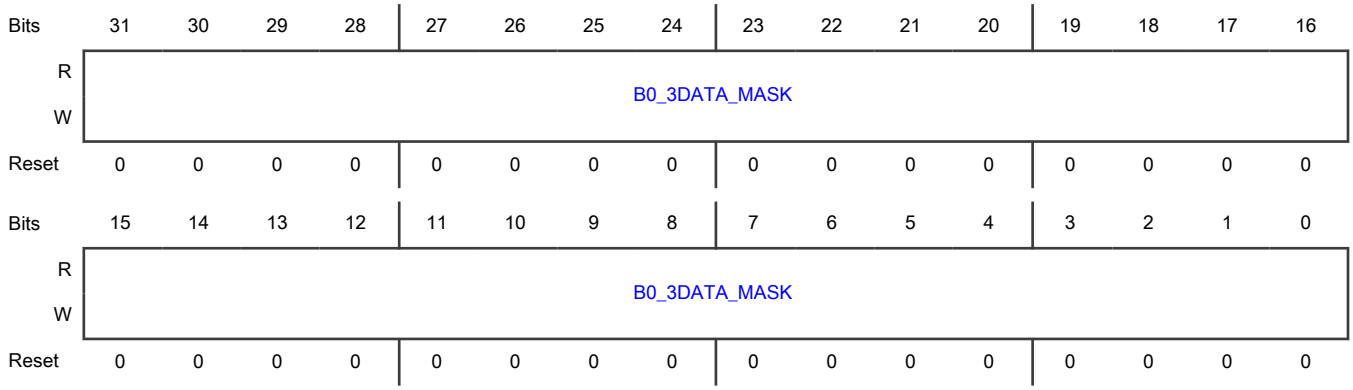
Offset

Register	Offset
EICHD6_WORD1	284h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD6_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD6_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD6_WORD1 on the read data bus is inverted.

50.8.18 Error Injection Channel Descriptor 6, Word2 (EICHD6_WORD2)

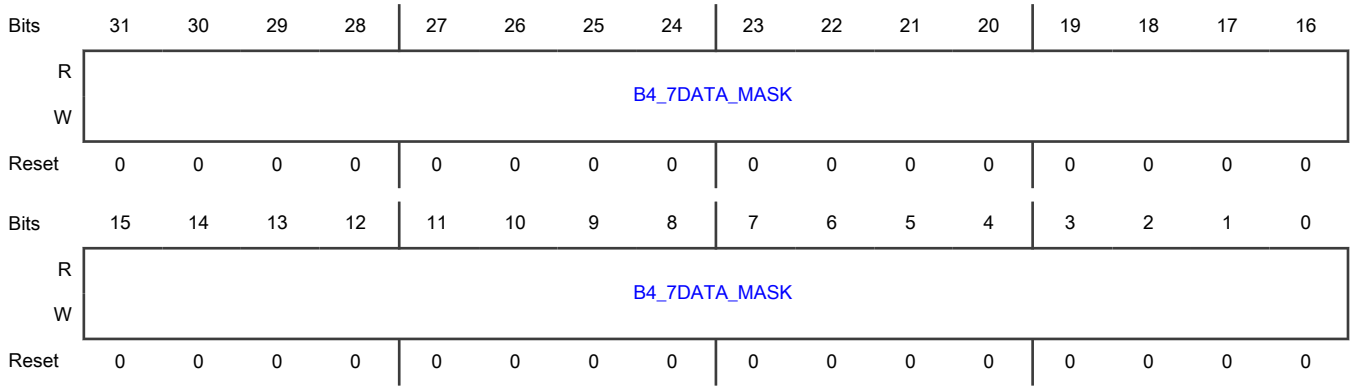
Offset

Register	Offset
EICHD6_WORD2	288h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to EICHD6_WORD2 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD6_WORD2 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD6_WORD2 on the read data bus is inverted.

50.8.19 Error Injection Channel Descriptor 7, Word1 (EICHD7_WORD1)

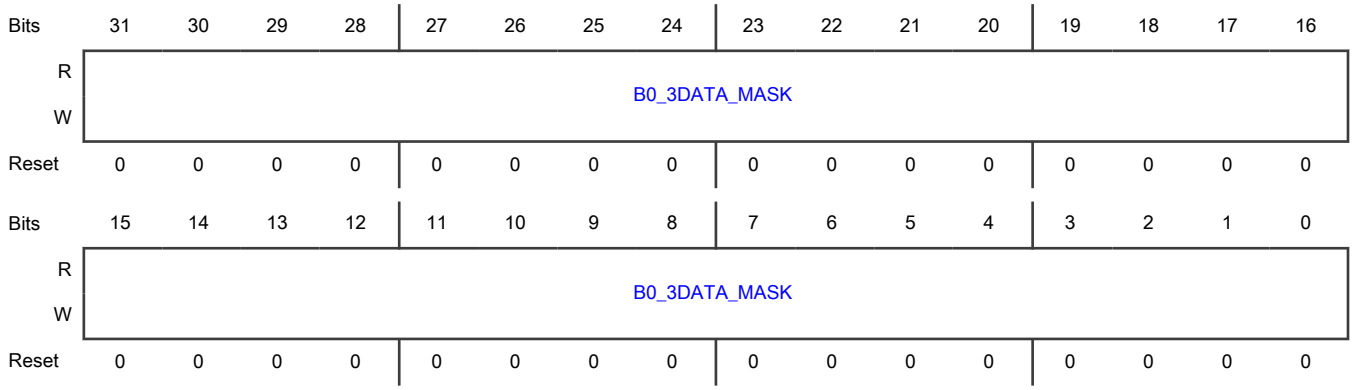
Offset

Register	Offset
EICHD7_WORD1	2C4h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD7_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD7_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD7_WORD1 on the read data bus is inverted.

50.8.20 Error Injection Channel Descriptor 8, Word1 (EICHD8_WORD1)

Offset

Register	Offset
EICHD8_WORD1	304h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD8_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	B0_3DATA_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	B0_3DATA_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD8_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD8_WORD1 on the read data bus is inverted.

50.8.21 Error Injection Channel Descriptor 9, Word1 (EICHD9_WORD1)

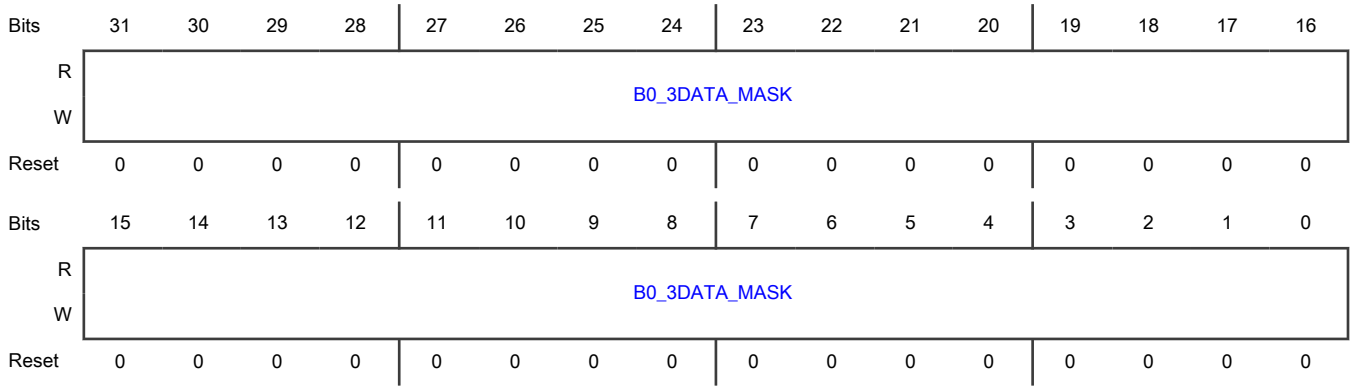
Offset

Register	Offset
EICHD9_WORD1	344h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD9_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD9_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD9_WORD1 on the read data bus is inverted.

50.8.22 Error Injection Channel Descriptor 10, Word1 (EICHD10_WORD1)

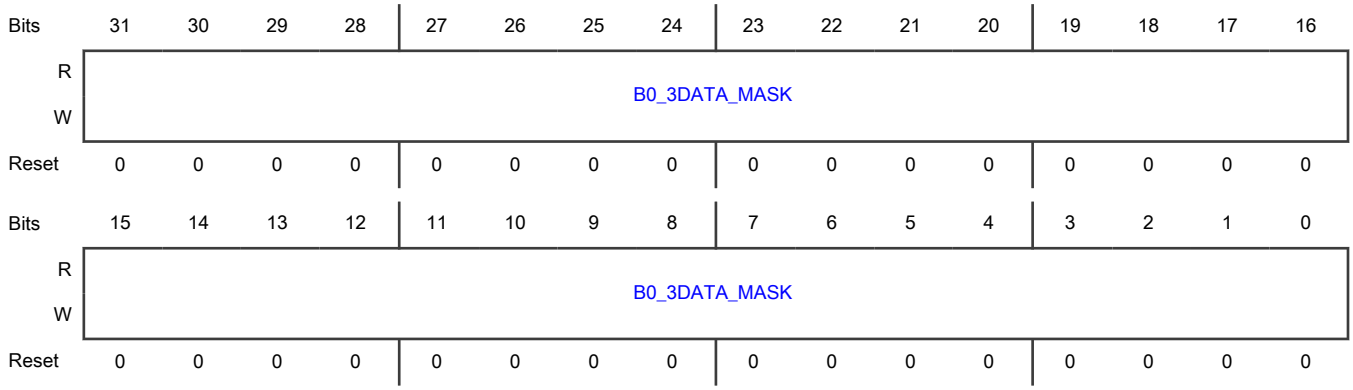
Offset

Register	Offset
EICHD10_WORD1	384h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to EICHD10_WORD1 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Field This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of EICHD10_WORD1 on the read data bus remains unmodified. 1b - The corresponding bit of EICHD10_WORD1 on the read data bus is inverted.

50.9 Glossary

SEC-DED	Single error correction – double error detection
ECC	Error correction code

Chapter 51
Error Reporting Module (ERM)

51.1 Chip-specific ERM information

51.1.1 ERM instances

This chip supports up to two instances of ERM:

- ERM_0
- ERM_1

Table 290. ERM instances

Instances	S32K358/S32K356/ S32K348/S32K338/ S32K328/S32K388/ S32K389	S32K322/S32K324/S32K344/S32K342/S32K341/S32K312/S32K311/ S32K310/S32K314
ERM_0	Yes	Yes
ERM_1	Yes	No

51.1.2 ERM channel mapping

ERM provides information on memory events associated with ECC and parity. It also provides you an option to enable interrupt notification for these events.

NOTE

A hard fault is generated when non-correctable errors are detected.

Table 291. ERM_0 channel mapping for all chips except S32K389

Channel #	Module	Captured status
00	SRAM0	Single-bit error, multi-bit error, syndrome, absolute error address aligned to double-word (64-bit) boundary
01	SRAM1 [1]	Single-bit error, multi-bit error, syndrome, absolute error address+18000h aligned to double-word (64-bit) boundary[2]
02	Cortex-M7_0 I-cache tag RAM	Single-bit error, multi-bit error[3]
03	Cortex-M7_0 I-cache data RAM	Single-bit error, multi-bit error[3]
04	Cortex-M7_0 D-cache tag RAM	Single-bit error, multi-bit error[3]
05	Cortex-M7_0 D-cache data RAM	Single-bit error, multi-bit error[3]
06	Cortex-M7_1 I-cache tag RAM [4]	Single-bit error, multi-bit error[3]
07	Cortex-M7_1 I-cache data RAM	Single-bit error, multi-bit error[3]
08	Cortex-M7_1 D-cache tag RAM	Single-bit error, multi-bit error[3]
09	Cortex-M7_1 D-cache data RAM	Single-bit error, multi-bit error[3]
10	Cortex-M7_0 ITCM	Single-bit error, multi-bit error, syndrome, offset error address

Table continues on the next page...

Table 291. ERM_0 channel mapping for all chips except S32K389 (continued)

Channel #	Module	Captured status
11	Cortex-M7_0 D0TCM	Single-bit error, multi-bit error, syndrome, offset error address
12	Cortex-M7_0 D1TCM	Single-bit error, multi-bit error, syndrome, offset error address[5]
13	Cortex-M7_1 ITCM	Single-bit error, multi-bit error, syndrome, offset error address
14	Cortex-M7_1 D0TCM	Single-bit error, multi-bit error, syndrome, offset error address
15	Cortex-M7_1 D1TCM	Single-bit error, multi-bit error, syndrome, offset error address[5]
16	DMA TCD	Single-bit error, multi-bit error, syndrome, offset error address[6]
17	Flash memory port p0	Single-bit error, multi-bit error, absolute error address
18	Flash memory port p1	Single-bit error, multi-bit error, absolute error address
19	Flash memory port p2 [7]	Single-bit error, multi-bit error, absolute error address

1. SRAM1 is not available for the S32K342, S32K341, S32K322, S32K312, and S32K311 product variants.
2. For variants: S32K314, S32K324, and S32K344 size of SRAM0 is 160 KB, and therefore, to align addresses to the next power of 2, an address space of 256 KB is reserved for error reporting. However, SRAM0 and SRAM1 are in contiguous locations in the memory map. So, you must subtract 18000h (256 KB-160 KB = 96 KB that corresponds to 18000h) from the reported address to get an absolute address. See the "Memory and Memory Interfaces" chapter for SRAM details on different S32K3xx variants.
3. The cache controller does not report error addresses and syndrome.
4. Cortex-M7_1 and its associated RAM and caches are not available in the S32K312 and S32K311 product variants.
5. For address reporting, bit 2 of the address is masked because this bit decides whether the access (read or write) is for D0TCM or D1TCM. For example, if the offset address is 0h or 8h, it is routed to D0TCM, but if the offset address is 4h or Ch, it is routed to D1TCM. The errors that are latched for these offset addresses are as follows:
   - Offset 0h : Address 0h is latched into the ERM channel corresponding to D0TCM.
   - Offset 4h : Address 0h is latched into the ERM channel corresponding to D1TCM.
   - Offset 8h : Address 8h is latched into the ERM channel corresponding to D0TCM.
   - Offset Ch : Address 8h is latched into the ERM channel corresponding to D1TCM.
6. Bits [31:10] and [2:0] are always 0 because they are not connected.

   Bits [9:5] indicate the corresponding TCD out of all the 32 implemented TCDs.

   Bits [4:3] indicate an offset of TCDs with respect to a 64-bit aligned boundary.
   - If [4:3] is 00, it indicates that an error is on offset address 20h.
   - If [4:3] is 01, it indicates that an error is on offset address 28h.
   - If [4:3] is 10, it indicates that an error is on offset address 30h.
   - If [4:3] is 11, it indicates that an error is on offset address 38h.
7. Flash memory port p2 is not applicable for the S32K312 and S32K311 product variants.

Table 292. ERM_0 channel mapping for S32K389

Channel #	Module	Captured status
00	SRAM0	Single-bit error, multi-bit error, syndrome, absolute error address aligned to double-word (64-bit) boundary
01	SRAM1	Single-bit error, multi-bit error, syndrome, absolute error address+18000h aligned to double-word (64-bit) boundary
02	Cortex-M7_0 I-cache tag RAM	Single-bit error, multi-bit error[1]

Table continues on the next page...

Table 292. ERM_0 channel mapping for S32K389 (continued)

Channel #	Module	Captured status
03	Cortex-M7_0 I-cache data RAM	Single-bit error, multi-bit error[1]
04	Cortex-M7_0 D-cache tag RAM	Single-bit error, multi-bit error[1]
05	Cortex-M7_0 D-cache data RAM	Single-bit error, multi-bit error[1]
06	Cortex-M7_1 I-cache tag RAM	Single-bit error, multi-bit error[1]
07	Cortex-M7_1 I-cache data RAM	Single-bit error, multi-bit error[3]
08	Cortex-M7_1 D-cache tag RAM	Single-bit error, multi-bit error[3]
09	Cortex-M7_1 D-cache data RAM	Single-bit error, multi-bit error[3]
10	Cortex-M7_0 ITCM	Single-bit error, multi-bit error, syndrome, offset error address
11	Cortex-M7_0 D0TCM	Single-bit error, multi-bit error, syndrome, offset error address
12	Cortex-M7_0 D1TCM	Single-bit error, multi-bit error, syndrome, offset error address[2]
13	Cortex-M7_1 ITCM	Single-bit error, multi-bit error, syndrome, offset error address
14	Cortex-M7_1 D0TCM	Single-bit error, multi-bit error, syndrome, offset error address
15	Cortex-M7_1 D1TCM	Single-bit error, multi-bit error, syndrome, offset error address[2]
16	DMA TCD	Single-bit error, multi-bit error, syndrome, offset error address[3]
17	Flash memory controller PFC0 port 0	Single-bit error, multi-bit error, absolute error address
18	Flash memory controller PFC0 port 1	Single-bit error, multi-bit error, absolute error address
19	Flash memory controller PFC1 port 0	Single-bit error, multi-bit error, absolute error address
20	Flash memory controller PFC1 port 1	Single-bit error, multi-bit error, absolute error address
21	SRAM3	Single-bit error, multi-bit error, syndrome, absolute error address+18000h aligned to double-word (64-bit) boundary

1. The cache controller does not report error addresses and syndrome.
2. For address reporting, bit 2 of the address is masked because this bit decides whether the access (read or write) is for D0TCM or D1TCM. For example, if the offset address is 0h or 8h, it is routed to D0TCM, but if the offset address is 4h or Ch, it is routed to D1TCM. The errors that are latched for these offset addresses are as follows:
   - Offset 0h : Address 0h is latched into the ERM channel corresponding to D0TCM.
   - Offset 4h : Address 0h is latched into the ERM channel corresponding to D1TCM.
   - Offset 8h : Address 8h is latched into the ERM channel corresponding to D0TCM.
   - Offset Ch : Address 8h is latched into the ERM channel corresponding to D1TCM.
3. Bits [31:10] and [2:0] are always 0 because they are not connected.

   Bits [9:5] indicate the corresponding TCD out of all the 32 implemented TCDs.

   Bits [4:3] indicate an offset of TCDs with respect to a 64-bit aligned boundary.
   - If [4:3] is 00, it indicates that an error is on offset address 20h.
   - If [4:3] is 01, it indicates that an error is on offset address 28h.
   - If [4:3] is 10, it indicates that an error is on offset address 30h.
   - If [4:3] is 11, it indicates that an error is on offset address 38h.

Table 293. ERM_1 channel mapping (S32K389)

Channel #	Module	Captured status
0	SRAM2	Single-bit error, multi-bit error, syndrome, absolute error address aligned to double, word (64-bit) boundary
1	-	-
2	CM7_2 I-cache tag RAM	Single-bit error, multi-bit error
3	CM7_2 I-cache data RAM	Single-bit error, multi-bit error
4	CM7_2 D-cache tag RAM	Single-bit error, multi-bit error
5	CM7_2 D-cache data RAM	Single-bit error, multi-bit error
6	CM7_3 I-cache tag RAM[1]	Single-bit error, multi-bit error
7	CM7_3 I-cache data RAM[1]	Single-bit error, multi-bit error
8	CM7_3 D-cache tag RAM[1]	Single-bit error, multi-bit error
9	CM7_3 D-cache data RAM[1]	Single-bit error, multi-bit error
10	CM7_2 ITCM	Single-bit error, multi-bit error, syndrome, offset error address
11	CM7_2 D0TCM	Single-bit error, multi-bit error, syndrome, offset error address
12	CM7_2 D1TCM	Single-bit error, multi-bit error, syndrome, offset error address
13	CM7_3 ITCM	Single-bit error, multi-bit error, syndrome, offset error address
14	CM7_3 D0TCM	Single-bit error, multi-bit error, syndrome, offset error address
15	CM7_3 D1TCM	Single-bit error, multi-bit error, syndrome, offset error address
16	-	-
17[2]	Flash memory port p3	Single-bit error, multi-bit error, syndrome, offset error address
18	ACE_ACCEL eDMA TCD 32-channel	Single-bit error, multi-bit error, syndrome, offset error address
19	ACE_ACCEL eDMA TCD 24-channel	Single-bit error, multi-bit error, syndrome, offset error address
20	-	-
21	-	-
22	-	-
23	-	-

1. CM7_3 ports are reserved for all chips except S32K388/S32K389.
2. Not applicable for S32K389.

NOTE

For S32K388/S32K389, channel 1 and channel 16 of ERM_1 are Reserved. Read and write to registers related to these channels will not give any transfer error.

NOTE

For S32K358/S32K356/S32K348/S32K338/S32K328, you have to enable both the ERM instances even if you are only using one otherwise software might hang during interrupt subroutine.

51.1.3 ECC error address remapping (S32K389)

The software driver for the ERM module converts the reported error addresses (physical) to the system logic addresses, as per the table below:

Table 294. ECC error address remapping (S32K389)

PFC0 (ERM Channel #17, #18)	
Physical Address (reported to ERM)	System Logical Address (to be converted to)
0x004x_xxxx	0x006x_xxxx
0x005x_xxxx	0x007x_xxxx
0x006x_xxxx	0x008x_xxxx
0x007x_xxxx	0x009x_xxxx
0x008x_xxxx	0x00Cx_xxxx
0x009x_xxxx	0x00Dx_xxxx
0x00Ax_xxxx	0x00Ex_xxxx
0x00Bx_xxxx	0x00Fx_xxxx
PFC1 (ERM Channel #19, #20)	
Physical Address (reported to ERM)	System Logical Address (to be converted to)
0x004x_xxxx	0x004x_xxxx
0x005x_xxxx	0x005x_xxxx
0x006x_xxxx	0x00Ax_xxxx
0x007x_xxxx	0x00Bx_xxxx

51.2 Overview

The Error Reporting Module (ERM) provides information and optional interrupt notification on memory error events associated with ECC and parity. The ERM collects error events on memory accesses for memory arrays, such as flash memory, system RAM, or peripheral RAMs. ERM supports various channels for memory sources where each ERM channel is associated with a different memory module. See the chip-specific ERM information for details about supported memory sources and specific memory channel assignments. If memory supports ECC then ERM syndrome and error address information is captured along with error event. ERM does not receive this information in case of cache or memory with parity along with error event.

51.2.1 Features

The ERM includes these features:

- Optional interrupt notification on captured error events
- Capturing of address and syndrome information on single-bit correction and non-correctable ECC events
- Support for error event capturing for memory sources, with individual reporting fields and interrupt configuration per memory channel
- Recording the count value of the number of corrected error events

51.2.2 Block diagram

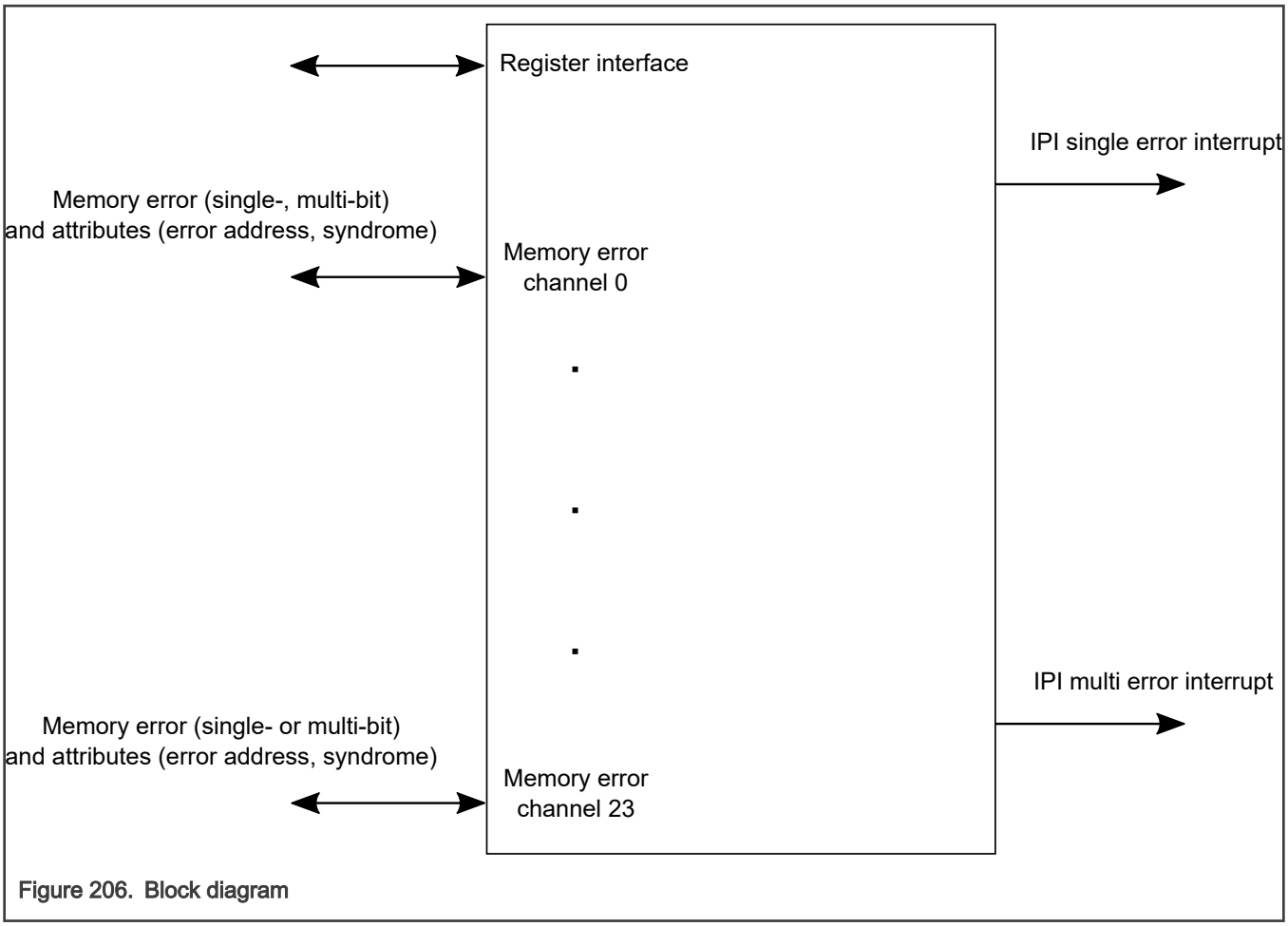


Figure 206. Block diagram

51.3 Functional description

51.3.1 Single-bit correction events

When a single-bit correction event on Memory *n* is detected, the ERM:

- Records the event by changing the value of the applicable Status Register bit SR*x*[SBC*n*] to 1.
- Increments the correctable error count value (until the counter reaches its maximum value): CORR_ERR_CNT*n*[COUNT].
- Records the corresponding access address that initiated the event in the Memory *n* Error Address Register: EAR*n* (if this register is present for the channel).
- Stores the corresponding ECC syndrome in the Memory *n* Error Syndrome Register: SYN*n* (if this register is present for the channel). This register identifies the bit position of the corrected data on single-bit data inversion.

The ERM holds event information only for the last reported event.

To clear the record of an event, write 1 to SR*x*[SBC*n*] to change its value to 0.

To reset the correctable error count value, write all zeros to CORR_ERR_CNT*n*[COUNT].

Optional interrupt notification for single-bit correction events

The ERM provides an option to generate an interrupt notification upon the report of a single-bit correction event. To enable single-bit correction interrupts for a channel:

1. To enable interrupt notification for single-bit correction events on Memory *n*, set CR*x*[ESCIE*n*] to 1.
2. Subsequently, a single-bit correction event on Memory *n* is detected, the ERM:
   - Records the event and address, and stores the ECC syndrome as usual.
   - Additionally sends an interrupt notification corresponding to the event.
3. To clear both the record of an event and the corresponding interrupt notification, write 1 to SR*x*[SBC*n*] to change its value to 0.

51.3.2 Non-correctable error events

When a non-correctable ECC error event on Memory *n* is detected, the ERM:

- Records the event by changing the value of the applicable Status Register bit: SR*x*[NCE*n*] to 1.
- Records the corresponding access address that initiated the event in the Memory *n* Error Address Register: EAR*n* (if this register is present for the channel).
- Stores the corresponding ECC syndrome in the Memory *n* Error Syndrome Register: SYN*n* (if this register is present for the channel).
  - In the event of a non-correctable address bit inversion, SYN*n* identifies the pertinent address bit position.
  - In the event of a non-correctable, multi-bit data inversion, the syndrome value does not provide any additional diagnostic information.

The ERM holds event information only for the last reported event.

To clear the record of an event, write 1 to SR*x*[NCE*n*] to change its value to 0.

Optional interrupt notification for non-correctable error events

The ERM provides an option to generate an interrupt notification upon the report of a non-correctable ECC event. To enable non-correctable error interrupts for a channel:

1. To enable interrupt notifications for non-correctable error events on Memory *n*, set CR*x*[ENCIE*n*] to 1.
2. Subsequently, when a non-correctable error event on Memory *n* is detected, the ERM:
   - Records the event and address and stores the ECC syndrome as usual.
   - Additionally sends an interrupt notification corresponding to the event.
3. To clear both the record of an event and the corresponding interrupt notification, write 1 to SR*x*[NCE*n*] to change its value to 0.

NOTE

Parity errors can be mapped to non-correctable errors where error attributes like SYNDROME, ADDRESS are not provided.

51.4 Initialization

For each ERM channel supporting memory with ECC, prepare the corresponding memory array before enabling ERM interrupts about errors for that memory.

1. Initialize the memory to a known value so that the correct corresponding ECC codeword is stored.
2. During the memory's initialization, if the ERM captures information about any ECC error event, clear the corresponding SR*x*[SBC*n*] or SR*x*[NCE*n*] field that stores the record of the event.
3. Program the applicable CR*x*[ESCIE*n*] and CR*x*[ENCIE*n*] fields to enable ERM interrupts as desired.

51.5 ERM_0 register descriptions

You can access the programming model:

- Only in supervisor mode
- Using only 32-bit (word) accesses

Any of the following attempted references to the programming model generates an error termination:

- In user mode
- Using non-32-bit access sizes

Based on the design implementation, the following XFR error behavior is evident at the IPS interface.

- Within the ERM memory map, an XFR error is evident at reserved addresses from location 20h to FFh.
- No XFR error is evident at reserved addresses in memory spaces allocated to each channel. For example: For channel 0, for read/write accesses to reserved address 10Ch, the XFR error is 0.
- For accesses to locations beyond the addresses allocated for the final channel, the XFR error is 1.

NOTE

- See the chip-specific ERM information at the beginning of this chapter for details on Memory channel mapping.
- To access the channel registers, corresponding memory channel clock must be enabled.

51.5.1 ERM_0 memory map

ERM_0 base address: 4025_C000h

Offset	Register	Width (In bits)	Access	Reset value
0h	ERM Configuration Register 0 (CR0)	32	RW	0000_0000h
4h	ERM Configuration Register 1 (CR1)	32	RW	0000_0000h
8h	ERM Configuration Register 2 (CR2)	32	RW	0000_0000h
10h	ERM Status Register 0 (SR0)	32	RW	0000_0000h
14h	ERM Status Register 1 (SR1)	32	RW	0000_0000h
18h	ERM Status Register 2 (SR2)	32	RW	0000_0000h
100h	ERM Memory 0 Error Address Register (EAR0)	32	R	0000_0000h
104h	ERM Memory 0 Syndrome Register (SYN0)	32	R	0000_0000h
108h	ERM Memory 0 Correctable Error Count Register (CORR_ERR_CNT0)	32	RW	0000_0000h
110h	ERM Memory 1 Error Address Register (EAR1)	32	R	0000_0000h
114h	ERM Memory 1 Syndrome Register (SYN1)	32	R	0000_0000h
118h	ERM Memory 1 Correctable Error Count Register (CORR_ERR_CNT1)	32	RW	0000_0000h
128h	ERM Memory 2 Correctable Error Count Register (CORR_ERR_CNT2)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
138h	ERM Memory 3 Correctable Error Count Register (CORR_ERR_CNT3)	32	RW	0000_0000h
148h	ERM Memory 4 Correctable Error Count Register (CORR_ERR_CNT4)	32	RW	0000_0000h
158h	ERM Memory 5 Correctable Error Count Register (CORR_ERR_CNT5)	32	RW	0000_0000h
168h	ERM Memory 6 Correctable Error Count Register (CORR_ERR_CNT6)	32	RW	0000_0000h
178h	ERM Memory 7 Correctable Error Count Register (CORR_ERR_CNT7)	32	RW	0000_0000h
188h	ERM Memory 8 Correctable Error Count Register (CORR_ERR_CNT8)	32	RW	0000_0000h
198h	ERM Memory 9 Correctable Error Count Register (CORR_ERR_CNT9)	32	RW	0000_0000h
1A0h	ERM Memory 10 Error Address Register (EAR10)	32	R	0000_0000h
1A4h	ERM Memory 10 Syndrome Register (SYN10)	32	R	0000_0000h
1A8h	ERM Memory 10 Correctable Error Count Register (CORR_ERR_CNT10)	32	RW	0000_0000h
1B0h	ERM Memory 11 Error Address Register (EAR11)	32	R	0000_0000h
1B4h	ERM Memory 11 Syndrome Register (SYN11)	32	R	0000_0000h
1B8h	ERM Memory 11 Correctable Error Count Register (CORR_ERR_CNT11)	32	RW	0000_0000h
1C0h	ERM Memory 12 Error Address Register (EAR12)	32	R	0000_0000h
1C4h	ERM Memory 12 Syndrome Register (SYN12)	32	R	0000_0000h
1C8h	ERM Memory 12 Correctable Error Count Register (CORR_ERR_CNT12)	32	RW	0000_0000h
1D0h	ERM Memory 13 Error Address Register (EAR13)	32	R	0000_0000h
1D4h	ERM Memory 13 Syndrome Register (SYN13)	32	R	0000_0000h
1D8h	ERM Memory 13 Correctable Error Count Register (CORR_ERR_CNT13)	32	RW	0000_0000h
1E0h	ERM Memory 14 Error Address Register (EAR14)	32	R	0000_0000h
1E4h	ERM Memory 14 Syndrome Register (SYN14)	32	R	0000_0000h
1E8h	ERM Memory 14 Correctable Error Count Register (CORR_ERR_CNT14)	32	RW	0000_0000h
1F0h	ERM Memory 15 Error Address Register (EAR15)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1F4h	ERM Memory 15 Syndrome Register (SYN15)	32	R	0000_0000h
1F8h	ERM Memory 15 Correctable Error Count Register (CORR_ERR_CNT15)	32	RW	0000_0000h
200h	ERM Memory 16 Error Address Register (EAR16)	32	R	0000_0000h
204h	ERM Memory 16 Syndrome Register (SYN16)	32	R	0000_0000h
208h	ERM Memory 16 Correctable Error Count Register (CORR_ERR_CNT16)	32	RW	0000_0000h
210h	ERM Memory 17 Error Address Register (EAR17)	32	R	0000_0000h
218h	ERM Memory 17 Correctable Error Count Register (CORR_ERR_CNT17)	32	RW	0000_0000h
220h	ERM Memory 18 Error Address Register (EAR18)	32	R	0000_0000h
228h	ERM Memory 18 Correctable Error Count Register (CORR_ERR_CNT18)	32	RW	0000_0000h
230h	ERM Memory 19 Error Address Register (EAR19)	32	R	0000_0000h
238h	ERM Memory 19 Correctable Error Count Register (CORR_ERR_CNT19)	32	RW	0000_0000h
240h	ERM Memory 20 Error Address Register (EAR20)	32	R	0000_0000h
248h	ERM Memory 20 Correctable Error Count Register (CORR_ERR_CNT20)	32	RW	0000_0000h
250h	ERM Memory 21 Error Address Register (EAR21)	32	R	0000_0000h
254h	ERM Memory 21 Syndrome Register (SYN21)	32	R	0000_0000h
258h	ERM Memory 21 Correctable Error Count Register (CORR_ERR_CNT21)	32	RW	0000_0000h

51.5.2 ERM Configuration Register 0 (CR0)

Offset

Register	Offset
CR0	0h

Function

This 32-bit control register configures the interrupt notification capability for available channels.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ESCIE0	ENCIE0	0		ESCIE1	ENCIE1	0		ESCIE2	ENCIE2	0		ESCIE3	ENCIE3	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ESCIE4	ENCIE4	0		ESCIE5	ENCIE5	0		ESCIE6	ENCIE6	0		ESCIE7	ENCIE7	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 ESCIE0	ESCIE0 Enable Memory 0 Single Correction Interrupt Notification 0b - Interrupt notification of Memory 0 single-bit correction events is disabled. 1b - Interrupt notification of Memory 0 single-bit correction events is enabled.
30 ENCIE0	ENCIE0 Enable Memory 0 Non-Correctable Interrupt Notification 0b - Interrupt notification of Memory 0 non-correctable error events is disabled. 1b - Interrupt notification of Memory 0 non-correctable error events is enabled.
29-28 —	Reserved
27 ESCIE1	ESCIE1 Enable Memory 1 Single Correction Interrupt Notification 0b - Interrupt notification of Memory 1 single-bit correction events is disabled. 1b - Interrupt notification of Memory 1 single-bit correction events is enabled.
26 ENCIE1	ENCIE1 Enable Memory 1 Non-Correctable Interrupt Notification 0b - Interrupt notification of Memory 1 non-correctable error events is disabled. 1b - Interrupt notification of Memory 1 non-correctable error events is enabled.
25-24 —	Reserved
23	ESCIE2

Table continues on the next page...

Table continued from the previous page...

Field	Function
ESCIE2	Enable Memory 2 Single Correction Interrupt Notification 0b - Interrupt notification of Memory 2 single-bit correction events is disabled. 1b - Interrupt notification of Memory 2 single-bit correction events is enabled.
22 ENCIE2	ENCIE2 Enable Memory 2 Non-Correctable Interrupt Notification 0b - Interrupt notification of Memory 2 non-correctable error events is disabled. 1b - Interrupt notification of Memory 2 non-correctable error events is enabled.
21-20 —	Reserved
19 ESCIE3	ESCIE3 Enable Memory 3 Single Correction Interrupt Notification 0b - Interrupt notification of Memory 3 single-bit correction events is disabled. 1b - Interrupt notification of Memory 3 single-bit correction events is enabled.
18 ENCIE3	ENCIE3 Enable Memory 3 Non-Correctable Interrupt Notification 0b - Interrupt notification of Memory 3 non-correctable error events is disabled. 1b - Interrupt notification of Memory 3 non-correctable error events is enabled.
17-16 —	Reserved
15 ESCIE4	ESCIE4 Enable Memory 4 Single Correction Interrupt Notification 0b - Interrupt notification of Memory 4 single-bit correction events is disabled. 1b - Interrupt notification of Memory 4 single-bit correction events is enabled.
14 ENCIE4	ENCIE4 Enable Memory 4 Non-Correctable Interrupt Notification 0b - Interrupt notification of Memory 4 non-correctable error events is disabled. 1b - Interrupt notification of Memory 4 non-correctable error events is enabled.
13-12 —	Reserved
11	ESCIE5 Enable Memory 5 Single Correction Interrupt Notification

Table continues on the next page...

Table continued from the previous page...

Field	Function
ESCIE5	0b - Interrupt notification of Memory 5 single-bit correction events is disabled. 1b - Interrupt notification of Memory 5 single-bit correction events is enabled.
10 ENCIE5	ENCIE5 Enable Memory 5 Non-Correctable Interrupt Notification 0b - Interrupt notification of Memory 5 non-correctable error events is disabled. 1b - Interrupt notification of Memory 5 non-correctable error events is enabled.
9-8 —	Reserved
7 ESCIE6	ESCIE6 Enable Memory 6 Single Correction Interrupt Notification 0b - Interrupt notification of Memory 6 single-bit correction events is disabled. 1b - Interrupt notification of Memory 6 single-bit correction events is enabled.
6 ENCIE6	ENCIE6 Enable Memory 6 Non-Correctable Interrupt Notification 0b - Interrupt notification of Memory 6 non-correctable error events is disabled. 1b - Interrupt notification of Memory 6 non-correctable error events is enabled.
5-4 —	Reserved
3 ESCIE7	ESCIE7 Enable Memory 7 Single Correction Interrupt Notification 0b - Interrupt notification of Memory 7 single-bit correction events is disabled. 1b - Interrupt notification of Memory 7 single-bit correction events is enabled.
2 ENCIE7	ENCIE7 Enable Memory 7 Non-Correctable Interrupt Notification 0b - Interrupt notification of Memory 7 non-correctable error events is disabled. 1b - Interrupt notification of Memory 7 non-correctable error events is enabled.
1-0 —	Reserved

51.5.3 ERM Configuration Register 1 (CR1)

Offset

Register	Offset
CR1	4h

Function

This 32-bit control register configures the interrupt notification capability for available channels.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ESCIE8	ENCIE8	0		ESCIE9	ENCIE9	0		ESCIE10	ENCIE10	0		ESCIE11	ENCIE11	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ESCIE12	ENCIE12	0		ESCIE13	ENCIE13	0		ESCIE14	ENCIE14	0		ESCIE15	ENCIE15	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 ESCIE8	ESCIE8 Enable Memory 8 Single Correction Interrupt Notification 0b - Interrupt notification of Memory 8 single-bit correction events is disabled. 1b - Interrupt notification of Memory 8 single-bit correction events is enabled.
30 ENCIE8	ENCIE8 Enable Memory 8 Non-Correctable Interrupt Notification 0b - Interrupt notification of Memory 8 non-correctable error events is disabled. 1b - Interrupt notification of Memory 8 non-correctable error events is enabled.
29-28 —	Reserved
27 ESCIE9	ESCIE9 Enable Memory 9 Single Correction Interrupt Notification 0b - Interrupt notification of Memory 9 single-bit correction events is disabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Interrupt notification of Memory 9 single-bit correction events is enabled.
26 ENCIE9	ENCIE9 Enable Memory 9 Non-Correctable Interrupt Notification 0b - Interrupt notification of Memory 9 non-correctable error events is disabled. 1b - Interrupt notification of Memory 9 non-correctable error events is enabled.
25-24 —	Reserved
23 ESCIE10	ESCIE10 Enable Memory 10 Single Correction Interrupt Notification 0b - Interrupt notification of Memory 10 single-bit correction events is disabled. 1b - Interrupt notification of Memory 10 single-bit correction events is enabled.
22 ENCIE10	ENCIE10 Enable Memory 10 Non-Correctable Interrupt Notification 0b - Interrupt notification of Memory 10 non-correctable error events is disabled. 1b - Interrupt notification of Memory 10 non-correctable error events is enabled.
21-20 —	Reserved
19 ESCIE11	ESCIE11 Enable Memory 11 Single Correction Interrupt Notification 0b - Interrupt notification of Memory 11 single-bit correction events is disabled. 1b - Interrupt notification of Memory 11 single-bit correction events is enabled.
18 ENCIE11	ENCIE11 Enable Memory 11 Non-Correctable Interrupt Notification 0b - Interrupt notification of Memory 11 non-correctable error events is disabled. 1b - Interrupt notification of Memory 11 non-correctable error events is enabled.
17-16 —	Reserved
15 ESCIE12	ESCIE12 Enable Memory 12 Single Correction Interrupt Notification 0b - Interrupt notification of Memory 12 single-bit correction events is disabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Interrupt notification of Memory 12 single-bit correction events is enabled.
14 ENCIE12	ENCIE12 Enable Memory 12 Non-Correctable Interrupt Notification 0b - Interrupt notification of Memory 12 non-correctable error events is disabled. 1b - Interrupt notification of Memory 12 non-correctable error events is enabled.
13-12 —	Reserved
11 ESCIE13	ESCIE13 Enable Memory 13 Single Correction Interrupt Notification 0b - Interrupt notification of Memory 13 single-bit correction events is disabled. 1b - Interrupt notification of Memory 13 single-bit correction events is enabled.
10 ENCIE13	ENCIE13 Enable Memory 13 Non-Correctable Interrupt Notification 0b - Interrupt notification of Memory 13 non-correctable error events is disabled. 1b - Interrupt notification of Memory 13 non-correctable error events is enabled.
9-8 —	Reserved
7 ESCIE14	ESCIE14 Enable Memory 14 Single Correction Interrupt Notification 0b - Interrupt notification of Memory 14 single-bit correction events is disabled. 1b - Interrupt notification of Memory 14 single-bit correction events is enabled.
6 ENCIE14	ENCIE14 Enable Memory 14 Non-Correctable Interrupt Notification 0b - Interrupt notification of Memory 14 non-correctable error events is disabled. 1b - Interrupt notification of Memory 14 non-correctable error events is enabled.
5-4 —	Reserved
3 ESCIE15	ESCIE15 Enable Memory 15 Single Correction Interrupt Notification 0b - Interrupt notification of Memory 15 single-bit correction events is disabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Interrupt notification of Memory 15 single-bit correction events is enabled.
2 ENCIE15	ENCIE15 Enable Memory 15 Non-Correctable Interrupt Notification 0b - Interrupt notification of Memory 15 non-correctable error events is disabled. 1b - Interrupt notification of Memory 15 non-correctable error events is enabled.
1-0 —	Reserved

51.5.4 ERM Configuration Register 2 (CR2)

Offset

Register	Offset
CR2	8h

Function

This 32-bit control register configures the interrupt notification capability for available channels.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ESCIE16	ENCIE16	0		ESCIE17	ENCIE17	0		ESCIE18	ENCIE18	0		ESCIE19	ENCIE19	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ESCIE20	ENCIE20	0		ESCIE21	ENCIE21	0		Reserved		0		Reserved		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 ESCIE16	ESCIE16 Enable Memory 16 Single Correction Interrupt Notification

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Interrupt notification of Memory 16 single-bit correction events is disabled. 1b - Interrupt notification of Memory 16 single-bit correction events is enabled.
30 ENCIE16	ENCIE16 Enable Memory 16 Non-Correctable Interrupt Notification 0b - Interrupt notification of Memory 16 non-correctable error events is disabled. 1b - Interrupt notification of Memory 16 non-correctable error events is enabled.
29-28 —	Reserved
27 ESCIE17	ESCIE17 Enable Memory 17 Single Correction Interrupt Notification 0b - Interrupt notification of Memory 17 single-bit correction events is disabled. 1b - Interrupt notification of Memory 17 single-bit correction events is enabled.
26 ENCIE17	ENCIE17 Enable Memory 17 Non-Correctable Interrupt Notification 0b - Interrupt notification of Memory 17 non-correctable error events is disabled. 1b - Interrupt notification of Memory 17 non-correctable error events is enabled.
25-24 —	Reserved
23 ESCIE18	ESCIE18 Enable Memory 18 Single Correction Interrupt Notification 0b - Interrupt notification of Memory 18 single-bit correction events is disabled. 1b - Interrupt notification of Memory 18 single-bit correction events is enabled.
22 ENCIE18	ENCIE18 Enable Memory 18 Non-Correctable Interrupt Notification 0b - Interrupt notification of Memory 18 non-correctable error events is disabled. 1b - Interrupt notification of Memory 18 non-correctable error events is enabled.
21-20 —	Reserved
19 ESCIE19	ESCIE19 Enable Memory 19 Single Correction Interrupt Notification

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Interrupt notification of Memory 19 single-bit correction events is disabled. 1b - Interrupt notification of Memory 19 single-bit correction events is enabled.
18 ENCIE19	ENCIE19 Enable Memory 19 Non-Correctable Interrupt Notification 0b - Interrupt notification of Memory 19 non-correctable error events is disabled. 1b - Interrupt notification of Memory 19 non-correctable error events is enabled.
17-16 —	Reserved
15 ESCIE20	ESCIE20 Enable Memory 20 Single Correction Interrupt Notification 0b - Interrupt notification of Memory 20 single-bit correction events is disabled. 1b - Interrupt notification of Memory 20 single-bit correction events is enabled.
14 ENCIE20	ENCIE20 Enable Memory 20 Non-Correctable Interrupt Notification 0b - Interrupt notification of Memory 20 non-correctable error events is disabled. 1b - Interrupt notification of Memory 20 non-correctable error events is enabled.
13-12 —	Reserved
11 ESCIE21	ESCIE21 Enable Memory 21 Single Correction Interrupt Notification 0b - Interrupt notification of Memory 21 single-bit correction events is disabled. 1b - Interrupt notification of Memory 21 single-bit correction events is enabled.
10 ENCIE21	ENCIE21 Enable Memory 21 Non-Correctable Interrupt Notification 0b - Interrupt notification of Memory 21 non-correctable error events is disabled. 1b - Interrupt notification of Memory 21 non-correctable error events is enabled.
9-8 —	Reserved
7-6 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
5-4 —	Reserved
3-2 —	Reserved
1-0 —	Reserved

51.5.5 ERM Status Register 0 (SR0)

Offset

Register	Offset
SR0	10h

Function

This 32-bit status register reports error events for available channels.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC0	NCE0	0		SBC1	NCE1	0		SBC2	NCE2	0		SBC3	NCE3	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SBC4	NCE4	0		SBC5	NCE5	0		SBC6	NCE6	0		SBC7	NCE7	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SBC0	SBC0 Memory 0 Single-Bit Correction Event Write 1 to clear this field.This write also clears the corresponding interrupt notification, if CR0[ESCIE0] is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No single-bit correction event on Memory 0 detected. 1b - Single-bit correction event on Memory 0 detected.
30 NCE0	NCE0 Memory 0 Non-Correctable Error Event Write 1 to clear this field.This write also clears the corresponding interrupt notification, if CR0[ENCIE0] is enabled. 0b - No non-correctable error event on Memory 0 detected. 1b - Non-correctable error event on Memory 0 detected.
29-28 —	Reserved
27 SBC1	SBC1 Memory 1 Single-Bit Correction Event Write 1 to clear this field.This write also clears the corresponding interrupt notification, if CR0[ESCIE1] is enabled. 0b - No single-bit correction event on Memory 1 detected. 1b - Single-bit correction event on Memory 1 detected.
26 NCE1	NCE1 Memory 1 Non-Correctable Error Event Write 1 to clear this field.This write also clears the corresponding interrupt notification, if CR0[ENCIE1] is enabled. 0b - No non-correctable error event on Memory 1 detected. 1b - Non-correctable error event on Memory 1 detected.
25-24 —	Reserved
23 SBC2	SBC2 Memory 2 Single-Bit Correction Event Write 1 to clear this field.This write also clears the corresponding interrupt notification, if CR0[ESCIE2] is enabled. 0b - No single-bit correction event on Memory 2 detected. 1b - Single-bit correction event on Memory 2 detected.
22 NCE2	NCE2 Memory 2 Non-Correctable Error Event

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Write 1 to clear this field.This write also clears the corresponding interrupt notification, if CR0[ENCIE2] is enabled. 0b - No non-correctable error event on Memory 2 detected. 1b - Non-correctable error event on Memory 2 detected.
21-20 —	Reserved
19 SBC3	SBC3 Memory 3 Single-Bit Correction Event Write 1 to clear this field.This write also clears the corresponding interrupt notification, if CR0[ESCIE3] is enabled. 0b - No single-bit correction event on Memory 3 detected. 1b - Single-bit correction event on Memory 3 detected.
18 NCE3	NCE3 Memory 3 Non-Correctable Error Event Write 1 to clear this field.This write also clears the corresponding interrupt notification, if CR0[ENCIE3] is enabled. 0b - No non-correctable error event on Memory 3 detected. 1b - Non-correctable error event on Memory 3 detected.
17-16 —	Reserved
15 SBC4	SBC4 Memory 4 Single-Bit Correction Event Write 1 to clear this field.This write also clears the corresponding interrupt notification, if CR0[ESCIE4] is enabled. 0b - No single-bit correction event on Memory 4 detected. 1b - Single-bit correction event on Memory 4 detected.
14 NCE4	NCE4 Memory 4 Non-Correctable Error Event Write 1 to clear this field.This write also clears the corresponding interrupt notification, if CR0[ENCIE4] is enabled. 0b - No non-correctable error event on Memory 4 detected. 1b - Non-correctable error event on Memory 4 detected.
13-12	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
11 SBC5	SBC5 Memory 5 Single-Bit Correction Event Write 1 to clear this field.This write also clears the corresponding interrupt notification, if CR0[ESCIE5] is enabled. 0b - No single-bit correction event on Memory 5 detected. 1b - Single-bit correction event on Memory 5 detected.
10 NCE5	NCE5 Memory 5 Non-Correctable Error Event Write 1 to clear this field.This write also clears the corresponding interrupt notification, if CR0[ENCIE5] is enabled. 0b - No non-correctable error event on Memory 5 detected. 1b - Non-correctable error event on Memory 5 detected.
9-8 —	Reserved
7 SBC6	SBC6 Memory 6 Single-Bit Correction Event Write 1 to clear this field.This write also clears the corresponding interrupt notification, if CR0[ESCIE6] is enabled. 0b - No single-bit correction event on Memory 6 detected. 1b - Single-bit correction event on Memory 6 detected.
6 NCE6	NCE6 Memory 6 Non-Correctable Error Event Write 1 to clear this field.This write also clears the corresponding interrupt notification, if CR0[ENCIE6] is enabled. 0b - No non-correctable error event on Memory 6 detected. 1b - Non-correctable error event on Memory 6 detected.
5-4 —	Reserved
3 SBC7	SBC7 Memory 7 Single-Bit Correction Event Write 1 to clear this field.This write also clears the corresponding interrupt notification, if CR0[ESCIE7] is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No single-bit correction event on Memory 7 detected. 1b - Single-bit correction event on Memory 7 detected.
2 NCE7	NCE7 Memory 7 Non-Correctable Error Event Write 1 to clear this field.This write also clears the corresponding interrupt notification, if CR0[ENCIE7] is enabled. 0b - No non-correctable error event on Memory 7 detected. 1b - Non-correctable error event on Memory 7 detected.
1-0 —	Reserved

51.5.6 ERM Status Register 1 (SR1)

Offset

Register	Offset
SR1	14h

Function

This 32-bit status register reports error events for available channels.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC8	NCE8	0		SBC9	NCE9	0		SBC10	NCE10	0		SBC11	NCE11	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SBC12	NCE12	0		SBC13	NCE13	0		SBC14	NCE14	0		SBC15	NCE15	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SBC8	SBC8 Memory 8 Single-Bit Correction Event Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE8] is enabled. 0b - No single-bit correction event on Memory 8 detected. 1b - Single-bit correction event on Memory 8 detected.
30 NCE8	NCE8 Memory 8 Non-Correctable Error Event Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE8] is enabled. 0b - No non-correctable error event on Memory 8 detected. 1b - Non-correctable error event on Memory 8 detected.
29-28 —	Reserved
27 SBC9	SBC9 Memory 9 Single-Bit Correction Event Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE9] is enabled. 0b - No single-bit correction event on Memory 9 detected. 1b - Single-bit correction event on Memory 9 detected.
26 NCE9	NCE9 Memory 9 Non-Correctable Error Event Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE9] is enabled. 0b - No non-correctable error event on Memory 9 detected. 1b - Non-correctable error event on Memory 9 detected.
25-24 —	Reserved
23 SBC10	SBC10 Memory 10 Single-Bit Correction Event Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE10] is enabled. 0b - No single-bit correction event on Memory 10 detected.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Single-bit correction event on Memory 10 detected.
22 NCE10	NCE10 Memory 10 Non-Correctable Error Event Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE10] is enabled. 0b - No non-correctable error event on Memory 10 detected. 1b - Non-correctable error event on Memory 10 detected.
21-20 —	Reserved
19 SBC11	SBC11 Memory 11 Single-Bit Correction Event Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE11] is enabled. 0b - No single-bit correction event on Memory 11 detected. 1b - Single-bit correction event on Memory 11 detected.
18 NCE11	NCE11 Memory 11 Non-Correctable Error Event Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE11] is enabled. 0b - No non-correctable error event on Memory 11 detected. 1b - Non-correctable error event on Memory 11 detected.
17-16 —	Reserved
15 SBC12	SBC12 Memory 12 Single-Bit Correction Event Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE12] is enabled. 0b - No single-bit correction event on Memory 12 detected. 1b - Single-bit correction event on Memory 12 detected.
14 NCE12	NCE12 Memory 12 Non-Correctable Error Event Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE12] is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No non-correctable error event on Memory 12 detected. 1b - Non-correctable error event on Memory 12 detected.
13-12 —	Reserved
11 SBC13	SBC13 Memory 13 Single-Bit Correction Event Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE13] is enabled. 0b - No single-bit correction event on Memory 13 detected. 1b - Single-bit correction event on Memory 13 detected.
10 NCE13	NCE13 Memory 13 Non-Correctable Error Event Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE13] is enabled. 0b - No non-correctable error event on Memory 13 detected. 1b - Non-correctable error event on Memory 13 detected.
9-8 —	Reserved
7 SBC14	SBC14 Memory 14 Single-Bit Correction Event Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE14] is enabled. 0b - No single-bit correction event on Memory 14 detected. 1b - Single-bit correction event on Memory 14 detected.
6 NCE14	NCE14 Memory 14 Non-Correctable Error Event Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE14] is enabled. 0b - No non-correctable error event on Memory 14 detected. 1b - Non-correctable error event on Memory 14 detected.
5-4 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
3 SBC15	SBC15 Memory 15 Single-Bit Correction Event Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE15] is enabled. 0b - No single-bit correction event on Memory 15 detected. 1b - Single-bit correction event on Memory 15 detected.
2 NCE15	NCE15 Memory 15 Non-Correctable Error Event Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE15] is enabled. 0b - No non-correctable error event on Memory 15 detected. 1b - Non-correctable error event on Memory 15 detected.
1-0 —	Reserved

51.5.7 ERM Status Register 2 (SR2)

Offset

Register	Offset
SR2	18h

Function

This 32-bit status register reports error events for available channels.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC16	NCE16	0		SBC17	NCE17	0		SBC18	NCE18	0		SBC19	NCE19	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SBC20	NCE20	0		SBC21	NCE21	0		0				0			
W	W1C	W1C			W1C	W1C										
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SBC16	SBC16 Memory 16 Single-Bit Correction Event Write 1 to clear this field.This write also clears the corresponding interrupt notification, if CR2[ESCIE16] is enabled. 0b - No single-bit correction event on Memory 16 detected. 1b - Single-bit correction event on Memory 16 detected.
30 NCE16	NCE16 Memory 16 Non-Correctable Error Event Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ENCIE16] is enabled. 0b - No non-correctable error event on Memory 16 detected. 1b - Non-correctable error event on Memory 16 detected.
29-28 —	Reserved
27 SBC17	SBC17 Memory 17 Single-Bit Correction Event Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ESCIE17] is enabled. 0b - No single-bit correction event on Memory 17 detected. 1b - Single-bit correction event on Memory 17 detected.
26 NCE17	NCE17 Memory 17 Non-Correctable Error Event Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ENCIE17] is enabled. 0b - No non-correctable error event on Memory 17 detected. 1b - Non-correctable error event on Memory 17 detected.
25-24 —	Reserved
23 SBC18	SBC18 Memory 18 Single-Bit Correction Event Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ESCIE18] is enabled. 0b - No single-bit correction event on Memory 18 detected.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Single-bit correction event on Memory 18 detected.
22 NCE18	NCE18 Memory 18 Non-Correctable Error Event Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ENCIE18] is enabled. 0b - No non-correctable error event on Memory 18 detected. 1b - Non-correctable error event on Memory 18 detected.
21-20 —	Reserved
19 SBC19	SBC19 Memory 19 Single-Bit Correction Event Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ESCIE19] is enabled. 0b - No single-bit correction event on Memory 19 detected. 1b - Single-bit correction event on Memory 19 detected.
18 NCE19	NCE19 Memory 19 Non-Correctable Error Event Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ENCIE19] is enabled. 0b - No non-correctable error event on Memory 19 detected. 1b - Non-correctable error event on Memory 19 detected.
17-16 —	Reserved
15 SBC20	SBC20 Memory 20 Single-Bit Correction Event Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ESCIE20] is enabled. 0b - No single-bit correction event on Memory 20 detected. 1b - Single-bit correction event on Memory 20 detected.
14 NCE20	NCE20 Memory 20 Non-Correctable Error Event Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ENCIE20] is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No non-correctable error event on Memory 20 detected. 1b - Non-correctable error event on Memory 20 detected.
13-12 —	Reserved
11 SBC21	SBC21 Memory 21 Single-Bit Correction Event Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ESCIE21] is enabled. 0b - No single-bit correction event on Memory 21 detected. 1b - Single-bit correction event on Memory 21 detected.
10 NCE21	NCE21 Memory 21 Non-Correctable Error Event Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ENCIE21] is enabled. 0b - No non-correctable error event on Memory 21 detected. 1b - Non-correctable error event on Memory 21 detected.
9-8 —	Reserved
7-4 —	Reserved
3-0 —	Reserved

51.5.8 ERM Memory a Error Address Register (EAR0 - EAR21)

Offset

Register	Offset
EAR0	100h
EAR1	110h
EAR10	1A0h
EAR11	1B0h
EAR12	1C0h

Table continues on the next page...

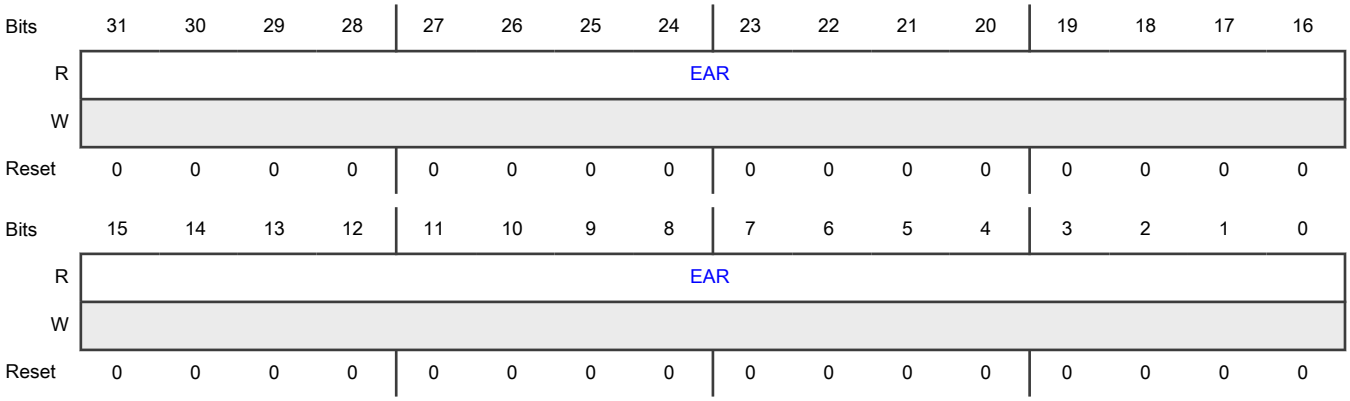
Table continued from the previous page...

Register	Offset
EAR13	1D0h
EAR14	1E0h
EAR15	1F0h
EAR16	200h
EAR17	210h
EAR18	220h
EAR19	230h
EAR20	240h
EAR21	250h

Function

Each ERM Memory n Error Address Register is a 32-bit register for capturing the address of the last ECC event in Memory*n*, where*n* denotes the memory channel. Any attempted write to EAR*n* is ignored. See the chip-specific ERM information for the ERM Memory a Error Address Register.

Diagram



Fields

Field	Function
31-0 EAR	EAR Memory*n* Error Address — This field contains the faulting system address of the last recorded ECC event on Memory*n*.

51.5.9 ERM Memory a Syndrome Register (SYN0 - SYN21)

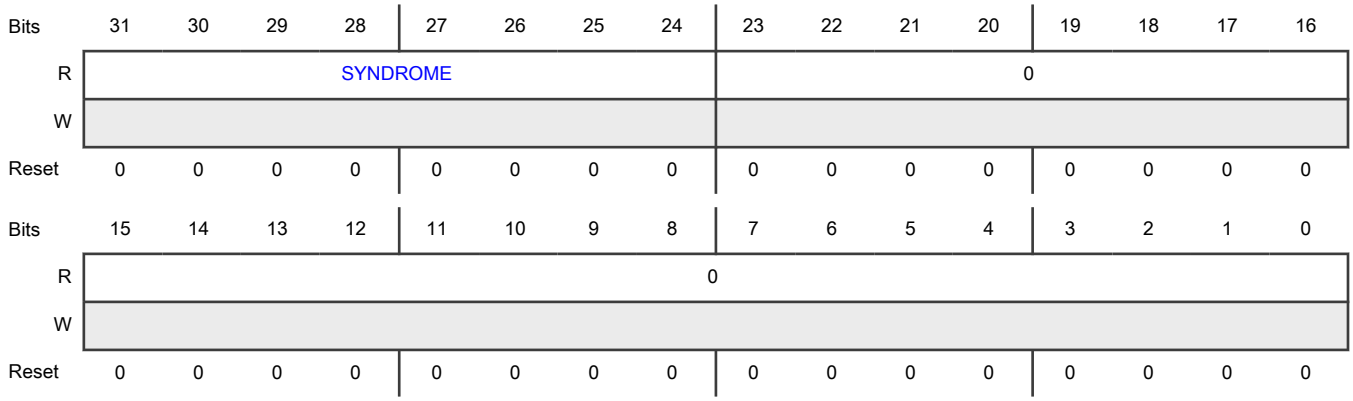
Offset

Register	Offset
SYN0	104h
SYN1	114h
SYN10	1A4h
SYN11	1B4h
SYN12	1C4h
SYN13	1D4h
SYN14	1E4h
SYN15	1F4h
SYN16	204h
SYN21	254h

Function

The ERM Memory *n* Syndrome Register is a 32-bit register for capturing the calculated syndrome of the last ECC event on Memory*n*, where*n* denotes the memory channel. Any attempted write to SYN*n* is ignored. The syndrome value identifies the pertinent bit position on a correctable, single-bit data inversion or a non-correctable, single-bit address inversion. The syndrome value does not provide any additional diagnostic information on non-correctable, multi-bit inversions.

Diagram



Fields

Field	Function
31-24 SYNDROME	SYNDROME Memory*n* Syndrome — This field contains the ECC syndrome associated with the last recorded ECC event on Memory*n*.

Table continues on the next page...

Table continued from the previous page...

Field	Function
23-0 —	Reserved

51.5.10 ERM Memory a Correctable Error Count Register (CORR_ERR_CNT0 - CORR_ERR_CNT21)

Offset

Register	Offset
CORR_ERR_CNT0	108h
CORR_ERR_CNT1	118h
CORR_ERR_CNT2	128h
CORR_ERR_CNT3	138h
CORR_ERR_CNT4	148h
CORR_ERR_CNT5	158h
CORR_ERR_CNT6	168h
CORR_ERR_CNT7	178h
CORR_ERR_CNT8	188h
CORR_ERR_CNT9	198h
CORR_ERR_CNT10	1A8h
CORR_ERR_CNT11	1B8h
CORR_ERR_CNT12	1C8h
CORR_ERR_CNT13	1D8h
CORR_ERR_CNT14	1E8h
CORR_ERR_CNT15	1F8h
CORR_ERR_CNT16	208h
CORR_ERR_CNT17	218h
CORR_ERR_CNT18	228h
CORR_ERR_CNT19	238h
CORR_ERR_CNT20	248h
CORR_ERR_CNT21	258h

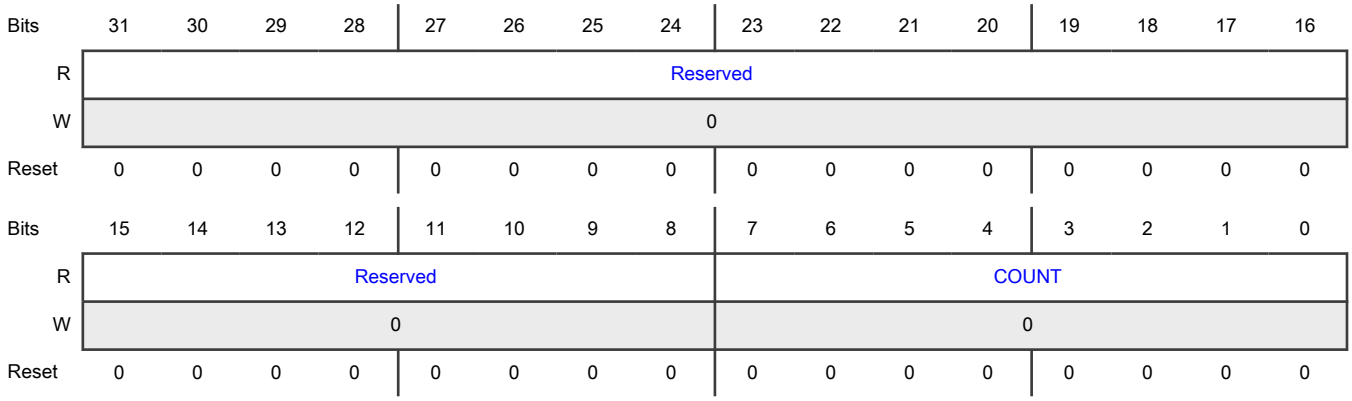
Function

Each 32-bit ERM Memory *n* Correctable Error Count Register records the count value of the number of correctable ECC error events for Memory*n*, where*n* denotes the memory channel.

NOTE

Non-correctable errors are considered a serious fault, so the ERM does not provide any mechanism to count non-correctable errors. Only correctable errors are counted.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 COUNT	Memory n Correctable Error Count For each correctable error event, the ERM increments this field's error count value until the counter reaches its maximum value FFh. COUNT value will stop when it reaches maximum value FFh and will not wrap even if additional errors occur. Read this field to determine the correctable error count value so far. Write all zeros to this field to reset the counter. Writing a non-zero value has no effect.

51.6 ERM_1 register descriptions

You can access the programming model:

- Only in supervisor mode
- Using only 32-bit (word) accesses

Any of the following attempted references to the programming model generates an error termination:

- In user mode
- Using non-32-bit access sizes

Based on the design implementation, the following XFR error behavior is evident at the IPS interface.

- Within the ERM memory map, an XFR error is evident at reserved addresses from location 20h to FFh.
- No XFR error is evident at reserved addresses in memory spaces allocated to each channel. For example: For channel 0, for read/write accesses to reserved address 10Ch, the XFR error is 0.
- For accesses to locations beyond the addresses allocated for the final channel, the XFR error is 1.

NOTE

- See the chip-specific ERM information at the beginning of this chapter for details on Memory channel mapping.
- To access the channel registers, corresponding memory channel clock must be enabled.

51.6.1 ERM_1 memory map

ERM_1 base address: 4000_C000h

Offset	Register	Width (In bits)	Access	Reset value
0h	ERM Configuration Register 0 (CR0)	32	RW	0000_0000h
4h	ERM Configuration Register 1 (CR1)	32	RW	0000_0000h
8h	ERM Configuration Register 2 (CR2)	32	RW	0000_0000h
10h	ERM Status Register 0 (SR0)	32	RW	0000_0000h
14h	ERM Status Register 1 (SR1)	32	RW	0000_0000h
18h	ERM Status Register 2 (SR2)	32	RW	0000_0000h
100h	ERM Memory 0 Error Address Register (EAR0)	32	R	0000_0000h
104h	ERM Memory 0 Syndrome Register (SYN0)	32	R	0000_0000h
108h	ERM Memory 0 Correctable Error Count Register (CORR_ERR_CNT0)	32	RW	0000_0000h
128h	ERM Memory 2 Correctable Error Count Register (CORR_ERR_CNT2)	32	RW	0000_0000h
138h	ERM Memory 3 Correctable Error Count Register (CORR_ERR_CNT3)	32	RW	0000_0000h
148h	ERM Memory 4 Correctable Error Count Register (CORR_ERR_CNT4)	32	RW	0000_0000h
158h	ERM Memory 5 Correctable Error Count Register (CORR_ERR_CNT5)	32	RW	0000_0000h
168h	ERM Memory 6 Correctable Error Count Register (CORR_ERR_CNT6)	32	RW	0000_0000h
178h	ERM Memory 7 Correctable Error Count Register (CORR_ERR_CNT7)	32	RW	0000_0000h
188h	ERM Memory 8 Correctable Error Count Register (CORR_ERR_CNT8)	32	RW	0000_0000h
198h	ERM Memory 9 Correctable Error Count Register (CORR_ERR_CNT9)	32	RW	0000_0000h
1A0h	ERM Memory 10 Error Address Register (EAR10)	32	R	0000_0000h
1A4h	ERM Memory 10 Syndrome Register (SYN10)	32	R	0000_0000h
1A8h	ERM Memory 10 Correctable Error Count Register (CORR_ERR_CNT10)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1B0h	ERM Memory 11 Error Address Register (EAR11)	32	R	0000_0000h
1B4h	ERM Memory 11 Syndrome Register (SYN11)	32	R	0000_0000h
1B8h	ERM Memory 11 Correctable Error Count Register (CORR_ERR_CNT11)	32	RW	0000_0000h
1C0h	ERM Memory 12 Error Address Register (EAR12)	32	R	0000_0000h
1C4h	ERM Memory 12 Syndrome Register (SYN12)	32	R	0000_0000h
1C8h	ERM Memory 12 Correctable Error Count Register (CORR_ERR_CNT12)	32	RW	0000_0000h
1D0h	ERM Memory 13 Error Address Register (EAR13)	32	R	0000_0000h
1D4h	ERM Memory 13 Syndrome Register (SYN13)	32	R	0000_0000h
1D8h	ERM Memory 13 Correctable Error Count Register (CORR_ERR_CNT13)	32	RW	0000_0000h
1E0h	ERM Memory 14 Error Address Register (EAR14)	32	R	0000_0000h
1E4h	ERM Memory 14 Syndrome Register (SYN14)	32	R	0000_0000h
1E8h	ERM Memory 14 Correctable Error Count Register (CORR_ERR_CNT14)	32	RW	0000_0000h
1F0h	ERM Memory 15 Error Address Register (EAR15)	32	R	0000_0000h
1F4h	ERM Memory 15 Syndrome Register (SYN15)	32	R	0000_0000h
1F8h	ERM Memory 15 Correctable Error Count Register (CORR_ERR_CNT15)	32	RW	0000_0000h
220h	ERM Memory 18 Error Address Register (EAR18)	32	R	0000_0000h
224h	ERM Memory 18 Syndrome Register (SYN18)	32	R	0000_0000h
228h	ERM Memory 18 Correctable Error Count Register (CORR_ERR_CNT18)	32	RW	0000_0000h
230h	ERM Memory 19 Error Address Register (EAR19)	32	R	0000_0000h
234h	ERM Memory 19 Syndrome Register (SYN19)	32	R	0000_0000h
238h	ERM Memory 19 Correctable Error Count Register (CORR_ERR_CNT19)	32	RW	0000_0000h

51.6.2 ERM Configuration Register 0 (CR0)

Offset

Register	Offset
CR0	0h

Function

This 32-bit control register configures the interrupt notification capability for available channels.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ESCIE0	ENCIE0	0		Reserved		0		ESCIE2	ENCIE2	0		ESCIE3	ENCIE3	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ESCIE4	ENCIE4	0		ESCIE5	ENCIE5	0		ESCIE6	ENCIE6	0		ESCIE7	ENCIE7	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 ESCIE0	ESCIE0 Enable Memory 0 Single Correction Interrupt Notification 0b - Interrupt notification of Memory 0 single-bit correction events is disabled. 1b - Interrupt notification of Memory 0 single-bit correction events is enabled.
30 ENCIE0	ENCIE0 Enable Memory 0 Non-Correctable Interrupt Notification 0b - Interrupt notification of Memory 0 non-correctable error events is disabled. 1b - Interrupt notification of Memory 0 non-correctable error events is enabled.
29-28 —	Reserved
27-26 —	Reserved
25-24 —	Reserved
23 ESCIE2	ESCIE2 Enable Memory 2 Single Correction Interrupt Notification 0b - Interrupt notification of Memory 2 single-bit correction events is disabled. 1b - Interrupt notification of Memory 2 single-bit correction events is enabled.
22	ENCIE2

Table continues on the next page...

Table continued from the previous page...

Field	Function
ENCIE2	Enable Memory 2 Non-Correctable Interrupt Notification 0b - Interrupt notification of Memory 2 non-correctable error events is disabled. 1b - Interrupt notification of Memory 2 non-correctable error events is enabled.
21-20 —	Reserved
19 ESCIE3	ESCIE3 Enable Memory 3 Single Correction Interrupt Notification 0b - Interrupt notification of Memory 3 single-bit correction events is disabled. 1b - Interrupt notification of Memory 3 single-bit correction events is enabled.
18 ENCIE3	ENCIE3 Enable Memory 3 Non-Correctable Interrupt Notification 0b - Interrupt notification of Memory 3 non-correctable error events is disabled. 1b - Interrupt notification of Memory 3 non-correctable error events is enabled.
17-16 —	Reserved
15 ESCIE4	ESCIE4 Enable Memory 4 Single Correction Interrupt Notification 0b - Interrupt notification of Memory 4 single-bit correction events is disabled. 1b - Interrupt notification of Memory 4 single-bit correction events is enabled.
14 ENCIE4	ENCIE4 Enable Memory 4 Non-Correctable Interrupt Notification 0b - Interrupt notification of Memory 4 non-correctable error events is disabled. 1b - Interrupt notification of Memory 4 non-correctable error events is enabled.
13-12 —	Reserved
11 ESCIE5	ESCIE5 Enable Memory 5 Single Correction Interrupt Notification 0b - Interrupt notification of Memory 5 single-bit correction events is disabled. 1b - Interrupt notification of Memory 5 single-bit correction events is enabled.
10	ENCIE5 Enable Memory 5 Non-Correctable Interrupt Notification

Table continues on the next page...

Table continued from the previous page...

Field	Function
ENCIE5	0b - Interrupt notification of Memory 5 non-correctable error events is disabled. 1b - Interrupt notification of Memory 5 non-correctable error events is enabled.
9-8 —	Reserved
7 ESCIE6	ESCIE6 Enable Memory 6 Single Correction Interrupt Notification 0b - Interrupt notification of Memory 6 single-bit correction events is disabled. 1b - Interrupt notification of Memory 6 single-bit correction events is enabled.
6 ENCIE6	ENCIE6 Enable Memory 6 Non-Correctable Interrupt Notification 0b - Interrupt notification of Memory 6 non-correctable error events is disabled. 1b - Interrupt notification of Memory 6 non-correctable error events is enabled.
5-4 —	Reserved
3 ESCIE7	ESCIE7 Enable Memory 7 Single Correction Interrupt Notification 0b - Interrupt notification of Memory 7 single-bit correction events is disabled. 1b - Interrupt notification of Memory 7 single-bit correction events is enabled.
2 ENCIE7	ENCIE7 Enable Memory 7 Non-Correctable Interrupt Notification 0b - Interrupt notification of Memory 7 non-correctable error events is disabled. 1b - Interrupt notification of Memory 7 non-correctable error events is enabled.
1-0 —	Reserved

51.6.3 ERM Configuration Register 1 (CR1)

Offset

Register	Offset
CR1	4h

Function

This 32-bit control register configures the interrupt notification capability for available channels.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ESCIE8	ENCIE8	0		ESCIE9	ENCIE9	0		ESCIE10	ENCIE10	0		ESCIE11	ENCIE11	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ESCIE12	ENCIE12	0		ESCIE13	ENCIE13	0		ESCIE14	ENCIE14	0		ESCIE15	ENCIE15	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 ESCIE8	ESCIE8 Enable Memory 8 Single Correction Interrupt Notification 0b - Interrupt notification of Memory 8 single-bit correction events is disabled. 1b - Interrupt notification of Memory 8 single-bit correction events is enabled.
30 ENCIE8	ENCIE8 Enable Memory 8 Non-Correctable Interrupt Notification 0b - Interrupt notification of Memory 8 non-correctable error events is disabled. 1b - Interrupt notification of Memory 8 non-correctable error events is enabled.
29-28 —	Reserved
27 ESCIE9	ESCIE9 Enable Memory 9 Single Correction Interrupt Notification 0b - Interrupt notification of Memory 9 single-bit correction events is disabled. 1b - Interrupt notification of Memory 9 single-bit correction events is enabled.
26 ENCIE9	ENCIE9 Enable Memory 9 Non-Correctable Interrupt Notification 0b - Interrupt notification of Memory 9 non-correctable error events is disabled. 1b - Interrupt notification of Memory 9 non-correctable error events is enabled.
25-24	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
23 ESCIE10	ESCIE10 Enable Memory 10 Single Correction Interrupt Notification 0b - Interrupt notification of Memory 10 single-bit correction events is disabled. 1b - Interrupt notification of Memory 10 single-bit correction events is enabled.
22 ENCIE10	ENCIE10 Enable Memory 10 Non-Correctable Interrupt Notification 0b - Interrupt notification of Memory 10 non-correctable error events is disabled. 1b - Interrupt notification of Memory 10 non-correctable error events is enabled.
21-20 —	Reserved
19 ESCIE11	ESCIE11 Enable Memory 11 Single Correction Interrupt Notification 0b - Interrupt notification of Memory 11 single-bit correction events is disabled. 1b - Interrupt notification of Memory 11 single-bit correction events is enabled.
18 ENCIE11	ENCIE11 Enable Memory 11 Non-Correctable Interrupt Notification 0b - Interrupt notification of Memory 11 non-correctable error events is disabled. 1b - Interrupt notification of Memory 11 non-correctable error events is enabled.
17-16 —	Reserved
15 ESCIE12	ESCIE12 Enable Memory 12 Single Correction Interrupt Notification 0b - Interrupt notification of Memory 12 single-bit correction events is disabled. 1b - Interrupt notification of Memory 12 single-bit correction events is enabled.
14 ENCIE12	ENCIE12 Enable Memory 12 Non-Correctable Interrupt Notification 0b - Interrupt notification of Memory 12 non-correctable error events is disabled. 1b - Interrupt notification of Memory 12 non-correctable error events is enabled.
13-12	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
11 ESCIE13	ESCIE13 Enable Memory 13 Single Correction Interrupt Notification 0b - Interrupt notification of Memory 13 single-bit correction events is disabled. 1b - Interrupt notification of Memory 13 single-bit correction events is enabled.
10 ENCIE13	ENCIE13 Enable Memory 13 Non-Correctable Interrupt Notification 0b - Interrupt notification of Memory 13 non-correctable error events is disabled. 1b - Interrupt notification of Memory 13 non-correctable error events is enabled.
9-8 —	Reserved
7 ESCIE14	ESCIE14 Enable Memory 14 Single Correction Interrupt Notification 0b - Interrupt notification of Memory 14 single-bit correction events is disabled. 1b - Interrupt notification of Memory 14 single-bit correction events is enabled.
6 ENCIE14	ENCIE14 Enable Memory 14 Non-Correctable Interrupt Notification 0b - Interrupt notification of Memory 14 non-correctable error events is disabled. 1b - Interrupt notification of Memory 14 non-correctable error events is enabled.
5-4 —	Reserved
3 ESCIE15	ESCIE15 Enable Memory 15 Single Correction Interrupt Notification 0b - Interrupt notification of Memory 15 single-bit correction events is disabled. 1b - Interrupt notification of Memory 15 single-bit correction events is enabled.
2 ENCIE15	ENCIE15 Enable Memory 15 Non-Correctable Interrupt Notification 0b - Interrupt notification of Memory 15 non-correctable error events is disabled. 1b - Interrupt notification of Memory 15 non-correctable error events is enabled.
1-0	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	

51.6.4 ERM Configuration Register 2 (CR2)

Offset

Register	Offset
CR2	8h

Function

This 32-bit control register configures the interrupt notification capability for available channels.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved		0		Reserved		0		ESCIE 18	ENCIE 18	0		ESCIE 19	ENCIE 19	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved		0		Reserved		0		Reserved		0		Reserved		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29-28 —	Reserved
27-26 —	Reserved
25-24 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
23 ESCIE18	ESCIE18 Enable Memory 18 Single Correction Interrupt Notification 0b - Interrupt notification of Memory 18 single-bit correction events is disabled. 1b - Interrupt notification of Memory 18 single-bit correction events is enabled.
22 ENCIE18	ENCIE18 Enable Memory 18 Non-Correctable Interrupt Notification 0b - Interrupt notification of Memory 18 non-correctable error events is disabled. 1b - Interrupt notification of Memory 18 non-correctable error events is enabled.
21-20 —	Reserved
19 ESCIE19	ESCIE19 Enable Memory 19 Single Correction Interrupt Notification 0b - Interrupt notification of Memory 19 single-bit correction events is disabled. 1b - Interrupt notification of Memory 19 single-bit correction events is enabled.
18 ENCIE19	ENCIE19 Enable Memory 19 Non-Correctable Interrupt Notification 0b - Interrupt notification of Memory 19 non-correctable error events is disabled. 1b - Interrupt notification of Memory 19 non-correctable error events is enabled.
17-16 —	Reserved
15-14 —	Reserved
13-12 —	Reserved
11-10 —	Reserved
9-8 —	Reserved
7-6 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
5-4 —	Reserved
3-2 —	Reserved
1-0 —	Reserved

51.6.5 ERM Status Register 0 (SR0)

Offset

Register	Offset
SR0	10h

Function

This 32-bit status register reports error events for available channels.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC0	NCE0	0		0				SBC2	NCE2	0		SBC3	NCE3	0	
W	W1C	W1C							W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SBC4	NCE4	0		SBC5	NCE5	0		SBC6	NCE6	0		SBC7	NCE7	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SBC0	SBC0 Memory 0 Single-Bit Correction Event Write 1 to clear this field.This write also clears the corresponding interrupt notification, if CR0[ESCIE0] is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No single-bit correction event on Memory 0 detected. 1b - Single-bit correction event on Memory 0 detected.
30 NCE0	NCE0 Memory 0 Non-Correctable Error Event Write 1 to clear this field.This write also clears the corresponding interrupt notification, if CR0[ENCIE0] is enabled. 0b - No non-correctable error event on Memory 0 detected. 1b - Non-correctable error event on Memory 0 detected.
29-28 —	Reserved
27-24 —	Reserved
23 SBC2	SBC2 Memory 2 Single-Bit Correction Event Write 1 to clear this field.This write also clears the corresponding interrupt notification, if CR0[ESCIE2] is enabled. 0b - No single-bit correction event on Memory 2 detected. 1b - Single-bit correction event on Memory 2 detected.
22 NCE2	NCE2 Memory 2 Non-Correctable Error Event Write 1 to clear this field.This write also clears the corresponding interrupt notification, if CR0[ENCIE2] is enabled. 0b - No non-correctable error event on Memory 2 detected. 1b - Non-correctable error event on Memory 2 detected.
21-20 —	Reserved
19 SBC3	SBC3 Memory 3 Single-Bit Correction Event Write 1 to clear this field.This write also clears the corresponding interrupt notification, if CR0[ESCIE3] is enabled. 0b - No single-bit correction event on Memory 3 detected. 1b - Single-bit correction event on Memory 3 detected.

Table continues on the next page...

Table continued from the previous page...

Field	Function
18 NCE3	NCE3 Memory 3 Non-Correctable Error Event Write 1 to clear this field.This write also clears the corresponding interrupt notification, if CR0[ENCIE3] is enabled. 0b - No non-correctable error event on Memory 3 detected. 1b - Non-correctable error event on Memory 3 detected.
17-16 —	Reserved
15 SBC4	SBC4 Memory 4 Single-Bit Correction Event Write 1 to clear this field.This write also clears the corresponding interrupt notification, if CR0[ESCIE4] is enabled. 0b - No single-bit correction event on Memory 4 detected. 1b - Single-bit correction event on Memory 4 detected.
14 NCE4	NCE4 Memory 4 Non-Correctable Error Event Write 1 to clear this field.This write also clears the corresponding interrupt notification, if CR0[ENCIE4] is enabled. 0b - No non-correctable error event on Memory 4 detected. 1b - Non-correctable error event on Memory 4 detected.
13-12 —	Reserved
11 SBC5	SBC5 Memory 5 Single-Bit Correction Event Write 1 to clear this field.This write also clears the corresponding interrupt notification, if CR0[ESCIE5] is enabled. 0b - No single-bit correction event on Memory 5 detected. 1b - Single-bit correction event on Memory 5 detected.
10 NCE5	NCE5 Memory 5 Non-Correctable Error Event Write 1 to clear this field.This write also clears the corresponding interrupt notification, if CR0[ENCIE5] is enabled. 0b - No non-correctable error event on Memory 5 detected.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Non-correctable error event on Memory 5 detected.
9-8 —	Reserved
7 SBC6	SBC6 Memory 6 Single-Bit Correction Event Write 1 to clear this field.This write also clears the corresponding interrupt notification, if CR0[ESCIE6] is enabled. 0b - No single-bit correction event on Memory 6 detected. 1b - Single-bit correction event on Memory 6 detected.
6 NCE6	NCE6 Memory 6 Non-Correctable Error Event Write 1 to clear this field.This write also clears the corresponding interrupt notification, if CR0[ENCIE6] is enabled. 0b - No non-correctable error event on Memory 6 detected. 1b - Non-correctable error event on Memory 6 detected.
5-4 —	Reserved
3 SBC7	SBC7 Memory 7 Single-Bit Correction Event Write 1 to clear this field.This write also clears the corresponding interrupt notification, if CR0[ESCIE7] is enabled. 0b - No single-bit correction event on Memory 7 detected. 1b - Single-bit correction event on Memory 7 detected.
2 NCE7	NCE7 Memory 7 Non-Correctable Error Event Write 1 to clear this field.This write also clears the corresponding interrupt notification, if CR0[ENCIE7] is enabled. 0b - No non-correctable error event on Memory 7 detected. 1b - Non-correctable error event on Memory 7 detected.
1-0 —	Reserved

51.6.6 ERM Status Register 1 (SR1)

Offset

Register	Offset
SR1	14h

Function

This 32-bit status register reports error events for available channels.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC8	NCE8	0		SBC9	NCE9	0		SBC10	NCE10	0		SBC11	NCE11	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SBC12	NCE12	0		SBC13	NCE13	0		SBC14	NCE14	0		SBC15	NCE15	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SBC8	SBC8 Memory 8 Single-Bit Correction Event Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE8] is enabled. 0b - No single-bit correction event on Memory 8 detected. 1b - Single-bit correction event on Memory 8 detected.
30 NCE8	NCE8 Memory 8 Non-Correctable Error Event Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE8] is enabled. 0b - No non-correctable error event on Memory 8 detected. 1b - Non-correctable error event on Memory 8 detected.
29-28 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
27 SBC9	SBC9 Memory 9 Single-Bit Correction Event Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE9] is enabled. 0b - No single-bit correction event on Memory 9 detected. 1b - Single-bit correction event on Memory 9 detected.
26 NCE9	NCE9 Memory 9 Non-Correctable Error Event Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE9] is enabled. 0b - No non-correctable error event on Memory 9 detected. 1b - Non-correctable error event on Memory 9 detected.
25-24 —	Reserved
23 SBC10	SBC10 Memory 10 Single-Bit Correction Event Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE10] is enabled. 0b - No single-bit correction event on Memory 10 detected. 1b - Single-bit correction event on Memory 10 detected.
22 NCE10	NCE10 Memory 10 Non-Correctable Error Event Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE10] is enabled. 0b - No non-correctable error event on Memory 10 detected. 1b - Non-correctable error event on Memory 10 detected.
21-20 —	Reserved
19 SBC11	SBC11 Memory 11 Single-Bit Correction Event Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE11] is enabled. 0b - No single-bit correction event on Memory 11 detected.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Single-bit correction event on Memory 11 detected.
18 NCE11	NCE11 Memory 11 Non-Correctable Error Event Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE11] is enabled. 0b - No non-correctable error event on Memory 11 detected. 1b - Non-correctable error event on Memory 11 detected.
17-16 —	Reserved
15 SBC12	SBC12 Memory 12 Single-Bit Correction Event Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE12] is enabled. 0b - No single-bit correction event on Memory 12 detected. 1b - Single-bit correction event on Memory 12 detected.
14 NCE12	NCE12 Memory 12 Non-Correctable Error Event Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE12] is enabled. 0b - No non-correctable error event on Memory 12 detected. 1b - Non-correctable error event on Memory 12 detected.
13-12 —	Reserved
11 SBC13	SBC13 Memory 13 Single-Bit Correction Event Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE13] is enabled. 0b - No single-bit correction event on Memory 13 detected. 1b - Single-bit correction event on Memory 13 detected.
10 NCE13	NCE13 Memory 13 Non-Correctable Error Event Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE13] is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No non-correctable error event on Memory 13 detected. 1b - Non-correctable error event on Memory 13 detected.
9-8 —	Reserved
7 SBC14	SBC14 Memory 14 Single-Bit Correction Event Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE14] is enabled. 0b - No single-bit correction event on Memory 14 detected. 1b - Single-bit correction event on Memory 14 detected.
6 NCE14	NCE14 Memory 14 Non-Correctable Error Event Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE14] is enabled. 0b - No non-correctable error event on Memory 14 detected. 1b - Non-correctable error event on Memory 14 detected.
5-4 —	Reserved
3 SBC15	SBC15 Memory 15 Single-Bit Correction Event Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE15] is enabled. 0b - No single-bit correction event on Memory 15 detected. 1b - Single-bit correction event on Memory 15 detected.
2 NCE15	NCE15 Memory 15 Non-Correctable Error Event Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE15] is enabled. 0b - No non-correctable error event on Memory 15 detected. 1b - Non-correctable error event on Memory 15 detected.
1-0 —	Reserved

51.6.7 ERM Status Register 2 (SR2)

Offset

Register	Offset
SR2	18h

Function

This 32-bit status register reports error events for available channels.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				0				SBC18	NCE18	0		SBC19	NCE19	0	
W									W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				0				0				0			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved
27-24 —	Reserved
23 SBC18	SBC18 Memory 18 Single-Bit Correction Event Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ESCIE18] is enabled. 0b - No single-bit correction event on Memory 18 detected. 1b - Single-bit correction event on Memory 18 detected.
22 NCE18	NCE18 Memory 18 Non-Correctable Error Event Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ENCIE18] is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No non-correctable error event on Memory 18 detected. 1b - Non-correctable error event on Memory 18 detected.
21-20 —	Reserved
19 SBC19	SBC19 Memory 19 Single-Bit Correction Event Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ESCIE19] is enabled. 0b - No single-bit correction event on Memory 19 detected. 1b - Single-bit correction event on Memory 19 detected.
18 NCE19	NCE19 Memory 19 Non-Correctable Error Event Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ENCIE19] is enabled. 0b - No non-correctable error event on Memory 19 detected. 1b - Non-correctable error event on Memory 19 detected.
17-16 —	Reserved
15-12 —	Reserved
11-8 —	Reserved
7-4 —	Reserved
3-0 —	Reserved

51.6.8 ERM Memory a Error Address Register (EAR0 - EAR19)

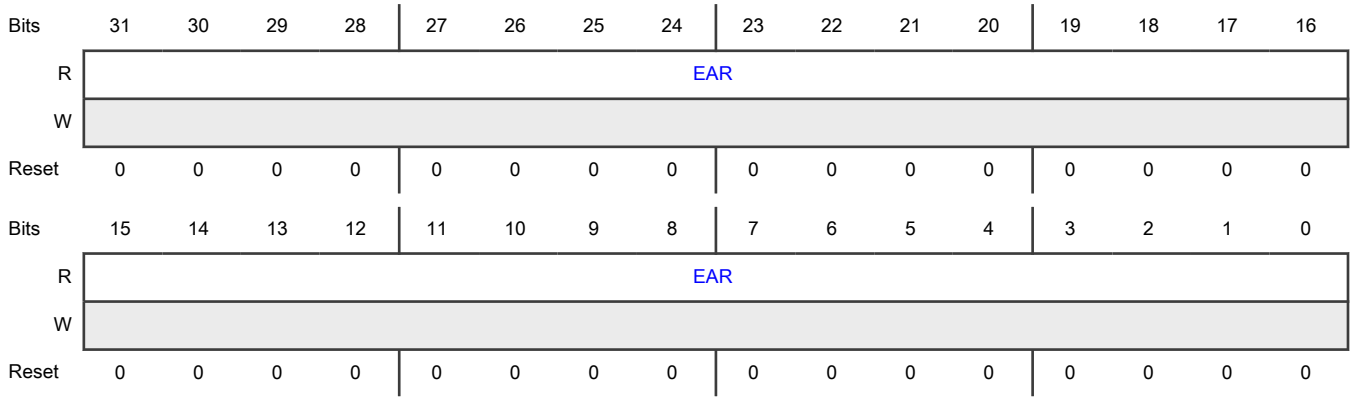
Offset

Register	Offset
EAR0	100h
EAR10	1A0h
EAR11	1B0h
EAR12	1C0h
EAR13	1D0h
EAR14	1E0h
EAR15	1F0h
EAR18	220h
EAR19	230h

Function

Each ERM Memory n Error Address Register is a 32-bit register for capturing the address of the last ECC event in Memory*n*, where*n* denotes the memory channel. Any attempted write to EAR*n* is ignored. See the chip-specific ERM information for the ERM Memory a Error Address Register.

Diagram



Fields

Field	Function
31-0 EAR	EAR Memory*n* Error Address — This field contains the faulting system address of the last recorded ECC event on Memory*n*.

51.6.9 ERM Memory a Syndrome Register (SYN0 - SYN19)

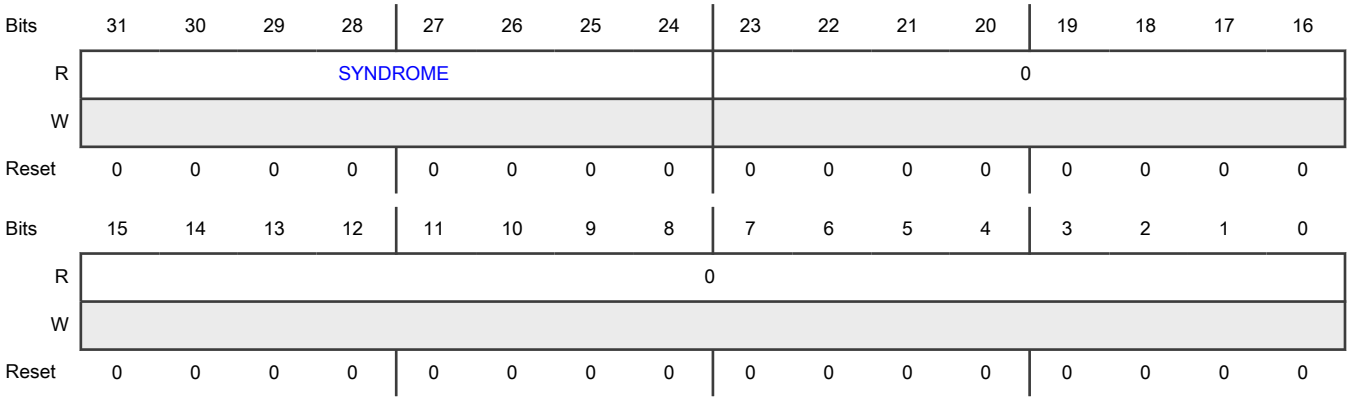
Offset

Register	Offset
SYN0	104h
SYN10	1A4h
SYN11	1B4h
SYN12	1C4h
SYN13	1D4h
SYN14	1E4h
SYN15	1F4h
SYN18	224h
SYN19	234h

Function

The ERM Memory *n* Syndrome Register is a 32-bit register for capturing the calculated syndrome of the last ECC event on Memory*n*, where*n* denotes the memory channel. Any attempted write to SYN*n* is ignored. The syndrome value identifies the pertinent bit position on a correctable, single-bit data inversion or a non-correctable, single-bit address inversion. The syndrome value does not provide any additional diagnostic information on non-correctable, multi-bit inversions.

Diagram



Fields

Field	Function
31-24 SYNDROME	SYNDROME Memory*n* Syndrome — This field contains the ECC syndrome associated with the last recorded ECC event on Memory*n*.
23-0 —	Reserved

51.6.10 ERM Memory a Correctable Error Count Register (CORR_ERR_CNT0 - CORR_ERR_CNT19)

Offset

Register	Offset
CORR_ERR_CNT0	108h
CORR_ERR_CNT2	128h
CORR_ERR_CNT3	138h
CORR_ERR_CNT4	148h
CORR_ERR_CNT5	158h
CORR_ERR_CNT6	168h
CORR_ERR_CNT7	178h
CORR_ERR_CNT8	188h
CORR_ERR_CNT9	198h
CORR_ERR_CNT10	1A8h
CORR_ERR_CNT11	1B8h
CORR_ERR_CNT12	1C8h
CORR_ERR_CNT13	1D8h
CORR_ERR_CNT14	1E8h
CORR_ERR_CNT15	1F8h
CORR_ERR_CNT18	228h
CORR_ERR_CNT19	238h

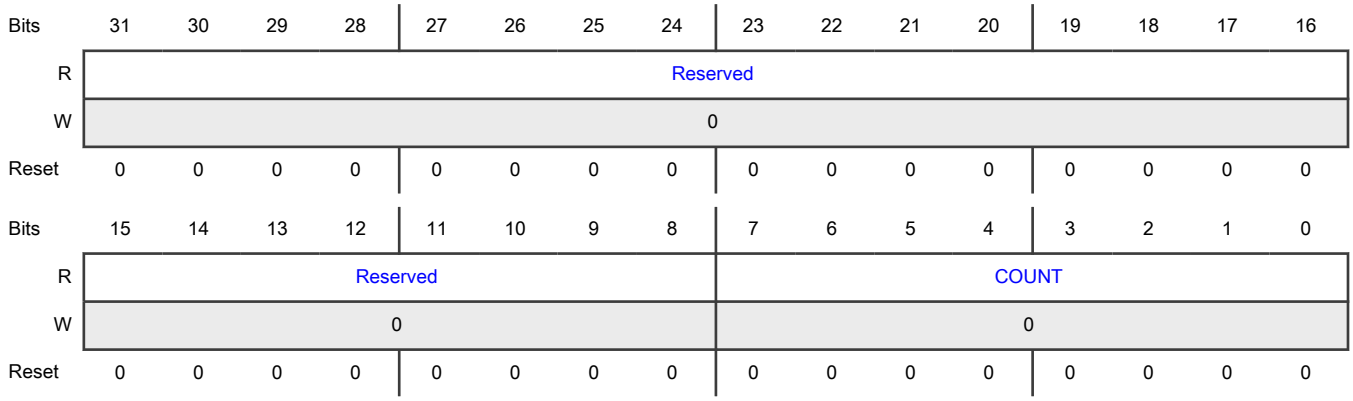
Function

Each 32-bit ERM Memory *n* Correctable Error Count Register records the count value of the number of correctable ECC error events for Memory*n*, where*n* denotes the memory channel.

NOTE

Non-correctable errors are considered a serious fault, so the ERM does not provide any mechanism to count non-correctable errors. Only correctable errors are counted.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 COUNT	Memory n Correctable Error Count For each correctable error event, the ERM increments this field's error count value until the counter reaches its maximum value FFh. COUNT value will stop when it reaches maximum value FFh and will not wrap even if additional errors occur. Read this field to determine the correctable error count value so far. Write all zeros to this field to reset the counter. Writing a non-zero value has no effect.

51.7 Glossary

ECC Error correction code

Chapter 52
Fault Collection and Control Unit (FCCU)

52.1 Chip-specific FCCU information

52.1.1 FCCU NCF slots

Table 295. FCCU NCF slots

Slot number	Source module (error type)
NCF[0]	• Cortex-M7 LS and core lockup
NCF[1]	Interconnect: • All EDC bus gaskets • XBIC monitors and platform gaskets
NCF[2]	ECC errors: • PRAMC • TCMs • Caches • eDMA • EDC after ECC • QuadSPI[1] • AES_ACCEL (include DMA TCD) errors [2]
NCF[3]	All flash memory errors: • FMU • PFLASH • DCM flash memory • Flash Address Remapping[3]
NCF[4]	Voltage-related errors: • PMC 1.1 V and 2.5 V GNG • PMC 1.1 V and 2.5 V GNG2[1] • Pad overvoltage
NCF[5]	Debug and test monitoring, STCU faults [4]
NCF[6]	INTM
NCF[7]	Software notification

1. For S32K358/S32K356/S32K348/S32K338/S32K328 only.
2. For S32K388/S32K389 only.
3. For S32K389 only.
4. The fault would be raised if DBGPWRUPREQ toggles while any core is in halted state. If this operation is expected, the application can ignore/disable such an NCF event.

52.1.2 Chip-boundary FCCU signals

Table 296. Chip-boundary FCCU signals

FCCU signal	Chip signal
EOUT0	FCCU_ERR0
EOUT1	FCCU_ERR1

52.1.3 FCCU clocking

Table 297. FCCU clocking

FCCU clock signal	Chip clock signal
CLKSAFE	FIRC_CLK
CLKPRIM	AIPS_PLAT_CLK

52.1.4 FOSU timer interval

The FOSU_COUNT value determines the FOSU timer interval. On this chip, FOSU_COUNT is 69780h.

52.1.5 Supported internal chip reactions

The short functional reset discussed in this chapter is equivalent to the chip functional reset. See the interrupt map file attached to this document for interrupts from FCCU to NVIC.

NOTE

After STCU2 completes the self-testing procedure, the chip reboots and FCCU resets.

52.1.6 Recommended reaction programming for faults

You can upgrade or downgrade the reaction of the faults according to the recommended reaction discussed in the fault map file attached to this document.

In case you upgrade a reaction, no issues are expected in the behavior. If you downgrade the reaction, the functionality is not guaranteed.

This is the recommendation for faults caused by lockstep errors:

1. The recommended reaction for the lockstep error fault is a functional reset.
2. Program the core to perform the following steps after rebooting:
    a. Initialize TCM (because TCMs can become corrupted)
    b. Invalidate the cache (because caches can become corrupted)
3. Write 1 to the corresponding bit in FCCU.NCF_S0 register to clear the non-critical fault status.
4. If the FCCU fault gets cleared, you have nothing else to do for fault handling (this means, the FCCU lockstep error is recovered).
5. If the error persists, it indicates that the internal registers of the two cores have different values. To bring them to the same value, you could perform any of these steps:
    a. Initialize destructive reset through software by using MC_ME.MODE_CONF[DEST_RST] to recover from lockstep. This initializes the FCCU with no pending faults.
    b. Reconfigure the Cortex-M7 debug configurations (which would have been lost in previous step due to destructive reset in the system).

52.1.7 FCCU NCF handling architecture

This chip supports eight NCFs. Therefore, multiple faults of similar nature are ORed together and then connected as a single NCF to FCCU. To control individual fault, there are enable/disable controls provided within DCM in registers DCMRWD*n*.

Similarly, the FCCU.NCF_S0 captures the status of fault within FCCU. This fault might have resulted due to any fault mapped on the corresponding FCCU NCF channel. The DCM status registers DCMROD*n* capture the status of individual faults which are mapped onto FCCU NCFs. Figure 207 shows this arrangement.

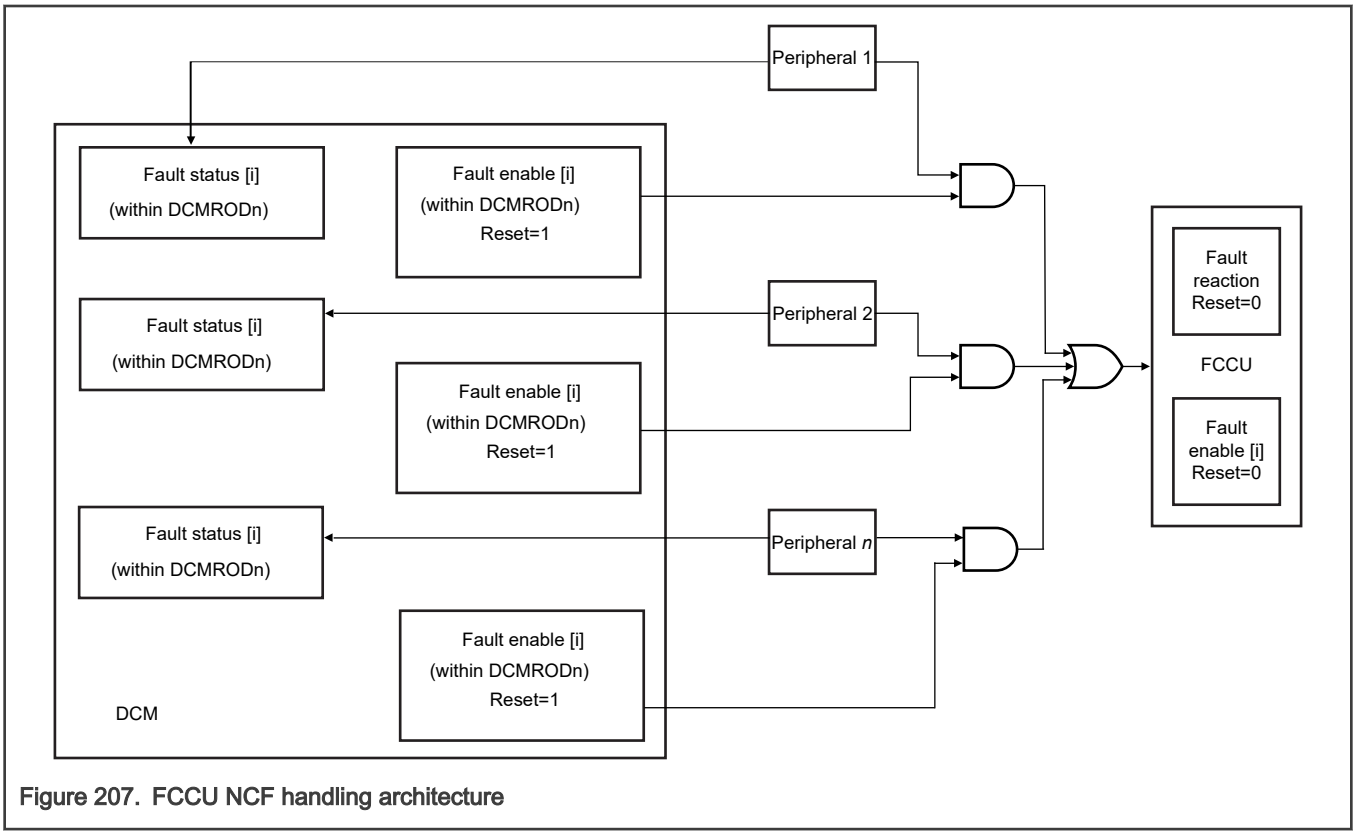


Figure 207. FCCU NCF handling architecture

52.2 Overview

The FCCU provides a hardware interface to collect faults and to place the device into a safe state when a failure is detected in the device. No CPU intervention is requested for collection and control operations. FCCU offers a systematic approach to fault collection and control.

52.2.1 Block diagram

The following figure represents a top-level diagram of the FCCU module.

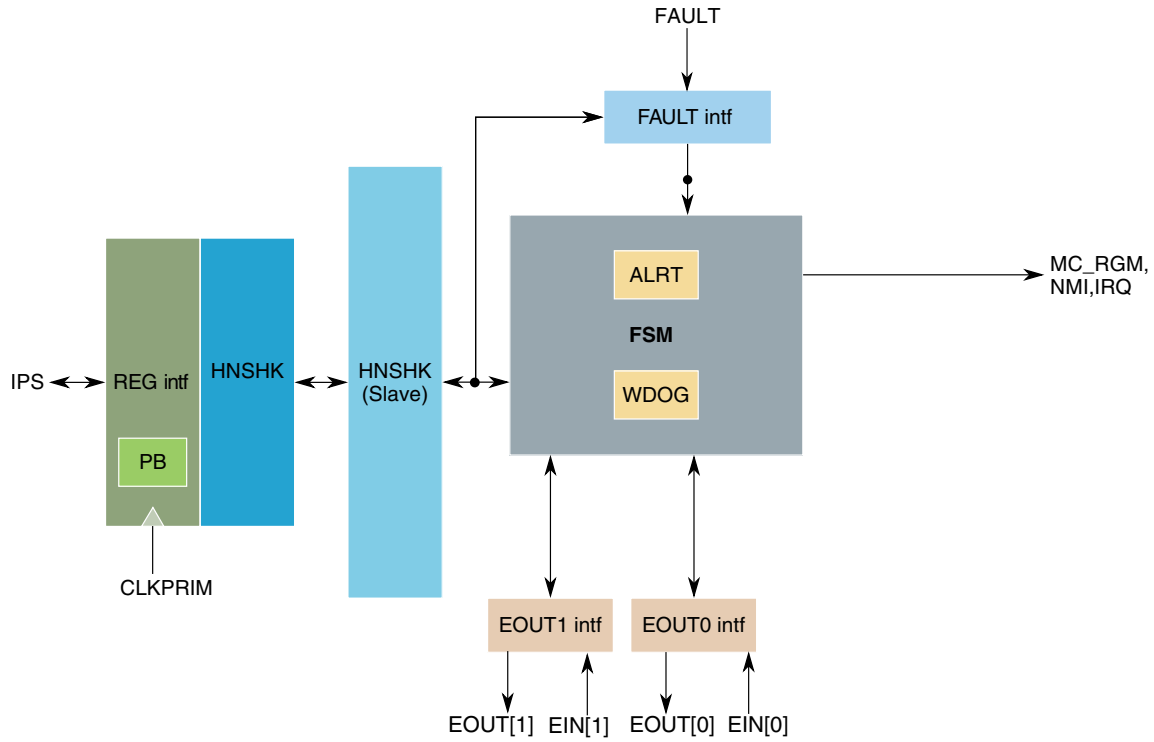


Figure 208. FCCU block diagram

This table describes the FCCU submodules.

Table 298. FCCU submodules

Submodule	Description
REG intf	Includes the register file, the IPS bus interface, the IRQ interface and the parity block (PB) for the configuration registers
HNSHK blocks (master and slave blocks)	Includes the FSM ability to support the handshake between the REG interface and the FSM unit because of the usage of two asynchronous clocks [CLKPRIM(module clock) and CLKSAFE(RC oscillator clock)]
FSM unit	Implements the main functions of FCCU. The FSM also includes the: • Watchdog timer (WDG) • Alarm timer (ALRT)
FAULT intf	Implements the interface for fault conditioning and management
EOUTx units	Implement the output stage to manage the EOUT interfaces

52.2.2 Features

The key features of the FCCU module are these:

- Management of non-critical faults
- HW or SW fault recovery management
- Fault collection from safety relevant modules on the device

- Fault injection (fake faults)
- Collection of test results
- Lockable configuration
  - Changes are only possible after entering the CONFIG state
  - Supports a transient and a permanent lock
  - Configuration changes observed by a watchdog timer
- Configurable fault control
- External reaction (FAULT state): EOUT signaling. Error indication via the pin(s) is controlled by FCCU.
- Internal chip reactions (ALARM state): interrupt request
- Configurable internal chip reactions for each NCF (FAULT state):
  - Short functional reset request pulse
  - NMI
  - No reaction
  - IRQ
- In Bi-Stable operational mode, one of the EOUT signals is high to indicate an OK operational state of FCCU.
- After power on, the EOUT signals have high impedance.[11] They indicate an operational state only after the software configures them.
- In case of a failure event or on software request for error pin indication, the pin(s) are set to faulty state for a minimum time T_{min} (see DELTA_T[DELTA_T]), even if the software tries to release it before (for the case of error pin configured in Bi-Stable mode only).

The self-checking procedure checks the FCCU circuitry at the start up. The FCCU is operational with the default configuration immediately after the completion of the self-checking procedure. Internal (short functional reset request pulse, interrupt request) and external (EOUT signaling) reactions are statically defined or programmable. The default configuration can be modified only in the Configuration (CONFIG) state. FCCU is designed to function when CLKPRIM is faster than the CLKSAFE clocks.

52.3 Functional description

52.3.1 Definitions

In general, the following definitions are applicable for fault management:

- HW recoverable fault: The fault indication is a level-sensitive signal that remains asserted until the fault cause is deasserted. That is, if logical 0 on the fault signal indicates fault, then the status flags are valid as long as the fault line stays at 0. The status is automatically cleared when the fault signal goes to 1. Typically the fault signal is latched external to the FCCU in the module where the fault occurred. The FCCU state transitions are consequently executed on the state changes of the input fault signal. No SW intervention in the FCCU is required to recover the fault condition.
- SW recoverable fault: The fault indication is a signal asserted without a defined time duration. The fault signal is latched in the FCCU. The fault recovery is executed following a SW recovery procedure (status/flag register clearing).

HW recoverable is an option to exclude the handling of error sources by FCCU management SW, in case it is known that the fault is recoverable by itself when the fault condition is corrected.

For details related to reset interface, see the reset interface section in the chip reference manual.

52.3.2 FSM description

The functionality of FCCU is depicted by the FSM state diagram (see Figure 209).

[11] Actual value depends on device-specific setting at pad level.

FCCU has four states that are identified with the following meaning:

- CONFIG: Used only to modify the configuration of FCCU from its default. A subset of the FCCU registers, dedicated to define the FCCU configuration (global configuration, reactions to fault, timeout, non-critical fault masking) can be accessed in write mode only in the CONFIG state.

  The CONFIG state is accessible only in the NORMAL state and if the configuration is not locked. A permanent configuration lock can be disabled by a reset that also resets the FCCU. The transient lock register is unlocked by writing BCh into it. FCCU gets transiently locked again if an invalid key is written into TRANS_LOCK[TRANSKEY] (that is, other than BCh). To lock FCCU for configuration, write FFh to PERMNT_LOCK[PERMNTKEY].

  After the release of reset, the state of the transient lock is locked, and the state of the permanent lock is unlocked.

  The locking feature only restricts the FSM movement into CONFIG state. After the user enters the CONFIG state and then tries to lock the configuration, the locking of configuration is effective only after FCCU moves to the NORMAL state; it will not be effective in the current CONFIG state.

  The CONFIG to NORMAL state transition can be executed by SW or automatically following a timeout condition of the watchdog. In case the timeout information and the SW request for state change to NORMAL appears at the same time, watchdog timeout has the priority and hence the configuration registers (those that are writable only in the CONFIG state) are reset to their default values. The movement to the NORMAL state is made.

  The incoming faults, occurring during the configuration phase (CONFIG state) are latched in order to process them when FCCU is moved to the NORMAL state, according to the new configuration.

  All pending faults that occur during the CONFIG state result in both of the following:

  — Highest-priority state transition

  — Interrupt generation (NMI or alarm IRQ)

  If the state transition occurs, it gives the reset reaction corresponding to the worst case based on all the faults (pending or non-pending faults) that occurred during the CONFIG state.

- NORMAL: This is FCCU's operating state when no faults are occurring. It is also the default state on the reset exit. Following state transitions occur on one of the following events:

  — Unmasked non-critical faults with the timeout disabled: FCCU moves to the FAULT state.

  — Unmasked non-critical faults with the timeout enabled: FCCU moves to the ALARM state.

  — Masked non-critical faults: FCCU stays in the NORMAL state.

- ALARM: FCCU moves into the ALARM state when an unmasked non-critical fault occurs and the timeout is enabled. Transition to the ALARM state goes along with an interrupt alarm, if enabled. By definition, this fault may be recovered within a programmable timeout period, before it generates a transition to the FAULT state. The timeout is reinitialized if FCCU enters the NORMAL state. The timeout restarts following the recovery from the FAULT state.

- FAULT: FCCU moves into the FAULT state when one of the following condition occurs:

  — Timeout related to a non-critical fault when FCCU is in the ALARM state

  — Unmasked non-critical faults with the timeout disabled

The transition from the NORMAL or ALARM to the FAULT state goes along with the generation of:

- Internal chip reaction—NMI interrupt (optional)
- External reaction—EOUT signaling (optional)
- Internal chip reaction—SW option: Soft reaction (Short functional reset request pulse if configured)
- Non Maskable Interrupt (NMI) is routed to all cores.

After moving to the FAULT state, if there is either a previous pending fault or a new fault for which NMI is enabled, NMI generation takes place.

Multiple faults can occur at the same time.

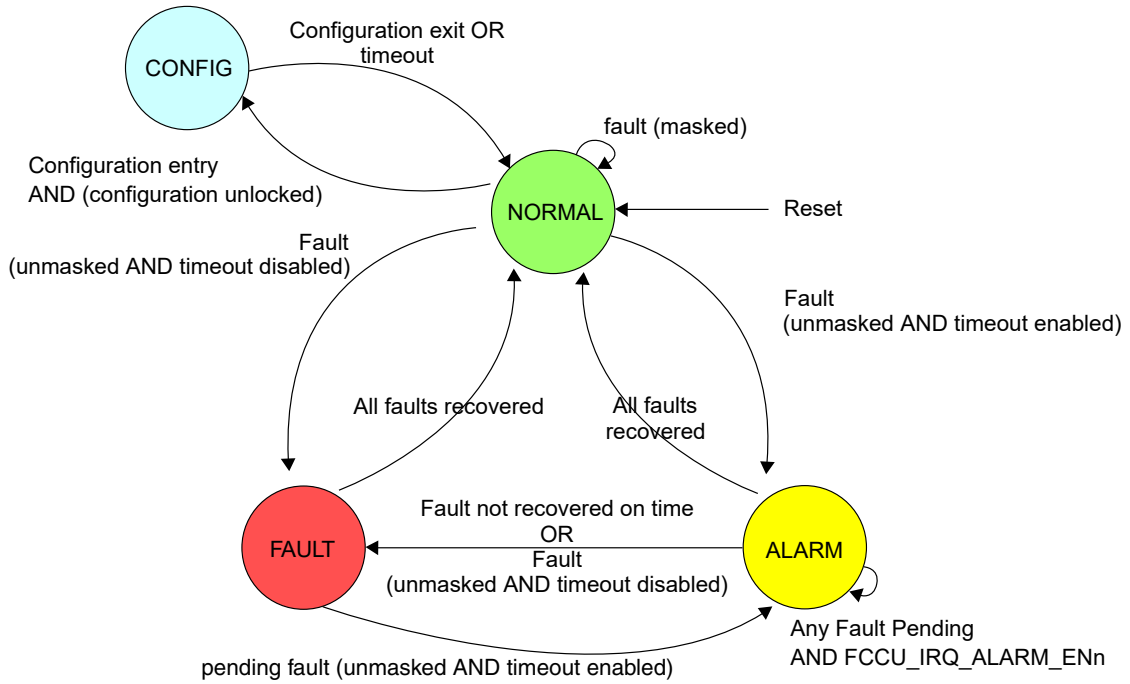


Figure 209. FCCU state diagram

52.3.3 Fault priority scheme and nesting

The FAULT state has a higher priority than the ALARM state in case of concurrent fault events (non-critical) that occur in the NORMAL state.

The ALARM to FAULT state transition occurs if a non-critical fault (unmasked and with timeout disabled) is asserted in the ALARM state.

The ALARM to NORMAL state transition occurs only if all the non-critical faults (including the faults that have been collected after the entry in the ALARM state) have been cleared (SW or HW recovery); otherwise the FCCU remains in the ALARM state.

The FAULT to NORMAL state transition occurs only if all the non-critical faults (including the faults that have been collected after the entry in the FAULT/ALARM state) have been cleared (SW or HW recovery); otherwise the FCCU moves to the ALARM state (if any non-critical fault is still pending and the timeout is not elapsed).

In general, no fault nesting is supported except for the non-critical faults that cause an ALARM to FAULT state transition. In this case, the NCF timer is stopped until the FAULT state is recovered. If FCCU is in the ALARM state and another fault occurs, which has its alarm timeout enabled, then the alarm timer shall not reload and shall not start again.

52.3.4 Fault recovery

The following timing diagrams describe the main use cases of FCCU in terms of fault events and related recovery.

A typical sequence related to non-critical fault management (ALARM state), see Figure 210 and Figure 211, is as follows:

1. Non-critical fault assertion
2. FCCU state transition (automatic): NORMAL to ALARM
   - Alarm interrupt request (if enabled)
   - Timeout running
3. System state: RUN
4. Alarm interrupt management: fault recovery

- FCCU state transition: ALARM to NORMAL

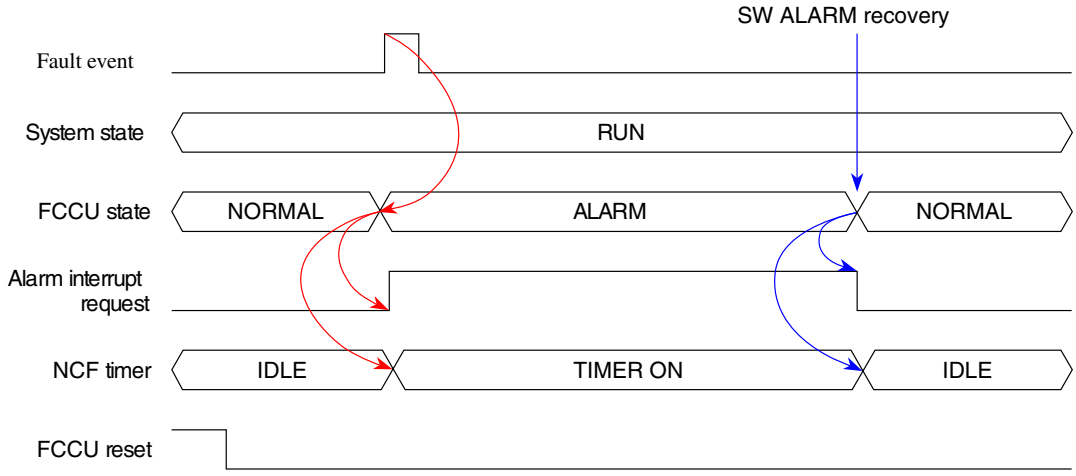


Figure 210. Non-critical fault (ALARM state) SW recovery

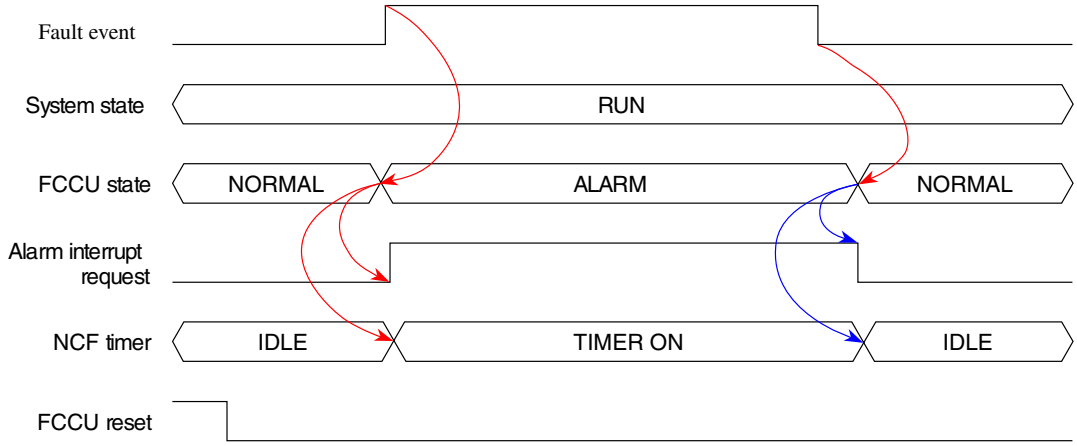


Figure 211. Non-critical fault (ALARM state) HW recovery

A typical sequence related to non-critical fault management (ALARM to FAULT state), see Figure 212, is as follows:

1. Non-critical fault assertion
2. FCCU state transition (automatic): NORMAL to ALARM
   - Alarm interrupt request (if enabled)
   - Timeout running
3. FCCU state transition (following the timeout trigger): ALARM to FAULT
   - NMI assertion (if enabled)
4. NMI interrupt management (if enabled)
   - Fault recovery (by software): FCCU state transition: FAULT to NORMAL

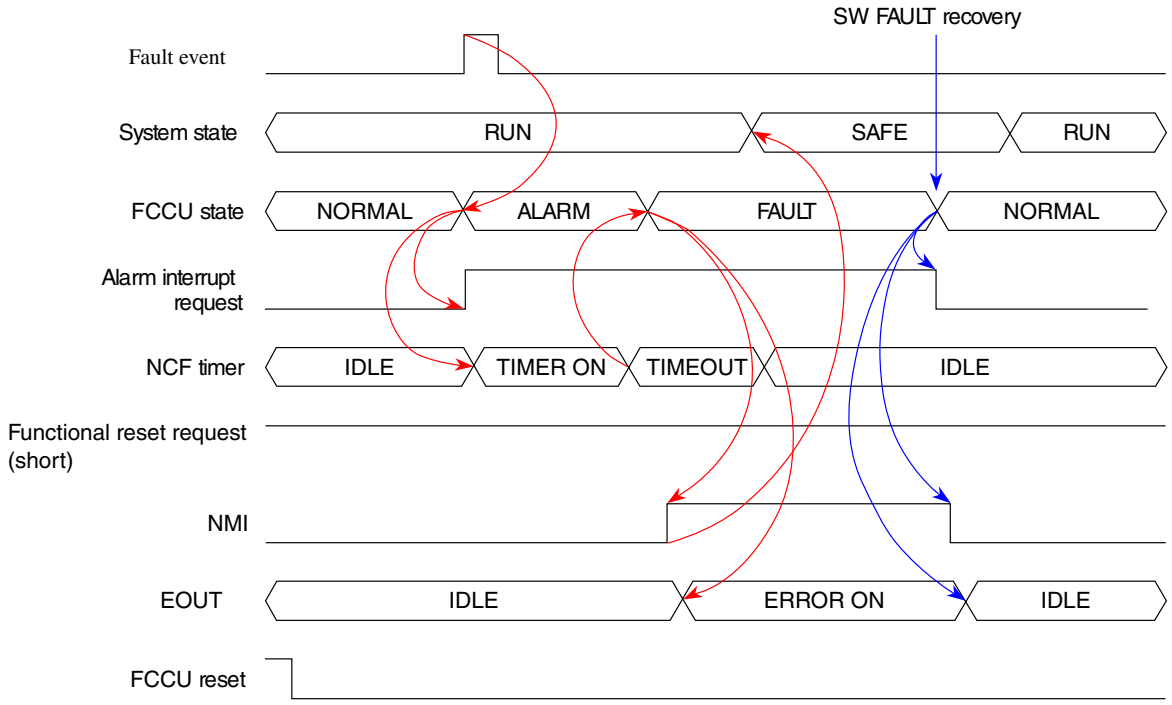


Figure 212. Non-critical fault (ALARM to FAULT state) recovery

52.3.5 EOUT interface

Introduction

You use the EOUT[1:0] signals to indicate FCCU's condition (or, for Fault-Toggle fault-output mode, faults only) to off-chip logic.

NOTE

For information on the availability and names of these FCCU signals on the boundary of this chip, see the chip-specific FCCU information.

The FCCU conditions

There are three FCCU conditions:

Condition	Description
Faulty	All of the following are true: • The fault-output (EOUT) timer is running (see How the fault-output (EOUT) timer works in Bi-Stable fault-output mode). • FCCU is in FAULT state.
Non-faulty	All of the following are true: • The fault-output (EOUT) timer is not running. • FCCU is in ALARM or NORMAL state.

Table continues on the next page...

Table continued from the previous page...

Condition	Description
Configuration	All of the following are true: • The fault-output (EOUT) timer is not running. • FCCU is in CONFIG state.

How the fault-output (EOUT) timer works in Bi-Stable fault-output mode

In Bi-Stable fault-output mode (FOM), FCCU starts the fault-output (EOUT) timer when all of the following are true:

- If the EOUT signals are in Bi-Stable FOM, and the EOUT signals are not programmed to be always low (CFG[FCCU_SET_CLEAR]).
- The EOUT timer is not already running.
- FCCU enters the FAULT state as the result of a fault.

When the fault-output (EOUT) timer is already running and a new fault occurs:

- If FCCU is in the CONFIG state: FCCU does not restart the EOUT timer.
- If FCCU is in the NORMAL or ALARM state:
  - And ALARM state is enabled for the fault (non-critical): FCCU enters (or remains in) the ALARM state but does not restart the EOUT timer.
  - And ALARM state is disabled for the fault (non-critical): FCCU enters the FAULT state and restarts the EOUT timer.
- If FCCU is in the FAULT state: FCCU restarts the fault-output (EOUT) timer.

FCCU stops and reinitializes the fault-output (EOUT) timer when all of the following are true:

- If the EOUT signals are in Bi-Stable fault-output mode (CFG[FOM]), and T_{min} (see DELTA_T[DELTA_T]) has expired.
- All faults that caused FCCU to enter or remain in the FAULT state since FCCU started the fault-output (EOUT) timer have been cleared, causing FCCU to return to the NORMAL state.

How the fault-output (EOUT) timer works in Fault-Toggle fault-output mode

In Fault-Toggle fault-output mode, FCCU initializes the fault-output (EOUT) timer with the value of T_{min} (see DELTA_T[DELTA_T]) and begins decrementing the timer when any of the following are true:

- FCCU enters the FAULT state as the result of a fault.
- In FAULT state, a fault occurs while the EOUT timer is 0 (idle).
- In FAULT state, a fault occurs while the EOUT timer is running because of a previous fault (also called a pending fault), and then in any state the EOUT timer reaches 0 (idle).

FCCU stops the EOUT timer when it reaches 0 (idle).

Prepare the EOUT signals to indicate FCCU's condition

- If the EOUT signals are in Bi-Stable or Fault-Toggle fault-output mode (CFG[FOM]), ensure that the EOUT signals are controlled by FCCU's FSM (CFG[FCCU_SET_CLEAR]).
- Ensure that the EOUT signals are active (CFG[FCCU_SET_AFTER_RESET]).

NOTE

If the EOUT signals are in Bi-Stable or Fault-Toggle fault-output mode, you must deactivate and then reactivate the EOUT signals (CFG[FCCU_SET_AFTER_RESET]) to correctly initialize them so they have opposite states.

More about the EOUT interface

Different fault-output modes (protocols) for the fault-output (EOUT) interface are supported (CFG[FOM]):

- Bi-Stable
- Fault-Toggle

NOTE

See the chip-specific FCCU information for the fault-output modes supported by this chip.

You can further configure the fault-output modes using the following attributes:

Attribute	Field	Setting used in the example diagrams and tables that follow
Configuration mode	CFG[CM]	Different
Polarity selection	CFG[PS]	For the faulty indication, EOUT1 is high, and EOUT0 is low.

EOUT frequency: This frequency is generated by dividing the CLKSAFE frequency by a fixed factor of 2^{18}.

$$EOUT_{freq} = \frac{CLKSAFE_{freq}}{2^{18}}$$

For example, with a CLKSAFE frequency of 16 MHz, this drives a signal of 61 Hz on EOUT.

In case of a failure event or on software request for EOUT indication, the signal(s) are set to the faulty state for a minimum time (T_{min}), even if software tries to release it before. If software configures the error pins to OK(1), and if a fault comes trying to drive the pin to NOK(0), then priority is given to the fault indication and the error signals indicate NOK, such as an incoming fault is not masked even when software has set the error signal to high. Also, if the error signals are forced to low by software by writing to CFG[FCCU_SET_CLEAR], then the signals shall remain low (or high) for the entire duration of T_{min}. During the T_{min} by a non-software fault, the FCCU FSM moves independently of this signal state (low), and as soon as the timer expires, the pin behavior is dictated by the state in which the FSM finds itself in, and it is not possible to set the signals to OK by software moving FCCU to the CONFIG state, as long as this timer is running. No software intervention is needed to bring the signal from the low state.

Software can bring the pin back to OK state by clearing the faults and waiting for the T_{min} interval to expire, after which the FCCU automatically enters the NORMAL state and the error signal indicates OK.

In case another failure event happens within T_{min} after a first one, the T_{min} counter is restarted.

52.3.5.1 Bi-Stable protocol

The encoding scheme is provided in Table 299 and the related timing diagram is shown in Figure 213.

Table 299. Bi-Stable encoding

Condition	EOUT[1:0] (CFG[PS] is 0)	EOUT[1:0] (CFG[PS] is 1)
Non-faulty	Static 01	Static 10
Faulty	Static 10	Static 01

Table continues on the next page...

Table 299. Bi-Stable encoding (continued)

Condition	EOUT[1:0] (CFG[PS] is 0)	EOUT[1:0] (CFG[PS] is 1)
Reset	High-impedance (no toggling)[1]	High-impedance (no toggling)
Configuration (CFG[CM] is 1 and CFG[FCCU_SET_AFTER_RESET] is 0)	High-impedance (no toggling)[2]	High-impedance (no toggling)
Configuration (When CFG[CM] is 1 and CFG[FCCU_SET_AFTER_RESET] is 1)	Static 01	Static 10

1. Final value depends on device specific settings at pad level.
2. Ensure that the EOUT signals are active (CFG[FCCU_SET_AFTER_RESET]); otherwise, the EOUT signals stay in a high-impedance state after reset lifts.

NOTE

Figure 213 is formatted to display the behavior in all four conditions (reset, non-faulty, faulty, and configuration), not to imply transitions between one condition and another. In particular, a transition from the faulty condition to the configuration condition is not possible.

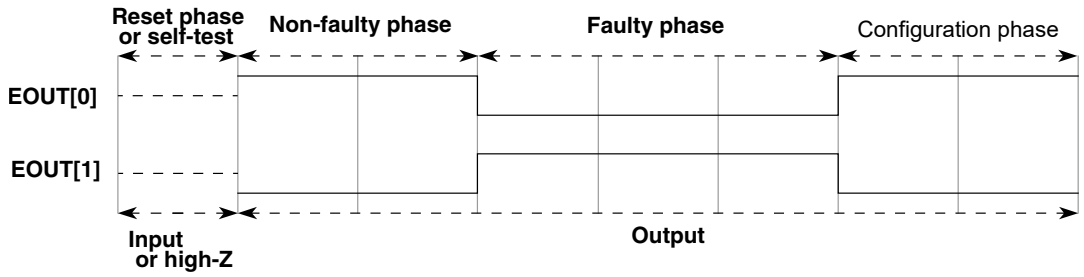


Figure 213. Bi-Stable protocol

52.3.5.2 Fault-Toggle protocol

The Fault-Toggle protocol uses the EOUT signals to indicate only the occurrences of faults, not FCCU's condition like the other protocols. In Fault-Toggle fault-output mode, when controlled by the FSM (CFG[FCCU_SET_CLEAR]), the EOUT signals:

- Have opposite states
- Toggle their states to indicate a fault and then maintain those new states for at least T_{min} (see DELTA_T[DELTA_T]), after which they are free to toggle to indicate another fault

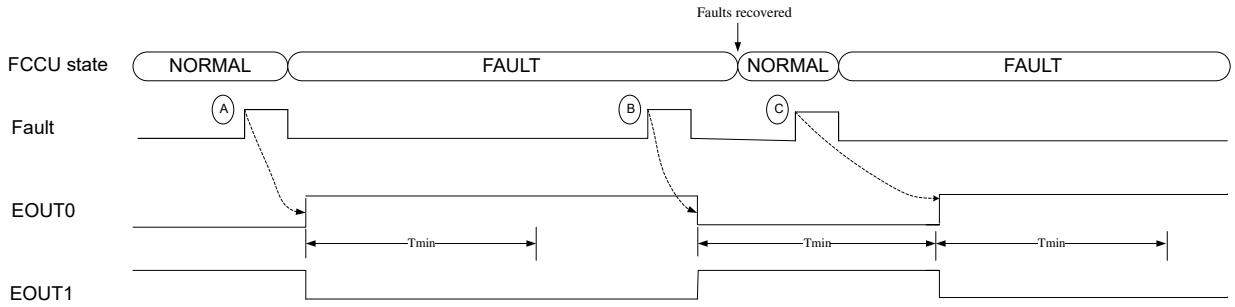
FCCU uses the fault-output (EOUT) timer to measure T_{min}. For more information, see How the fault-output (EOUT) timer works in Fault-Toggle fault-output mode.

The EOUT polarity selection (CFG[PS]) determines the initial states of the EOUT signals, which FCCU applies after you deactivate and then reactivate the EOUT signals (CFG[FCCU_SET_AFTER_RESET]).

If two or more faults occur while the EOUT timer is running, then the EOUT signals toggle only once after the EOUT timer reaches zero (idle). In other words, FCCU can pipeline only one fault.

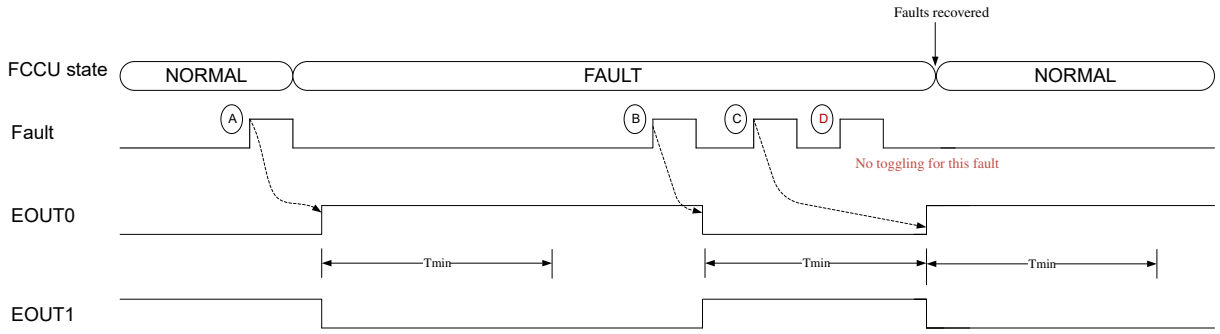
Note that FCCU does not toggle the EOUT signals in CONFIG state. FCCU toggles the EOUT signals only when FCCU enters the FAULT state or when a fault occurs in the FAULT state.

Example 1



Stage	Description
1	A fault source indicates fault A to FCCU.
2	FCCU enters the FAULT state.
3	FCCU toggles the EOUT signals in response to fault A, initializes the fault-output (EOUT) timer with the value of T_{min} (see DELTA_T[DELTA_T]), and begins decrementing the timer.
4	A fault source indicates fault B to FCCU while FCCU is still in the FAULT state.
5	The EOUT timer has already reached 0 (idle), so FCCU toggles the EOUT signals in response to fault B, initializes the EOUT timer with the value of T_{min}, and begins decrementing the timer.
6	The source (for hardware-recoverable faults) and software (for software-recoverable faults) recovers the faults.
7	FCCU enters the NORMAL state.
8	A fault source indicates fault C to FCCU.
9	The EOUT timer has not yet reached 0 (idle), so FCCU leaves the EOUT signals as they are.
10	When the EOUT timer reaches 0 (idle), FCCU toggles the EOUT signals in response to fault C, initializes the EOUT timer with the value of T_{min}, and begins decrementing the timer.

Example 2



Stage	Description
1	A fault source indicates fault A to FCCU.
2	FCCU enters the FAULT state.
3	FCCU toggles the EOUT signals in response to fault A, initializes the fault-output (EOUT) timer with the value of T_{min} (see DELTA_T[DELTA_T]), and begins decrementing the timer.
4	A fault source indicates fault B to FCCU while FCCU is still in the FAULT state.
5	The EOUT timer has already reached 0 (idle), so FCCU toggles the EOUT signals in response to fault B, initializes the EOUT timer with the value of T_{min}, and begins decrementing the timer.
6	A fault source indicates fault C to FCCU while FCCU is still in the FAULT state.
7	The EOUT timer has not yet reached 0 (idle), so FCCU leaves the EOUT signals as they are. Fault C is now *pending*.
8	A fault source indicates fault D to FCCU while FCCU is still in the FAULT state. (FCCU can pipeline only one fault, so FCCU ignores fault D.)
9	The EOUT timer has not yet reached 0 (idle), so FCCU leaves the EOUT signals as they are.
10	When the EOUT timer reaches 0 (idle), FCCU toggles the EOUT signals in response to fault C, initializes the EOUT timer with the value of T_{min}, and begins decrementing the timer. FCCU never toggles the EOUT signals in response to fault D because FCCU can pipeline only one fault.

52.3.6 Modes of operation

52.3.6.1 Put FCCU in the NORMAL state

52.3.6.1.1 Introduction

You put FCCU in the NORMAL state to save changes to the configuration, and to allow FCCU to enter the ALARM or FAULT state when a fault occurs on an enabled fault channel.

52.3.6.1.2 About putting FCCU in NORMAL state

When putting FCCU in the NORMAL state:

- If you attempt to lock the configuration while FCCU is in CONFIG state, FCCU does not actually lock the configuration until FCCU leaves CONFIG state—that is, either you put FCCU in the NORMAL state, or FCCU puts itself in the NORMAL state because the Configuration-state timeout interval (CFG_TO[TO]) expires.
- After you permanently lock the configuration, you must reset FCCU before you can put FCCU in the CONFIG state.

52.3.6.1.3 Put FCCU in the NORMAL state

1. Check the FCCU status (STAT[STATUS]).
   - If the FCCU status is NORMAL, go to step 6.
   - If the FCCU status is CONFIG, go to step 2.
   - If the FCCU status is ALARM or FAULT, go to step 4.
2. Run the OP2 operation (see Run an operation).
3. Check the operation status (CTRL[OPS]).
   - If the operation status is Successful, FCCU is in the NORMAL state. Go to step 6.
   - If the operation status is Aborted, go to step 2.
4. Recover all faults (see Fault recovery).
5. Go to step 1.
6. Lock the configuration if you want to prevent any changes to it:
   - To require a key to unlock the configuration, from Supervisor mode, temporarily lock the configuration (TRANS_LOCK[TRANSKEY]).
   - To require a reset of FCCU to unlock the configuration, from Supervisor mode, permanently lock the configuration (PERMNT_LOCK[PERMNTKEY]).

     The configuration is permanently locked until FCCU is reset.

52.3.6.2 Manage faults

52.3.6.2.1 Introduction

After saving changes to the configuration, you are ready to use FCCU to manage faults.

52.3.6.2.2 Determine if there are any unrecovered non-critical faults

Check the unrecovered-fault indicators for the non-critical faults (NCF_Sa[NCFSn]).

52.3.6.2.3 Recover a software-recoverable non-critical fault

1. Resolve the source of the software-recoverable non-critical fault.
2. Unlock the NCF_Sa registers (NCFK[NCFK]) using a 32-bit write.

3. Initiate clearing of the unrecovered-fault indicator for the software-recoverable non-critical fault (NCF_Sa[NCFSn]) using a 32-bit write.

   FCCU initiates the OP12 operation.

   NOTE

   If you want to clear multiple unrecovered-fault indicators and those indicators reside in different NCF_Sa registers, you must perform steps 2 and 3 for each individual register.

4. Check the operation status (CTRL[OPS]).
   - If the operation status is In Progress, go to step 4.
   - If the operation status is Successful, go to step 5.
   - If the operation status is Aborted, go to step 2.
5. Check the unrecovered-fault indicator for the software-recoverable non-critical fault (NCF_Sa[NCFSn]).
   - If the indicator indicates no unrecovered fault, the fault has been recovered. Stop.
   - If the indicator still indicates an unrecovered fault, go to step 2.

52.3.6.2.4 Clear the freeze-status indicators

1. Run the OP13 operation (see Run an operation).
2. Check the operation status (CTRL[OPS]).
   - If the operation status is Successful, stop .
   - If the operation status is Aborted, go to step 1.

52.3.6.3 Run operations

52.3.6.3.1 Introduction

You run operations to perform actions such as putting FCCU in the CONFIG state or setting the operation status to Idle. For a complete list of operations you can run, see CTRL[OPR].

52.3.6.3.2 About running operations

When running operations:

- FCCU ignores any operations initiated while the operation status is In Progress.
- Certain operations must be unlocked before you can initiate them. After you initiate them, they are locked again.

52.3.6.3.3 Run an operation

1. Check the operation status (CTRL[OPS]).
2. Go to step 1 if the operation status is In Progress.
3. Unlock the operation (CTRLK[CTRLK]) using a 32-bit write, if the operation must be unlocked before you can initiate it.

   NOTE

   The Control Key (CTRLK) register used in this step and the Control (CTRL) register used in the next step must be written with consecutive instructions. Do not use read-modify-write instructions, such as bit-field instructions, to modify these registers.

4. Initiate the operation (CTRL[OPR]) using a 32-bit write.
5. Check the operation status (CTRL[OPS]).

6. Go to step 5 if the operation status is In Progress.

The operation status is now Idle (OP15 only), Aborted, or Successful.

52.3.7 FOSU

The FOSU provides a supervision of the primary fault notification path by analyzing FCCU's behavior for correctness. It waits for any reaction of the FCCU in a fixed time window after a fault is signaled.

The intention of the FOSU is to provide a secondary fault reaction path in most cases when the FCCU fails but not to needlessly propagate a fault which is already handled by the FCCU in a full chip reset. Only a failed primary fault reaction (that is, FCCU's failure) is a reason for the secondary reaction to take over (and generate a destructive reset request).

There is a 'do nothing' input coming from FCCU which indicates that the FCCU is programmed for no reaction for ALL FAULTS. It is a "static" input in the sense that it does not change after FCCU configuration. The FOSU masks the incoming faults with the 'do nothing' control from the FCCU, meaning that a fault is not captured by the FOSU if the 'do nothing' signal is asserted (that is, a disabled fault). There is no minimum pulse width requirement on the fault indication other than what is required by the technology, which is the same as that of the FCCU. FOSU does not monitor FCCU for the case of faults occurring during the CONFIG state.

The FOSU contains a timer with a duration of FOSU_COUNT, driven by CLKSAFE. The timer is initialized and started on any captured, enabled fault. While the timer is running, any subsequent captured fault will neither restart nor reinitialize the timer. The timer is stopped when the FCCU shows any of the following reactions (the FOSU does not check whether the reaction is the configured one for the faults which occurred):

- Reset: short functional reset
- IRQ (triggered by ALARM state)
- NMI
- Error out triggered (by FCCU or by SW)

When the timer is stopped, the fault capture logic is cleared to ensure that the timer is not restarted because of faults still 'stuck' in the capture logic. The timer is then restarted by the next new failure indication. When the timer expires, the FOSU's failure indicator output is asserted after it ensures that the fault is enabled and the static "fccu program to do nothing" signal is deasserted. This is because FCCU uses settings after it exits CONFIG state, even if fault captured before the exit.

The FOSU's failure indicator output is connected to one of the MC_RGM's 'destructive' reset inputs, so its assertion will cause a reset sequence to be initiated starting at DEST0. The FOSU module is reset with the same reset as is used by the FCCU. When this reset is asserted, the FOSU's capture logic is cleared, its timer is kept stopped and in a non-expired state, and its failure indicator output is deasserted.

NOTE

FOSU is triggered on assertion of enabled fault. In case the triggering fault is disabled, FOSU times out without reaction.

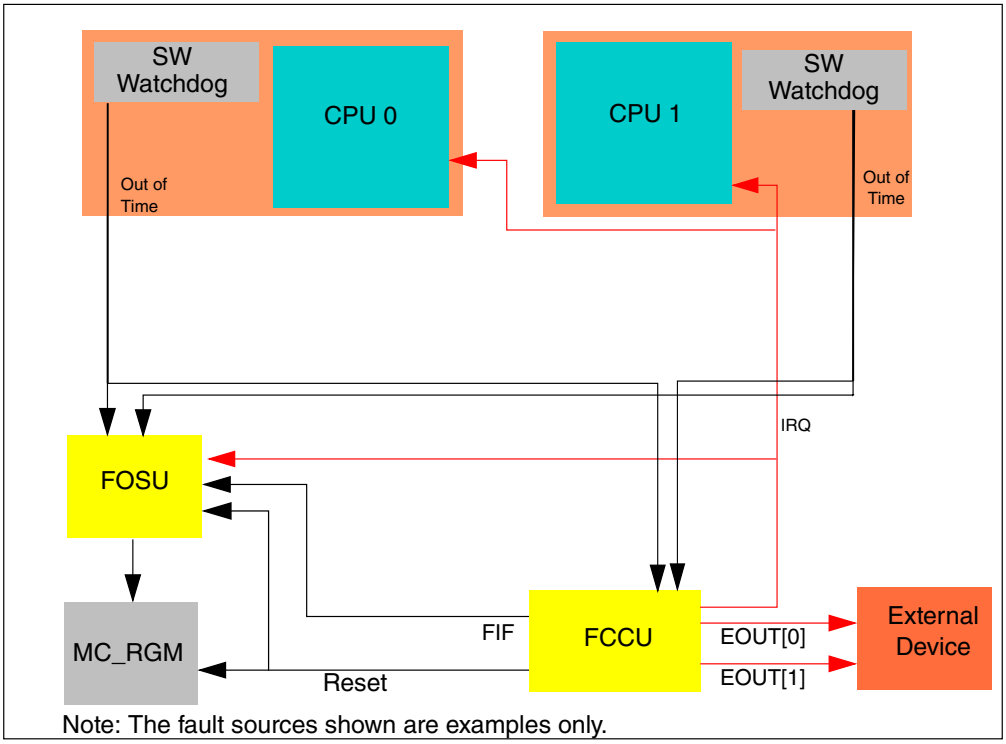


Figure 214. FOSU connections to the FCCU and MC_RGM

52.3.8 Clocking

This module has no clocking considerations.

52.3.9 Interrupts

This module has no interrupts.

52.4 External signals

FCCU interfaces with the EOUT pin. Signaling on the EOUT pin depends on whether the module is processing an error or is idle.

52.5 Initialization

52.5.1 Prepare FCCU for configuration

52.5.1.1 Introduction

You prepare FCCU for configuration by first configuring the CONFIG state and then putting FCCU in that state.

52.5.1.2 About preparing FCCU for configuration

When preparing FCCU for configuration, keep the following in mind:

- To put FCCU in CONFIG state, FCCU must be in NORMAL state.
- After FCCU is reset, the configuration is temporarily locked. You must temporarily unlock the configuration before you can put FCCU in the CONFIG state.

- When FCCU is in the CONFIG state, FCCU does not actually save the changes you make to the configuration. To save changes to the configuration, you must manually put FCCU in the NORMAL state. If FCCU automatically leaves the CONFIG state and enters the NORMAL state because the Configuration-state timeout interval (CFG_TO[TO]) expires (called a Configuration-state timeout), FCCU changes the value of the Configuration (CFG) register to its Configuration-state-timeout value and the value of each of the other configuration registers to its reset value. FCCU also changes the value of the Configuration-State Timeout Interval (CFG_TO) register to its reset value. For information on the Configuration-state timeout value, see CFG register bit value at different events. For a list of configuration registers, see Configuration registers.

52.5.1.3 Configure the CONFIG state

1. Set the Configuration-state timeout interval (CFG_TO[TO]).
2. Enable the Configuration-state-timeout interrupt signal (IRQ_EN[CFG_TO_IEN]), if you want FCCU to request an interrupt when a Configuration-state timeout occurs.

52.5.1.4 Put FCCU in the CONFIG state

1. Unlock the configuration temporarily (TRANS_LOCK[TRANSKEY]) in Supervisor mode.
2. Check the FCCU status (STAT[STATUS]).
   - If the FCCU status is CONFIG, FCCU is in the CONFIG state. Stop.
   - If the FCCU status is NORMAL, go to step 3.
   - If the FCCU status is ALARM or FAULT, go to step 5.
3. Run the OP1 operation (see Run an operation).
4. Check the operation status (CTRL[OPS]).
   - If the operation status is Successful, FCCU is in the CONFIG state. Stop.
   - If the operation status is Aborted, the configuration is probably permanently locked. Go to step 7.
5. Recover all faults (see Fault recovery).
6. Go to step 2.
7. Reset FCCU.

52.5.2 Configure FCCU

52.5.2.1 Introduction

You configure FCCU so it functions according to the needs of your particular application.

52.5.2.2 About configuring FCCU

When configuring FCCU:

- If you enable a non-critical fault channel but disable all reactions for that channel, FCCU changes state when necessary but does not perform any reaction because reactions are disabled. If you enable reactions for a non-critical fault channel but disable that channel, and FCCU is in the NORMAL state when a fault occurs on the channel, FCCU does not enter the ALARM or FAULT state and therefore does not perform any reaction.

52.5.2.3 Configure the non-critical fault channels

For each non-critical fault channel that you want FCCU to monitor:

1. Set the recovery type (NCF_CFGa[NCFCn]).
2. Enable at least one type of Fault-state reaction:
   - Chip functional reset (NCFS_CFGa[NCFSCn])

- Non-maskable interrupt (NMI_ENa[NMIENn])
- EOUT signaling (EOUT_SIG_ENa[EOUTENn])

NOTE

If you enable the chip functional reset as the type of Fault-state reaction for a channel, enable at least one other type of Fault-state reaction for the channel or enable the ALARM state (step 4) for the channel.

3. Set the Alarm-state timeout interval (NCF_TO[TO]), if you plan to enable the ALARM state for any non-critical fault channel.

NOTE

Ensure that the Alarm-state timeout interval is less than the FOSU module's timeout interval; otherwise, FOSU generates a chip reset every time a fault occurs on the channel. The FOSU timeout interval (FOSU_COUNT) is a chip-specific value. See the chip-specific FCCU information.

4. Enable the ALARM state (NCF_TOEa[NCFTOEn]) for any non-critical fault channel for which you want FCCU to enter the ALARM state before entering the FAULT state.
5. Enable the Alarm-state reaction (IRQ_ALARM_ENa[IRQENn]) for each non-critical fault channel for which you enabled the ALARM state.
6. Enable the corresponding NCF_Ea[NCFEn] field for each non-critical fault channel that you want FCCU to monitor.

52.6 Application information

52.6.1 Use cases and limitations

Configuration guidelines

Follow these guidelines to configure FCCU:

- If you want FCCU to react to a fault on a non-critical fault channel:
  - Enable the channel (Non-critical Fault Enable (NCF_E0)).
  - Enable at least one type of Fault-state reaction for the channel: chip reset (Non-critical Fault-State Configuration (NCFS_CFG0)), fault-output (EOUT) signaling (Non-critical Fault-State EOUT Signaling Enable (EOUT_SIG_EN0)), or non-maskable interrupt (Non-critical Fault-State Non-maskable-Interrupt-Request Enable (NMI_EN0)).
  - If you enable chip reset as the type of Fault-state reaction for the channel (Non-critical Fault-State Configuration (NCFS_CFG0)), enable either ALARM state (Non-critical-Fault Alarm-State Timeout Enable (NCF_TOE0)) or at least one other type of Fault-state reaction for the channel: fault-output (EOUT) signaling (Non-critical Fault-State EOUT Signaling Enable (EOUT_SIG_EN0)) or non-maskable interrupt (Non-critical Fault-State Non-maskable-Interrupt-Request Enable (NMI_EN0)).
  - If you enable ALARM state for the channel (Non-critical-Fault Alarm-State Timeout Enable (NCF_TOE0)), enable the Alarm-state reaction (Non-critical Alarm-State Interrupt-Request Enable (IRQ_ALARM_EN0)).
  - If you enable ALARM state for the channel (Non-critical-Fault Alarm-State Timeout Enable (NCF_TOE0)), make sure the Alarm-state timer interval (Non-critical-Fault Alarm-State Timeout Interval (NCF_TO)) is less than the FOSU module's timer interval; otherwise, FOSU generates a chip reset every time a fault occurs on the channel. The FOSU timer interval (FOSU_COUNT) is chip-specific. See the chip-specific FCCU information.

Recommendations to configure FCCU

1. After a power on, or 'destructive' reset (when initiated by the assertion of the chip reset pin, RESET_B), where both system and FCCU are reset, the following steps could be followed to configure FCCU:
   a. Check and clear any pending fault status
   b. Verify FCCU is in NORMAL state, else repeat step(a) above

c. Configure FCCU

2. After any 'functional' reset of the system, arising out of a reset request from FCCU or other sources, the following steps could be followed to reconfigure FCCU:
   a. If active, wait for the Error out T_{min} to expire
   b. Check and clear fault status
   c. Error pin moves to "non faulty" state, once fault status is cleared and T_{min} expires
   d. Verify FCCU is in NORMAL state, else repeat step(a) above
   e. Read and verify value in NCF_En
   f. Reconfigure FCCU, if necessary

52.7 Register descriptions

52.7.1 FCCU register descriptions

The FCCU registers are listed in the table below. Any address offset not explicitly mentioned in this table is reserved.

The FCCU supports word (32-bit), half-word (16-bit), and byte (8-bit) read and write accesses.

Follow these register-access guidelines:

- Do not read from or write to any addresses that are not shown in the following table. Doing so may or may not result in a transfer error.
- Do not write to any of the configuration registers unless FCCU is in the CONFIG state. Doing so results in a transfer error.
- Do not write to the Transient Configuration Lock (TRANS_LOCK) or Permanent Configuration Lock (PERMNT_LOCK) registers unless your code runs in the Supervisor mode. Doing so results in a transfer error.

For each possible NCF failure source, a different reaction—including no reaction—is configurable through the use of NMI, IRQ, and short reset selection registers. It is not possible for a single event upset to switch off all reactions on failures as implementation is per fault source (but it will be possible to switch them all off by SW if intended). Failures themselves are not able to disable all reactions and indications.

52.7.1.1 FCCU memory map

FCCU base address: 4038_4000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Control (CTRL)	32	RW	0000_00C0h
4h	Control Key (CTRLK)	32	W	0000_0000h
8h	Configuration (CFG)	32	RW	0000_0000h
1Ch	Non-critical Fault Configuration (NCF_CFG0)	32	RW	0000_00FFh
4Ch	Non-critical Fault-State Configuration (NCFS_CFG0)	32	RW	0000_0000h
80h	Non-critical Fault Status (NCF_S0)	32	RW	0000_0000h
90h	Non-critical Fault Key (NCFK)	32	RW	0000_0000h
94h	Non-critical Fault Enable (NCF_E0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
A4h	Non-critical-Fault Alarm-State Timeout Enable (NCF_TOE0)	32	RW	0000_00FFh
B4h	Non-critical-Fault Alarm-State Timeout Interval (NCF_TO)	32	RW	0003_A980h
B8h	Configuration-State Timeout Interval (CFG_TO)	32	RW	0000_0005h
BCh	IO Control (EINOUT)	32	RW	See section
C0h	Status (STAT)	32	R	0000_0010h
C4h	Normal-to-Alarm Freeze Status (N2AF_STATUS)	32	R	0000_0000h
C8h	Alarm-to-Fault Freeze Status (A2FF_STATUS)	32	R	0000_0000h
CCh	Normal-to-Fault Freeze Status (N2FF_STATUS)	32	R	0000_0000h
D0h	Fault-to-Alarm Freeze Status (F2AF_STATUS)	32	R	0000_0000h
DCh	Non-critical Fault Fake (NCFF)	32	W	0000_0000h
E0h	IRQ Status (IRQ_STAT)	32	RW	0000_0000h
E4h	IRQ Enable (IRQ_EN)	32	RW	0000_0000h
F0h	Transient Configuration Lock (TRANS_LOCK)	32	W	0000_0000h
F4h	Permanent Configuration Lock (PERMNT_LOCK)	32	W	0000_0000h
F8h	Delta T (DELTA_T)	32	RW	0000_0000h
FCh	Non-critical Alarm-State Interrupt-Request Enable (IRQ_ALARM_EN0)	32	RW	0000_0000h
10Ch	Non-critical Fault-State Non-maskable-Interrupt-Request Enable (NMI_EN0)	32	RW	0000_0000h
11Ch	Non-critical Fault-State EOUT Signaling Enable (EOUT_SIG_EN0)	32	RW	0000_0000h
12Ch	Alarm-State Timer (TMR_ALARM)	32	R	0003_A980h
134h	Configuration-State Timer (TMR_CFG)	32	R	000F_FFFFh
138h	Fault-Output Timer (TMR_ETMR)	32	R	0000_0000h

52.7.1.2 Control (CTRL)

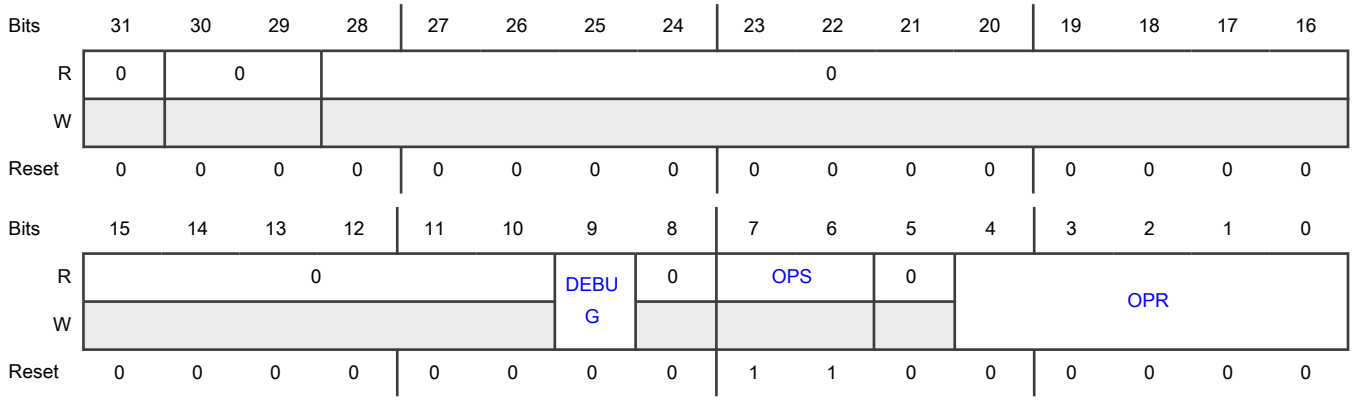
Offset

Register	Offset
CTRL	0h

Function

Initiates and indicates the status of operations—and enables the Debug mode.

Diagram



Fields

Field	Function
31 —	Reserved
30-29 —	Reserved
28-10 —	Reserved
9 DEBUG	Debug Mode Enable Specifies whether the Debug mode is enabled. If so, FCCU enters the Debug mode when the Debug signal is asserted. When FCCU enters the Debug mode, it halts operation and remains in the state it was in before it entered this mode. **NOTE** FOSU does not halt when FCCU enters the Debug mode. Therefore, FOSU can still cause a reset if a fault occurs while FCCU is in the Debug mode. 0b - Disabled 1b - Enabled
8 —	Reserved
7-6 OPS	Operation Status This field can be read and cleared (via OP15 operation) by the software. 00b - Idle 01b - In progress 10b - Aborted

Table continues on the next page...

Table continued from the previous page...

Field	Function
	11b - Successful
5 —	Reserved
4-0 OPR	Operation Run Initiates operations that perform actions such as putting FCCU in the CONFIG state or setting the operation status to Idle. For information on how to run operations, see Run operations. FCCU ignores any write to this field while the operation status (CTRL[OPS]) is "In progress". After completion of an operation, FCCU sets this field to OP0. The following events result in an operation status (CTRL[OPS]) of "Aborted": • Writing to a NCF_Sa register (which automatically initiates the OP12 operation) without first successfully unlocking the register (NCFK[NCFK]) • Initiating an OP1 operation when FCCU is not in the NORMAL state or the configuration is locked • Initiating an OP1, or OP2, operation without first unlocking the operation (CTRLK[CTRLK]) 00000 OP0—No operation 00001 OP1—Applies only when the configuration is unlocked, when FCCU is in the NORMAL state, and immediately after you unlock the operation (CTRLK[CTRLK]). Put FCCU in the CONFIG state. 00010 OP2—Applies only immediately after you unlock the operation (CTRLK[CTRLK]). Put FCCU in the NORMAL state. 00011 Reserved 00100 Reserved 00101 Reserved 00110 Reserved 00111 Reserved 01000 Reserved 01001 Reserved 01010 Reserved 01011 Reserved 01100 OP12—Do not initiate this operation; it is automatically initiated by the FCCU. A NCF_Sa register status clear operation is in progress. 01101 OP13—Clear the freeze status registers. 01110 OP14—Do not initiate this operation; it is automatically initiated by the FCCU. A Configuration-state timeout is in progress. For more information, see Configuration registers. 01111 OP15—Set the operation status (CTRL[OPS]) to Idle. 10000 Reserved 10001 Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10010 Reserved 10011 Reserved 10100 Reserved 10101—11110 Forbidden. Writing any of these values returns an operation status (CTRL[OPS]) of "Aborted" with no side effect. 11111 Reserved

52.7.1.3 Control Key (CTRLK)

Offset

Register	Offset
CTRLK	4h

Function

See CTRLK[CTRLK].

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	CTRLK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	CTRLK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 CTRLK	Locked-Operation Control Key Writable only with a 32-bit write. Unlocks locked operations (CTRL[OPR]) so you can initiate them. For information on how to unlock locked operations before you initiate them, see Run an operation.

Table continues on the next page...

Field	Function
	NOTE • You must initiate an operation in the FCCU register access that immediately follows the one that unlocks it; otherwise, the operation is again locked. • Reading from this register always returns the value 0000_0000h. Operations not listed here are not locked and do not need to be unlocked. 9137_56AFh: Unlock OP1. 825A_132Bh: Unlock OP2. Any other value: Do nothing.

52.7.1.4 Configuration (CFG)

Offset

Register	Offset
CFG	8h

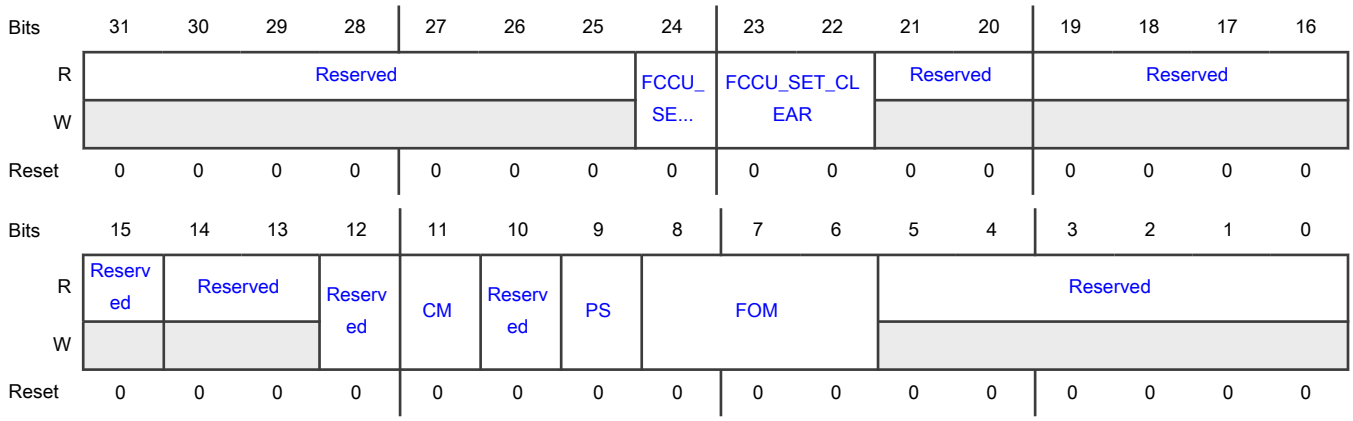
Function

Writable only when FCCU is in the CONFIG state. Changed by FCCU to another value when the chip resets FCCU, a Configuration-state timeout occurs, or you run an OP31 operation. See CFG register bit value at different events for more information. Specifies the global configuration for FCCU.

NOTE

If you specify a new value for any of the fields in this register that affect the EOUT signals while the fault-output (EOUT) timer is running (FCCU is indicating a fault on the EOUT signals), FCCU does not use the new settings you specified until after the fault-output (EOUT) timer expires (FCCU stops indicating a fault on the EOUT signals).

Diagram



Fields

Field	Function
31-25 —	Reserved
24 FCCU_SET_AFTER_RESET	Fault-Output (EOUT) Activate For fault-output (EOUT) signaling, controls whether the EOUT signals are active. 0b - Inactive (the EOUT signals are in a high-impedance state) 1b - Active (the EOUT signals indicate FCCU's condition)
23-22 FCCU_SET_CLEAR	Fault-Output (EOUT) Control Applies only to Bi-Stable and Fault-Toggle fault-output modes (CFG[FOM]) and when the EOUT signals are active (CFG[FCCU_SET_AFTER_RESET]). Controls whether the fault-output (EOUT) signals are managed by FCCU's FSM. **NOTE** When the EOUT signals are in the Fault-Toggle fault-output mode, if you change the value of this field, you must also deactivate and then reactivate the EOUT signals (CFG[FCCU_SET_AFTER_RESET]) to correctly initialize them so they have opposite states. 00b - Controlled by the FSM 01b - Always low 10b - Controlled by the FSM 11b - High until a fault occurs on a channel, regardless of whether that fault is disabled; thereafter, controlled by the FSM. Note: FCCU ignores an attempt to write this value if the fault-output (EOUT) timer is already running.
21-20 —	Reserved
19-16 —	Reserved
15 —	Reserved
14-13 —	Reserved
12 —	Reserved Always write the reset value to this field.
11 CM	Fault-Output (EOUT) Configuration-Indication Mode

Table continues on the next page...

Table continued from the previous page...

Field	Function
	For fault-output (EOUT) signaling, this field controls whether the configuration indication is the same as the non-faulty indication. 0b - Different 1b - Same
10 —	Reserved
9 PS	Fault-Output (EOUT) Polarity Selection Applies to fault-output (EOUT) signaling and controls the polarity of the signals for fault-output mode indications that hold the signals low or high (versus toggling them or placing them in a high-impedance state). Applies only to Bi-Stable fault-output mode (for all indications). 0b - For the faulty indication, EOUT1 is high, and EOUT0 is low. 1b - For the faulty indication, EOUT1 is low, and EOUT0 is high.
8-6 FOM	Fault-Output (EOUT) Mode For fault-output (EOUT) signaling, controls the protocol of the signaling. **NOTE** See the chip-specific FCCU information for the fault-output modes supported by this chip. 000b - Reserved 001b - Reserved 010b - Bi-Stable 011b - Fault-Toggle 100b - Reserved 101b - Test 0 (controlled by the EINOUT register; EOUT1 is an output; EOUT0 is an input) 110b - Test 1 (controlled by the EINOUT register; EOUT1 and EOUT0 are both outputs) 111b - Test 2 (controlled by the EINOUT register; EOUT1 is an input; EOUT0 is an output)
5-0 —	Reserved

52.7.1.5 Non-critical Fault Configuration (NCF_CFG0)

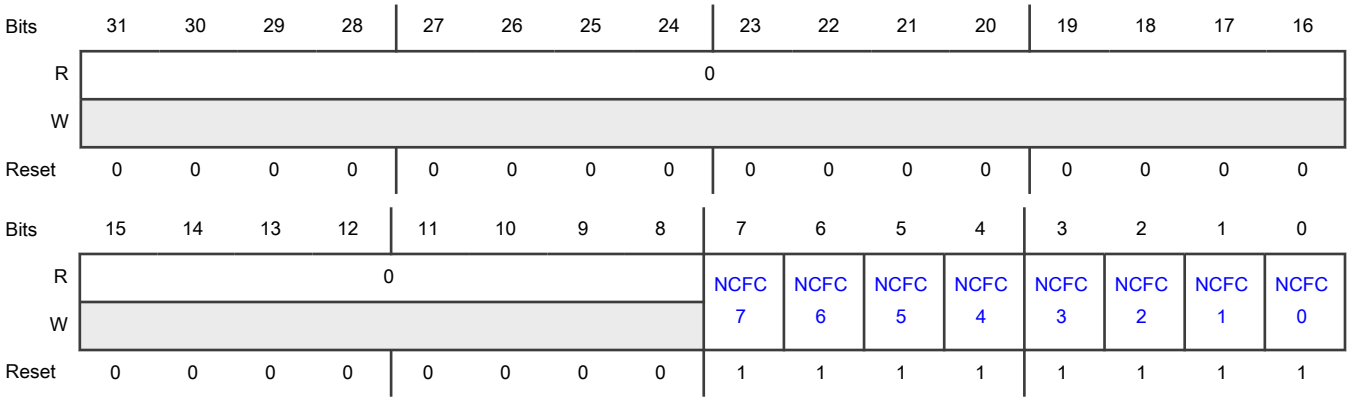
Offset

Register	Offset
NCF_CFG0	1Ch

Function

See NCF_CFGa[NCFCn].

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 NCFCn	Non-critical Fault Configuration n Writable only when FCCU is in the CONFIG state. Changed by FCCU to its reset value when a Configuration-state timeout occurs. Controls the recovery type (HW or SW) of the associated non-critical fault channel (n). For information on how to configure the non-critical fault channels, see Configure the non-critical fault channels. **NOTE** Configure a non-critical fault channel as hardware-recoverable only if the source continues to indicate a fault on the fault channel's input (NCFn) until the condition that caused the fault is no longer true; otherwise, configure the non-critical fault channel as software-recoverable. 0b - Hardware-recoverable 1b - Software-recoverable

52.7.1.6 Non-critical Fault-State Configuration (NCFS_CFG0)

Offset

Register	Offset
NCFS_CFG0	4Ch

Function

See NCFS_CFGa[NCFSCn].

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	NCFSC7		NCFSC6		NCFSC5		NCFSC4		NCFSC3		NCFSC2		NCFSC1		NCFSC0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-14: NCFSC7 13-12: NCFSC6 11-10: NCFSC5 9-8: NCFSC4 7-6: NCFSC3 5-4: NCFSC2 3-2: NCFSC1 1-0: NCFSC0	Non-critical Fault-State Configuration n Writable only when FCCU is in the CONFIG state. Changed by FCCU to its reset value when a Configuration-state timeout occurs. Controls whether the chip functional reset is enabled as a Fault-state reaction for the associated non-critical fault channel (n). When the chip functional reset is enabled for an enabled non-critical fault channel, a fault on that channel causes FCCU to assert the rst_sfunc_b signal when FCCU enters the FAULT state. For information on how to configure the non-critical fault channels, see Configure the non-critical fault channels. 00b - Disabled 01b - Enabled (rst_sfunc_b) (short) 10b - Reserved 11b - Disabled

52.7.1.7 Non-critical Fault Status (NCF_S0)

Offset

Register	Offset
NCF_S0	80h

Function

See NCF_Sa[NCFSn].

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								NCFS 7	NCFS 6	NCFS 5	NCFS 4	NCFS 3	NCFS 2	NCFS 1	NCFS 0
W									W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-8 —	Reserved
7-0 NCFSn	Non-critical Fault Status n Indicates whether there is an unrecovered fault on the associated non-critical fault channel (n). **NOTE** To recover a software-recoverable non-critical fault, which includes clearing its unrecovered-fault indicator, see Recover a software-recoverable non-critical fault. FCCU clears the unrecovered-fault indicator for a hardware-recoverable non-critical fault automatically when the source no longer indicates a fault on the fault channel's input signal; if you attempt to clear the unrecovered-fault indicator for a hardware-recoverable non-critical fault, FCCU does not clear the indicator and does not indicate an error. 0b - No unrecovered fault 1b - Unrecovered fault

52.7.1.8 Non-critical Fault Key (NCFK)

Offset

Register	Offset
NCFK	90h

Function

See NCFK[NCFK].

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W	NCFK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W	NCFK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 NCFK	Non-critical Fault Key Writable only with a 32-bit write. Unlocks the NCF_Sa registers so you can write to them while recovering a software-recoverable non-critical fault. For information on how to unlock the NCF_Sa registers before writing to them, see Recover a software-recoverable non-critical fault. **NOTE** • You must write to one of the NCF_Sa registers immediately after unlocking it (that is, in the FCCU register access that immediately follows the one that unlocks them); otherwise the registers are again locked. If you want to write to multiple NCF_Sa registers, you must unlock each register immediately before you write to it. • Reading from this register always returns the value 0000_0000h. AB34_98FEh: Unlock. Any other value: Do nothing.

52.7.1.9 Non-critical Fault Enable (NCF_E0)

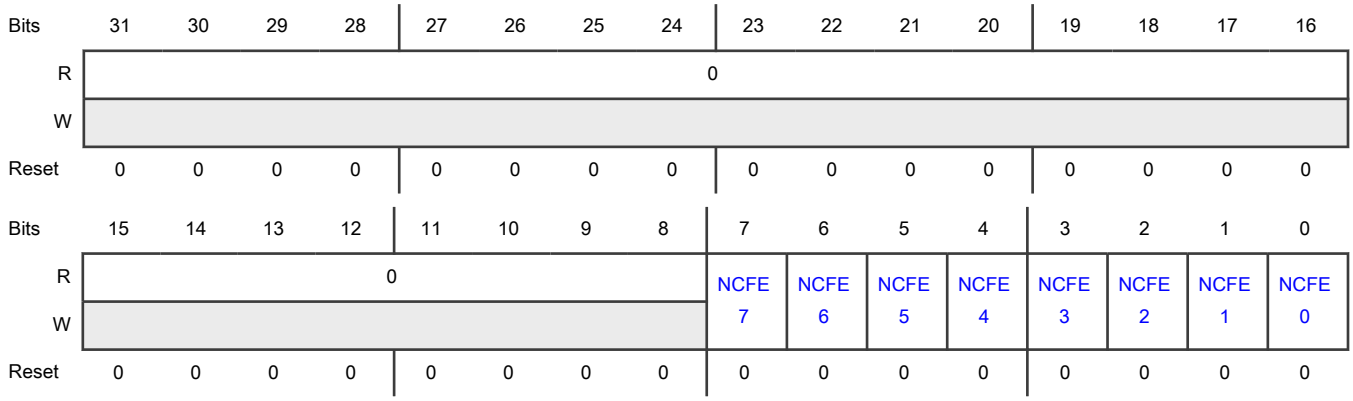
Offset

Register	Offset
NCF_E0	94h

Function

See NCF_Ea[NCFEn].

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 NCFEn	Non-critical Fault Enable n Writable only when FCCU is in the CONFIG state. Changed by FCCU to its reset value when a Configuration-state timeout occurs. Controls whether the associated non-critical fault channel (n) is enabled. When a non-critical fault channel is enabled, a fault on that channel causes FCCU to leave the NORMAL state and enter the FAULT state (or ALARM state if enabled for the channel). For information on how to configure the non-critical fault channels, see Configure the non-critical fault channels. 0b - Disabled 1b - Enabled

52.7.1.10 Non-critical-Fault Alarm-State Timeout Enable (NCF_TOE0)

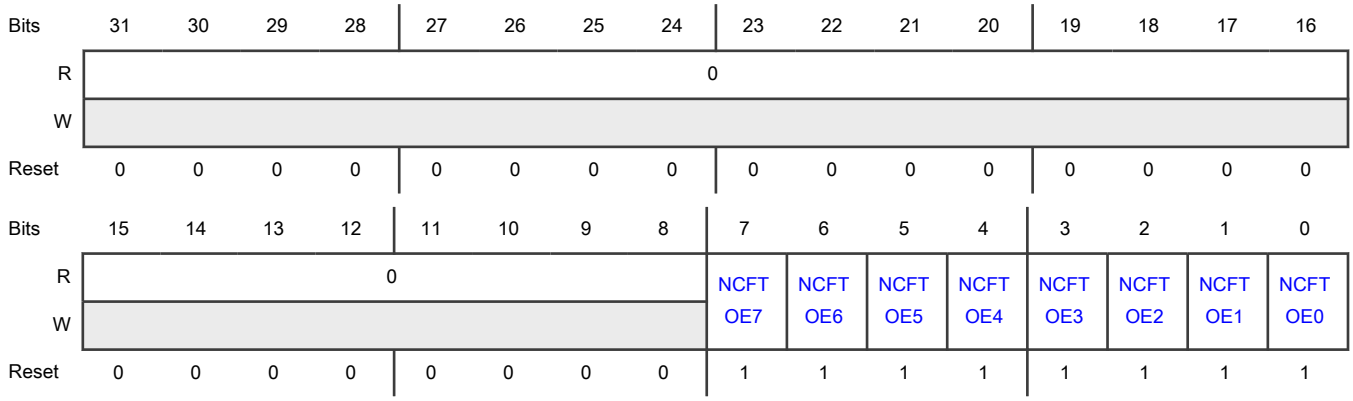
Offset

Register	Offset
NCF_TOE0	A4h

Function

See NCF_TOEa[NCFTOEn].

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 NCFTOEn	Non-critical-Fault Alarm-State Timeout Enable n Writable only when FCCU is in the CONFIG state. Changed by FCCU to its reset value when a Configuration-state timeout occurs. Controls whether the ALARM state is enabled for the associated non-critical fault channel (n). When the ALARM state is enabled for an enabled non-critical fault channel, a fault on that channel causes FCCU to leave the NORMAL state and enter the ALARM state instead of FAULT state. If the fault is not recovered within the Alarm-state timeout interval, then FCCU leaves the ALARM state and enters the FAULT state. For information on how to configure the non-critical fault channels, see Configure the non-critical fault channels. 0b - Disabled 1b - Enabled

52.7.1.11 Non-critical-Fault Alarm-State Timeout Interval (NCF_TO)

Offset

Register	Offset
NCF_TO	B4h

Function

See NCF_TO[TO]

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	TO															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	TO															
Reset	1	0	1	0	1	0	0	1	1	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TO	Non-critical-Fault Alarm-State Timeout Interval Writable only when FCCU is in the CONFIG state. Changed by FCCU to its reset value when a Configuration-state timeout occurs. Controls the maximum amount of time that FCCU can be in the ALARM state ($T_{Max\ Alarm}$) according to this equation: $T_{Max\ Alarm} = TO \times T_{CLKSAFE}$ where $T_{CLKSAFE}$ is the safe-clock period. If FCCU enters the ALARM state (because a fault occurs on an enabled non-critical fault channel for which the ALARM state is enabled) and this timeout interval expires (called an Alarm-state timeout), then FCCU leaves the ALARM state and enters the FAULT state. **NOTE** Make sure the Alarm-state timeout interval is less than the FOSU module's timeout interval; otherwise, a fault that occurs while FCCU is in the ALARM state can cause FOSU to generate a chip reset. The FOSU timeout interval (FOSU_COUNT) is chip-specific. See the chip-specific FCCU information.

52.7.1.12 Configuration-State Timeout Interval (CFG_TO)

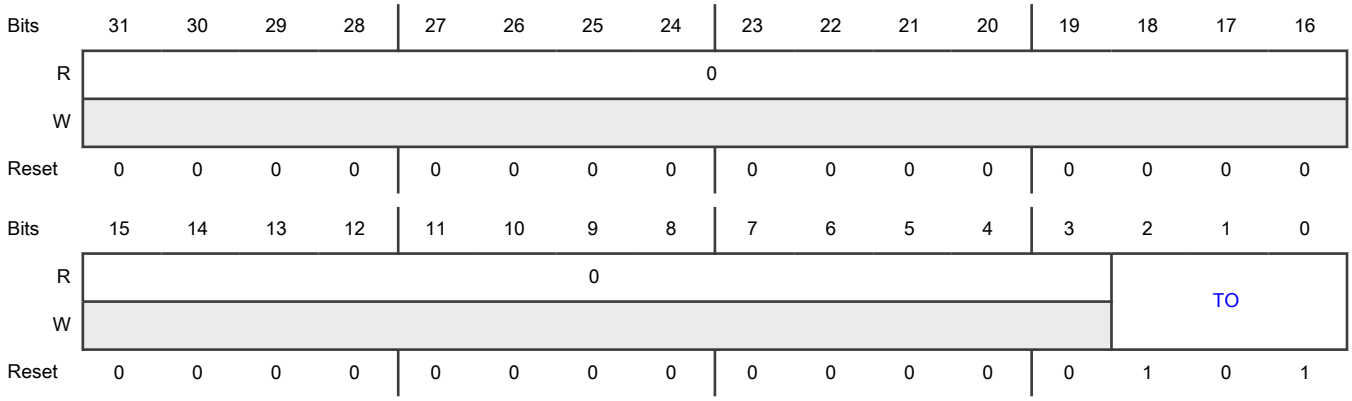
Offset

Register	Offset
CFG_TO	B8h

Function

See CFG_TO[TO]

Diagram



Fields

Field	Function
31-3 —	Reserved
2-0 TO	Configuration-State Timeout Interval Writable only when FCCU is in the NORMAL, ALARM, or FAULT state (not in the CONFIG state). Changed by FCCU to its reset value when a Configuration-state timeout occurs. Not accessible while a Configuration-state timeout (OP14 operation) is in progress. Controls the maximum amount of time that FCCU can be in the CONFIG state ($T_{\text{Max configuration}}$) according to this equation: $T_{\text{Max configuration}} = T_{\text{CLKSAFE}} \times 2^{(\text{TO} + 13)}$ where T_{CLKSAFE} is the safe-clock period. If you put FCCU in the CONFIG state and this timeout interval expires (called a Configuration-state timeout), then FCCU: • Automatically leaves the CONFIG state and enters the NORMAL state • Changes the value of the Configuration (CFG) register to its Configuration-state-timeout value and the value of each of the other configuration registers to its reset value. For information on the Configuration (CFG) register's Configuration-state-timeout value, see CFG register bit value at different events. For a list of configuration registers, see Configuration registers.

52.7.1.13 IO Control (EINOUT)

Offset

Register	Offset
EINOUT	BCh

Function

The EINOUT register allows the following operations typically in the NORMAL state:

- To control the EOUT[1] output level when the FCCU is configured in "Test1" or "Test0" fault output mode (CFG[FOM])

- To control the EOUT[0] output level when the FCCU is configured in "Test1" or "Test2" fault output mode (CFG[FOM])
- to observe the state of signals at EIN[1:0] pins

The following table shows Bi-Stable encoding.

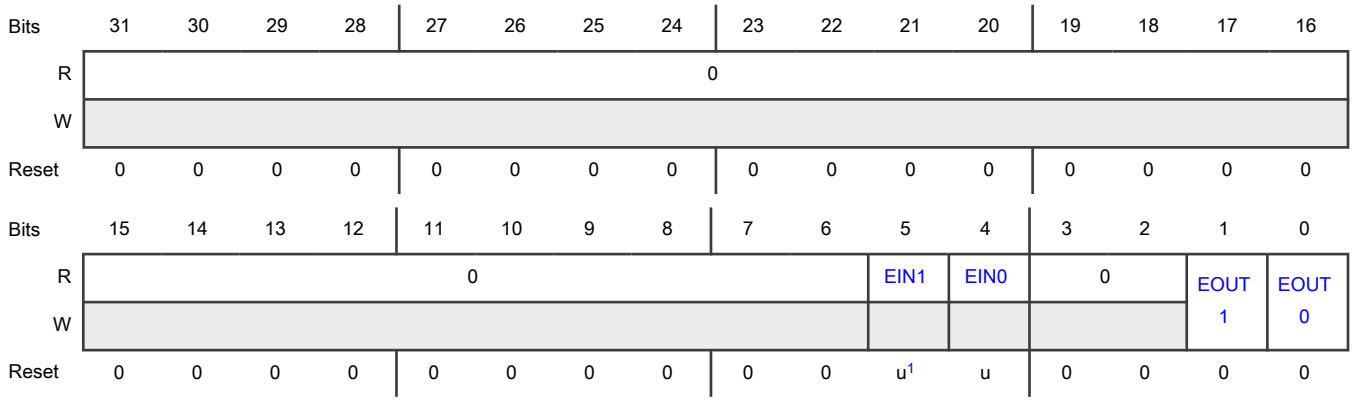
Table 300. Bi-Stable encoding

Mode = CFG[FOM]	EOUT[0]	EOUT[1]
Test1	output	output
Test2	output	input
Test0	input	output

NOTE

Because of the resynchronization stage of the EOUT interface, there is a latency of a few CLKSAFE cycles following a write/read operation of the EINOUT register.

Diagram



1. Reset value varies as per the corresponding EIN signal.

Fields

Field	Function
31-6 —	Reserved
5 EIN1	Error Input 1 Indicates the state of the EIN1 signal. **NOTE** When IP's SET_AFTER_RESET bit is 0, then corresponding pad is in High-Z state, in this case values sampled on EIN1 will depend on board/pad pull up/down values. 0b - Low 1b - High

Table continues on the next page...

Table continued from the previous page...

Field	Function
4 EIN0	Error Input 0 Indicates the state of the EIN0 signal. **NOTE** When IP's SET_AFTER_RESET bit is 0, then corresponding pad is in High-Z state, in this case values sampled on EIN0 will depend on board/pad pull up/down values. 0b - Low 1b - High
3-2 —	Reserved
1 EOUT1	EOUT1 Error out 1 (significant only if the CFG.FOM = Test1 or Test0 => EOUT[1] configured in output mode). The EOUT1 set/clear the respective EOUT[1] output signal if CFG.FOM = 110 or 101, otherwise it is a "don't-care" value. **NOTE** When the configuration watchdog timer expires, FCCU changes the value of this field to its reset value. 0b - force EOUT[1] = 0 1b - force EOUT[1] = 1
0 EOUT0	EOUT0 Error out 0 (significant only if the CFG.FOM = Test1 or Test2 => EOUT[0] configured in output mode). The EOUT0 set/clear the respective EOUT[0] output signal if CFG.FOM = 110 or 111, otherwise it is a "don't care" value. **NOTE** When the configuration watchdog timer expires, FCCU changes the value of this field to its reset value. 0b - force EOUT[0] = 0 1b - force EOUT[0] = 1

52.7.1.14 Status (STAT)

Offset

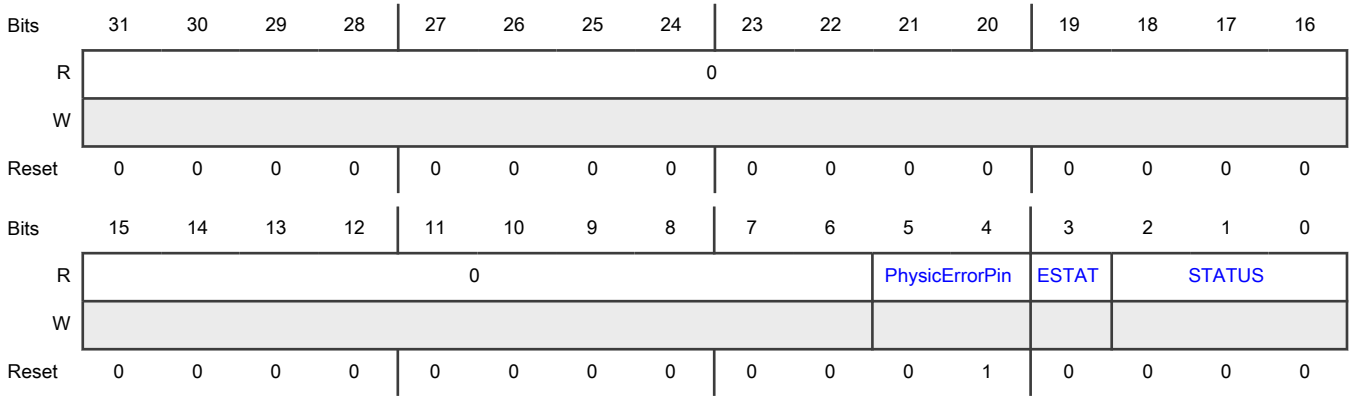
Register	Offset
STAT	C0h

Function

This register indicates the following:

- States that FCCU is driving on the EOUT signals
- Whether FCCU is in a faulty condition
- Current state of FCCU

Diagram



Fields

Field	Function
31-6 —	Reserved
5-4 PhysicErrorPin	EOUT Signal States Applies only when the EOUT signals are active (CFG[FCCU_SET_AFTER_RESET]). Indicates the states that FCCU is driving on the EOUT signals. 00b - EOUT1 is low; EOUT0 is low. 01b - EOUT1 is low; EOUT0 is high. 10b - EOUT1 is high; EOUT0 is low. 11b - EOUT1 is high; EOUT0 is high.
3 ESTAT	FCCU Faulty Condition Indicates whether FCCU is in faulty condition (as indicated by the EOUT signals). For more information, see The FCCU conditions. 0b - Not in faulty condition (in non-faulty or configuration condition) 1b - In faulty condition
2-0 STATUS	FCCU State Indicates the current state of FCCU 000b - NORMAL

Table continues on the next page...

Table continued from the previous page...

Field	Function
	001b - CONFIG 010b - ALARM 011b - FAULT 100b - Reserved 101b - Reserved 110b - Reserved 111b - Reserved

52.7.1.15 Normal-to-Alarm Freeze Status (N2AF_STATUS)

Offset

Register	Offset
N2AF_STATUS	C4h

Function

See N2AF_STATUS[NAFS].

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								NAFS							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-8 —	Reserved
7-0	Normal-to-Alarm Freeze Status

Table continues on the next page...

Table continued from the previous page...

Field	Function
NAFS	Used only for testing and debugging. Indicates whether FCCU left the NORMAL state and entered the ALARM state since the last time this register was cleared and, if so, which non-critical fault caused FCCU to do so. **NOTE** To clear this register and the other freeze status registers, see Clear the freeze-status indicators. 00h: No Normal-to-Alarm-state transition (cleared) 01h: NCF0 10h: NCF1 ... 7Fh: NCF126 80h: NCF127 ... FFh: Multiple Normal-to-Alarm-state transitions

52.7.1.16 Alarm-to-Fault Freeze Status (A2FF_STATUS)

Offset

Register	Offset
A2FF_STATUS	C8h

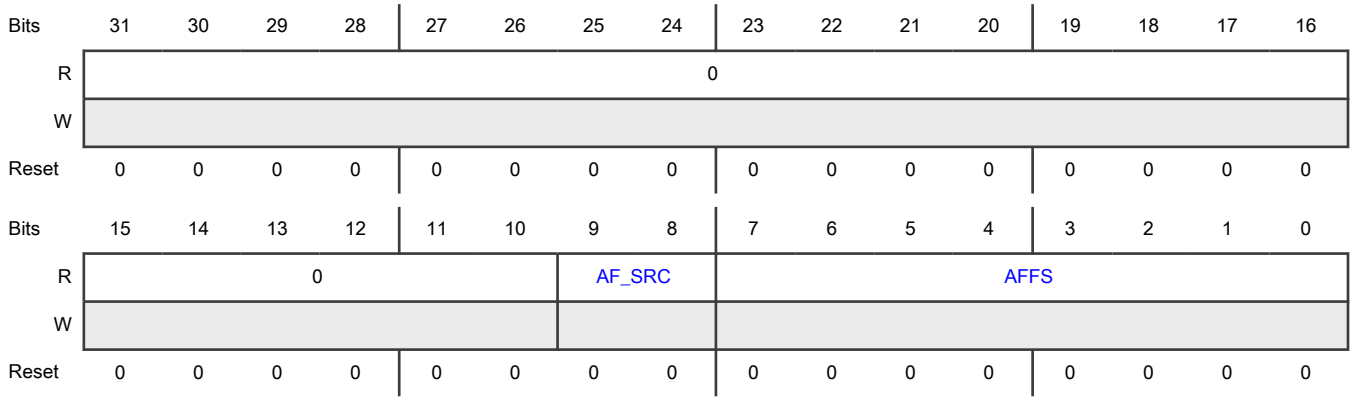
Function

Used only for testing and debugging. Indicates whether FCCU left the ALARM state and entered the FAULT state since the last time this register was cleared and, if so, which type of fault caused FCCU to do so.

NOTE

To clear this register and the other freeze status registers, see Clear the freeze-status indicators.

Diagram



Fields

Field	Function
31-10 —	Reserved
9-8 AF_SRC	Alarm-to-Fault Source Used only for testing and debugging. Indicates the type of fault that caused FCCU to leave the ALARM state and enter the FAULT state since the last time this register was cleared. **NOTE** To clear this register and the other freeze status registers, see Clear the freeze-status indicators. 00b - No Alarm-to-Fault-state fault 01b - Reserved 10b - Non-critical fault 11b - Multiple Alarm-to-Fault-state faults
7-0 AFFS	Alarm-to-Fault Freeze Status Used only for testing and debugging. Indicates whether FCCU left the ALARM state and entered the FAULT state since the last time this register was cleared and, if so, which fault caused FCCU to do so. **NOTE** To clear this register and the other freeze status registers, see Clear the freeze-status indicators. 00h: No Alarm-to-Fault-state transition (cleared) 01h: NCF0 (due to an Alarm-state timeout) 10h: NCF1 (due to an Alarm-state timeout) ... 7Fh: NCF126 (due to an Alarm-state timeout)

Table continues on the next page...

Table continued from the previous page...

Field	Function
	80h: NCF127 (due to an Alarm-state timeout) ... FFh: Multiple Alarm-to-Fault-state transitions

52.7.1.17 Normal-to-Fault Freeze Status (N2FF_STATUS)

Offset

Register	Offset
N2FF_STATUS	CCh

Function

Used only for testing and debugging. Indicates whether FCCU left the NORMAL state and entered the FAULT state since the last time this register was cleared and, if so, which type of fault caused FCCU to do so.

NOTE

To clear this register and the other freeze status registers, see Clear the freeze-status indicators.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0						NF_SRC		NFFS							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-10 —	Reserved
9-8 NF_SRC	Normal-to-Fault Source Used only for testing and debugging. Indicates the type of fault that caused FCCU to leave the NORMAL state and enter the FAULT state since the last time this register was cleared.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE To clear this register and the other freeze status registers, see Clear the freeze-status indicators. 00b - No Normal-to-Fault-state fault 01b - Reserved 10b - Non-critical fault 11b - Multiple Normal-to-Fault-state faults
7-0 NFFS	Normal-to-Fault Freeze Status Used only for testing and debugging. Indicates whether FCCU left the NORMAL state and entered the FAULT state since the last time this register was cleared and, if so, which fault caused FCCU to do so. **NOTE** To clear this register and the other freeze status registers, see Clear the freeze-status indicators. 00h: No Normal-to-Fault-state transition (cleared) 01h: NCF0 10h: NCF1 ... 7Fh: NCF126 80h: NCF127 ... FFh: Multiple Normal-to-Fault-state transitions

52.7.1.18 Fault-to-Alarm Freeze Status (F2AF_STATUS)

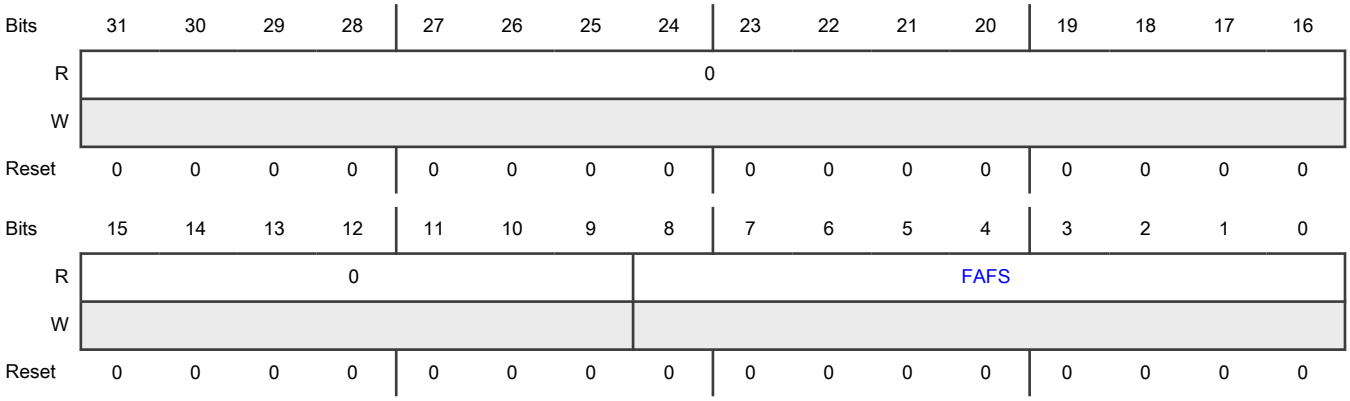
Offset

Register	Offset
F2AF_STATUS	D0h

Function

See F2AF_STATUS[FAFS].

Diagram



Fields

Field	Function
31-9 —	Reserved
8-0 FAFS	Fault-to-Alarm Freeze Status Used only for testing and debugging. Indicates whether FCCU left the FAULT state and entered the ALARM state since the last time this register was cleared and, if so, which non-critical fault caused FCCU to do so. **NOTE** To clear this register and the other freeze status registers, see Clear the freeze-status indicators. 00h: No Fault-to-Alarm-state transition (cleared) 01h: NCF0 10h: NCF1 ... 7Fh: NCF126 80h: NCF127 ... FFh: Multiple Fault-to-Alarm-state transitions

52.7.1.19 Non-critical Fault Fake (NCFF)

Offset

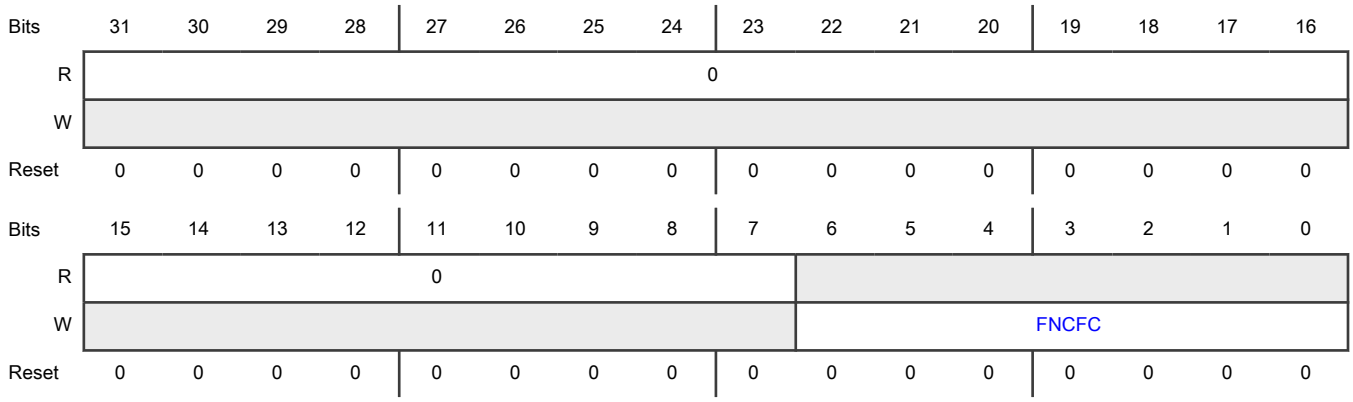
Register	Offset
NCFF	DCh

Function

This register contains a unique code to set a non-critical fault in mutually exclusive mode by the external FAULT interface (signal setting). It allows the SW emulation of the non-critical faults, by injecting the fault directly in the FAULT root, to verify the entire path and reaction. The reaction following a fake non-critical fault cannot be masked.

This is a write-only register with a set of codes corresponding to each non-critical fault injection.

Diagram



Fields

Field	Function
31-7 —	Reserved
6-0 FNCFC	FNCFC Fake non-critical fault code **NOTE** Writing to this field injects fake faults; writing 00 and the default value being 0 renders different results. 00h: Fake non-critical fault injection at non-critical fault source 0 01h: Fake non-critical fault injection at non-critical fault source 1 02h: Fake non-critical fault injection at non-critical fault source 2 ... 7Fh: Fake non-critical fault injection at non-critical fault source 127

52.7.1.20 IRQ Status (IRQ_STAT)

Offset

Register	Offset
IRQ_STAT	E0h

Function

This register provides the FCCU interrupt status related to the following events:

- Configuration-state timeout error
- Alarm interrupt
- NMI interrupt

The configuration-state timeout interrupt is asserted if both IRQ_STAT[CFG_TO_STAT] and IRQ_EN[CFG_TO_IEN] bits are asserted. It is cleared when a 1 is written to the IRQ_STAT[CFG_TO_STAT] bit.

The NMI and ALARM interrupts are asserted and cleared according to the FCCU state. The status bits of the IRQ_STAT trace the status of the related interrupt lines.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0											Reserved	Reserved	NMI_STAT	ALRM_ST...	CFG_TO_...
W																W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-5 —	Reserved
4 —	Reserved
3 —	Reserved
2 NMI_STAT	NMI Interrupt Status 0b - NMI interrupt is OFF 1b - NMI interrupt is ON
1 ALRM_STAT	Alarm Interrupt Status 0b - Alarm interrupt is OFF 1b - Alarm interrupt is ON
0	Configuration-State Timeout Status

Table continues on the next page...

Table continued from the previous page...

Field	Function
CFG_TO_STAT	0b - No configuration-stat timeout error 1b - Configuration-state timeout error

52.7.1.21 IRQ Enable (IRQ_EN)

Offset

Register	Offset
IRQ_EN	E4h

Function

This register is used to configure enabling of interrupt related to the "Configuration-state timeout error".

The configuration-state timeout interrupt is asserted if both the IRQ_STAT[CFG_TO_STAT] and IRQ_EN[CFG_TO_IEN] fields are set to 1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	0		CFG_TO_...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-4 —	Reserved
3 —	Reserved
2-1 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 CFG_TO_IEN	Configuration-State Timeout Interrupt Enable 0b - Configuration-state timeout interrupt disabled 1b - Configuration-state timeout interrupt enabled

52.7.1.22 Transient Configuration Lock (TRANS_LOCK)

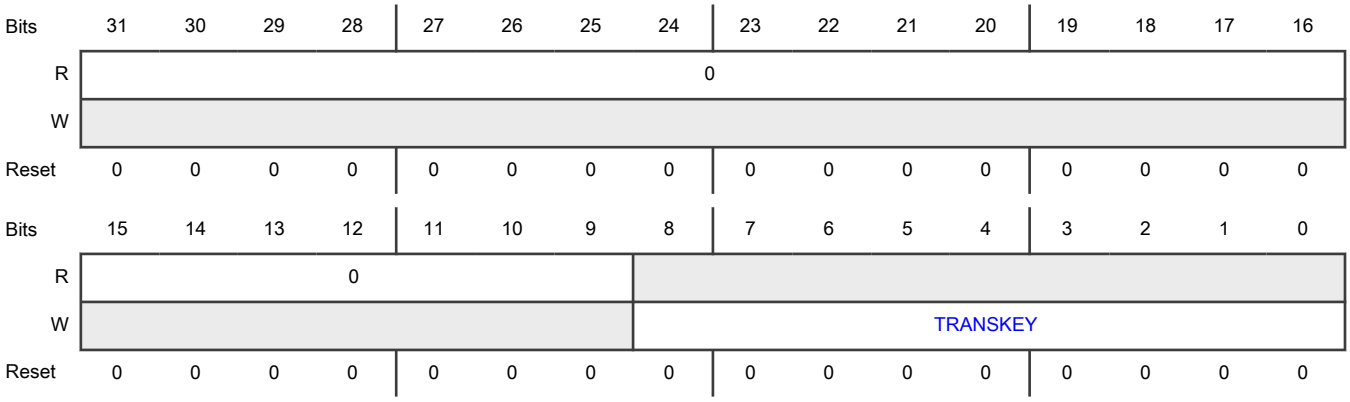
Offset

Register	Offset
TRANS_LOCK	F0h

Function

See TRANS_LOCK[TRANSKEY]

Diagram



Fields

Field	Function
31-9 —	Reserved
8-0 TRANSKEY	Transient Configuration Lock Writable only by code running in Supervisor mode. Temporarily locks and unlocks the configuration. Locking the configuration prevents FCCU from entering the CONFIG state. For information about putting FCCU in configuration, see Prepare FCCU for configuration and Configure FCCU.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE You can write to this field when FCCU is in any state, but the lock will not get into effect until FCCU is in the NORMAL state. BCh: Unlock. Any other value: Lock.

52.7.1.23 Permanent Configuration Lock (PERMNT_LOCK)

Offset

Register	Offset
PERMNT_LOCK	F4h

Function

See PERMNT_LOCK[PERMNTKEY]

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W								PERMNTKEY								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-9 —	Reserved
8-0 PERMNTKEY	Permanent Configuration Lock Writable only by code running in the Supervisor mode. Permanently locks the configuration, which prevents FCCU from entering the CONFIG state until FCCU is reset. For information about putting FCCU in configuration, see Prepare FCCU for configuration and Configure FCCU.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE You can write to this field when FCCU is in any state, but the lock will not get into effect until FCCU is in NORMAL state. FFh: Lock. Any other value: Do nothing.

52.7.1.24 Delta T (DELTA_T)

Offset

Register	Offset
DELTA_T	F8h

Function

The DELTA_T register is used for programming the value of delta_T constant (see DELTA_T), in microseconds.

NOTE
This register can be written only when the FCCU is in the CONFIG state.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved		Reserved													
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved		DELTA_T													
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29-16	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
15-14 —	Reserved
13-0 DELTA_T	Minimum Fault-Output (EOUT) Timer Interval Applies only to Bi-Stable and Fault-Toggle fault-output modes (CFG[FOM]). Controls the minimum amount of time (T_{min}) that the fault-output (EOUT) timer runs according to this equation: T_{min} = (250+ DELTA_T) µs * (48000/CLKSAFE freq KHz) **NOTE** The durations shown for the DELTA_T values depend on CLKSAFE signals. For frequency of CLKSAFE signals see the chip-specific FCCU information). Also see chip's data sheet for the trimmed frequency variation (for example, δF_{var}).

52.7.1.25 Non-critical Alarm-State Interrupt-Request Enable (IRQ_ALARM_EN0)

Offset

Register	Offset
IRQ_ALARM_EN0	FCh

Function

See IRQ_ALARM_ENa[IRQENn].

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								IRQEN 7	IRQEN 6	IRQEN 5	IRQEN 4	IRQEN 3	IRQEN 2	IRQEN 1	IRQEN 0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-8 —	Reserved
7-0 IRQENn	Non-critical Alarm-State Interrupt-Request Enable n Writable only when FCCU is in the CONFIG state. Changed by FCCU to its reset value when a Configuration-state timeout occurs. Controls whether the interrupt request is enabled as the Alarm-state reaction for the associated non-critical fault channel (n). When the ALARM state and the Alarm-state interrupt request are enabled for an enabled non-critical fault channel, a fault on that channel causes FCCU to assert the irq_alarm signal when FCCU enters the ALARM state; irq_alarm remains asserted until FCCU is in the NORMAL state. For information on how to configure the non-critical fault channels, see Configure the non-critical fault channels. 0b - Disabled 1b - Enabled

52.7.1.26 Non-critical Fault-State Non-maskable-Interrupt-Request Enable (NMI_EN0)

Offset

Register	Offset
NMI_EN0	10Ch

Function

See NMI_ENa[NMIENn].

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								NMIEN 7	NMIEN 6	NMIEN 5	NMIEN 4	NMIEN 3	NMIEN 2	NMIEN 1	NMIEN 0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-8 —	Reserved
7-0 NMIENn	Non-critical Fault-State Non-maskable-Interrupt-Request Enable n Writable only when FCCU is in the CONFIG state. Changed by FCCU to its reset value when a Configuration-state timeout occurs. Controls whether the non-maskable interrupt request is enabled as a Fault-state reaction for the associated non-critical fault channel (n). When the non-maskable interrupt request is enabled for an enabled non-critical fault channel, a fault on that channel causes FCCU to assert the NMIOUT signal when FCCU enters the FAULT state; NMIOUT remains asserted until FCCU exits FAULT state. For information on how to configure the non-critical fault channels, see Configure the non-critical fault channels. 0b - Disabled 1b - Enabled

52.7.1.27 Non-critical Fault-State EOUT Signaling Enable (EOUT_SIG_EN0)

Offset

Register	Offset
EOUT_SIG_EN0	11Ch

Function

See EOUT_SIG_ENa[EOUTENn].

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								EOUT EN7	EOUT EN6	EOUT EN5	EOUT EN4	EOUT EN3	EOUT EN2	EOUT EN1	EOUT EN0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-8 —	Reserved
7-0 EOUTENn	Non-critical Fault-State EOUT Signaling Enable n Writable only when FCCU is in the CONFIG state. Changed by FCCU to its reset value when a Configuration-state timeout occurs. Applies only when the EOUT signals are active (CFG[FCCU_SET_AFTER_RESET]). When FCCU is configured for Bi-Stable fault-output mode (CFG[FOM]), controls whether EOUT signaling is enabled as a Fault-state reaction for the associated non-critical fault channel (n). (For other fault-output modes, EOUT signaling is always enabled, regardless of the value of this field.) When EOUT signaling is enabled for an enabled non-critical fault channel, a fault on that channel causes FCCU to indicate the faulty condition on the EOUT[1:0] signals when FCCU enters the FAULT state. For information on how to configure the non-critical fault channels, see Configure the non-critical fault channels. For all fault-output modes, also controls whether FCCU asserts the FIF signal when a fault on the associated non-critical fault channel (n) causes FCCU to enter the FAULT state. **NOTE** For all fault-output modes, you must set this field to enabled to ensure that FCCU asserts the FIF signal when FCCU enters FAULT state as the result of a fault on the associated non-critical fault channel (n) so the FOSU module does not mistakenly generate a destructive chip reset. 0b - In Bi-Stable fault-output mode, both EOUT signaling and FIF assertion are disabled; in other fault-output modes, EOUT signaling is enabled and FIF assertion is disabled. 1b - Both EOUT signaling and FIF assertion are enabled in all fault-output modes.

52.7.1.28 Alarm-State Timer (TMR_ALARM)

Offset

Register	Offset
TMR_ALARM	12Ch

Function

See TMR_ALARM[COUNT].

This table shows how the Alarm-state timer's state and value vary by FCCU state:

Table 301. TMR_ALARM reset value

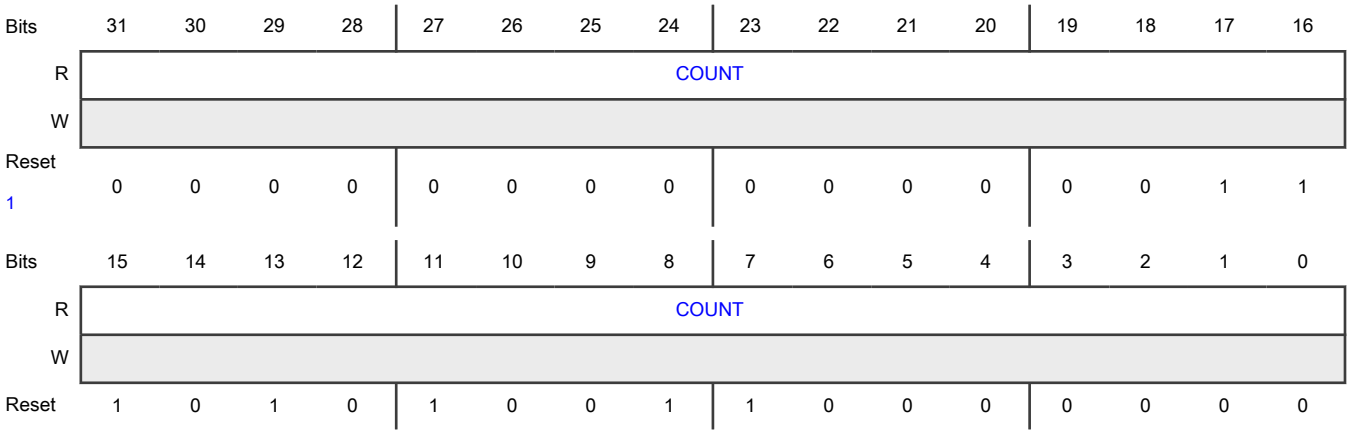
FCCU state	Timer state	Timer value
CONFIG	Idle	0000_0000h
NORMAL	Idle	Initial value: NCF_TO[TO]

Table continues on the next page...

Table 301. TMR_ALARM reset value (continued)

FCCU state	Timer state	Timer value
ALARM	Running	Value when read
FAULT	Idle	End of count

Diagram



1. The default reset value is provided by NCF_TO[TO]. See Table 301 for the reset value at different FCCU states.

Fields

Field	Function
31-0 COUNT	Alarm-State Timer Count Specifies the value of the Alarm-state timer in CLKSAFE periods.

52.7.1.29 Configuration-State Timer (TMR_CFG)

Offset

Register	Offset
TMR_CFG	134h

Function

See TMR_CFG[COUNT].

This table shows how the Configuration-state timer's state and value vary by FCCU state:

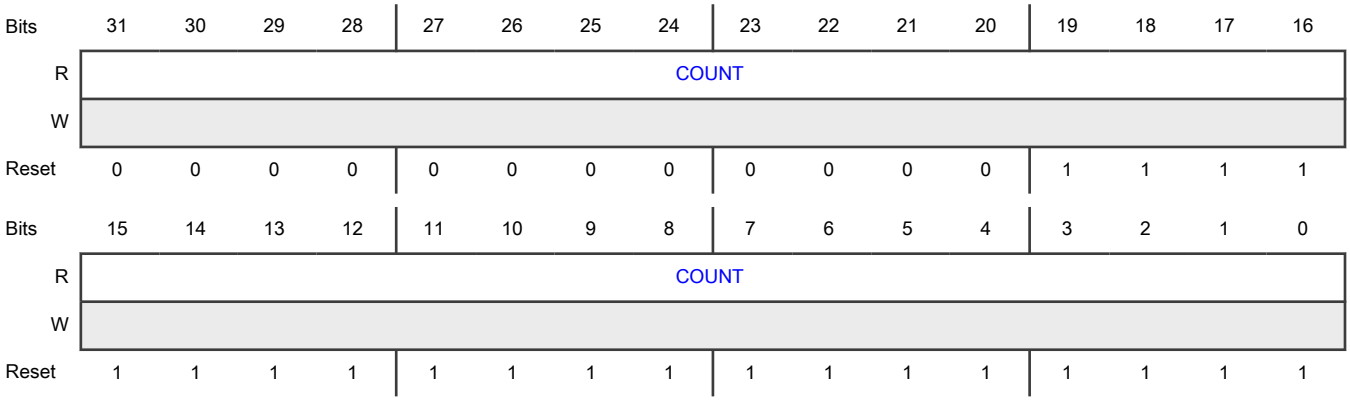
FCCU state	Timer state	Timer value
CONFIG	Running	Value when read

Table continues on the next page...

Table continued from the previous page...

FCCU state	Timer state	Timer value
NORMAL	Idle	000F_FFFFh
ALARM	Idle	000F_FFFFh
FAULT	Idle	000F_FFFFh

Diagram



Fields

Field	Function
31-0 COUNT	Configuration-State Timer Count Specifies the value of the Configuration-state timer in CLKSAFE periods.

52.7.1.30 Fault-Output Timer (TMR_ETMR)

Offset

Register	Offset
TMR_ETMR	138h

Function

See TMR_ETMR[COUNT].

This table shows how the fault-output timer's state and value vary by FCCU state and fault-output mode:

FCCU state	Timer state (value)
CONFIG	Not Fault-Toggle: Idle (0000_0000h)

Table continues on the next page...

Table continued from the previous page...

FCCU state	Timer state (value)
	Fault-Toggle: Running (value when read) or idle (0000_0000h)
NORMAL	Not Fault-Toggle: Idle (0000_0000h) Fault-Toggle: Running (value when read) or idle (0000_0000h)
ALARM	Not Fault-Toggle: Idle (0000_0000h) Fault-Toggle: Running (value when read) or idle (0000_0000h)
FAULT	Not Fault-Toggle: Running (value when read. It is an up-counter which rollbacks to zero after reaching maximum value and then it begins counting again.) Fault-Toggle: Running (value when read) or idle (0000_0000h)

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	COUNT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COUNT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 COUNT	Fault-Output Timer Count Specifies the value of the fault-output timer in CLKSAFE periods.

52.7.2 Configuration registers

52.7.2.1 Definition

Configuration registers are registers that:

- Let you configure FCCU's Alarm-state timer interval, fault channels, and fault-output (EOUT) signals.
- You can write to only when the configuration is not locked and FCCU is in CONFIG state (see Put FCCU in the CONFIG state).
- Save the values you write to them while FCCU is in CONFIG state only after you manually put FCCU in NORMAL state. If FCCU automatically leaves CONFIG state and enters NORMAL state because the configuration-timer interval (CFG_TO[TO]) expires (called a Configuration-state timeout), FCCU changes the value of the Configuration (CFG) register to its Configuration-state-timeout value and the value of each of the other configuration registers to its reset value; FCCU also

changes the value of the Configuration-State Timeout Interval (CFG_TO) register to its reset value. For information on the Configuration-state timeout value, see CFG register bit value at different events.

52.7.2.2 Configuration registers

Following is the list of configuration registers in this module. They are listed in an offset order from lowest to highest.

- Configuration (CFG)
- Non-critical Fault Configuration (NCF_CFGa)
- Non-critical Fault-State Configuration (NCFS_CFG0)
- Non-critical Fault Enable (NCF_E0)
- Non-critical-Fault Alarm-State Timeout Enable (NCF_TOE0)
- Non-critical-Fault Alarm-State Timeout Interval (NCF_TO)
- Delta T (DELTA_T)
- Non-critical Alarm-State Interrupt-Request Enable (IRQ_ALARM_EN0)
- Non-critical Fault-State Non-maskable-Interrupt-Request Enable (NMI_EN0)
- Non-critical Fault-State EOUT Signaling Enable (EOUT_SIG_EN0)

52.7.3 CFG register bit value at different events

In this chip, there are no events that affect the bits in the configuration register.

52.8 Glossary

CF	Critical fault
EOUT	Error out
FOSU	FCCU output supervision unit
FSM	Finite state machine
intf	Interface
IRQ	Interrupt request
NCF	Non-critical fault
NMI	Non-maskable interrupt

Chapter 53
Self-test General-Purpose Registers (SELFTEST_GPR)

53.1 Introduction

SELFTEST_GPR configures the STCU2 LBIST shift cycles. Programming self-test GPRs is a part of the self-test programming sequence. See the "Self-test programming sequence" section in the STCU2 chapter.

NXP supports only STCU2 Built-In Self-Test (BIST) sequences that have been validated by NXP for in-field testing usage.

The detailed programming sequences for supported configurations are available in a separate application note (contact your NXP sales representative).

The values for SELFTEST_GPR registers must be aligned with the NXP supported configurations only.

NOTE

SELFTEST_GPR is not present in S32K312 and S32K311.

NOTE

While accessing SELFTEST_GPR space, software must ensure that the STCU2 block has its clock enabled, i.e., MC_ME.PRTN1_COFB3_CLKEN[REQ104] configured as 1'b1. Not ensuring this might result in unpredictable device behaviour.

53.2 Peripheral shift clock switching (PCS)

The PCS feature is used to avoid current surges at the start and/or end of LBIST. Since LBIST can cause sudden current surges which might lead to chip malfunction.

With PCS enabled, there are additional (SELFTEST_GPR.CONFIG_GPR[PCS_STEP_SIZE] + 1)*16 patterns beyond the configured count. If enabled at the start of LBIST (using SELFTEST_GPR.CONFIG_GPR[PCS_ENABLE_START]), it would cause the shift clock to scale up in this duration from divide by 16 to the original frequency (by changing divider in each step).

Similarly, if configured at the end of LBIST (using SELFTEST_GPR.CONFIG_GPR[PCS_ENABLE_END]), the shift clock frequency will be scaled down from original frequency to divide by 16.

53.3 SELFTEST_GPR register descriptions

53.3.1 SELFTEST_GPR memory map

SELFTEST_GPR base address: 403B_0000h

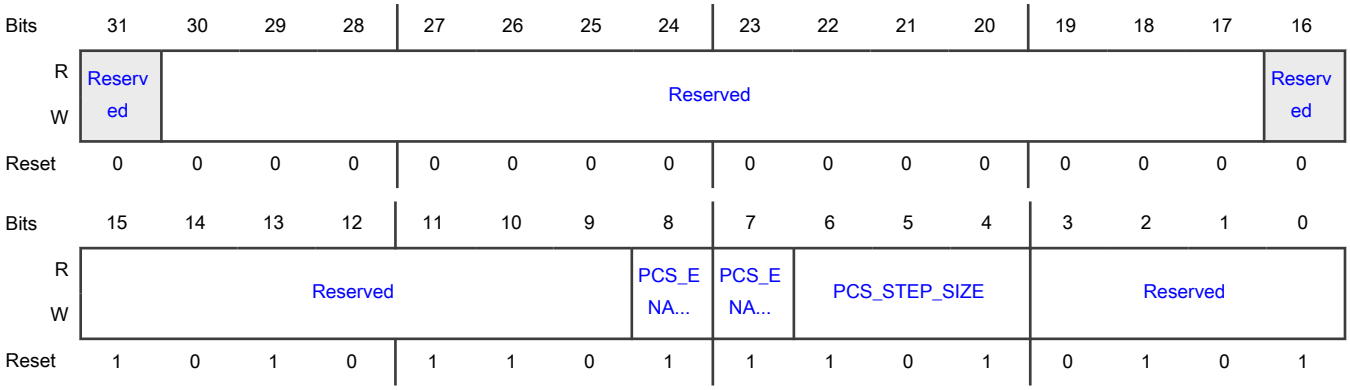
Offset	Register	Width (In bits)	Access	Reset value
0h	Configuration register (CONFIG_REG)	32	RW	0000_ADD5h
14h	LBIST Program (LBIST_PROG_REG)	32	RW	0003_0050h

53.3.2 Configuration register (CONFIG_REG)

Offset

Register	Offset
CONFIG_REG	0h

Diagram



Fields

Field	Function
31 —	Reserved
30-17 —	Reserved
16 —	Reserved
15-9 —	Reserved
8 PCS_ENABLE_END	PCS Enable End Enables Progressive Shift Clock Switching for LBIST at end. The feature gets enabled when this bit is set to 1. It is recommended to run LBIST patterns by enabling this feature.
7 PCS_ENABLE_START	PCS Enable Start Enables Progressive Shift Clock Switching for LBIST at start. The feature gets enabled when this bit is set to 1. It is recommended to run LBIST patterns by enabling this feature.
6-4	PCS Step Size pcs_step_size[2:0] controls the step size for progressive increase of shift frequency:

Table continues on the next page...

Table continued from the previous page...

Field	Function
PCS_STEP_SIZE	000 - step size of 1 pattern 001 - step size of 2 patterns ... 111 - step size of 8 patterns
3-0 —	Reserved

53.3.3 LBIST Program (LBIST_PROG_REG)

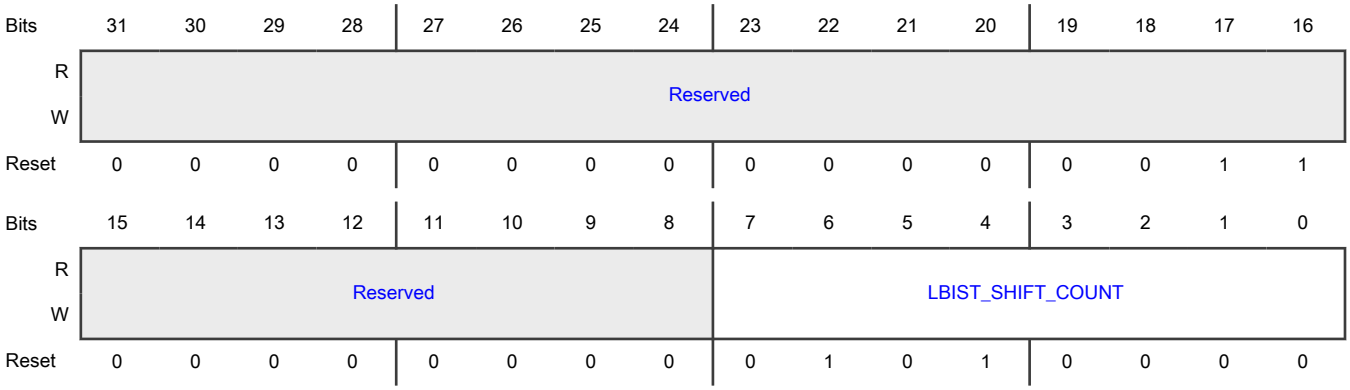
Offset

Register	Offset
LBIST_PROG_REG	14h

Function

Configures the LBIST shift count.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 LBIST_SHIFT_COUNT	LBIST Shift Count Specifies the number of shift cycles in a scan chain in one pattern count for the LBIST partition.

53.4 Glossary

BIST clock	The BIST clock refers to the BIST operational clock which is the clock source used while LBIST is operational by the LBIST controller, which is the MC_CGM clock node LBIST_CLK for the chip. See the system block diagram for details.
LBIST	Logic built-in self-test
Pattern count	The pattern count refers to an empirically determined value of shift patterns run in LBIST on a scan chain to result in predetermined fault coverage, based on device safety integrity level. Since the LBIST scheme is Pseudo Random Testing, a sufficiently good value of pattern count is used which provides the required fault coverage.
Scan chain	A sequential chain of flops in scan mode over which one scan pattern (generated by PRPG, that is, Pseudo Random Pattern Generator) in LBIST will be shifted and the output MISR (Multi Input Signature Register) will be matched with earlier known value to determine pass or fail criterion of that particular chain.

Chapter 54
Self-Test Control Unit (STCU2)

54.1 Chip-specific STCU2 information

54.1.1 Supported BIST sequences

This chip supports only STCU2 BIST sequences that NXP has validated for in-field testing usage. The detailed programming sequences for supported configurations are available in a separate application note (contact your NXP sales representative for this information).

54.1.2 Self-test overview

STCU2 provides the overall control of both LBIST and MBIST, in serial or parallel, depending on the power, timing, or coverage constraints.

NOTE

LBIST is not available for S32K312, S32K311, and S32K310.

The application software initiates the self-test sequence or procedure (please use whichever term is correct), and has provisions to bypass the self-test configuration to directly move to normal Run mode.

There is a programmable BIST-sequence-execution watchdog timer that specifies the maximum time allowed for the execution of a BIST sequence. In case the selected LBISTs or MBISTs are not complete during the assigned time, the current LBISTs or MBISTs execution is interrupted and a failure is flagged into ERR_STAT[WDTOSW] and MBESWx or LBESW.

The STCU2 execution includes three phases:

- Software configuration. Load or program the test configuration parameters to be executed, depending on ASIL target and performance requirements. The self-test configuration parameters allow the self-test sequence to run with different times depending on pattern numbers and types, achieving different levels of coverage.
- Start of BIST execution. This involves software register configuration in STCU2. The BIST completion is followed by a functional reset.
- End of BIST execution. After BIST execution and functional reset are complete, the BIST results are made available and you decide how to continue (trigger a new reset or start a safety application). The BIST execution completion is signalized in an internal STCU2 register (LBESWn) bit which is not affected by the functional reset. The BIST test results are also not affected by the functional reset.

The following figure shows a pictorial representation of two LBIST partitions having two and one MBIST groups, respectively. STCU2 interacts with LBIST_CTRL for the respective partitions to configure and control the LBIST sequence and compare results. It also interacts with MTR controller (MCT) for MBIST. Each LBIST partition has its corresponding controller whereas a single MCT can cater to multiple MBIST groups.

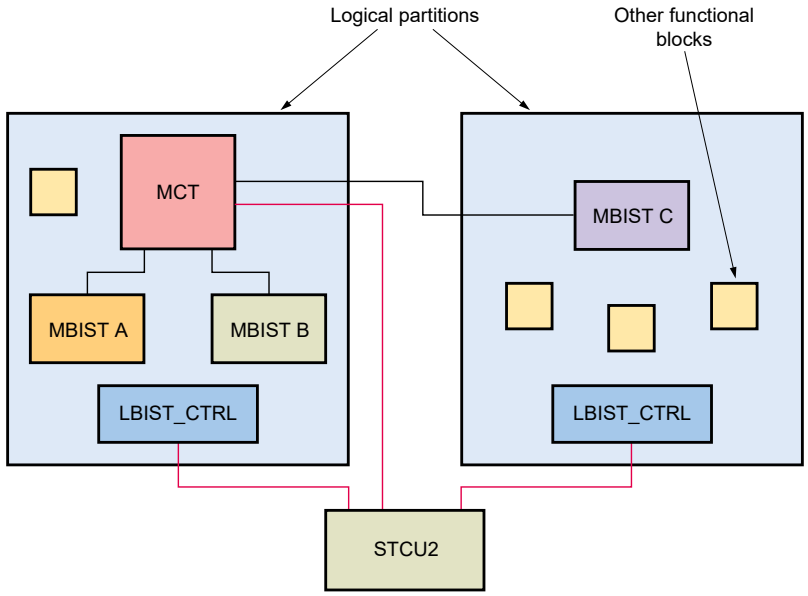


Figure 215. Self-test overview

NOTE

LBIST_CTRL is not available for S32K312, S232K311, and S32K310.

Each LBIST partition also includes a block called Self-test GPR involving a set of registers that provide miscellaneous configuration and parameter details for an LBIST operation.

When you trigger a self-test sequence, the chip becomes unavailable for software application during the self-test run.

54.1.3 STCU2 LBIST/MBIST mapping

This topic covers STCU2 LBIST/MBIST mapping.

LBIST mapping

The below table lists the LBIST mapping of the chip. The NLBIST value is 1.

NOTE

For module instance availability. See the respective chip-specific section for details.

Table 302. LBIST mapping

BIST ID (NLBIST)	BIST instance name	Modules/Instances
0	LBIST	EIM: • EIM_2 • EIM_3
		ERM: • ERM_0 • ERM_1

Table continues on the next page...

Table 302. LBIST mapping (continued)

BIST ID (NLBIST)	BIST instance name	Modules/Instances
		XRDC-MDAC: • MDAC0 • MDAC1 • MDAC4 • MDAC5 • MDAC6 • MDAC7 • MDAC8 • MDAC9 XRDC-MRC: • MRC0 • MRC1 • MRC2 • MRC3 • MRC4 • MRC5 XRDC-PDAC: • PDAC0 • PDAC1 • PDAC2
		CRC
		FCCU
		REG_PROT
		FOSU
		FAR_PFC: • FAR_PFC0 • FAR_PFC1
		PFLASH_CTL
		PFLASH_CTL1
		PRAM: • PRAMC_0 • PRAMC_1

Table continues on the next page...

Table 302. LBIST mapping (continued)

BIST ID (NLBIST)	BIST instance name	Modules/Instances
		• PRAMC_2 • PRAMC_3
		SWT: • SWT_1 • SWT_2 • SWT_3
		CMU: • CMU_FM_1 • CMU_FM_2 • CMU_FC_4 • CMU_FC_5 • CMU_FC_6
		iAHB Gaskets (1:1, 2:1): **NOTE** See the block diagrams in the "Introduction" chapter for details. • CM7_0 AHBM GSKT • CM7_0 AHBP GSKT • CM7_0 AHBS GSKT • CM7_1 AHBM GSKT • CM7_1 AHBP GSKT • CM7_1 AHBS GSKT • CM7_2 AHBM GSKT • CM7_2 AHBP GSKT • CM7_2 AHBS GSKT • CM7_3 AHBM GSKT • CM7_3 AHBP GSKT • CM7_3 AHBS GSKT • DMA_S0_GSKT • DMA_S1_GSKT • HSE_GSKT • ACE_GSKT • AIPS0_GSKT

Table continues on the next page...

Table 302. LBIST mapping (continued)

BIST ID (NLBIST)	BIST instance name	Modules/Instances
		• AIPS1_GSKT • AIPS2_GSKT • QSPI_GSKT • TCM_GSKT • EMAC_GSKT • GMAC_0_GSKT • GMAC_1_GSKT • EDC master checker gaskets • EDC slave checker gasket
		Crossbars • AXBS_0 (main) • AXBS_1 (peripheral) • AXBS_2 (eDMA) • AXBS_3 (Cortex-M7 TCM) • AXBS_4 (TCM PRAM) • AXBS_5 (ACE HSE_B AXBS) • AXBS_6 (ACE AXBS) • AIPS_Lite_0 • AIPS_Lite_1 • AIPS_Lite_2

MBIST mapping

The below tables lists the MBIST mapping of different variants of this chip.

Table 303. S32K314/S32K324/S32K344 MBIST mapping

BIST index (NMCUT)	BIST name	Memory index	Memory instance name
0	SYS0_RAMS	0	PRAM0 block 1
0	SYS0_RAMS	1	PRAM0 block 2
1	SYS1_RAMS	0	PRAM1 block 1
1	SYS1_RAMS	1	PRAM1 block 2
2	DMA_TCD_RAM	0	DMA TCD RAM
3	CM7_0_TOP	0	CM7_0 instruction cache data block 1
3	CM7_0_TOP	1	CM7_0 instruction cache data block 2

Table continues on the next page...

Table 303. S32K314/S32K324/S32K344 MBIST mapping (continued)

BIST index (NMCUT)	BIST name	Memory index	Memory instance name
3	CM7_0_TOP	2	CM7_0 instruction cache tag block 1
3	CM7_0_TOP	3	CM7_0 instruction cache tag block 2
3	CM7_0_TOP	4	CM7_0 data cache data block 1
3	CM7_0_TOP	4	CM7_0 data cache data block 2
3	CM7_0_TOP	5	CM7_0 data cache data block 3
3	CM7_0_TOP	5	CM7_0 data cache data block 4
3	CM7_0_TOP	6	CM7_0 data cache data block 5
3	CM7_0_TOP	6	CM7_0 data cache data block 6
3	CM7_0_TOP	7	CM7_0 data cache data block 7
3	CM7_0_TOP	7	CM7_0 data cache data block 8
3	CM7_0_TOP	8	CM7_0 data cache tag block 1
3	CM7_0_TOP	9	CM7_0 data cache tag block 2
3	CM7_0_TOP	10	CM7_0 data cache tag block 3
3	CM7_0_TOP	11	CM7_0 data cache tag block 4
3	CM7_0_TOP	12	CM7_0 instruction TCM (Tightly Coupled Memory)
3	CM7_0_TOP	13	CM7_0 data TCM block 1
3	CM7_0_TOP	13	CM7_0 data TCM block 2
4	CM7_1_TOP	0	CM7_1 instruction cache data block 1
4	CM7_1_TOP	1	CM7_1 instruction cache data block 2
4	CM7_1_TOP	2	CM7_1 instruction cache tag block 1
4	CM7_1_TOP	3	CM7_1 instruction cache tag block 2
4	CM7_1_TOP	4	CM7_1 data cache data block 1
4	CM7_1_TOP	4	CM7_1 data cache data block 2
4	CM7_1_TOP	5	CM7_1 data cache data block 3
4	CM7_1_TOP	5	CM7_1 data cache data block 4
4	CM7_1_TOP	6	CM7_1 data cache data block 5
4	CM7_1_TOP	6	CM7_1 data cache data block 6
4	CM7_1_TOP	7	CM7_1 data cache data block 7
4	CM7_1_TOP	7	CM7_1 data cache data block 8
4	CM7_1_TOP	8	CM7_0 data cache tag block 1
4	CM7_1_TOP	9	CM7_0 data cache tag block 2
4	CM7_1_TOP	10	CM7_0 data cache tag block 3

Table continues on the next page...

Table 303. S32K314/S32K324/S32K344 MBIST mapping (continued)

BIST index (NMCUT)	BIST name	Memory index	Memory instance name
4	CM7_1_TOP	11	CM7_0 data cache tag block 4
4	CM7_1_TOP	12	CM7_0 instruction TCM
4	CM7_1_TOP	13	CM7_0 data TCM block 1
4	CM7_1_TOP	13	CM7_0 data TCM block 2
5	FLEX_CAN_RAMS	0	FLEXCAN0 MB memory
5	FLEX_CAN_RAMS	1	FLEXCAN1 MB memory
5	FLEX_CAN_RAMS	2	FLEXCAN2 MB memory
5	FLEX_CAN_RAMS	3	FLEXCAN3 MB memory
5	FLEX_CAN_RAMS	4	FLEXCAN4 MB memory
5	FLEX_CAN_RAMS	5	FLEXCAN5 MB memory
6	QSPI_PERI_RAMS	0	QSPI RAM
6	QSPI_PERI_RAMS	1	QSPI TX memory
7	EMAC_TSN_RAM	0	EMAC Timestamp Memory
8	EMAC_RAMS	0	EMAC RXParser
8	EMAC_RAMS	1	EMAC TX block 1
8	EMAC_RAMS	2	EMAC TX block 1
8	EMAC_RAMS	3	EMAC TX block 2
8	EMAC_RAMS	4	EMAC RX block 2
9	b03_ETF_RAMS	0	HTM ETF
9	b03_ETF_RAMS	1	Shared ETF
9	b03_ETF_RAMS	2	CM7 cluster Instruction ETF
9	b03_ETF_RAMS	3	CM7 cluster Data ETF
10	HSE_RAMS	0	HSE secure RAM
10	HSE_RAMS	1	HSE PKC memory block 1
10	HSE_RAMS	2	HSE PKC memory block 2
10	HSE_RAMS	3	HSE MTB memory

Table 304. S32K342/S32K341/S32K322 MBIST mapping

BIST index (NMCUT)	BIST name	Memory index	Memory instance name
1	HSE_RAMS	0	HSE secure RAM
1	HSE_RAMS	1	HSE PKC memory block 1
1	HSE_RAMS	2	HSE PKC memory block 2

Table continues on the next page...

Table 304. S32K342/S32K341/S32K322 MBIST mapping (continued)

BIST index (NMCUT)	BIST name	Memory index	Memory instance name
1	HSE_RAMS	3	HSE MTB memory
2	SYS0_DMA_RAMS	0	PRAM0 block 1
2	SYS0_DMA_RAMS	1	PRAM0 block 2
2	SYS0_DMA_RAMS	2	DMA TCD RAM
3	CM7_0_TOP	0	CM7_0 instruction cache data block 1
3	CM7_0_TOP	1	CM7_0 instruction cache data block 2
3	CM7_0_TOP	2	CM7_0 instruction cache tag block 1
3	CM7_0_TOP	3	CM7_0 instruction cache tag block 2
3	CM7_0_TOP	4	CM7_0 data cache data block 1
3	CM7_0_TOP	4	CM7_0 data cache data block 2
3	CM7_0_TOP	5	CM7_0 data cache data block 3
3	CM7_0_TOP	5	CM7_0 data cache data block 4
3	CM7_0_TOP	6	CM7_0 data cache data block 5
3	CM7_0_TOP	6	CM7_0 data cache data block 6
3	CM7_0_TOP	7	CM7_0 data cache data block 7
3	CM7_0_TOP	7	CM7_0 data cache data block 8
3	CM7_0_TOP	8	CM7_0 data cache tag block 1
3	CM7_0_TOP	9	CM7_0 data cache tag block 2
3	CM7_0_TOP	10	CM7_0 data cache tag block 3
3	CM7_0_TOP	11	CM7_0 data cache tag block 4
3	CM7_0_TOP	12	CM7_0 instruction TCM (Tightly Coupled Memory)
3	CM7_0_TOP	13	CM7_0 data TCM block 1
3	CM7_0_TOP	13	CM7_0 data TCM block 2
4	FLEX_CAN_RAMS	0	FLEXCAN0 MB memory
4	FLEX_CAN_RAMS	1	FLEXCAN1 MB memory
4	FLEX_CAN_RAMS	2	FLEXCAN2 MB memory
4	FLEX_CAN_RAMS	3	FLEXCAN3 MB memory
5	CM7_1_TOP	0	CM7_1 instruction cache data block 1
5	CM7_1_TOP	1	CM7_1 instruction cache data block 2
5	CM7_1_TOP	2	CM7_1 instruction cache tag block 1
5	CM7_1_TOP	3	CM7_1 instruction cache tag block 2
5	CM7_1_TOP	4	CM7_1 data cache data block 1

Table continues on the next page...

Table 304. S32K342/S32K341/S32K322 MBIST mapping (continued)

BIST index (NMCUT)	BIST name	Memory index	Memory instance name
5	CM7_1_TOP	4	CM7_1 data cache data block 2
5	CM7_1_TOP	5	CM7_1 data cache data block 3
5	CM7_1_TOP	5	CM7_1 data cache data block 4
5	CM7_1_TOP	6	CM7_1 data cache data block 5
5	CM7_1_TOP	6	CM7_1 data cache data block 6
5	CM7_1_TOP	7	CM7_1 data cache data block 7
5	CM7_1_TOP	7	CM7_1 data cache data block 8
5	CM7_1_TOP	8	CM7_0 data cache tag block 1
5	CM7_1_TOP	9	CM7_0 data cache tag block 2
5	CM7_1_TOP	10	CM7_0 data cache tag block 3
5	CM7_1_TOP	11	CM7_0 data cache tag block 4
5	CM7_1_TOP	12	CM7_0 instruction TCM
5	CM7_1_TOP	13	CM7_0 data TCM block 1
5	CM7_1_TOP	13	CM7_0 data TCM block 2
6	QSPI_PERI_RAMS	0	QSPI RAM
6	QSPI_PERI_RAMS	1	QSPI TX memory
7	EMAC_TSN_RAM	0	EMAC Timestamp Memory
8	EMAC_RAMS	0	EMAC RXParser
8	EMAC_RAMS	1	EMAC TX block 1
8	EMAC_RAMS	2	EMAC TX block 1
8	EMAC_RAMS	3	EMAC TX block 2
8	EMAC_RAMS	4	EMAC RX block 2

Table 305. S32K312 MBIST mapping

BIST index (NMCUT)	BIST name	Memory index	Memory instance name
1	HSE_RAMS	0	HSE secure RAM
1	HSE_RAMS	1	HSE PKC memory block 1
1	HSE_RAMS	2	HSE PKC memory block 2
1	HSE_RAMS	3	HSE MTB memory
2	SYS0_DMA_RAMS	0	PRAM0 block 1
2	SYS0_DMA_RAMS	1	PRAM0 block 2
2	SYS0_DMA_RAMS	2	DMA TCD RAM

Table continues on the next page...

Table 305. S32K312 MBIST mapping (continued)

BIST index (NMCUT)	BIST name	Memory index	Memory instance name
3	CM7_0_TOP	0	CM7_0 instruction cache data block 1
3	CM7_0_TOP	1	CM7_0 instruction cache data block 2
3	CM7_0_TOP	2	CM7_0 instruction cache tag block 1
3	CM7_0_TOP	3	CM7_0 instruction cache tag block 2
3	CM7_0_TOP	4	CM7_0 data cache data block 1
3	CM7_0_TOP	4	CM7_0 data cache data block 2
3	CM7_0_TOP	5	CM7_0 data cache data block 3
3	CM7_0_TOP	5	CM7_0 data cache data block 4
3	CM7_0_TOP	6	CM7_0 data cache data block 5
3	CM7_0_TOP	6	CM7_0 data cache data block 6
3	CM7_0_TOP	7	CM7_0 data cache data block 7
3	CM7_0_TOP	7	CM7_0 data cache data block 8
3	CM7_0_TOP	8	CM7_0 data cache tag block 1
3	CM7_0_TOP	9	CM7_0 data cache tag block 2
3	CM7_0_TOP	10	CM7_0 data cache tag block 3
3	CM7_0_TOP	11	CM7_0 data cache tag block 4
3	CM7_0_TOP	12	CM7_0 instruction TCM (Tightly Coupled Memory)
3	CM7_0_TOP	13	CM7_0 data TCM block 1
3	CM7_0_TOP	13	CM7_0 data TCM block 2
4	FLEX_CAN_RAMS	0	FLEXCAN0 MB memory
4	FLEX_CAN_RAMS	1	FLEXCAN1 MB memory
4	FLEX_CAN_RAMS	2	FLEXCAN2 MB memory
4	FLEX_CAN_RAMS	3	FLEXCAN3 MB memory
4	FLEX_CAN_RAMS	4	FLEXCAN4 MB memory
4	FLEX_CAN_RAMS	5	FLEXCAN5 MB memory

Table 306. S32K311/S32K310 MBIST mapping

BIST index (NMCUT)	BIST name	Memory index	Memory instance name
1	HSE_RAMS	0	HSE secure RAM
1	HSE_RAMS	1	HSE PKC memory block 1
1	HSE_RAMS	2	HSE PKC memory block 2
1	HSE_RAMS	3	HSE MTB memory

Table continues on the next page...

Table 306. S32K311/S32K310 MBIST mapping (continued)

BIST index (NMCUT)	BIST name	Memory index	Memory instance name
2	SYS0_DMA_RAMS	0	PRAM0 block 1
2	SYS0_DMA_RAMS	1	DMA TCD RAM
3	CM7_0_TOP	0	CM7_0 instruction cache data block 1
3	CM7_0_TOP	1	CM7_0 instruction cache data block 2
3	CM7_0_TOP	2	CM7_0 instruction cache tag block 1
3	CM7_0_TOP	3	CM7_0 instruction cache tag block 2
3	CM7_0_TOP	4	CM7_0 data cache data block 1
3	CM7_0_TOP	4	CM7_0 data cache data block 2
3	CM7_0_TOP	5	CM7_0 data cache data block 3
3	CM7_0_TOP	5	CM7_0 data cache data block 4
3	CM7_0_TOP	6	CM7_0 data cache data block 5
3	CM7_0_TOP	6	CM7_0 data cache data block 6
3	CM7_0_TOP	7	CM7_0 data cache data block 7
3	CM7_0_TOP	7	CM7_0 data cache data block 8
3	CM7_0_TOP	8	CM7_0 data cache tag block 1
3	CM7_0_TOP	9	CM7_0 data cache tag block 2
3	CM7_0_TOP	10	CM7_0 data cache tag block 3
3	CM7_0_TOP	11	CM7_0 data cache tag block 4
3	CM7_0_TOP	12	CM7_0 instruction TCM (Tightly Coupled Memory)
3	CM7_0_TOP	13	CM7_0 data TCM block 1
3	CM7_0_TOP	13	CM7_0 data TCM block 2
4	FLEX_CAN_RAMS	0	FLEXCAN0 MB memory
4	FLEX_CAN_RAMS	1	FLEXCAN1 MB memory
4	FLEX_CAN_RAMS	2	FLEXCAN2 MB memory

Table 307. S32K358/S32K356/S32K348/S32K338/S32K328 MBIST mapping

BIST index (NMCUT)	BIST name	Memory index	Memory instance name
1	HSE_RAMS	0	HSE secure RAM
1	HSE_RAMS	1	HSE PKC memory block 1
1	HSE_RAMS	2	HSE PKC memory block 2
1	HSE_RAMS	3	HSE MTB memory

Table continues on the next page...

Table 307. S32K358/S32K356/S32K348/S32K338/S32K328 MBIST mapping (continued)

BIST index (NMCUT)	BIST name	Memory index	Memory instance name
2	SYS0_DMA_RAMS	0	PRAM0 block 1
2	SYS0_DMA_RAMS	1	PRAM0 block 2
2	SYS0_DMA_RAMS	2	PRAM0 block 3
2	SYS0_DMA_RAMS	3	PRAM0 block 4
3	SYS1_RAMS	0	PRAM1 block 1
3	SYS1_RAMS	1	PRAM1 block 2
4	SYS2_RAMS	0	PRAM2 block 1
4	SYS2_RAMS	1	PRAM2 block 2
4	SYS2_RAMS	2	PRAM2 block 3
5	CM7_0_TOP	0	CM7_0 instruction cache data block 1
5	CM7_0_TOP	1	CM7_0 instruction cache data block 2
5	CM7_0_TOP	2	CM7_0 instruction cache tag block 1
5	CM7_0_TOP	3	CM7_0 instruction cache tag block 2
5	CM7_0_TOP	4	CM7_0 data cache data block 1
5	CM7_0_TOP	4	CM7_0 data cache data block 5
5	CM7_0_TOP	5	CM7_0 data cache data block 2
5	CM7_0_TOP	5	CM7_0 data cache data block 6
5	CM7_0_TOP	6	CM7_0 data cache data block 3
5	CM7_0_TOP	6	CM7_0 data cache data block 7
5	CM7_0_TOP	7	CM7_0 data cache data block 4
5	CM7_0_TOP	7	CM7_0 data cache data block 8
5	CM7_0_TOP	8	CM7_0 data cache tag block 1
5	CM7_0_TOP	9	CM7_0 data cache tag block 2
5	CM7_0_TOP	10	CM7_0 data cache tag block 3
5	CM7_0_TOP	11	CM7_0 data cache tag block 4
5	CM7_0_TOP	12	CM7_0 instruction TCM
5	CM7_0_TOP	13	CM7_0 data TCM block 1
5	CM7_0_TOP	13	CM7_0 data TCM block 2
6	CM7_1_TOP	0	CM7_1 instruction cache data block 1

Table continues on the next page...

Table 307. S32K358/S32K356/S32K348/S32K338/S32K328 MBIST mapping (continued)

BIST index (NMCUT)	BIST name	Memory index	Memory instance name
6	CM7_1_TOP	1	CM7_1 instruction cache data block 2
6	CM7_1_TOP	2	CM7_1 instruction cache tag block 1
6	CM7_1_TOP	3	CM7_1 instruction cache tag block 2
6	CM7_1_TOP	4	CM7_1 data cache data block 1
6	CM7_1_TOP	4	CM7_1 data cache data block 5
6	CM7_1_TOP	5	CM7_1 data cache data block 2
6	CM7_1_TOP	5	CM7_1 data cache data block 6
6	CM7_1_TOP	6	CM7_1 data cache data block 3
6	CM7_1_TOP	6	CM7_1 data cache data block 7
6	CM7_1_TOP	7	CM7_1 data cache data block 4
6	CM7_1_TOP	7	CM7_1 data cache data block 8
6	CM7_1_TOP	8	CM7_1 data cache tag block 1
6	CM7_1_TOP	9	CM7_1 data cache tag block 2
6	CM7_1_TOP	10	CM7_1 data cache tag block 3
6	CM7_1_TOP	11	CM7_1 data cache tag block 4
6	CM7_1_TOP	12	CM7_1 instruction TCM
6	CM7_1_TOP	13	CM7_1 data block 1
6	CM7_1_TOP	13	CM7_1 data block 2
7	CM7_2_TOP	0	CM7_2 instruction cache data block 1
7	CM7_2_TOP	1	CM7_2 instruction cache data block 2
7	CM7_2_TOP	2	CM7_2 instruction cache tag block 1
7	CM7_2_TOP	3	CM7_2 instruction cache tag block 2
7	CM7_2_TOP	4	CM7_2 data cache data block 1
7	CM7_2_TOP	4	CM7_2 data cache data block 5
7	CM7_2_TOP	5	CM7_2 data cache data block 2
7	CM7_2_TOP	5	CM7_2 data cache data block 6
7	CM7_2_TOP	6	CM7_2 data cache data block 3
7	CM7_2_TOP	6	CM7_2 data cache data block 7
7	CM7_2_TOP	7	CM7_2 data cache data block 4
7	CM7_2_TOP	7	CM7_2 data cache data block 8
7	CM7_2_TOP	8	CM7_2 data cache tag block 1
7	CM7_2_TOP	9	CM7_2 data cache tag block 2

Table continues on the next page...

Table 307. S32K358/S32K356/S32K348/S32K338/S32K328 MBIST mapping (continued)

BIST index (NMCUT)	BIST name	Memory index	Memory instance name
7	CM7_2_TOP	10	CM7_2 data cache tag block 3
7	CM7_2_TOP	11	CM7_2 data cache tag block 4
7	CM7_2_TOP	12	CM7_2 instruction TCM
7	CM7_2_TOP	13	CM7_2 data TCM block 1
7	CM7_2_TOP	13	CM7_2 data TCM block 2
8	FLEX_CAN_RAMS	0	FLEXCAN0 MB
8	FLEX_CAN_RAMS	1	FLEXCAN1 MB
8	FLEX_CAN_RAMS	2	FLEXCAN2 MB
8	FLEX_CAN_RAMS	3	FLEXCAN3 MB
8	FLEX_CAN_RAMS	4	FLEXCAN4 MB
8	FLEX_CAN_RAMS	5	FLEXCAN5 MB
8	FLEX_CAN_RAMS	6	FLEXCAN6 MB
8	FLEX_CAN_RAMS	7	FLEXCAN7 MB
9	QSPI_PERI_RAMS	0	QSPI RAM
9	QSPI_PERI_RAMS	1	QSPI TX memory
10	GMAC_TSN_RAM	0	GMAC Timestamp Memory
11	GMAC_RAMS	0	GMAC RXParser
11	GMAC_RAMS	1	GMAC TX block 1
11	GMAC_RAMS	2	GMAC RX block 1
11	GMAC_RAMS	3	GMAC TX block 2
11	GMAC_RAMS	4	GMAC RX block 2
12	ETF_RAMS	0	HTM ETF HTM_ETF
12	ETF_RAMS	1	Shared ETF Shared_system_ETF
12	ETF_RAMS	2	CM7 cluster Instruction ETF CM7_cluster_ETF_ETMI
12	ETF_RAMS	3	CM7 cluster Data ETF CM7_cluster_ETF_ETMD

Table 308. S32K388 MBIST mapping

BIST index (NMCUT)	BIST name	Memory index	Memory instance name
1	HSE_RAMS	0	HSE secure RAM
1	HSE_RAMS	1	HSE PCK memory block 1
1	HSE_RAMS	2	HSE PCK memory block 2
1	HSE_RAMS	3	HSE MTB memory
2	SYS0_RAMS	0	PRAM0 block 1
2	SYS0_RAMS	1	PRAM0 block 2
2	SYS0_RAMS	2	PRAM0 block 3
2	SYS0_RAMS	3	PRAM0 block 4
3	SYS1_RAMS	0	PRAM1 block 1
3	SYS1_RAMS	1	PRAM1 block 2
3	SYS1_RAMS	2	PRAM1 block 3
3	SYS1_RAMS	3	PRAM1 block 4
4	SYS2_DMA_RAMS	0	PRAM2 block 1
4	SYS2_DMA_RAMS	1	PRAM2 block 2
4	SYS2_DMA_RAMS	2	PRAM2 block 3
4	SYS2_DMA_RAMS	3	PRAM2 block 4
4	SYS2_DMA_RAMS	4	DMA TCD RAM
5	CM7_0_TOP	0	CM7_0 instruction cache data block 1
5	CM7_0_TOP	1	CM7_0 instruction cache data block 2
5	CM7_0_TOP	2	CM7_0 instruction cache tag block 1
5	CM7_0_TOP	3	CM7_0 instruction cache tag block 2
5	CM7_0_TOP	4	CM7_0 data cache data block 1
5	CM7_0_TOP	4	CM7_0 data cache data block 5
5	CM7_0_TOP	5	CM7_0 data cache data block 2
5	CM7_0_TOP	5	CM7_0 data cache data block 6
5	CM7_0_TOP	6	CM7_0 data cache data block 3
5	CM7_0_TOP	6	CM7_0 data cache data block 7
5	CM7_0_TOP	7	CM7_0 data cache data block 4
5	CM7_0_TOP	7	CM7_0 data cache data block 8
5	CM7_0_TOP	8	CM7_0 data cache tag block 1
5	CM7_0_TOP	9	CM7_0 data cache tag block 2
5	CM7_0_TOP	10	CM7_0 data cache tag block 3

Table continues on the next page...

Table 308. S32K388 MBIST mapping (continued)

BIST index (NMCUT)	BIST name	Memory index	Memory instance name
5	CM7_0_TOP	11	CM7_0 data cache tag block 4
5	CM7_0_TOP	12	CM7_0 instruction TCM(Tightly Coupled Memory)
5	CM7_0_TOP	13	CM7_0 data TCM block 1
5	CM7_0_TOP	13	CM7_0 data TCM block 2
6	CM7_1_TOP	0	CM7_1 instruction cache data block 1
6	CM7_1_TOP	1	CM7_1 instruction cache data block 2
6	CM7_1_TOP	2	CM7_1 instruction cache tag block 1
6	CM7_1_TOP	3	CM7_1 instruction cache tag block 2
6	CM7_1_TOP	4	CM7_1 data cache data block 1
6	CM7_1_TOP	4	CM7_1 data cache data block 5
6	CM7_1_TOP	5	CM7_1 data cache data block 2
6	CM7_1_TOP	5	CM7_1 data cache data block 6
6	CM7_1_TOP	6	CM7_1 data cache data block 3
6	CM7_1_TOP	6	CM7_1 data cache data block 7
6	CM7_1_TOP	7	CM7_1 data cache data block 4
6	CM7_1_TOP	7	CM7_1 data cache data block 8
6	CM7_1_TOP	8	CM7_1 data cache tag block 1
6	CM7_1_TOP	9	CM7_1 data cache tag block 2
6	CM7_1_TOP	10	CM7_1 data cache tag block 3
6	CM7_1_TOP	11	CM7_1 data cache tag block 4
6	CM7_1_TOP	12	CM7_1 instruction TCM
6	CM7_1_TOP	13	CM7_1 data TCM block 1
6	CM7_1_TOP	13	CM7_1 data TCM block 2
7	CM7_2_TOP	0	CM7_2 instruction cache data block 1
7	CM7_2_TOP	1	CM7_2 instruction cache data block 2
7	CM7_2_TOP	2	CM7_2 instruction cache tag block 1
7	CM7_2_TOP	3	CM7_2 instruction cache tag block 2
7	CM7_2_TOP	4	CM7_2 data cache data block 1
7	CM7_2_TOP	4	CM7_2 data cache data block 5
7	CM7_2_TOP	5	CM7_2 data cache data block 2
7	CM7_2_TOP	5	CM7_2 data cache data block 6
7	CM7_2_TOP	6	CM7_2 data cache data block 3

Table continues on the next page...

Table 308. S32K388 MBIST mapping (continued)

BIST index (NMCUT)	BIST name	Memory index	Memory instance name
7	CM7_2_TOP	6	CM7_2 data cache data block 7
7	CM7_2_TOP	7	CM7_2 data cache data block 4
7	CM7_2_TOP	7	CM7_2 data cache data block 8
7	CM7_2_TOP	8	CM7_2 data cache tag block 1
7	CM7_2_TOP	9	CM7_2 data cache tag block 2
7	CM7_2_TOP	10	CM7_2 data cache tag block 3
7	CM7_2_TOP	11	CM7_2 data cache tag block 4
7	CM7_2_TOP	12	CM7_2 instruction TCM
7	CM7_2_TOP	13	CM7_2 data TCM block 1
7	CM7_2_TOP	13	CM7_2 data TCM block 2
8	FLEX_CAN_RAMS	0	FLEXCAN0 MB memory
8	FLEX_CAN_RAMS	1	FLEXCAN1 MB memory
8	FLEX_CAN_RAMS	2	FLEXCAN2 MB memory
8	FLEX_CAN_RAMS	3	FLEXCAN3 MB memory
8	FLEX_CAN_RAMS	4	FLEXCAN4 MB memory
8	FLEX_CAN_RAMS	5	FLEXCAN5 MB memory
8	FLEX_CAN_RAMS	6	FLEXCAN5 MB memory
8	FLEX_CAN_RAMS	7	FLEXCAN5 MB memory
9	QSPI_PERI_RAMS	0	QSPI RAM
9	QSPI_PERI_RAMS	1	QSPI TX memory
10	GMAC_0_TSN_RAM	0	GMAC_0 Timestamp memory
11	GMAC_0_RAM	0	GMAC_0 RXParser
11	GMAC_0_RAM	1	GMAC_0 TX block 1
11	GMAC_0_RAM	2	GMAC_0 RX block 1
11	GMAC_0_RAM	3	GMAC_0 TX block 2
11	GMAC_0_RAM	4	GMAC_0 RX block 2
12	ETF_RAMS	0	Shared ETF Shared_system_ETF
12	ETF_RAMS	1	CM7 cluster Intstruction ETF
12	ETF_RAMS	2	CM7 cluster Data ETF block 1
12	ETF_RAMS	3	CM7 cluster Data ETF block 2
13	HTM_ETF	0	HTM ETF HTM_ETF
14	GMAC_1_TSN_RAM	0	GMAC_1 Timestamp memory

Table continues on the next page...

Table 308. S32K388 MBIST mapping (continued)

BIST index (NMCUT)	BIST name	Memory index	Memory instance name
15	GMAC_1_RAM	0	GMAC_1 RXParser
15	GMAC_1_RAM	1	GMAC_1 TX block 1
15	GMAC_1_RAM	2	GMAC_1 RX block 1
15	GMAC_1_RAM	3	GMAC_1 TX block 2
15	GMAC_1_RAM	4	GMAC_1 RX block 2
16	CM7_3_TOP	0	CM7_3 instruction cache data block 1
16	CM7_3_TOP	1	CM7_3 instruction cache data block 2
16	CM7_3_TOP	2	CM7_3 instruction cache tag block 1
16	CM7_3_TOP	3	CM7_3 instruction cache tag block 2
16	CM7_3_TOP	4	CM7_3 data cache data block 1
16	CM7_3_TOP	4	CM7_3 data cache data block 5
16	CM7_3_TOP	5	CM7_3 data cache data block 2
16	CM7_3_TOP	5	CM7_3 data cache data block 6
16	CM7_3_TOP	6	CM7_3 data cache data block 3
16	CM7_3_TOP	6	CM7_3 data cache data block 7
16	CM7_3_TOP	7	CM7_3 data cache data block 4
16	CM7_3_TOP	7	CM7_3 data cache data block 8
16	CM7_3_TOP	8	CM7_3 data cache tag block 1
16	CM7_3_TOP	9	CM7_3 data cache tag block 2
16	CM7_3_TOP	10	CM7_3 data cache tag block 3
16	CM7_3_TOP	11	CM7_3 data cache tag block 4
16	CM7_3_TOP	12	CM7_3 instruction TCM
16	CM7_3_TOP	13	CM7_3 data TCM block 1
16	CM7_3_TOP	13	CM7_3 data TCM block 2
17	DMA_ACE	0	DMA ACE 1
17	DMA_ACE	1	DMA ACE 2

Table 309. S32K389 MBIST mapping

BIST index (NMCUT)	BIST name	Memory index	Memory instance name
1	HSE_RAMS	0	HSE secure RAM
1	HSE_RAMS	1	HSE PCK memory block 1
1	HSE_RAMS	2	HSE PCK memory block 2

Table continues on the next page...

Table 309. S32K389 MBIST mapping (continued)

BIST index (NMCUT)	BIST name	Memory index	Memory instance name
1	HSE_RAMS	3	HSE MTB memory
2	SYS0_RAMS	0	PRAM0 block 1
2	SYS0_RAMS	1	PRAM0 block 2
2	SYS0_RAMS	1	PRAM0 block 3
2	SYS0_RAMS	2	PRAM0 block 4
2	SYS0_RAMS	2	PRAM0 block 5
2	SYS0_RAMS	3	PRAM0 block 6
2	SYS0_RAMS	3	PRAM0 block 7
2	SYS0_RAMS	4	PRAM0 block 8
3	SYS1_RAMS	0	PRAM1 block 1
3	SYS1_RAMS	0	PRAM1 block 2
3	SYS1_RAMS	1	PRAM1 block 3
3	SYS1_RAMS	1	PRAM1 block 4
3	SYS1_RAMS	2	PRAM1 block 5
3	SYS1_RAMS	2	PRAM1 block 6
3	SYS1_RAMS	3	PRAM1 block 7
3	SYS1_RAMS	3	PRAM1 block 8
4	SYS2_RAMS	0	PRAM2 block 1
4	SYS2_RAMS	0	PRAM2 block 2
4	SYS2_RAMS	1	PRAM2 block 3
4	SYS2_RAMS	1	PRAM2 block 4
4	SYS2_RAMS	2	PRAM2 block 5
4	SYS2_RAMS	2	PRAM2 block 6
4	SYS2_RAMS	3	PRAM2 block 7
4	SYS2_RAMS	3	PRAM2 block 8
5	SYS3_DMA_RAMS	0	PRAM3 block 1
5	SYS3_DMA_RAMS	0	PRAM3 block 2
5	SYS3_DMA_RAMS	1	PRAM3 block 3
5	SYS3_DMA_RAMS	1	PRAM3 block 4
5	SYS3_DMA_RAMS	2	PRAM3 block 5
5	SYS3_DMA_RAMS	2	PRAM3 block 6
5	SYS3_DMA_RAMS	3	DMA TCD RAM

Table continues on the next page...

Table 309. S32K389 MBIST mapping (continued)

BIST index (NMCUT)	BIST name	Memory index	Memory instance name
6	CM7_0_TOP	0	CM7_0 instruction cache data block 1
6	CM7_0_TOP	1	CM7_0 instruction cache data block 2
6	CM7_0_TOP	2	CM7_0 instruction cache tag block 1
6	CM7_0_TOP	3	CM7_0 instruction cache tag block 2
6	CM7_0_TOP	4	CM7_0 data cache data block 1
6	CM7_0_TOP	4	CM7_0 data cache data block 5
6	CM7_0_TOP	5	CM7_0 data cache data block 2
6	CM7_0_TOP	5	CM7_0 data cache data block 6
6	CM7_0_TOP	6	CM7_0 data cache data block 3
6	CM7_0_TOP	6	CM7_0 data cache data block 7
6	CM7_0_TOP	7	CM7_0 data cache data block 4
6	CM7_0_TOP	7	CM7_0 data cache data block 8
6	CM7_0_TOP	8	CM7_0 data cache tag block 1
6	CM7_0_TOP	9	CM7_0 data cache tag block 2
6	CM7_0_TOP	10	CM7_0 data cache tag block 3
6	CM7_0_TOP	11	CM7_0 data cache tag block 4
6	CM7_0_TOP	12	CM7_0 instruction TCM(Tightly Coupled Memory)
6	CM7_0_TOP	13	CM7_0 data TCM block 1
6	CM7_0_TOP	13	CM7_0 data TCM block 2
7	CM7_1_TOP	0	CM7_1 instruction cache data block 1
7	CM7_1_TOP	1	CM7_1 instruction cache data block 2
7	CM7_1_TOP	2	CM7_1 instruction cache tag block 1
7	CM7_1_TOP	3	CM7_1 instruction cache tag block 2
7	CM7_1_TOP	4	CM7_1 data cache data block 1
7	CM7_1_TOP	4	CM7_1 data cache data block 5
7	CM7_1_TOP	5	CM7_1 data cache data block 2
7	CM7_1_TOP	5	CM7_1 data cache data block 6
7	CM7_1_TOP	6	CM7_1 data cache data block 3
7	CM7_1_TOP	6	CM7_1 data cache data block 7
7	CM7_1_TOP	7	CM7_1 data cache data block 4
7	CM7_1_TOP	7	CM7_1 data cache data block 8
7	CM7_1_TOP	8	CM7_1 data cache tag block 1

Table continues on the next page...

Table 309. S32K389 MBIST mapping (continued)

BIST index (NMCUT)	BIST name	Memory index	Memory instance name
7	CM7_1_TOP	9	CM7_1 data cache tag block 2
7	CM7_1_TOP	10	CM7_1 data cache tag block 3
7	CM7_1_TOP	11	CM7_1 data cache tag block 4
7	CM7_1_TOP	12	CM7_1 instruction TCM
7	CM7_1_TOP	13	CM7_1 data TCM block 1
7	CM7_1_TOP	13	CM7_1 data TCM block 2
8	CM7_2_TOP	0	CM7_2 instruction cache data block 1
8	CM7_2_TOP	1	CM7_2 instruction cache data block 2
8	CM7_2_TOP	2	CM7_2 instruction cache tag block 1
8	CM7_2_TOP	3	CM7_2 instruction cache tag block 2
8	CM7_2_TOP	4	CM7_2 data cache data block 1
8	CM7_2_TOP	4	CM7_2 data cache data block 5
8	CM7_2_TOP	5	CM7_2 data cache data block 2
8	CM7_2_TOP	5	CM7_2 data cache data block 6
8	CM7_2_TOP	6	CM7_2 data cache data block 3
8	CM7_2_TOP	6	CM7_2 data cache data block 7
8	CM7_2_TOP	7	CM7_2 data cache data block 4
8	CM7_2_TOP	7	CM7_2 data cache data block 8
8	CM7_2_TOP	8	CM7_2 data cache tag block 1
8	CM7_2_TOP	9	CM7_2 data cache tag block 2
8	CM7_2_TOP	10	CM7_2 data cache tag block 3
8	CM7_2_TOP	11	CM7_2 data cache tag block 4
8	CM7_2_TOP	12	CM7_2 instruction TCM
8	CM7_2_TOP	13	CM7_2 data TCM block 1
8	CM7_2_TOP	13	CM7_2 data TCM block 2
9	FLEX_CAN_RAMS	0	FLEXCAN0 MB memory
9	FLEX_CAN_RAMS	1	FLEXCAN1 MB memory
9	FLEX_CAN_RAMS	2	FLEXCAN2 MB memory
9	FLEX_CAN_RAMS	3	FLEXCAN3 MB memory
9	FLEX_CAN_RAMS	4	FLEXCAN4 MB memory
9	FLEX_CAN_RAMS	5	FLEXCAN5 MB memory
9	FLEX_CAN_RAMS	6	FLEXCAN6 MB memory

Table continues on the next page...

Table 309. S32K389 MBIST mapping (continued)

BIST index (NMCUT)	BIST name	Memory index	Memory instance name
9	FLEX_CAN_RAMS	7	FLEXCAN7 MB memory
9	FLEX_CAN_RAMS	8	FLEXCAN8 MB memory
9	FLEX_CAN_RAMS	9	FLEXCAN9 MB memory
9	FLEX_CAN_RAMS	10	FLEXCAN10 MB memory
9	FLEX_CAN_RAMS	11	FLEXCAN11 MB memory
10	QSPI_PERI_RAMS	0	QSPI RAM
10	QSPI_PERI_RAMS	1	QSPI TX memory
11	GMAC_0_TSN_RAM	0	GMAC_0 Timestamp memory
12	GMAC_0_RAM	0	GMAC_0 RXParser
12	GMAC_0_RAM	1	GMAC_0 TX block 1
12	GMAC_0_RAM	2	GMAC_0 RX block 1
12	GMAC_0_RAM	3	GMAC_0 TX block 2
12	GMAC_0_RAM	4	GMAC_0 RX block 2
13	ETF_RAMS	0	Shared ETF Shared_system_ETF
13	ETF_RAMS	1	CM7 cluster Intstruction ETF
13	ETF_RAMS	2	CM7 cluster Data ETF block 1
13	ETF_RAMS	3	CM7 cluster Data ETF block 2
14	HTM_ETF	0	HTM ETF HTM_ETF
15	GMAC_1_TSN_RAM	0	GMAC_1 Timestamp memory
16	GMAC_1_RAM	0	GMAC_1 RXParser
16	GMAC_1_RAM	1	GMAC_1 TX block 1
16	GMAC_1_RAM	2	GMAC_1 RX block 1
16	GMAC_1_RAM	3	GMAC_1 TX block 2
16	GMAC_1_RAM	4	GMAC_1 RX block 2
17	CM7_3_TOP	0	CM7_3 instruction cache data block 1
17	CM7_3_TOP	1	CM7_3 instruction cache data block 2
17	CM7_3_TOP	2	CM7_3 instruction cache tag block 1
17	CM7_3_TOP	3	CM7_3 instruction cache tag block 2
17	CM7_3_TOP	4	CM7_3 data cache data block 1
17	CM7_3_TOP	4	CM7_3 data cache data block 5
17	CM7_3_TOP	5	CM7_3 data cache data block 2
17	CM7_3_TOP	5	CM7_3 data cache data block 6

Table continues on the next page...

Table 309. S32K389 MBIST mapping (continued)

BIST index (NMCUT)	BIST name	Memory index	Memory instance name
17	CM7_3_TOP	6	CM7_3 data cache data block 3
17	CM7_3_TOP	6	CM7_3 data cache data block 7
17	CM7_3_TOP	7	CM7_3 data cache data block 4
17	CM7_3_TOP	7	CM7_3 data cache data block 8
17	CM7_3_TOP	8	CM7_3 data cache tag block 1
17	CM7_3_TOP	9	CM7_3 data cache tag block 2
17	CM7_3_TOP	10	CM7_3 data cache tag block 3
17	CM7_3_TOP	11	CM7_3 data cache tag block 4
17	CM7_3_TOP	12	CM7_3 instruction TCM
17	CM7_3_TOP	13	CM7_3 data TCM block 1
17	CM7_3_TOP	13	CM7_3 data TCM block 2
18	DMA_ACE	0	DMA ACE 1
18	DMA_ACE	1	DMA ACE 2

There are different number of MBIST indices (NMCUT) across different parts. The below table shows the variation across different chips.

Table 310. NMCUT

Chip	NMCUT
S32K314/S32K324/S32K344	12
S32K341/S32K342/S32K322	9
S32K310/S32K311/S32K312	5
S32K358/S32K356/S32K348/S32K338/S32K328	13
S32K388	18
S32K389	19

54.1.4 Clock connectivity

STCU2 works on the system clock domain. The scan operations while LBIST are done on TCK clock.

Table 311. Clock connectivity

STCU2 clock	Clock connected to the chip	Remarks
STCU2 CORE_CLK	AIPS_SLOW_CLK	System clock divider output clock node for STCU2 operations and register interface accesses
REG_INTF_CLK		
SHIFT_CLK	JTAG_TCK	JTAG_TCK used for shift operations when self-test is in progress

54.1.5 Memory map for self-test GPRs

See the memory map file attached to this document and 'SELFTEST_GPR' chapter for details.

54.1.6 Handling unrecoverable faults

Only recoverable and non-critical faults from STCU2 are mapped to FCCU. The unrecoverable and critical faults are mapped to MC_RGM. Therefore, ignore any references to mapping of unrecoverable and critical faults with FCCU in this chapter.

54.1.7 Indicating fault state during the main reset domain Self-Test

FCCU EOUT indicates the fault state during the Self-Test as soon as the UTEST_MISC[FCCU_EOUT_ DEDICATED] is enabled and DCMRWD2[EOUT_STAT_DUR_STEST] is enabled at the time of Self-Test setup steps.

The fault state remains asserted until the bit DCMRWD2[EOUT_STAT_DUR_STEST] is cleared post Self-Test reset sequence.

In case it is not required to indicate the fault state during the Self-Test, then the bit DCMRWD2[EOUT_STAT_DUR_STEST] should not be programmed.

54.1.8 Error events management in main reset domain Self-Test

Error events should be mapped to unrecoverable faults in STCU2_ERR_FM register for main reset domain Self-Test.

54.1.9 AUTOLOCK_VALUE for register write access via STCU2_SKC

Register write-access watchdog timer explains that access to STCU2 registers via the STCU2_SKC security-key mechanism times out after a number of STCU2 clock cycles equal to AUTOLOCK_VALUE. On this chip, AUTOLOCK_VALUE is 32'hFFFF_FFFF.

54.1.10 PLL loss of lock during Self-Test

Detection of PLL loss of lock during LBIST or MBIST causes a destructive reset in the system.

CMU0 programming is required before the Self-Test programming begins.

All the registers present in LBIST domains do not retain their content after Self-Test reset.

54.1.11 STCU2 BIST start (BSTART) register description

Attempted accesses with the undefined reserved values in the BSTART register can result in undefined behavior. The following table defines the BSTART field on this chip.

Table 312. STCU2 BIST start (BSTART) field description

Field	Description
31 CLKEN	IPS clock enable, enables the IPS clock for BISTs. 0b - If cleared, disables the clock 1b - Enables the IPS clock for BISTs
30 - 3	Reserved
2 – 0 BSTART	BSTART 000b – Reserved 001b – Reserved 010b – Reserved

Table continues on the next page...

Table 312. STCU2 BIST start (BSTART) field description (continued)

Field	Description
	011b – Reserved 100b - Reserved 101b – Program BISTs and start them 110b – Reserved 111b – Reserved

54.1.12 STCU2 algorithm select (ALGOSEL) register description

This register selects any one of the predefined algorithms. The value used in this register is mapped to a predefined algorithm register in BIST. After MCT has programmed information on this register and starts some BIST operation, this value is transferred to the predefined algorithm register in every BIST that is selected.

If multiple predefined algorithms are selected, the order of execution is from LSB to MSB.

Any field that does not have a predefined algorithm associated with it is a reserved bit. No data is written to reserved bits and when you read these bits, they always return 0. These predefined algorithms include:

- Backgrounds
- Address modes and other register programming that may be necessary
- The number of BIST runs needed to perform the exact coverage requested.

MCT decodes this information through a series of look-up tables. The following table describes the high-level BIST algorithms that can be invoked through the MCT. The columns in that table have the following meaning:

- Index: The position in this MCT register enables this algorithm.
- Name: The symbolic name for this algorithm.
- Description: The intent and use model for this algorithm.
- BIST runs: The sequence of BIST invocations comprising this algorithm.
- Sequences: The ordered list of march elements to run. Each march element consists of a direction indicator (for increasing addresses and for decreasing addresses) and a set of march phases. A march phase consists of reading (R) or writing (W), the non-inverted (0) or inverted (1) background pattern.
- Used by default: Whether this algorithm is selected in the post-reset state of this register.

Table 313. ALGOSEL description

Index	Name	Description	BIST runs	Sequences	Used by default
3	March C+ single	March C+ with column fast and solid background	MarchC+ column fast 1x (#4)	W0, R0W1, R1W0R0W1, R1W0, R0W1R1W0, R0	Yes
13	BasicChk	Basic selfcontained checkerboard, using column fast	BasicChk (#18)	R0W1R1	-

54.1.13 Self-Test programming sequence

This section describes a high level programming sequence for executing Self-Test. The detailed programming sequences for supported configurations are available in a separate application note (contact your NXP sales representative).

- Program clock sources and MC_CGMs as per the clock configuration.

NOTE

While running MBIST on EMAC timestamp memory, MC_CGM.MUX_9_DC_0[DIV] should be appropriately configured to ensure EMAC_CLK_TS should be atleast 1.5 times the AIPS_SLOW_CLK frequency.

- Program SELFTEST_GPR[CONFIG_REG] and SELFTEST_GPR[LBIST_PROG_REG] as per the LBIST configurations.

NOTE

This is not applicable for S32K312, S32K311, and S32K310 wherein LBIST is not supported.

- Program STCU2 as per NXP recommended.
- Program dcf_client_utest_misc[FCCU_EOUT_DEDICATED] and DCMRWD2[EOUT_STAT_DUR_STEST] to configure EOUT pins as dedicated and to indicate FCCU error states on EOUT pins during Self-Test respectively.
- Configure ERCTRL[ERASSERT] register bit in MC_RGM to achieve desired behaviour of Reset Pin as documented in Reset Section and Pad States Section.
- Disable functional reset sources to avoid false Self-Test abort scenarios. For example, software watchdog timers cannot be serviced while Self-Test because the software is not accessible and can cause invalid timeout reset. See Reset events and configurations while Self-Test.
- Program the PCS_ENABLE[8:7] fields of the CONFIG_REG register (SELFTEST_GPR) to 0 to disable PCS.
- Trigger Self-Test execution by programming RUNSW register in STCU2.
- At the end of the Self-Test run, a functional reset is generated. Out of reset, the software can check the status registers inside STCU2 and take actions accordingly.

NOTE

EIM should be disabled when STCU based Self-Test is running.

54.1.14 Modules operation during/post Self-Test

The clock, reset, power generation modules, such as RGM, PMC, CGM, FIRC, PLL, and SIRC will be available during and post Self-Test. However, because the chip undergoes the reset sequence, PLL and CGM will be reset.

CMU0 and CMU3 will be available during Self-Test if enabled prior to Self-Test, but will be Reset post Self-Test.

STCU2 will also be available during Self-Test and the results can be read post Self-Test.

54.1.15 Reset events and configurations while Self-Test

As Self-Test is a software initiated Self-Test, it is expected by software to ensure all the peripherals are disabled as documented in Software considerations before Self-Test.

The functional reset sources should be appropriately configured to avoid any functional reset while Self-Test is running (disable SWT_RST and HSE_SWT_RST. JTAGC should not be in EXTEST, HIGHZ or CLAMP instructions). This ensures no selftest abort while running selftest due to functional reset source. The below table specifies the state of functional reset sources for Self-Test.

Table 314. Reset events and configurations while self-test

Reset source	Reset description	Configurations/remarks for selftest
FCCU_RST	FCCU reset reaction	The FCCU is a part of logic which is under Self-Test and thus it would be disabled during the Self-Test
ST_DONE	Self-Test done reset	The selftest done indication resets the device post selftest. While Self-Test is running, the Self-Test done indication will be in disabled state

Table continues on the next page...

Table 314. Reset events and configurations while self-test (continued)

Reset source	Reset description	Configurations/remarks for selftest
SWT_RST	SWT reset request	SWT should be disabled in Self-Test configuration sequence.
JTAG_RST	JTAG Reset	JTAG communication should be inactive while Self-Test and JTAGC state machine should not be in EXTEST, HIGHZ or CLAMP instructions
HSE_SWT_RST	HSE SWT timeout	HSE_SWT should be disabled in Self-Test configuration sequence.
SW_FUNC	software 'functional' reset	The software is inactive during Self-Test and hence software functional reset will not arrive while the chip is under Self-Test.
DEBUG_FUNC	debug 'functional' reset	MDM DAP should not be configured for debug functional reset generation while Self-Test is running.

The destructive reset sources are available while in Self-Test. In case of any destructive reset event, the device undergoes a destructive reset sequence, thereby resetting the STCU2 and Self-Test logic itself.

After completion of Self-Test (LBIST or MBIST), a functional reset sequence is executed by MC_RGM.

54.1.16 Software considerations before Self-Test

1. As all the communication peripherals are not in LBIST partitions; hence, it is required by SW to stop all communications and disable all the PCTLs of the communication peripherals before initiating Self-Test as the device will be unusable in Self-Test. This is needed so that there is graceful safe stating of inputs of non-lbisted peripheral modules.
2. Configure RGM functional reset sources as interrupts before Self-Test as per the considerations mentioned in Reset events and configurations while Self-Test, i.e., the SWTs should be disabled, the JTAGC should not be in states which can cause a JTAG reset event (EXTEST, HIGHZ, CLAMP instructions) and the software should not trigger a functional reset event via MC_ME or MDM_AP while the selftest is ongoing. This is to avoid any abort scenario due to any functional reset source as Self-Test is initiated by SW.

There is no issue with Reset pin as Reset pin is a source of destructive reset and will reset STCU2 and Self-Test related logic etc.

54.1.17 End of Self-Test

The Self-Test completion is a source of functional reset to the device, by which the device executes a functional reset sequence. The logic which is LBISTed additionally undergoes a complete POR sequence to ensure that the random content shifted during BIST sequence get cleared.

At the end of MBIST, the software should execute below steps:

1. Software reads the MBIST end flag to identify that the MBIST sequence had ended properly without any interruption or abort, by reading STCU2.MBESWn.
2. Software reads the MBIST status to identify whether the MBIST was successful or not, by reading STCU2.MBSSWn.

NOTE

The MBIST status should always be read in accordance with MBIST end flag. The STCU2.MBSSWn status is irrelevant if STCU2.MBESWn indicates that MBIST execution is incomplete.

At the end of LBIST, the software should execute below steps:

1. Software reads the LBIST end flag to identify that the LBIST sequence had ended properly without any interruption or abort, by reading STCU2.LBESWn.
2. Software reads the LBIST status to identify whether the LBIST was successful or not, by reading STCU2.LBSSWn.

NOTE

The LBIST status should always be read in accordance with LBIST end flag. The STCU2.LBSSWn status is irrelevant if STCU2.LBESWn indicates that LBIST execution is incomplete.

54.2 Introduction

STCU2 is a comprehensive programmable hardware module that:

- Manages the execution of BISTs
- Indicates whether each BIST passed
- Manages the chip's LBIST and the MBIST blocks

54.3 Main features

STCU2 includes:

- System interface for reading from and writing to the STCU2 registers (online) via the CPU
- Programmable scheduler for BIST execution
- Control over LBIST concurrent or sequential execution
- Control over MBIST concurrent or sequential execution
- Programmable LBIST delayed concurrent start
- Programmable internal clock prescaler to reduce internal and BIST clocks
- PLL lock signal monitoring during BIST sequences
- Programmable BIST-sequence-execution watchdog timer that specifies the maximum time allowed for the execution of a BIST sequence
- Fixed register-write-access watchdog timer that specifies the maximum amount of time that STCU2 allows you to write to its registers after you unlock them
- Fields that indicate the pass/fail indication of each BIST, see Types of BIST sequences and Types of BIST partitions
- Fields that indicate the status of each type of online STCU2 internal error condition
- Programmable fault mapping of each BIST for controlling the type of fault that STCU2 reports (recoverable or unrecoverable) when the BIST fails to execute
- Programmable fault mapping of each STCU2 internal error condition for controlling the type of fault that STCU2 reports (recoverable or unrecoverable) when the condition occurs
- Signals to report recoverable and unrecoverable faults to the FCCU module
- Redundant recoverable and unrecoverable fault-generation logic to improve reliability
- FCCU recoverable and unrecoverable fault-injection mechanism
- Global register write-protection mechanism that requires two security key codes
- Global automatic power saving after a BIST sequence is completed when watchdog timer time-out is detected
- Watchdog automatic clock wake-up mechanism when software unlocks the STCU2 registers for write access
- Watchdog automatic power saving when the fixed register-write-access watchdog timer times out

54.4 Block diagram

This diagram shows the parts of STCU2:

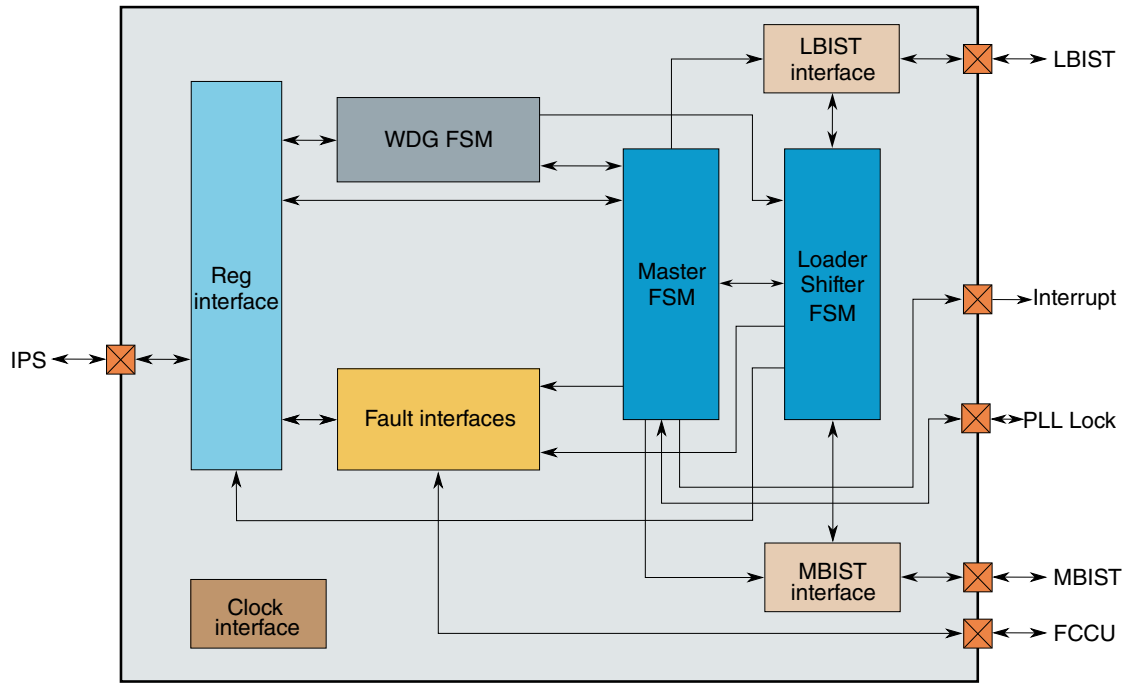


Figure 216. STCU2 block diagram

This table describes the parts of STCU2:

Table 315. STCU2 parts

Part	Function
Reg interface	Provides access to: • Registers • The security key logic • The IPS interface
Fault interfaces	Performs the following tasks: • Collects the fault conditions caused by STCU2 internal error conditions and each BIST executed in a sequence • Sets the global recoverable or unrecoverable status flag, depending on the fault mapping for a given BIST • Manages the recoverable and unrecoverable fault lines to and from the FCCU module • Manages the set/clear injection mechanism provided by the FCCU module To improve the intrinsic reliability of this critical logic, the generation logic is duplicated.
Clock interface	Manages the internal and the BIST clock prescaler, the internal clock-gating power saving, and the wake-up clock feature

Table continues on the next page...

Table 315. STCU2 parts (continued)

Part	Function
WDG FSM	Provides the following: • A programmable BIST-sequence-execution watchdog timer that specifies the maximum time allowed for the execution of a BIST sequence • A fixed register-write-access watchdog timer that specifies the maximum amount of time that STCU2 allows you to write to its registers after you unlock them
Master FSM	Coordinates and schedules all of the operations performed during a BIST sequence
Loader Shifter FSM	Programs the BIST registers and reads back the data to be checked at the end of each test operation
LBIST interface	Provides the interface between the chip's LBIST engines and the STCU2 controllers
MBIST interface	Provides the interface between the MBIST controller and the STCU2 controller

54.5 Peripheral bus interface

The peripheral bus interface is a slave bus used for configuration purposes via CPU. The module supports the following bus read operations:

- Word (32 bits) data read operations to any registers
- Any other operation is not supported.

Word, low and high half-words, byte operations are supported in write mode only:

- Write access is allowed by software only when CFG[WRP] is cleared. The default value of CFG[WRP] is cleared, exceptions are the following read/write fields: CFG[WRP], ERR_STAT[UFSF] and ERR_STAT[RFSF].
- STCU2 Configuration (CFG) should only be accessed in 32 bit (partial access not allowed).

The STCU2 module generates a transfer error in the following cases:

- Any read access to the registers after writing Key1 and before writing Key2 (in online) and before register write-access watchdog timer expires.
- Any write/read access to the register addresses not mapped on the peripheral but included in the address space of the peripheral
- Any write/read operation different from byte/halfword/word (free byte enables or other operations) on each register
- Any write operation on Double Security Key register applying a wrong sequence of keys (the two write operations cannot be interleaved with other access to STCU2 registers)
- Any write operation performed on a register when the Double Security keys have not been applied
- Any write operation performed on registers when CFG[WRP] is set and the access is performed through software (internal peripheral interface). The exceptions are SKC register (for valid keys value and sequence) in which STCU2 does not assert transfer error.
- Any write operation performed on Read Only registers

The registers of the STCU2 module are accessible (read/write) in each access mode: user, supervisor, or test.

In case there are write operations on bits marked as reserved, the transfer error is not generated.

NOTE

See the chip-specific STCU2 information for the wait time necessary to write to the online registers.

54.6 BISTs and BIST partitions

54.6.1 Definition: BIST

A BIST is a test that the chip can execute to verify the functional integrity of a part of itself. The chip uses STCU2 and other on-chip hardware to execute a BIST.

54.6.2 Definition: BIST partition

A BIST partition is a part of a chip for which a BIST has been defined. The hardware that is included in a given BIST partition is chip-specific. For a list of the hardware included in each of the BIST partitions on this chip, see the chip-specific STCU2 information.

54.6.3 Example: BIST partitions on a chip

This is an example of a chip with six BIST partitions:

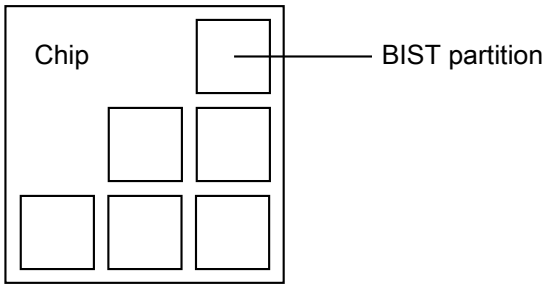


Figure 217. BIST partitions on a chip

54.6.4 Types of BIST partitions

The STCU2 module on this chip supports the following types of BIST partitions:

Table 316. Types of BIST partitions

BIST partition type	Description
MBIST partition	An SRAM or ROM block
LBIST partition	One or more digital modules

54.7 BIST sequences

54.7.1 Definition: BIST sequence

A BIST sequence is a programmable series of one or more phases, each of which executes one or more individual BISTs.

54.7.2 STCU2 executes MBISTs before LBISTs

If a BIST sequence includes both MBISTs and LBISTs; STCU2 executes the MBISTs first.

54.7.3 Example: Single-phase BIST sequence

This is an example of a single-phase BIST sequence in which STCU2 executes only one BIST:

Table 317. Example: Single-phase BIST sequence

Phase	BIST executed
0	MBIST 16

54.7.4 Example: Multi-phase BIST sequence

This is an example of a multiphase BIST sequence in which STCU2 executes more than one BIST in parallel in some phases:

Table 318. Example: Multi-phase BIST sequence

Phase	BISTs executed
0	MBISTs 7, 30
1	MBIST 23
2	MBISTs 6, 16, 29, 23
3	LBIST 12
4	LBISTs 13, 8, 11
5	LBISTs 19, 6

54.7.5 Types of BIST sequences

The STCU2 module on this chip supports these types of BIST sequences:

Table 319. Example: Types of BIST sequences

BIST sequence type	Description
Online BIST sequence	A BIST sequence that STCU2 executes during runtime at the request of software. Application software configures and initiates execution of the BIST sequence by loading values into STCU2 registers.

54.7.6 Supported BIST sequences

NXP supports only a specific set of BIST sequences for this chip. For the list of supported BIST sequences, see the chip-specific STCU2 information.

54.8 Functional description

54.8.1 FSM description

The module has three FSM that work together: Master FSM, Loader/Shifter FSM, and the Watchdog FSM. Basically the Master FSM is the core unit of the STCU2 module. It coordinates all the self-test operations and the other FSM. The Loader/Shifter FSM is used to program the MBIST and the LBIST parameters and to retrieve the related results depending on the parameters stored into the STCU2 registers and under the control of the Master FSM. The Watchdog FSM evaluates all the schedule time for MBIST and LBIST and the time-out in case of wrong STCU2 programming.

54.8.2 BIST scheduling

STCU2 is designed to program the parallel/serial execution of the MBIST or LBIST depending on the power, timing, and coverage constraints.

The mechanism used to provide this flexibility is a linked list of BIST descriptors where the starting pointer is defined in CFG[PTR]. The first LBIST is mapped on 0, the second on 1, and so on. A BIST descriptor identifies a LBIST or MBIST via an index that is associated with its control register (STCU2 LBIST Control (LB_CTRL0) or STCU2 MBIST Control (MB_CTRL0 - MB_CTRL18)). The first MBIST is mapped on (00000080h + 0), the second on (00000080h + 1), and so on. The additional pointers of the linked list are in LB_CTRL*n*[PTR] for LBIST *n* (where *n* is the selected LBIST) and MB_CTRL*m*[PTR] for MBIST *m* (where *m* is the selected MBIST) and must be populated depending on the selected run sequence. The additional fields LB_CTRL[CSM] and MB_CTRL[CSM] provide the flexibility to run concurrently or sequentially the chosen set of the LBIST or MBIST or to close the linked list by setting the NIL pointer.

54.8.3 FCCU interface

The FCCU interface is the hardware flag mechanism towards the system, to indicate the occurrence of an Unrecoverable fault and/or a Recoverable fault failure during the Self Test sequence. Two independent signals have been used to mitigate the common cause faults.

To diagnose physical defects on the two fault signals, a fault injection mechanism is also provided. In this case, the FCCU interface allows the user application to check the integrity of the UF and RF connection lines between the STCU2 and the FCCU. Refer to the description of FCCU fault injection mechanism to understand how the UF/RF set/clear mechanism works.

54.8.4 Watchdogs

The STCU2 implements different watchdogs to ensure that operations are finished in time.

54.8.4.1 BIST watchdog timer

The LBIST and MBIST execution time has to be configured as described in STCU2 Watchdog Granularity (WDG), to account for the overall execution time of the self test sequence. In case the selected LBISTs or MBISTs are not yet completed during assigned time, the current LBISTs or MBISTs execution is interrupted and a failure is flagged into ERR_STAT[WDTOSW] and MBESWx or LBESW.

In case of multiple sequential run in the same online session and the time-out happens in the middle of a sequential run, the next sequential run will be skipped and the execution ends with the current updated status of the registers reported above.

54.8.4.2 Register write-access watchdog timer

As explained in the STCU2 SK Code (SKC):

- A key mechanism protects STCU2 registers during the self-test configuration phase by preventing any unwanted access.
- A hardware watchdog timer locks register-write access after a number of STCU2 clock cycles. To refresh the hardware watchdog timer before it times out, write Key1 and Key2 sequences.

AUTOLOCK_VALUE is the number of STCU2 clock cycles after which the hardware watchdog timer locks register-write access. See the chip-specific STCU2 information for the value of AUTOLOCK_VALUE.

The hardware watchdog timer is particularly useful in case CFG[WRP] is 0 during the software self-test configuration. In this case, the software application might enable write access to the STCU2 registers.

54.9 STCU2 register descriptions

The STCU2 registers are listed in this section.

NOTE

Always write the reset value for the Reserved fields.

During online self-test, after the STCU2 registers are configured, they must not be overridden via internal peripheral system until the self-test is complete.

54.9.1 STCU2 memory map

STCU base address: 403A_0000h

Offset	Register	Width (In bits)	Access	Reset value
4h	STCU2 Run Software (RUNSW)	32	RW	0000_0000h
8h	STCU2 SK Code (SKC)	32	W	0000_0000h
Ch	STCU2 Configuration (CFG)	32	RW	0000_0000h
14h	STCU2 Watchdog Granularity (WDG)	32	RW	0000_FFFFh
24h	STCU2 Error (ERR_STAT)	32	RW	0000_0000h
28h	STCU2 Error FM (ERR_FM)	32	RW	0000_0000h
4Ch	STCU2 Online LBIST Status (LBSSW0)	32	R	0000_0000h
5Ch	STCU2 Online LBIST End Flag (LBESW0)	32	R	0000_0000h
7Ch	STCU2 Online LBIST Unrecoverable FM (LBUFM0)	32	RW	0000_0000h
10Ch	STCU2 Online MBIST Status (MBSSW0)	32	R	0000_0000h
14Ch	STCU2 Online MBIST End Flag (MBESW0)	32	R	0000_0000h
18Ch	STCU2 MBIST Unrecoverable FM (MBUFM0)	32	RW	0000_0000h
200h	STCU2 LBIST Control (LB_CTRL0)	32	RW	0000_0000h
204h	STCU2 LBIST PC Stop (LB_PCS0)	32	RW	0000_0000h
220h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW0)	32	RW	FFFF_FFFFh
224h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW0)	32	RW	FFFF_FFFFh
228h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW0)	32	R	0000_0000h
22Ch	STCU2 Online LBIST MISR Read High (LB_MISRRHSW0)	32	R	0000_0000h
2200h	STCU2 Algorithm Select (ALGOSEL)	32	RW	0000_0000h
220Ch	STCU2 MBIST Stagger (STGGR)	32	RW	0000_0000h
2210h	STCU2 BIST Start (BSTART)	32	RW	0000_0000h
2214h - 225Ch	STCU2 MBIST Control (MB_CTRL0 - MB_CTRL18)	32	RW	0000_0000h

54.9.2 STCU2 Run Software (RUNSW)

Offset

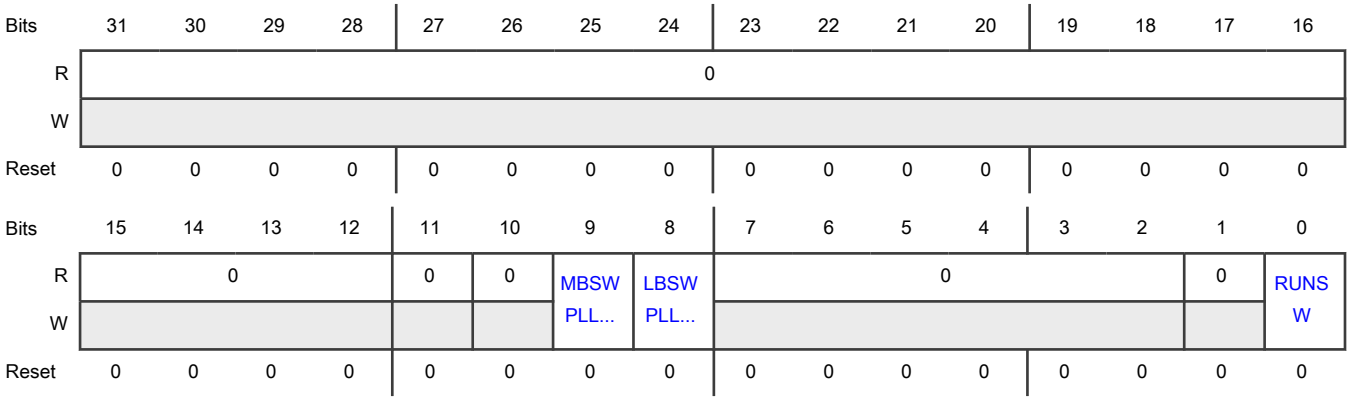
Register	Offset
RUNSW	4h

Function

The RUNSW register defines the RUNSW bit to start the online self-testing procedure.

The R/W fields in this register are readable at any time. However, you can write to these fields only when CFG[WRP] = 0.

Diagram



Fields

Field	Function
31-12 —	Reserved
11 —	Reserved
10 —	Reserved
9 MBSWPLLEN	Online MBIST with PLL Enabled 0b - Online MBIST is executed without using the on-chip PLL. 1b - Online MBIST is executed using the PLL configuration provided by software. STCU2 does not take the PLL control but monitors the PLL lock signal to check if PLL is working correctly.
8 LBSWPLLEN	Online LBIST with PLL Enabled 0b - Online LBIST is executed without using the on-chip PLL. 1b - Online LBIST is executed using the PLL configuration provided by the software. STCU2 does not take the PLL control but monitors the PLL lock signal to check if PLL is working correctly.
7-2 —	Reserved
1 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 RUNSW	The RUNSW bit is automatically cleared by STCU2 when the online self-testing procedure is complete. 0b - Idle 1b - Online self-testing procedure is running

54.9.3 STCU2 SK Code (SKC)

Offset

Register	Offset
SKC	8h

Function

The SKC register implements the security key code mechanism needed to access in write mode to the other STCU2 registers.

To unlock STCU2 access after the STCU2 asynchronous reset and at the end of the STCU2 run, the software (IPS bus) need to apply the following sequence:

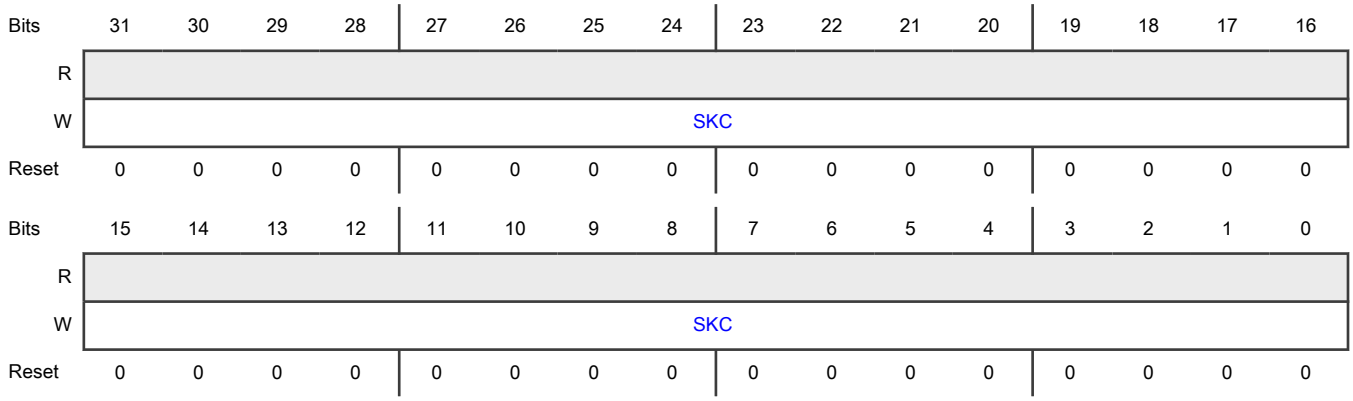
- write the key1 into the SKC register
- write the key2 into the SKC register

Depending on the online test step, the two keys are different. Byte write operation is not allowed because the full key has to be recognized as one unit.

A hard-coded WDG counter (Register write-access watchdog timer) starts decrementing its value right after POR is deasserted. Therefore, user must ensure to complete unlocking and programming sequence before its expiry. In case the STCU2 register access lasts more cycles than the one defined in the hard-coded WDG timeout, the STCU2 write access is locked and the WDG and register interface clocks are switched off. Also, in this case, to enable the write access to the STCU2 again and the WDG and register interface clocks, it is required to apply the sequence again. In case of invalid access or sequence (Key1/2 have to be applied consecutively), a transfer error on the IPS is asserted. The STCU2 write access is locked and to unlock the access, the sequence has to be applied again. This can also happen if watchdog expires between key1 and key2. In this case, user needs to apply the key1-key2 sequence again. In case it is required to extend the STCU2 register access cycles before the hard-coded WDG timeout expires, Key1 and Key2 sequences need to be applied. The effect of this write operation is to re-initialize the WDG timeout counter. The STCU2 write access is locked and to unlock the access, the sequence has to be applied again.

The SKC register is not readable. The value 00000000h is always returned in case of read operation.

Diagram



Fields

Field	Function
31-0 SKC	STCU2 SK Code STCU2 security key code for online test = 753F924Eh: Key1 to unlock the write access the STCU2 = 8AC06DB1h: Key2 to unlock the write access the STCU2

54.9.4 STCU2 Configuration (CFG)

Offset

Register	Offset
CFG	Ch

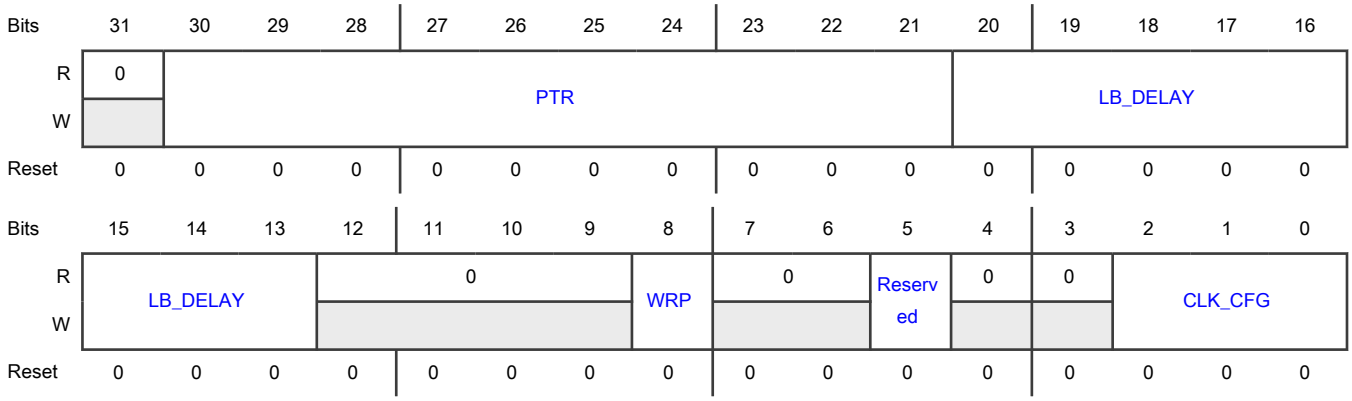
Function

The CFG register includes the global configuration of the STCU2 and can be updated in the online test steps.

The access to this register is described in the following figure. It further depends on the state of the WRP bit as follows:

- When WRP = 0: The register can be written during the online self-test case without restrictions.
- When WRP = 1:
  - The only bit that can be written without any restriction is WRP.
  - In case there are software operations that write all the register's bits, a transfer error is raised only if the value of the selected byte written differs from the current status of the register. This functionality has been implemented to prevent potential compilation behavior that might invalidate this single bit clean capability.

Diagram



Fields

Field	Function
31 —	Reserved
30-21 PTR	First LBIST or MBIST pointer PTR defines the logical pointer to the first LBIST or MBIST to be scheduled when the self-testing procedure is enabled. PTR is the entry pointer to a linked list of BIST descriptors. If PTR = 0 then the LBIST0 is initially scheduled. See BIST scheduling for details. 0h to (01h - 1): pointer to LBIST 00000080h to (00000080h + 013h - 1): pointer to MBIST 3FFh: pointer to NIL. No BIST execution. others: invalid pointer => an error is set into the ERR_STAT[INVPSW].
20-13 LB_DELAY	Delay LBIST run LB_DELAY defines the delay between the LBIST starts when more than a single LBIST is selected to be executed concurrently with the purpose of smoothing the power consumption transient. The allowed delay time are these: 00h: No delay 01h: 1 x 16 STCU2 CORE_CLK cycles 02h: 2 x 16 STCU2 CORE_CLK cycles 03h: 3 x 16 STCU2 CORE_CLK cycles ... FDh: 253 x 16 STCU2 CORE_CLK cycles FEh: 254 x 16 STCU2 CORE_CLK cycles FFh: 255 x 16 STCU2 CORE_CLK cycles
12-9	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
8 WRP	Write Protection 0: Specific STCU2 registers can be written through IPS bus interface 1: STCU2 registers cannot be written through IPS, preventing any user application write operation
7-6 —	Reserved
5 —	Reserved
4 —	Reserved
3 —	Reserved
2-0 CLK_CFG	Logic, Memory BIST, and STCU2 CORE_CLK configuration CLK_CFG defines the ratio between the sys_clk and the internal clock used to program both the LBIST and the MBIST and the STCU2 CORE_CLK. The punch-out mechanism is used to generate the derived clocks. The following configurations are allowed: 000b - sys_clk/1 001b - sys_clk/2 010b - sys_clk/3 011b - sys_clk/4 100b - sys_clk/5 101b - sys_clk/6 110b - sys_clk/7 111b - sys_clk/8

54.9.5 STCU2 Watchdog Granularity (WDG)

Offset

Register	Offset
WDG	14h

Function

The WDG register defines the time budget of LBIST and MBIST execution providing a protection mechanism in case of dead-lock or endless conditions during the self-test procedure.

In case online self-test sequence is run, it defines the timeout of the execution run.

The R/W fields in this register are readable at any time. You can write to these fields when online self-test phase is active and CFG[WRP] = 0.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	WDGEOC															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	WDGEOC															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-0 WDGEOC	Watchdog End of Count Timer This value has to be set to define the time budget related to the online self-test execution and check that everything is correctly working within this slot of time. The delay time slots that are allowed are as follows: 0000 0000h: 1 x 16 STCU2 CORE_CLK cycles 0000 0001h: 2 x 16 STCU2 CORE_CLK cycles 0000 0002h: 3 x 16 STCU2 CORE_CLK cycles ... FFFF FFFDh: 4294967294 x 16 STCU2 CORE_CLK cycles FFFF FFFEh: 4294967295 x 16 STCU2 CORE_CLK cycles FFFF FFFFh: 4294967296 x 16 STCU2 CORE_CLK cycles

54.9.6 STCU2 Error (ERR_STAT)

Offset

Register	Offset
ERR_STAT	24h

Function

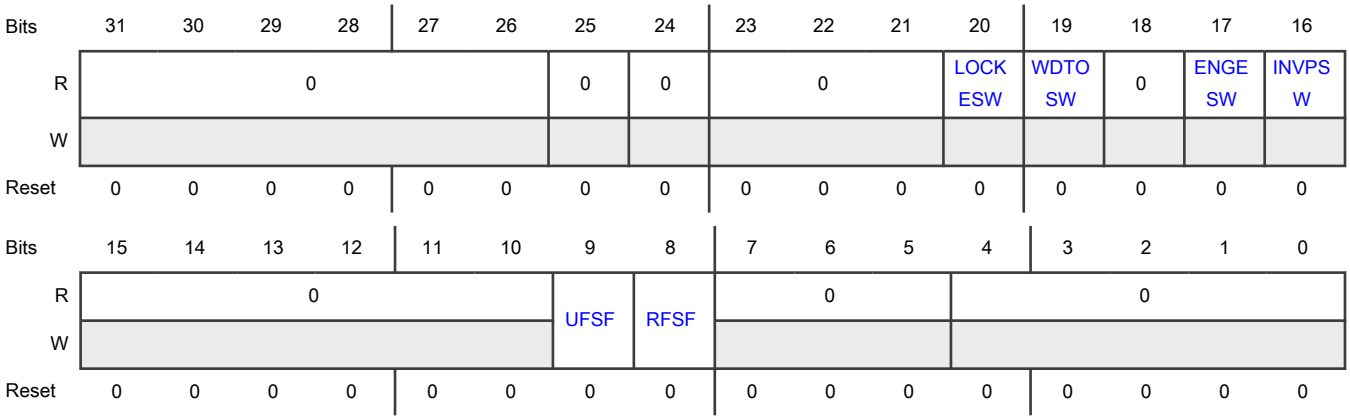
The ERR_STAT register includes the status flags related to the STCU2 internal error conditions occurred during the configuration or the online self-testing execution.

The UFSF and RFSF can be set/cleared using the FCCU dedicated channels.

The access to this register is described in the following figure and as follows:

- If you select the byte write capability to write only UFSF and RFSF, then there is no restriction in writing these bits.
- If your software performs the write operations on other bits besides UFSF/RFSF, then a transfer error is generated only if the value you are writing to the other bits differs from their value currently stored in the register. This functionality has been implemented to prevent potential compilation behavior that might invalidate the UFSF/RFSF single bit set/clean capability.

Diagram



Fields

Field	Function
31-26 —	Reserved
25 —	Reserved
24 —	Reserved
23-21 —	Reserved
20 LOCKESW	Online LOCK error You can always read this field. The content of this field is initialized to its reset value when RUNSW[RUNSW] is set to 1. 0b - In case PLL is enabled, it is correctly locked during the self-test sequence 1b - When the PLL is enabled, this flag highlights that there has been an unexpected PLL unlock(loss-of-lock) event during the online self-test sequence execution. The online self-test run is stopped and the status of the currently running LBISTs or MBISTs is saved in the related registers. The LOCK signal is monitored during the LBIST run when RUNSW[LBSWPLLEN] is set and/or during the MBIST run when RUNSW[MBSWPLLEN] = 1.
19 WDTOSW	Online watchdog timeout

Table continues on the next page...

Table continued from the previous page...

Field	Function
	You can always read this field. The content of this field is initialized to its reset value when RUNSW[RUNSW] is set to 1. 0b - LBIST and MBIST time slots completed within the assigned watchdog time. 1b - LBIST and MBIST time slots not completed within the assigned watchdog time or there are internal mismatches among End of Execution signals.
18 —	Reserved
17 ENGESW	Online engine error You can always read this field. The content of this field is initialized to its reset value when RUNSW[RUNSW] is set to 1. 0b - Valid engine execution 1b - Invalid engine execution. The error conditions that set this bit are FSM, protocol error, and so on.
16 INVPSW	Online invalid pointer You can always read this field. The content of this field is initialized to its reset value when RUNSW[RUNSW] is set to 1. 0b - Valid linked pointer list 1b - Invalid linked pointer list. The following conditions set this bit: Initial LBIST or MBIST pointer is out of range; LBIST is selected when MBIST is concurrently running or vice versa; Error in the LBIST/MBIST linking (execution generates an infinite loop).
15-10 —	Reserved
9 UFSF	Unrecoverable Faults Status Flag This flag reports the global status of the Unrecoverable Faults(UF). This field can be set or cleared using the FCCU dedicated channel, and can also be set or cleared by software. 0b - No errors that trigger the UF condition. 1b - There are errors that trigger the UF condition.
8 RFSF	Recoverable Faults Status Flag This flag reports the global status of the Recoverable Fault (RF). This field can be set or cleared using the FCCU dedicated channel, and can also be set or cleared by software. 0b - No errors that trigger the Recoverable Faults condition 1b - There are errors that trigger the Recoverable Faults condition
7-5 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
4-0 —	Reserved Write may occur on these fields but writes have no effect.

54.9.7 STCU2 Error FM (ERR_FM)

Offset

Register	Offset
ERR_FM	28h

Function

The ERR_FM register defines the fault mapping of the STCU2 faults described in the register ERR_STAT in terms of UF or RF. All sources of internal faults can be routed to UF and RF.

The R/W fields in this register are readable at any time. You can write to these fields when online self-test phase is active and CFG[WRP] = 0.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0											LOCKEUFM	WDTOUFM	Reserved	ENGEUFM	INVPUFM
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-5 —	Reserved
4 LOCKEUFM	PLL LOCK Unrecoverable Fault Mapping 0b - Recoverable Fault Mapping 1b - Unrecoverable Fault Mapping

Table continues on the next page...

Table continued from the previous page...

Field	Function
3 WDTOUFM	Watchdog Timeout Unrecoverable Fault Mapping 0b - Recoverable Fault Mapping 1b - Unrecoverable Fault Mapping
2 —	Reserved
1 ENGEUFM	Engine Error Unrecoverable Fault Mapping 0b - Recoverable Fault Mapping 1b - Unrecoverable Fault Mapping
0 INVPUFM	Invalid Pointer Unrecoverable Fault Mapping 0b - Recoverable Fault Mapping 1b - Unrecoverable Mapping

54.9.8 STCU2 Online LBIST Status (LBSSW0)

Offset

Register	Offset
LBSSW0	4Ch

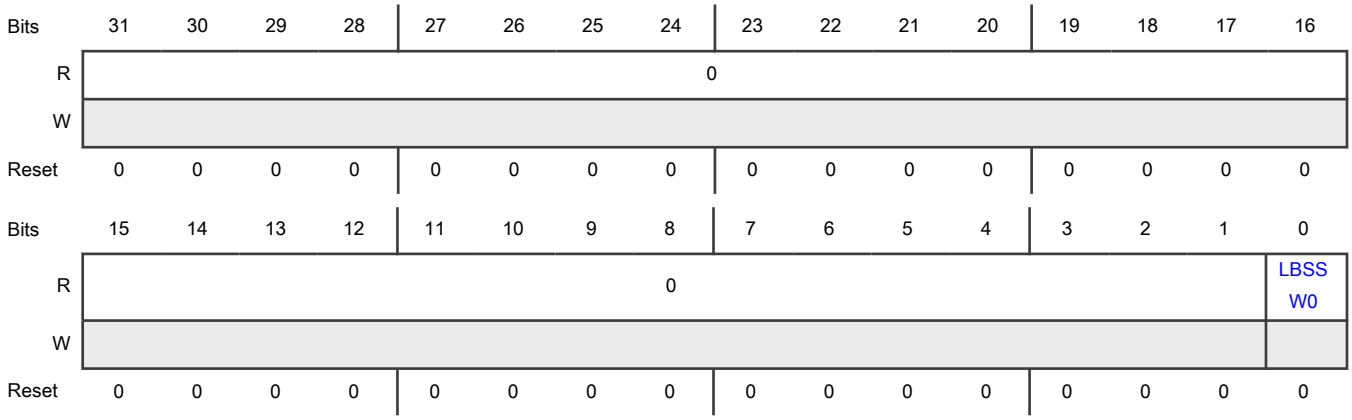
Function

This register includes the results corresponding to the execution of the selected online LBIST.

The size of the register depends on the number of LBIST .

The content of this register is initialized to its reset value when RUNSW[RUNSW] is set to 1.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 LBSSWn	LBSSWn online status of the selected LBIST 0b - Failed LBIST execution 1b - Successful LBIST execution

54.9.9 STCU2 Online LBIST End Flag (LBESW0)

Offset

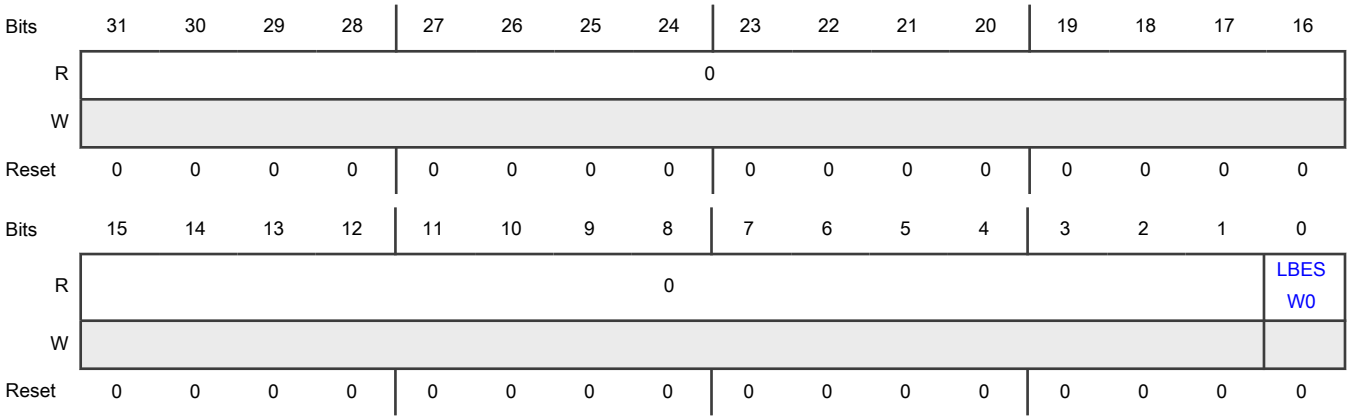
Register	Offset
LBESW0	5Ch

Function

This register includes the results corresponding to the execution of the selected online LBIST.

The size of the register depends on the number of LBIST .

Diagram



Fields

Field	Function
31-1 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 LBESWn	LBESW LBESWx: online LBIST end status 0b - LBIST execution not yet completed 1b - LBIST execution finished

54.9.10 STCU2 Online LBIST Unrecoverable FM (LBUFM0)

Offset

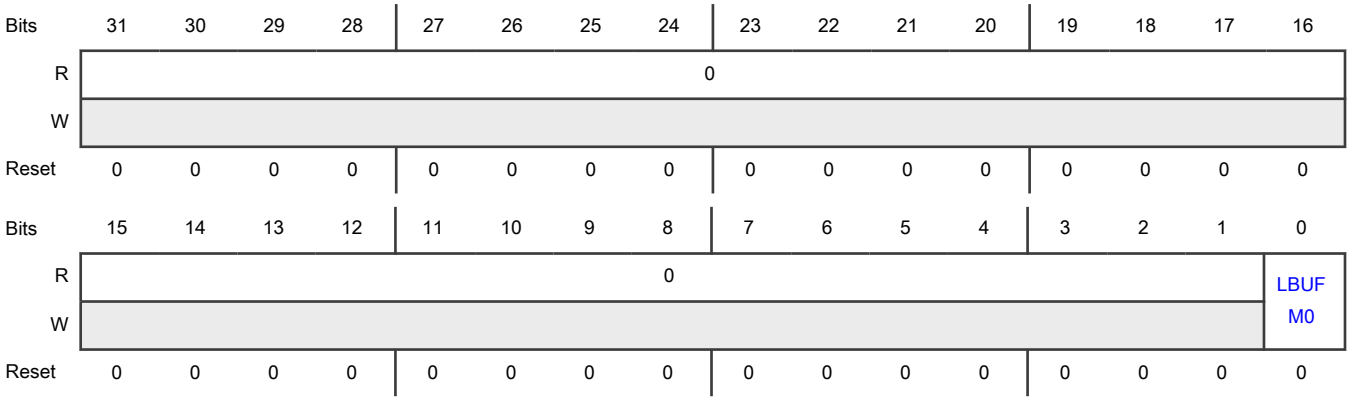
Register	Offset
LBUFM0	7Ch

Function

This register defines the fault mapping of each LBIST in terms of UF or RF.

The size of the register 0 depends on the number of LBIST .

Diagram



Fields

Field	Function
31-1 —	Reserved
0 LBUFMn	LBIST Unrecoverable Fault Mapping 0b - Recoverable Fault mapping 1b - Unrecoverable Fault mapping

54.9.11 STCU2 Online MBIST Status (MBSSW0)

Offset

Register	Offset
MBSSW0	10Ch

Function

This register includes the results corresponding to the execution of the selected online MBIST in the range 0:18.

The size of the register depends on the number of BISTed RAMs/ROMs.

The content of this register is initialized to its reset value when RUNSW[RUNSW] is set to 1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0													MBSSW18	MBSSW17	MBSSW16
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MBSSW15	MBSSW14	MBSSW13	MBSSW12	MBSSW11	MBSSW10	MBSSW9	MBSSW8	MBSSW7	MBSSW6	MBSSW5	MBSSW4	MBSSW3	MBSSW2	MBSSW1	MBSSW0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-19 —	Reserved
18-0 MBSSWn	MBSSW Online status of the selected MBISTn (where n = 18:0). 0b - Failed MBIST execution 1b - Successful MBIST execution

54.9.12 STCU2 Online MBIST End Flag (MBESW0)

Offset

Register	Offset
MBESW0	14Ch

Function

This register includes the End Flag related to the execution of the selected online MBIST in the range 0:18.

The size of the register depends on the number of BISTed RAMs/ROMs.

The content of this register is initialized to its reset value when RUNSW[RUNSW] is set to 1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0													MBESW18	MBESW17	MBESW16
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MBESW15	MBESW14	MBESW13	MBESW12	MBESW11	MBESW10	MBESW9	MBESW8	MBESW7	MBESW6	MBESW5	MBESW4	MBESW3	MBESW2	MBESW1	MBESW0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-19 —	Reserved
18-0 MBESWn	Online MBISTn (where n = 18:0) end status. 0b - MBIST execution not yet completed 1b - MBIST execution finished

54.9.13 STCU2 MBIST Unrecoverable FM (MBUFM0)

Offset

Register	Offset
MBUFM0	18Ch

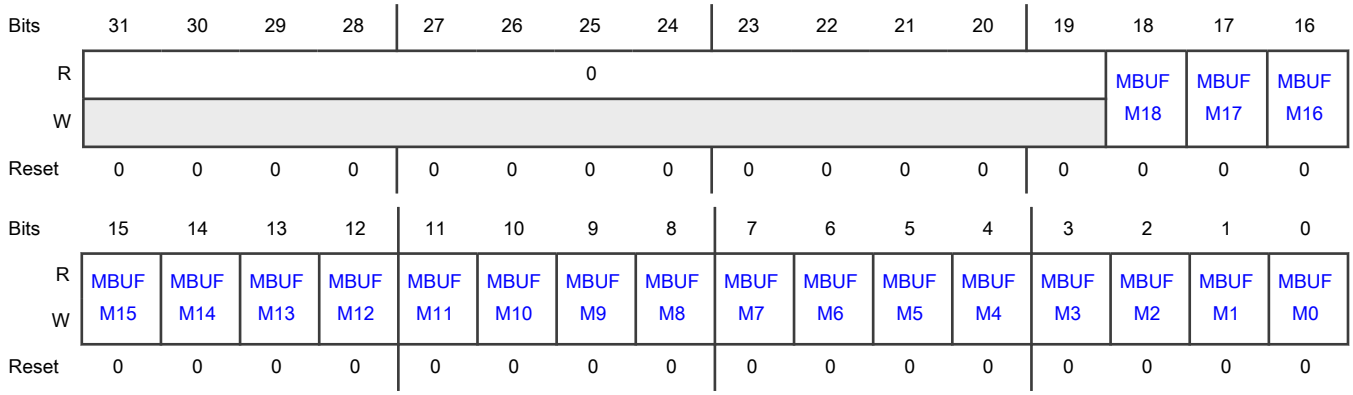
Function

This register defines the fault mapping, in terms of UF or RF, of the MBIST in the range 0:18

The size of the register depends on the number of BISTed RAMs/ROMs.

The R/W fields in this register are readable at any time. You can write to these fields when online self-test phase is active and CFG[WRP] = 0.

Diagram



Fields

Field	Function
31-19 —	Reserved
18-0 MBUFMn	MBESW Online end status of MBISTn (where n = 18:0). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping

54.9.14 STCU2 LBIST Control (LB_CTRL0)

Offset

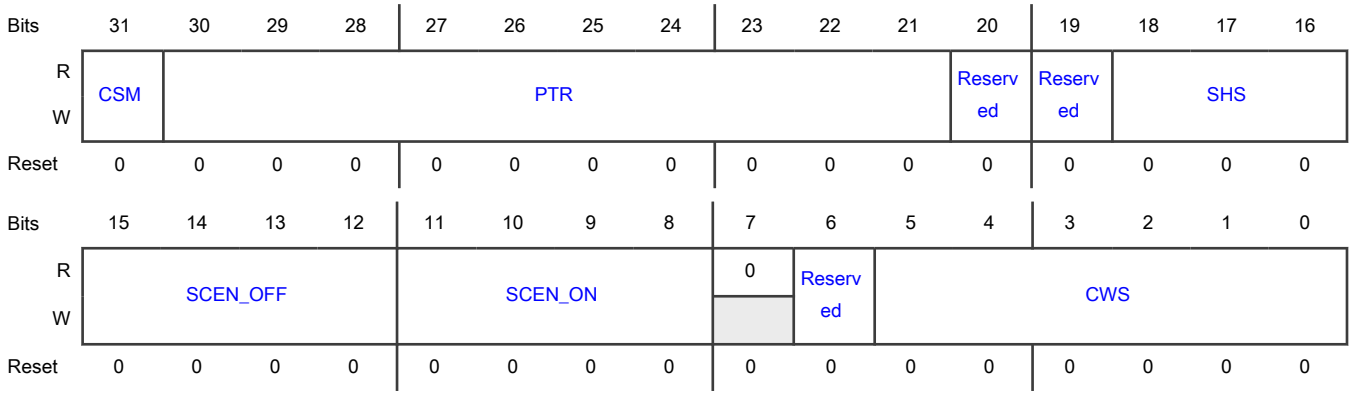
Register	Offset
LB_CTRL0	200h

Function

This register defines the control setting of each LBIST controller.

The R/W fields in this register are readable at any time. You can write to these fields when online self-test phase is active and CFG[WRP] = 0.

Diagram



Fields

Field	Function
31 CSM	Concurrent/sequential mode The next LBIST is scheduled concurrently to the current one if the CSM bit is set to 1; otherwise, it is scheduled sequentially to the completion of the current LBIST execution. 0b - Sequential mode 1b - Concurrent mode
30-21 PTR	Next LBIST or MBIST pointer PTR defines the logical pointer to the next LBIST or MBIST to be scheduled. The next LBIST or MBIST is scheduled concurrently to the current one if the CSM bit is set to 1, otherwise it is scheduled sequentially to the completion of the current LBIST execution. In case of NIL pointer, the CSM bit has to be set Sequential (0) to define this is the end of the list. The self-testing procedure stops after the last BIST in the configuration chain is complete. See BIST scheduling for details. 0h to (01h - 1): pointer to NLBIST-1 00000080h to (00000080h + 013h - 1): pointer to MBIST 3FFh: pointer to NIL. No BIST execution. others: invalid pointer => an error is set into the ERR_STAT[INVPSW].
20 —	Reserved **NOTE** This reserved field is writable but do not write any value to it other than its reset value
19 —	Reserved **NOTE** This reserved field is writable but do not write any value to it other than its reset value
18-16 SHS	Shift speed SHS defines the shift speed

Table continues on the next page...

Table continued from the previous page...

Field	Function
	000b - Shift at full rate (BIST clock). 001b - Shift at 1/2 rate (BIST clock). 010b - Shift at 1/3 rate (BIST clock). 011b - Shift at 1/4 rate (BIST clock). 100b - Shift at 1/5 rate (BIST clock). 101b - Shift at 1/6 rate (BIST clock). 110b - Shift at 1/7 rate (BIST clock). 111b - Shift at 1/8 rate (BIST clock).
15-12 SCEN_OFF	Scan enable OFF SCEN_OFF information is used to configure the lbist controller hardware to generate off_cycles, delay cycles during the scan enable off transition. **NOTE** SCEN_OFF must be programmed to a value >=1. 0000b - 0 delay cycles 0001b - 1 delay cycle 0010b - 2 delay cycles 0011b - 3 delay cycles 0100b - 4 delay cycles 0101b - 5 delay cycles 0110b - 6 delay cycles 0111b - 7 delay cycles 1000b - 8 delay cycles 1001b - 9 delay cycles 1010b - 10 delay cycles 1011b - 11 delay cycles 1100b - 12 delay cycles 1101b - 13 delay cycles 1110b - 14 delay cycles 1111b - 15 delay cycles
11-8 SCEN_ON	Scan enable ON SCEN_ON information is used to configure the lbist controller hardware to generate on_cycles, delay cycles during the scan enable on transition,

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE SCEN_ON delay register value must be programmed to a value >=1 0000b - 0 delay cycles 0001b - 1 delay cycle 0010b - 2 delay cycles 0011b - 3 delay cycles 0100b - 4 delay cycles 0101b - 5 delay cycles 0110b - 6 delay cycles 0111b - 7 delay cycles 1000b - 8 delay cycles 1001b - 9 delay cycles 1010b - 10 delay cycles 1011b - 11 delay cycles 1100b - 12 delay cycles 1101b - 13 delay cycles 1110b - 14 delay cycles 1111b - 15 delay cycles
7 —	Reserved
6 —	Reserved
5-0 CWS	Capture window size CWS defines the capture window size. 00_0000b - Illegal 00_0001b - Controller waits 1 shift cycle for capture to finish. 00_0010b - Controller waits 2 shift cycles for capture to finish. 00_0011b - Controller waits 3 shift cycles for capture to finish. 00_0100b - Controller waits 4 shift cycles for capture to finish. 00_0101b - Controller waits 5 shift cycles for capture to finish. 00_0110b - Controller waits 6 shift cycles for capture to finish. 00_0111b - Controller waits 7 shift cycles for capture to finish.

54.9.15 STCU2 LBIST PC Stop (LB_PCS0)

Offset

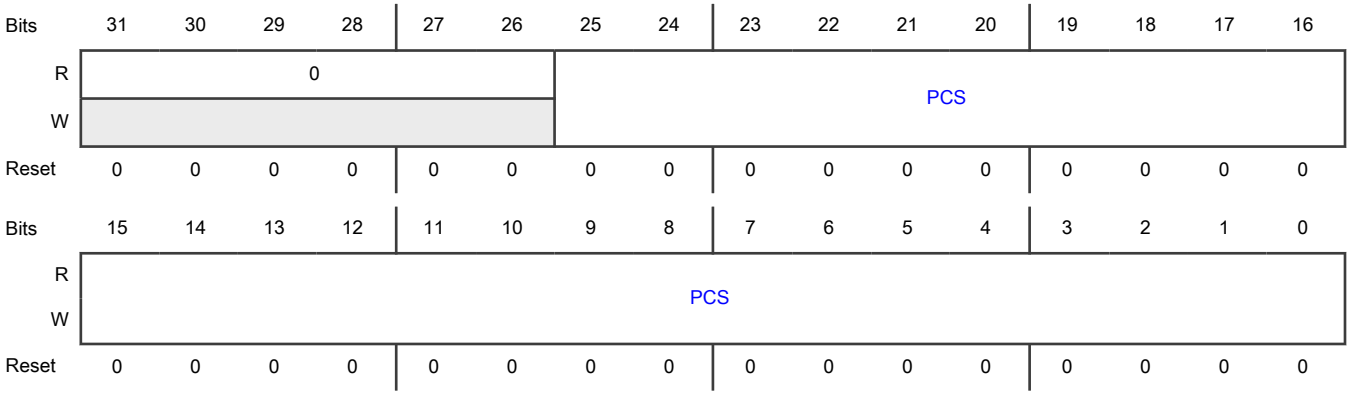
Register	Offset
LB_PCS0	204h

Function

This register defines the pattern counter stop of each LBIST controller.

The R/W fields in this register are readable at any time. You can write to these fields when online self-test phase is active and CFG[WRP] = 0.

Diagram



Fields

Field	Function
31-26 —	Reserved
25-0 PCS	PCS Pattern counter stop PCS defines the pattern counter stop value.

54.9.16 STCU2 Online LBIST MISR Expected Low (LB_MISRELSW0)

Offset

Register	Offset
LB_MISRELSW0	220h

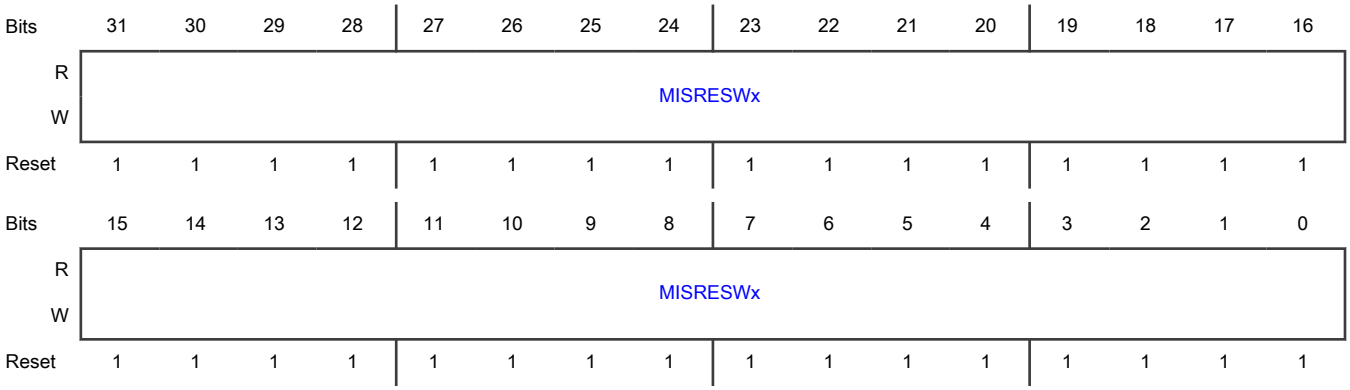
Function

This register defines bits 32 of the expected MISR of the online LBIST controller.

The size of the register depends on the number of MISR bits of the related LBIST. .

The R/W fields in this register are readable at any time. You can write to these fields when online self-test phase is active and CFG[WRP] = 0.

Diagram



Fields

Field	Function
31-0 MISRESWx	Online MISR expected low bits This field defines 32 bits of the expected MISR.

54.9.17 STCU2 Online LBIST MISR Expected High (LB_MISREHSW0)

Offset

Register	Offset
LB_MISREHSW0	224h

Function

The size of the register depends on the number of MISR bits of the related LBIST. .

The R/W fields in this register are readable at any time. You can write to these fields when online self-test phase is active and CFG[WRP] = 0.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MISRESWx															
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MISRESWx															
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-0 MISRESWx	Online MISR Expected High Bits This field defines the 32 bits of the expected MISR.

54.9.18 STCU2 Online LBIST MISR Read Low (LB_MISRRLSW0)

Offset

Register	Offset
LB_MISRRLSW0	228h

Function

This register reports 32 bits of the MISR obtained at the end of the online LBIST controller execution.

The size of the register depends on the number of MISR bits of the related LBIST. .

The content of this register is initialized to its reset value when RUNSW[RUNSW] is set to 1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MISRRSWx															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MISRRSWx															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 MISRRSWx	MISRRSWx Online MISR Read Low Bin This field is equivalent to 32 bits of the MISR obtained at the end of the online LBIST execution.

54.9.19 STCU2 Online LBIST MISR Read High (LB_MISRRHSW0)

Offset

Register	Offset
LB_MISRRHSW0	22Ch

Function

The size of the register depends on the number of MISR bits of the related LBIST. .

The content of this register is initialized to its reset value when RUNSW[RUNSW] is set to 1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MISRRSWx															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MISRRSWx															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 MISRRSWx	MISRRSWx Online MISR Read High Bits This field is equivalent to 32 bits of the MISR obtained at the end of the online LBIST execution.

54.9.20 STCU2 Algorithm Select (ALGOSEL)

Offset

Register	Offset
ALGOSEL	2200h

Function

This is a 32-bit register intended to be programmed by the user to select algorithms to be run on BIST. See the chip-specific STCU2 information for details of this register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	ALGO SEL...	ALGO SEL...	ALGO SEL...	ALGO SEL...	ALGO SEL...	ALGO SEL...	ALGO SEL...	ALGO SEL...	ALGO SEL...	ALGO SEL...	ALGO SEL...	ALGO SEL...	ALGO SEL...	ALGO SEL...	ALGO SEL...	ALGO SEL...
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ALGO SEL...	ALGO SEL...	ALGO SEL...	ALGO SEL...	ALGO SEL...	ALGO SEL...	ALGO SEL9	ALGO SEL8	ALGO SEL7	ALGO SEL6	ALGO SEL5	ALGO SEL4	ALGO SEL3	ALGO SEL2	ALGO SEL1	ALGO SEL0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 ALGOSELn	Algorithm Select

54.9.21 STCU2 MBIST Stagger (STGGR)

Offset

Register	Offset
STGGR	220Ch

Function

This register allows one to program number of clock cycles between execution of one BIST and the next one.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	STAG															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	STAG															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 STAG	STAG Number of STCU2 CORE_CLK cycles between execution of one BIST and the next one.

54.9.22 STCU2 BIST Start (BSTART)

Offset

Register	Offset
BSTART	2210h

Function

This is a 32-bit register intended to be programmed by the user to run BISTs with different configuration. See the chip-specific STCU2 information for details of this register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BSTART31	BSTART30	BSTART29	BSTART28	BSTART27	BSTART26	BSTART25	BSTART24	BSTART23	BSTART22	BSTART21	BSTART20	BSTART19	BSTART18	BSTART17	BSTART16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BSTART15	BSTART14	BSTART13	BSTART12	BSTART11	BSTART10	BSTART9	BSTART8	BSTART7	BSTART6	BSTART5	BSTART4	BSTART3	BSTART2	BSTART1	BSTART0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 BSTARTn	BIST Start

54.9.23 STCU2 MBIST Control (MB_CTRL0 - MB_CTRL18)

Offset

For a = 0 to 18:

Register	Offset
MB_CTRLa	2214h + (a × 4h)

Function

The MB_CTRL register defines the control setting of MBIST controller.

The R/W fields in this register are readable at any time. You can write to these fields when online self-test phase is active and CFG[WRP] = 0.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CSM	PTR											0			
W												BSEL				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 CSM	CSM Concurrent/sequential mode 0b - Sequential mode 1b - Concurrent mode
30-21	PTR

Table continues on the next page...

Table continued from the previous page...

Field	Function
PTR	Next LBIST or MBIST pointer PTR defines the logical pointer to the next LBIST or MBIST to be scheduled. The next LBIST or MBIST is scheduled concurrently to the current one if the CSM bit is set to 1; otherwise it is scheduled sequentially to the completion of the current MBIST execution. In case of NIL pointer the CSM bit must be set Sequential (0) to define this is the end of the list. The self-testing procedure stops after the last BIST in the configuration chain is complete. See BIST scheduling for details. **NOTE** If the pointer is invalid and MBIST is scheduled to run concurrently than this invalid scenario is handled by watchdog timeout feature and corresponding watchdog timeout status bit will be updated in ERR_STAT[WDTOSW]. In this particular case ERR_STAT[INVPSW] will not be set. 0h to (01h - 1): pointer to LBIST 00000080h to (00000080h + 013h - 1): pointer to MBIST 3FFh: pointer to NIL. No next BIST execution. others: invalid pointer => an error is set into the ERR_STAT[INVPSW].
20 BSEL	BSEL BIST Select 0b - Selected BIST is not selected for execution. 1b - Selected BIST is selected for execution.
19-0 —	Reserved

54.10 Glossary

BIST	Built-in self-test
BIST Clock	BIST controller clock corresponding to the specific BIST
CORE_CLK	Clock specified by the CFG register
FSM	Finite state machine
IPS	Internal peripheral system
LBIST	Logic BIST
MBIST	Memory BIST
NLBIST	Number of logic built-in self-test controller
NMCUT	Number of memory checked using memory built-in self-test controller
Online self-test phase	In this condition, system is alive and the SW can program STCU2
RF	Recoverable faults
UF	Unrecoverable faults

WDG FSM	Watchdog finite state machine

Chapter 55
Register Protection (REG_PROT)

55.1 Chip-specific REG_PROT information

55.1.1 REG_PROT configuration

The MCU safety relevant configuration registers are protected against unauthorized HW/S changes during read/write/clear access by implementing them with register protection module. See the REG_PROT details file attached to this document.

55.1.1.1 CMU_BRIDGE register protection

The CMUs in the device reside on a common peripheral slot as CMU_BRIDGE in the system memory map. Figure 218 indicates the locations of the CMUs alongwith the memory map.

Based on the CMU location in system peripheral memory map, the below locations in the CMU 16KB space are reserved and any access on these locations results in a bus transfer error.

402B_C018 – 402B_C01F

402B_C030 – 402B_C03F

402B_C050 – 402B_C05F

402B_C078 – 402B_C07F

402B_C098 – 402B_C09F

402B_C0B8 – 402B_C0BF

402B_C0D8 - 402B_CFFF

Since the register protection is available in the device for CMU_BRIDGE, the 4KB offset locations (refers as the Area 2 in register protection) are also reserved and accesses over these regions also result in bus transfer error.

402B_D018 – 402B_D01F

402B_D030 – 402B_D03F

402B_D050 – 402B_D05F

402B_D078 – 402B_D07F

402B_D098 – 402B_D09F

402B_D0B8 – 402B_DFFF

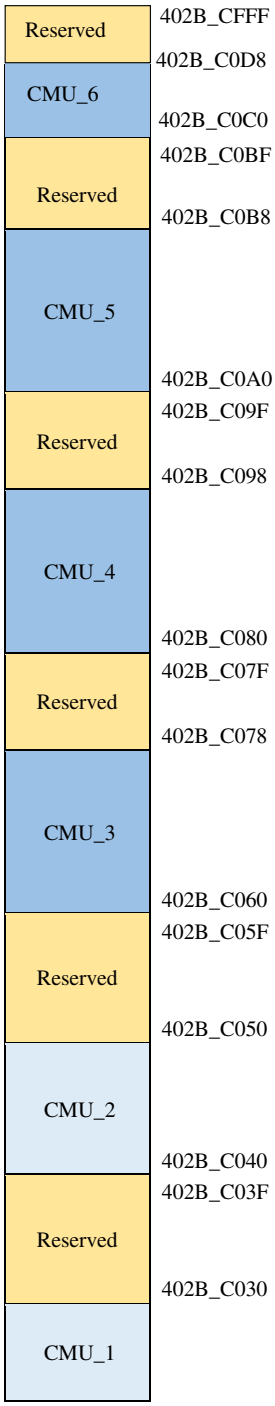


Figure 218. Location of CMUs

55.2 Overview

REG_PROT offers a mechanism to protect defined memory-mapped address locations, in a module under protection, from being written.

REG_PROT is a protection module that is located between the module under protection and the peripheral interface. The address locations to be protected exist in the module under protection and the address locations that can be protected are module specific.

Writes to these protected address locations are locked using a Soft Lock Bit Register (SLBR*n*). Any access to address locations in the module under protection is restricted if their corresponding Soft lock fields in a Soft Lock Bit Register (SLBR*n*) are 1.

The configured soft-lock fields can also be write-restricted by using GCR[HLB]. When GCR[HLB]=1, you cannot write to the soft-lock fields. The address locations in the module under protection can be restricted to Supervisor mode access using GCR[UAA].

55.2.1 Block Diagram

The following figure shows the block diagram of this module.

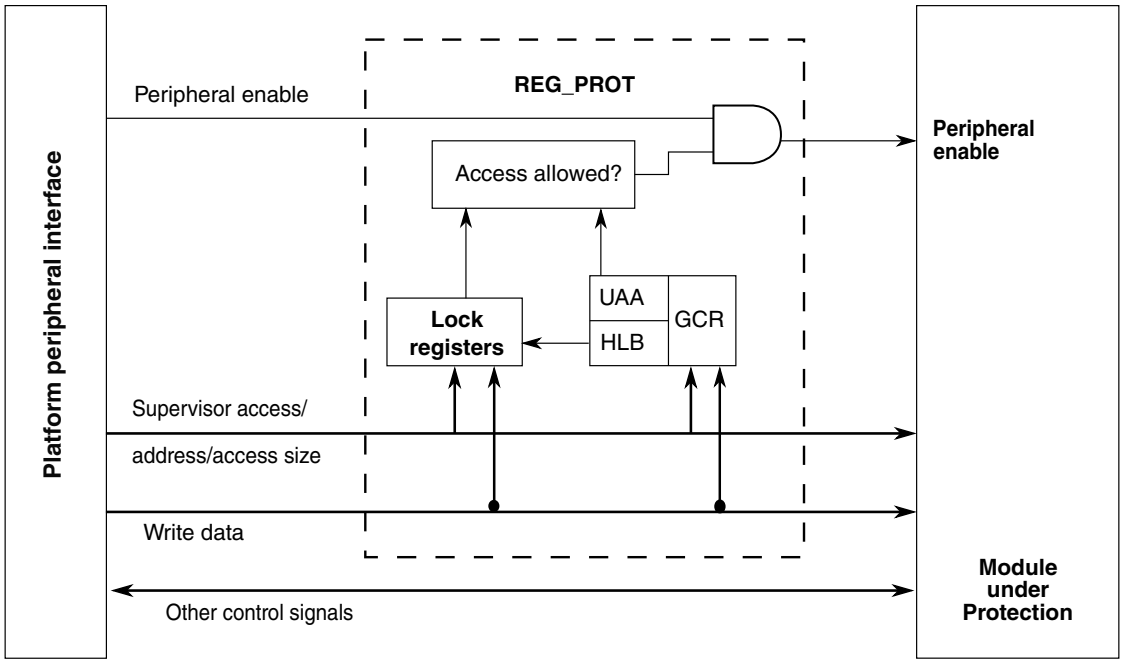


Figure 219. REG_PROT Block diagram

55.2.2 Features

REG_PROT includes these distinctive features:

- Write accesses for the module under protection can be restricted to the supervisor mode only.
- Registers of modules with their register slot size occupying 1 KB to 32 KB of memory-mapped address space, depending on the PROT_MEM parameter, can be dynamically locked. See Memory space and Protection size.
- Multiple ways are present to set the lock bits.
- After the lock bits are configured, they can be protected from changes.

55.3 Functional description

The following sections describe the functional characteristics of the REG_PROT module.

55.3.1 Modes of operation

The mode of REG_PROT depends on the Module Under Protection (MUP). The REG_PROT is operable when the MUP is operable.

The below table shows the write operations in user mode.

Table 320. User mode access table

GCR[UAA]	SLBRn[SLBm]	Write Operation on MUP	Write Operation on REG_PROT
0	0	Transfer error generated, write operation unsuccessful.	Transfer error generated, write operation unsuccessful.
0	1	Transfer error generated, write operation unsuccessful.	Transfer error generated, write operation unsuccessful.
1	0	• If register is implemented and write operation allowed on accessed address location then write operation is successful. • If register is not implemented or write operation is not allowed on accessed address location then transfer error is generated, write operation is unsuccessful.	• If register is implemented and write operation allowed on accessed address location then write operation is successful. • If register is not implemented or write operation is not allowed on accessed address location then transfer error is generated, write operation is unsuccessful.
1	1	Transfer error generated, write operation unsuccessful.	• If register is implemented and write operation allowed on accessed address location then write operation is successful. • If register is not implemented or write operation is not allowed on accessed address location then transfer error is generated.

Note:

- Read operation is always allowed
- n >= 0, m = [0,3]

55.3.2 Memory space and Protection size

This section provides a detailed description of the memory space and protection size of a module using REG_PROT.

The PROT_MEM size varies per chips. It can be:

- 1 KB
- 2 KB
- 4 KB
- 8 KB
- 16 KB
- 32 KB

The resulting memory space is divided into four areas. Memory locations within area #1 and area #2 are a part of the module under protection. Area #3 and area #4 are a part of REG_PROT.

The proper register slot size for each protected module is mentioned in the REG_PROT details file attached to this document.

Following is the memory space for a module that is protected by the REG_PROT module. Reserved registers in area #1 are handled as specified in the protected module's documentation. There is a single register in configuration space that corresponds

to area #4 while the size mentioned in Area size column is to turn the whole area multiple of 4 KB. See the Offset column to know the exact location of the GCR register.

Table 321. Area offset based on protection size

PROT_MEM	Area	Area size	Use	Offset
1 KB	Area 1	1 KB	Module Register(MR0—MR1023)	0000h—03FFh
	Area 2	1 KB	Module Register and Set Soft Lock Bit (LMR0—LMR1023)	0400h—07FFh
	Area 3	256 B	Soft Lock Bit Register (SLBR0—255)	0800h—08FFh
	Area 4	256 B	Global Configuration Register (GCR)	0900h—09FFh
		1.5 KB	Reserved	0A00h— 0FFFh
2 KB	Area 1	2 KB	Module Register (MR0—MR2047)	0000h—07FFh
	Area 2	2KB	Module Register and Set Soft Lock Bit (LMR0—LMR2047)	0800h—0FFFh
	Area 3	512 B	Soft Lock Bit Register (SLBR0—511)	1000h—11FFh
	Area 4	256 B	Global Configuration Register (GCR)	1200h—12FFh
		3.25 KB	Reserved	1300h—1FFFh
4 KB	Area 1	4 KB	Module Register (MR0—MR4095)	0000h—0FFFh
	Area 2	4 KB	Module Register and Set Soft Lock Bit (LMR0—LMR4095)	1000h—1FFFh
	Area 3	1 KB	Soft Lock Bit Register (SLBR0—1023)	2000h—23FFh
	Area 4	256 B	Global Configuration Register (GCR)	2400h—24FFh
		6.75 KB	Reserved	2500h—3FFFh
8 KB	Area 1	8 KB	Module Register (MR0—MR8191)	0000h—1FFFh
	Area 2	8 KB	Module Register and Set Soft Lock Bit (LMR0—LMR8191)	2000h—3FFFh
	Area 3	2 KB	Soft Lock Bit Register (SLBR0—2047)	4000h—47FFh
	Area 4	256 B	Global Configuration Register (GCR)	4800h—48FFh
		1.75 KB	Reserved	4900h—4FFFh
16 KB	Area 1	16 KB	Module Register (MR0—MR16383)	0000h—3FFFh
	Area 2	16 KB	Module Register and Set Soft Lock Bit (LMR0—LMR16383)	4000h—7FFFh
	Area 3	4 KB	Soft Lock Bit Register (SLBR0—4095)	8000h—8FFFh
	Area 4	4 KB	Global Configuration Register (GCR)	9000h—9FFFh
32 KB	Area 1	32 KB	Module Register (MR0—MR32767)	0000h—7FFFh
	Area 2	32 KB	Module Register and Set Soft Lock Bit (LMR0—LMR32767)	8000h—FFFFh
	Area 3	8 KB	Soft Lock Bit Register (SLBR0—8191)	10000h—11FFFh
	Area 4	4 KB	Global Configuration Register (GCR)	12000h—12FFFh

- Area #1 can take the memory space of 1 KB/2 KB/4 KB/8 KB/16 KB/32 KB, which holds the normal functional module registers and is transparent for all read/write operations.
- Area #2 can take the memory space of 1 KB/2 KB/4 KB/8 KB/16 KB/32 KB and is a mirror of area #1. For an example, if area #1 takes 8 KB of memory space, then read/write access to an offset 2000+X reads/writes the register at offset X. However, a write access to offset 2000+X additionally sets the optional soft lock bits for this offset X in the same cycle

as the register at offset X is written. This provides for an automatic write and lock operation. Not all registers in area #1 need to have protection defined by the associated soft lock bits. For unprotected registers at offset Y, accesses to offset 2000+Y are identical to accesses at offset Y.

- Area #3 can take the memory space of 256 B/512 B/1 KB/2 KB/4 KB/8 KB and hold soft lock bits, one bit per byte in area #1. The four soft lock bits associated with one module register word are arranged at byte boundaries in the memory map. The Soft Lock Bit registers can be directly written using a bit mask.
- Area #4 can take the memory space of 1.75 KB/3.5 KB/7 KB/2 KB/4 KB/4 KB and holds the configuration bits of the protection mode. There is one configuration Hard lock bit per module that prevents all further modifications to the soft lock bits and can only be cleared by a system reset after it is set. When user access allowed bit is set, it allows user access to the protected module.

If you access a locked byte with a write transaction, a bus error is issued to the system and the write transaction is not executed. This is true even if not all accessed bytes are locked.

Accessing unimplemented 32—bit registers in area #3 results in a bus transfer error. In area #4, there will not be any transfer error for the address range mentioned in the table above because these address spaces of area #4 are mapped to the GCR register. Accesses in area #4, beyond the address range mentioned in the Module memory map table, are prohibited and might or might not result into a bus abort.

55.3.2.1 Module register (MR)

Each functional module has its own unique set of registers. This chapter refers to these registers generically as module registers (MR), which exist in the lower module memory space (memory area 1), and receive protection from the REG_PROT module.

55.3.2.2 Module Register and Set Soft Lock Bit (LMR)

This is memory area #2 that provides mirrored access to module registers with the side-effect of setting soft lock bits in case of a write access to a register that is defined as protectable by the locking mechanism. Each MR byte is protectable by one associated bit in SLBR*n* [SLB*m*], according to the mapping described in Soft Lock Bit Register (SLBR*n*).

55.3.3 General

This module provides a generic register (address) write-protection mechanism. The register protection size can be:

- 32 bits (address == multiples of 4)
- 16 bits (address == multiples of 2)
- 8 bits (address == multiples of 1)

The addresses that are protected and the register protection size depends on the chip and/or module.

For all addresses that are protected, there are SLBR*n*[SLB*m*] bits that specify whether the address is locked. When an address is locked, it can only be read but not written in any mode (supervisor/normal). If an address is unprotected, the corresponding SLBR*n*[SLB*m*] bit is always 0, regardless of what the software writes to that field.

NOTE

For more information on register protection specification, see the REG_PROT details file attached to this document.

55.3.4 Change lock settings

To change the setting whether an address is locked or unlocked, the corresponding SLBR*n*[SLB*m*] bit needs to be changed. This can be done using the following methods:

- Set the SLBR*n*[SLB*m*] bit(s) by writing to area #2
- Modify the SLBR*n*[SLB*m*] directly by writing to area #3

Both methods are explained in the following sections.

55.3.4.1 Change lock settings via area #2

You can lock a register after writing to it. Note that reading area #2 does not affect soft lock bits. To do so, you must use area # 3.

The following shows an example with PROT_MEM = 8 KB where SLBs are modified using area #2.

When writing 16-bit to address 0008h, MR9 and MR8 in the protected module are updated. The corresponding lock fields remain unchanged (see the left part of Figure 220).

When writing 16-bit to address 2008h, MR9 and MR8 in the protected module are updated. The corresponding lock fields SLBR2[SLB0:1] become 1, and the lock fields SLBR2[SLB2:3] remain unchanged (see the right part of Figure 220).

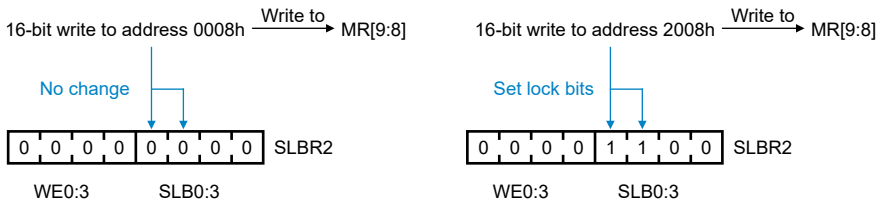


Figure 220. Enable locking via area #2

The following figure shows an example where some addresses are protected and some are not.

In Figure 221, addresses 000Ch and 000Dh are unprotected. Therefore, their corresponding lock fields SLBR3[SLB0:1] are always 0 (shown in bold). During a 32-bit write access to address 200Ch, the lock fields in SLBR3 change as follows:

- SLBR3[SLB2:3] become 1.
- SLBR3[SLB0:1] remain 0.

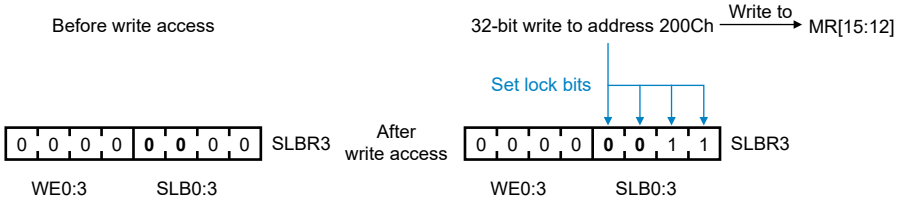


Figure 221. Enable locking for protected and unprotected addresses

55.3.4.2 Change lock settings via area #3

The lock bits are located in area #3 (see Table 321). You can modify them by writing to them. Each SLB*m* field in Soft Lock Bit Register (SLBR*n*) has a mask field, WE*m*, that protects SLB*m* from modification. This masking makes read-modify-write operations unnecessary.

Figure 222 shows two modification examples for registers with 8-bit protection. In part A, a write access to SLBR*n* specifies a mask value that allows modification of all SLBR*n*[SLB0:3] fields. Part B specifies a mask that allows modification only to SLBR*n*[SLB1:3].

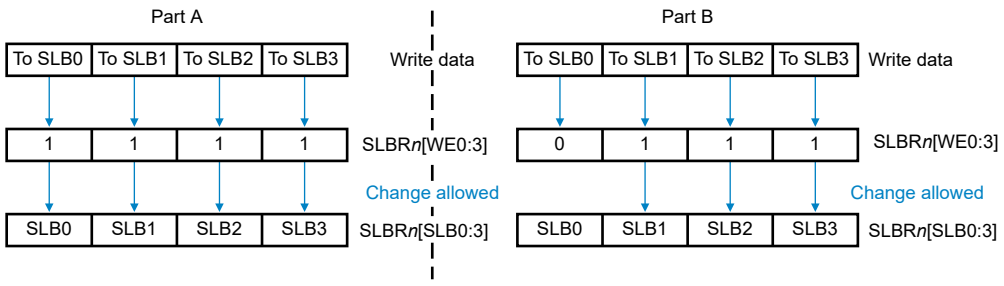


Figure 222. Change lock settings directly via area #3

Figure 223 shows examples for registers with 16-bit protection.

Part A of Figure 223 shows that for SLBR*n*:

- The data written to SLBR*n*[SLB0] is automatically written to SLBR*n*[SLB1].
- The data written to SLBR*n*[SLB2] is automatically written to SLBR*n*[SLB3].

This is because the address reflected by SLBR*n*[SLB0] and SLBR*n*[SLB2] are protected 16-bit wise. Note that in this case, the write enable SLBR*n*[WE0] and SLBR*n*[WE2] must be set while SLBR*n*[WE1] and SLBR*n*[WE3] don't matter.

Part B of Figure 223 shows that the data written to SLBR*n*[SLB0] is automatically written to SLBR*n*[SLB1]. This is done because the address reflected by SLBR*n*[SLB0] has 16-bit protection. In this case, SLBR*n*[WE0] must be 1 and SLBR*n*[WE1] does not matter. SLBR*n*[SLB2] and SLBR*n*[SLB3] remain unchanged because SLBR*n*[WE2] and SLBR*n*[WE3] are 0.

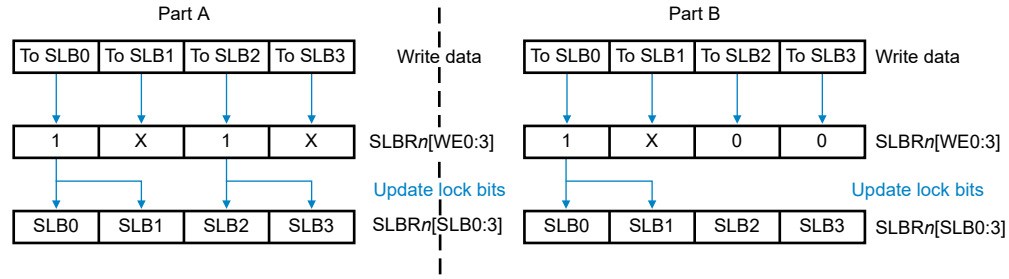


Figure 223. Change lock settings for addresses with 16-bit protection

Figure 224 shows a register with 32-bit protection. In SLBR*n*:

- When SLBR*n*[WE0]=1, the data written to SLBR*n*[SLB0] is automatically written to SLBR*n*[SLB1:3] as well.
- When SLBR*n*[WE0]=0, then SLBR*n*[SLB0:3] remain unchanged. Note that in this case, the write enable SLBR*n*[WE0] must be set while SLBR*n*[WE1:3] does not matter.

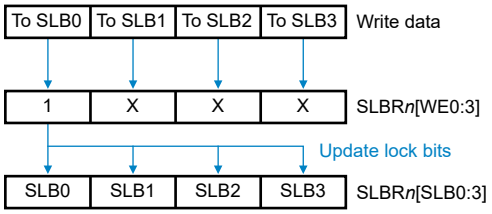


Figure 224. Change lock settings for addresses with 32-bit protection

The following figure shows a mixed protection size configuration.

In SLBR*n*:

- The data written to SLBR*n*[SLB0] is mirrored to SLBR*n*[SLB1] because the corresponding register has 16-bit protection.
- The data written to SLBR*n*[SLB2] is blocked because the corresponding register is unprotected.
- The data written to SLBR*n*[SLB3] is successfully written to SLBR*n*[SLB3] because the corresponding register has 8-bit protection.

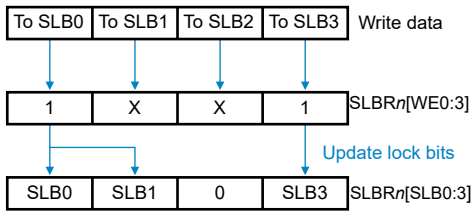


Figure 225. Change lock settings for mixed protection

55.3.4.3 Write protection for locking bits

Changing the locking bits through any of the procedures mentioned in Change lock settings via area #3 and Change lock settings via area #2 is only possible as long as the field GCR.HLB is cleared. After you write 1 to this field, the locking bits cannot be modified until there is a system reset.

55.3.5 Access errors

REG_PROT generates transfer errors under several circumstances, as described below. For the area definition, see Memory space and Protection size.

- If accessing area #1 or area #2, REG_PROT sends any access error from the underlying module under protection.
- If User mode is not allowed, the attempted User mode writes to all areas cause a transfer error, and the writes are blocked.
- If accessing the reserved area, a transfer error is asserted.
- If accessing unimplemented 32-bit registers in areas #3 and #4, a transfer error is asserted.
- If writing to a register in areas #1 and #2, and an SLB field is 1 for any of the affected bytes, then:
  - A transfer error is asserted.
  - The write is blocked.
  - The complete write operation to non-protected bytes in this word is ignored.
- If writing to an SLBR in area #3 with GCR[HLB]=1, a transfer error is asserted.
- Any write operation in any access mode to area #2, when GCR[HLB]=1, is not allowed.

NOTE

The transfer error generated by REG_PROT can only be observed on the CPU that initiates the target register write and has defined the strongly ordered memory region (through its MPU or MMU) covering the target register address.

55.4 External Signals

There are no external signals.

55.5 Initialization

The following sections contain information related to initialization of the Register Protection module.

55.5.1 Sequence

You must configure the REG_PROT first. And then, you need to write 1 to the HLB field in REG_PROT. This ensures that registers under protection are not updated in the user access mode. You need to follow these steps:

1. Enable XRDC protection for the module to enable writes in the supervisor mode only. This step is mandatory, if protected registers of the module are expected to be configured multiple times.
2. Configure REG_PROT to enable write on registers of module under protection.
3. Configure the registers of module under protection.
4. Configure the REG_PROT for the required SLB fields.
5. Set the HLB field in REG_PROT. This step is mandatory, if protected registers of module are expected to be configured only once.

55.5.2 Reset

It is a single clock IP, with single reset.

55.6 REG_PROT register descriptions

This register information documents memory area #3 and area #4.

55.6.1 REG_PROT memory map

REG_PROT base address: base address of protected module instance

Table 322. REG_PROT memory map

PROT_MEM	Offset	Register	Width (In bits)	Access	Reset value
1kB	0x800 – 0x8FF	Soft Lock Bit Register (SLBR0-255)	8	RW	00h
	0x900 – 0x9FF	Global Configuration Register (GCR)	32	RW	0000_0000h
2kB	0x1000 – 0x11FF	Soft Lock Bit Register (SLBR0-511)	8	RW	00h
	0x1200 – 0x12FF	Global Configuration Register (GCR)	32	RW	0000_0000h
4kB	0x2000 – 0x23FF	Soft Lock Bit Register (SLBR0-1023)	8	RW	00h
	0x2400 – 0x24FF	Global Configuration Register (GCR)	32	RW	0000_0000h
8kB	0x4000 – 0x47FF	Soft Lock Bit Register (SLBR0-2047)	8	RW	00h
	0x4800 – 0x48FF	Global Configuration Register (GCR)	32	RW	0000_0000h
16kB	0x8000 – 0x8FFF	Soft Lock Bit Register (SLBR0-4095)	8	RW	00h
	0x9000-0x9FFF	Global Configuration Register (GCR)	32	RW	0000_0000h
32kB	0x10000-0x11FFF	Soft Lock Bit Register (SLBR0-8191)	8	RW	00h
	0x12000 – 0x12FFF	Global Configuration Register (GCR)	32	RW	0000_0000h

55.6.2 Soft Lock Bit Register (SLBR*n*)

Offset

See REG_PROT memory map for SLBR*n* offset ranges. For SLBR*n* addresses, see the REG_PROT details file attached to this document.

Function

These registers hold the Soft Lock Bits (SLBs) for the protected registers in memory area #1, which is the normal register address space of the protected module. Multiple Soft Lock Bit Registers (SLBR*n*) can be implemented, depending on the number of protected module register bytes. Each SLBR*n* has four soft lock fields (SLB0-SLB3), each of which controls write access to a byte in memory area #1. Each soft lock field also has a corresponding write enable field in the same register that controls whether the soft lock field can be written. The following table shows the mapping between the soft lock fields to the bytes in memory area #1.

Table 323. SLBs vs. protected address

SLB	Protected address
SLBR0[SLB0]	MR0
SLBR0[SLB1]	MR1
SLBR0[SLB2]	MR2
SLBR0[SLB3]	MR3
SLBR1[SLB0]	MR4
SLBR1[SLB1]	MR5
SLBR1[SLB2]	MR6
SLBR1[SLB3]	MR7
SLBR2[SLB0]	MR8
...	...

Diagram

Bits	7	6	5	4	3	2	1	0
R	0	0	0	0	SLB0	SLB1	SLB2	SLB3
W	WE0	WE1	WE2	WE3				
Reset	0	0	0	0	0	0	0	0

Fields

Field	Function
7 WE0	Write enable fields for SLB WE0 enables writing to SLB0. 0b - SLB is not modified. 1b - Value is written to SLB.
6 WE1	Write enable fields for SLB WE1 enables writing to SLB1. 0b - SLB is not modified. 1b - Value is written to SLB.
5	Write enable fields for SLB

Table continues on the next page...

Table continued from the previous page...

Field	Function
WE2	WE2 enables writing to SLB2. 0b - SLB is not modified. 1b - Value is written to SLB.
4 WE3	Write enable bits for SLB WE3 enables writing to SLB3. 0b - SLB is not modified. 1b - Value is written to SLB.
3 SLB0	Soft lock fields for one MRn register SLB0 can block accesses to MR[n *4 + 0]. 0b - Associated MRn byte is unprotected and writable. 1b - Associated MRn byte is locked against write accesses.
2 SLB1	Soft lock bits for one MRn register SLB1 can block accesses to MR[n *4 + 1]. 0b - Associated MRn byte is unprotected and writable. 1b - Associated MRn byte is locked against write accesses.
1 SLB2	Soft lock fields for one MRn register SLB2 can block accesses to MR[n *4 + 2]. 0b - Associated MRn byte is unprotected and writable. 1b - Associated MRn byte is locked against write accesses.
0 SLB3	Soft lock fields for one MRn register SLB3 can block accesses to MR[n *4 + 3]. 0b - Associated MRn byte is unprotected and writable. 1b - Associated MRn byte is locked against write accesses.

55.6.3 Global Configuration Register (GCR)

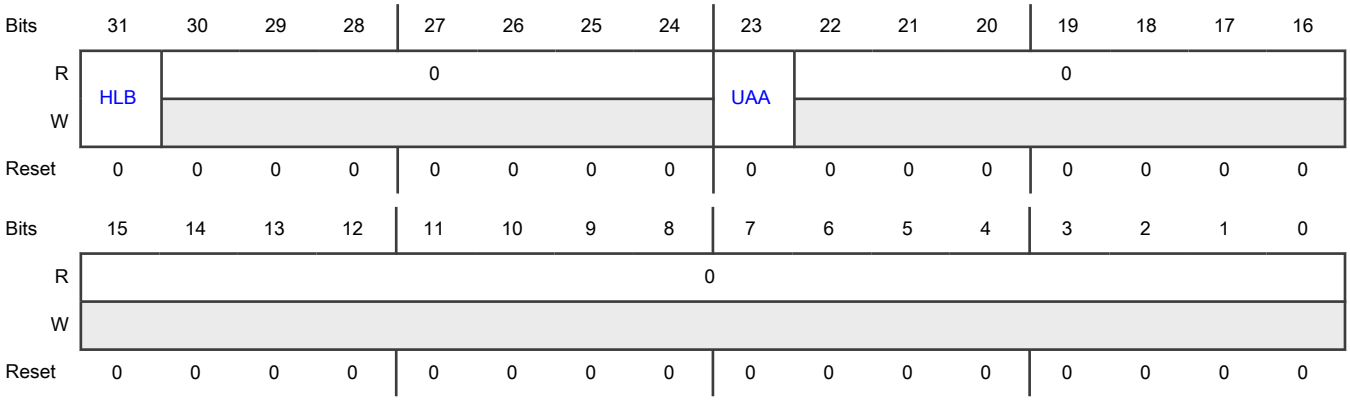
Offset

See REG_PROT memory map for GCR offset.

Function

This register controls module level configurations.

Diagram



Fields

Field	Function
31 HLB	Hard Lock Bit This field cannot be cleared after it is set by the software. It can only be cleared by a system reset. 0b - All SLB fields are accessible and can be modified. 1b - All SLB fields are write protected and cannot be modified. **NOTE** Once this bit is set, it will set all registers under the same PROT_MEM region.
30-24 —	Reserved
23 UAA	User Access Allowed Controls the user and supervisor mode access to registers of the module under protection. 0b - The registers in the module under protection can only be written in the supervisor mode (user access is not allowed). All the write accesses in user (non-supervisor) mode are not executed and a transfer error is issued. This access restriction is in addition to any access restrictions imposed by the protected IP module. 1b - The registers in the module under protection can be accessed in the mode defined for the module registers without any additional restrictions. It can be modified in both user (non-supervisor) and supervisory mode.
22-0 —	Reserved

55.7 Glossary

PROT_MEM Integration parameter that specifies the size of the module register slot protected. The PROT_MEM size depends on the chip or module.

Hard lock A lock that restricts access to any register bit. You can unlock or clear this lock through the hardware reset.

Soft lock A lock that restricts access to any register bit. You can unlock or clear this lock through the software.

Chapter 56
Clock Monitoring Unit – Frequency Check (CMU_FC)

56.1 Chip-specific CMU_FC information

56.1.1 CMU_FC configuration

The device consists of a robust clock monitoring architecture targeted to monitor various clock nodes to ensure device clock robustness. There are upto seven clock monitoring units present in the device out of which five are of type CMU_FC. See 'Clock monitoring' section in Clocking chapter for details on device clock monitoring architecture. Register interface clock is referred as 'bus_clock' in the below sections. See Clocking chapter for details.

NOTE

CMU_5 is accessible by Cortex-M0+ core only.

CMU_FC combined spec variation: The combined spec variation for CMU (used for configuring CMU_FCx.HTCR[HFREF] and CMU_FCx.HTCR[LFREF]) is the absolute sum of frequency variations of the clock sources. Refer device datasheet for frequency variations for the corresponding clock sources.

56.2 Overview

CMU_FC ensures the clock integrity of selected clocks and allows continuous clock integrity verification.

CMU_FC checks if the frequency of a monitored clock (*monitored_clock*) is within a programmable frequency range specified by the user. If the frequency is outside the specified limits, it could lead to performance, protocol, and timing failures. CMU_FC requires a clock used as a base reference for the frequency check operation. This reference clock (*reference_clock*) must be within documented limits for correct CMU_FC operation. See chip-specific information for details.

56.2.1 Block diagram

CMU_FC counts clock cycles of the monitored clock during a user programmable time duration of *n* clock cycles of the reference clock. There is a comparison between the final count of the monitored clock and the user programmable upper and lower threshold count limits. The control logic determines whether the cycle count is within the programmed limits. If not it sets the Frequency Higher than High frequency reference threshold event status (FHH) field or the Frequency Lower than Low frequency reference threshold event status (FLL) field in the Status Register (SR). If you enable interrupts, an interrupt asserts when the status flag sets.

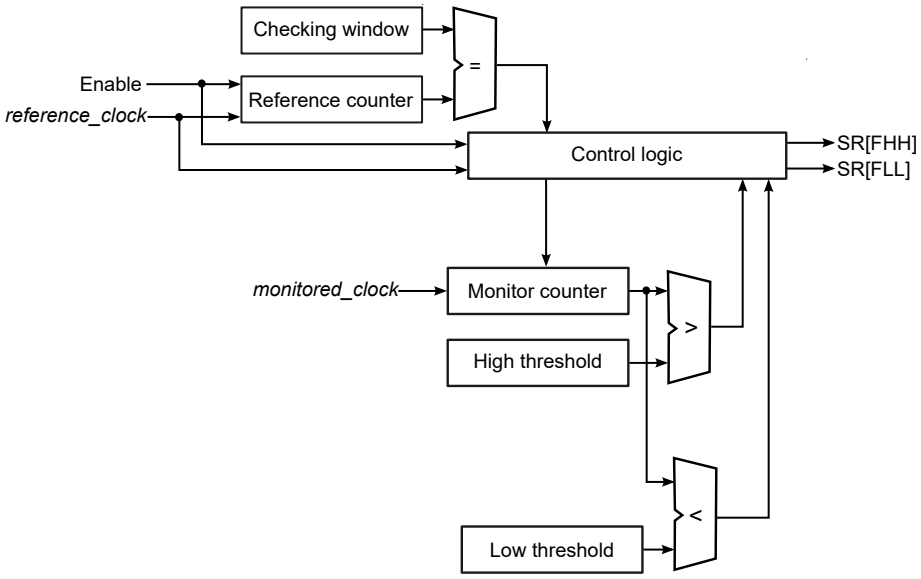


Figure 226. CMU_FC block diagram

CMU_FC cannot detect the following:

- Monitored clock duty cycle variations
- Instantaneous monitored clock frequency variations that do not cross upper or lower threshold limits

56.2.2 Features

The CMU_FC features are:

- Programmable duration of reference clock cycles
- Initiates an event if a monitored clock frequency is higher than high frequency reference (FHH)
- Initiates an event if a monitored clock frequency is lower than low frequency reference (FLL)
- FLL event (if enabled) is generated when monitored clock is very slow or stops functioning
- Masking of FHH event interrupt
- Masking of FLL event interrupt

56.3 Functional description

56.3.1 Operating modes

CMU_FC supports the following operating modes:

- Frequency checking

See the succeeding sections for detailed descriptions of the functions.

56.3.2 Frequency checking

Frequency checking starts when software writes GCR[FCE] = 1. The Run Status field SR[RS] shows 1 after frequency check operation starts.

CMU_FC performs continuous periodic clock checking of the monitored clock. Each checking window involves the following operations:

- Stage 1 – Reference clock counter runs for RCCR[REF_CNT] reference clock cycles. Monitored clock counter runs in parallel for the same time duration as reference clock counter.
- Stage 2 – After stage 1, the module compares the final monitored clock count against the programmed thresholds.

If the monitored clock count is greater than the high threshold value in HTCR[HFREF], an FHH event occurs.

If the monitored clock count is lower than the low threshold value in LTCR[LFREF], an FLL event occurs.

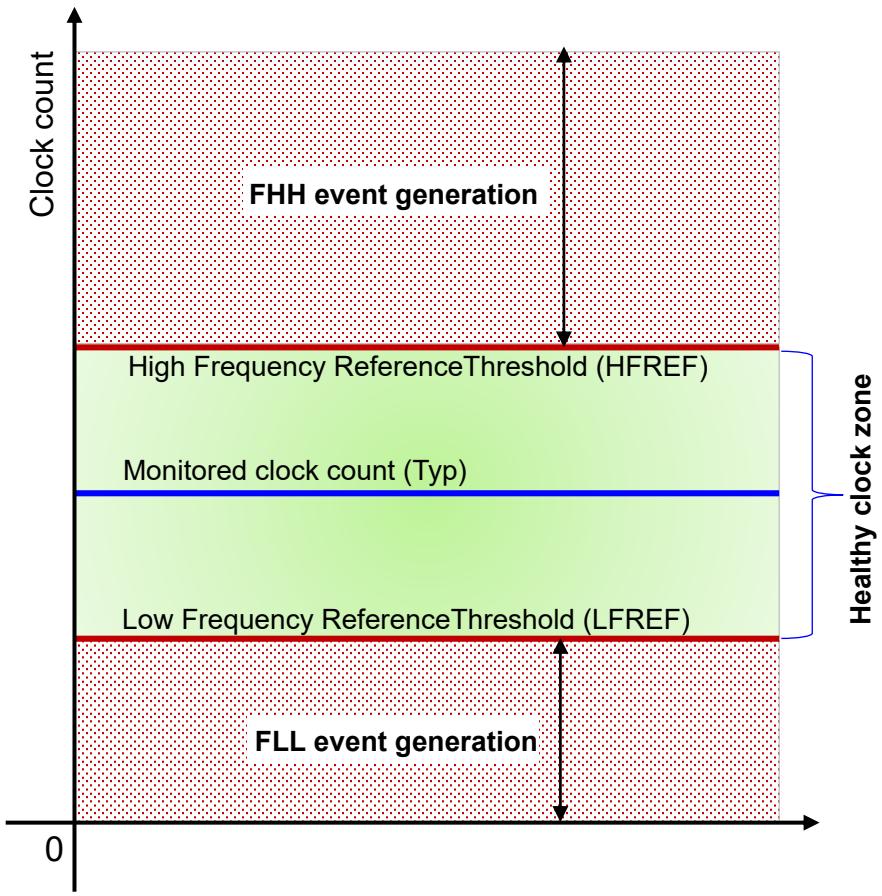


Figure 227. Frequency checking HFREF and LFREF thresholds

56.3.2.1 Monitored clock lost

If a monitored clock ceases operation before evaluation of that clock (Stage 2, see Frequency checking), CMU_FC waits for RCCR[REF_CNT] reference clock periods by extending Stage 2 before triggering an FLL event.

- If *monitored_clock* is not running when enabling CMU_FC:

Worst case FLL response time = (2 × RCCR[REF_CNT] + 12) *reference_clock* cycles + 15 *bus_clock* cycles

- If *monitored_clock* stops when the CMU_FC is running:

Worst case FLL response time = (2 × RCCR[REF_CNT] + 2) *reference_clock* cycles + 3 *bus_clock* cycles

56.3.3 Clocking

This module has the following clock sources.

Table 324. Clock description

Clock	I/O	Description
reference_clock	I	Clock signal that the module uses as a reference to evaluate *monitored_clock*
monitored_clock	I	Clock signal on which frequency check is performed
bus_clock	I	Clock signal on which register read/write operation is performed

NOTE

See the Clocking chapter for details of monitored and reference clocks used on this chip.

56.3.4 Reset

This module has the following reset sources.

- Global hard reset input
- Reset port synchronized to reference clock domain input
- Reset port synchronized to monitored clock domain input

56.4 External signals

This module has no external signals.

56.5 Programming guidelines

56.5.1 Programming RCCR[REF_CNT]

RCCR[REF_CNT] defines the duration of the checking operation in number of *reference_clock* cycles. The minimum RCCR[REF_CNT] value can be calculated using the following formula:

Minimum RCCR [REF_CNT] = CEILING value of MAX $\left(3 \text{ X } \left(\frac{f_{reference_clock}}{f_{bus_clock}}\right), 8 + 5 \text{ X } \left(\frac{f_{reference_clock}}{f_{monitored_clock}}\right)\right)$

...where:

- $f_{reference_clock}$ is the frequency of the reference clock
- f_{bus_clock} is the frequency of the bus clock
- $f_{monitored_clock}$ is the frequency of the monitored clock

Example to determine RCCR[REF_CNT]:

- f_{bus_clock} = 16 MHz
- $f_{reference_clock}$ = 8 MHz
- $f_{monitored_clock}$ = 48 MHz

Formula #1:	$3 \text{ X } \left(\frac{f_{reference_clock}}{f_{bus_clock}} \right)$
Inserting the values:	3 × (8 / 16) = 1.5
Formula #2:	$8 + 5 \text{ X } \left(\frac{f_{reference_clock}}{f_{monitored_clock}} \right)$
Inserting the values:	8 + 5 × (8 / 48) ≈ 8.83
MAX (1.5, 8.83):	8.83
CEILING value of MAX:	9
Therefore, minimum RCCR[REF_CNT]:	9

Programmed RCCR[REF_CNT] value must be greater than or equal to the calculated minimum value and should be decided after considering application requirements of frequency check completion response time and overall accuracy required from CMU_FC.

- Higher values of RCCR[REF_CNT] results in longer measurement window, leading to better accuracy in monitored clock check.
- Lower values of RCCR[REF_CNT] results in shorter measurement window, leading to faster FHH and FLL event response, but higher inaccuracy in reported result.

For measurement window calculation, see step 8 in Programming HFREF and LFREF

56.5.2 Programming HFREF and LFREF

You must consider the following when programming HFREF/LFREF:

- Tolerable monitored clock frequency variation
- Maximum reference clock frequency variation
- Inherent module inaccuracy

CMU_FC has an expected maximum deviation of ± 3 *monitored_clock* cycles (CMU_FC_{VAR+} = 3, CMU_FC_{VAR-} = –3).

The following procedure shows how to calculate optimum values for HTCR[HFREF] and LTCR[LFREF] (numbers shown are from example in table):

1. Determine the ideal *monitored_clock* frequency ($f_{monitored_clock}$ (ideal), 48 MHz).
2. Determine the specified variation of the *monitored_clock* (1.1%).
3. Calculate *monitored_clock* frequency variation: .

   $$f_{monitored_clock}\,(ideal) \text{ X} \left(1 \pm variation\,(\text{step 2})\right)$$

   - $f_{monitored_clock}$ (max) = 48 MHz × (1 + (1.1 ÷ 100)) = 48.53 MHz
   - $f_{monitored_clock}$ (min)= 48 MHz × (1 – (1.1 ÷ 100)) = 47.47 MHz
4. Determine the ideal *reference_clock* frequency ($f_{reference_clock}$ (ideal), 8 MHz).
5. Determine the specified variation of the *reference_clock* (3.3%).

6. Calculate ideal *reference_clock* frequency variation:

$$f_{reference_clock}\ (ideal) \times (1 \pm variation\ (\text{step 5}))$$

   - $f_{reference_clock}$ (max) = 8 MHz × (1 + (3.3 ÷ 100)) = 8.26 MHz
   - $f_{reference_clock}$ (min) = 8 MHz × (1 – (3.3 ÷ 100)) = 7.74 MHz
7. Select the value of RCCR[REF_CNT] needed (80). See Programming RCCR[REF_CNT].
8. Calculate ideal measurement window:

$$\frac{RCCR\,[REF_CNT]}{f_{reference_clock}\ (ideal)}$$

   - 80 ÷ 8 MHz = 10000 ns
9. Calculate ideal *monitored_clock* count:

$$\frac{RCCR[REF_CNT]}{f_{reference_clock}(ideal)} \times f_{monitored_clock}\ (ideal)$$

   - 80 ÷ 8 MHz × 48 MHz = 480.00 (round off this value, 480 in this case)
10. Calculate high threshold value (HTCR[HFREF]):

$$\text{Ceiling value of } \left(\frac{f_{monitored_clock}\ (max)}{f_{reference_clock}\ (min)} \times RCCR[REF_CNT]\right) + CMU_FC_{VAR+}$$

    - HTCR[HFREF] = Ceiling(48.53 MHz ÷ 7.74 MHz × 80 + 3) = 505
11. Calculate low threshold value (LTCR[LFREF]):

$$\text{Floor value of } \left(\frac{f_{monitored_clock}\ (min)}{f_{reference_clock}\ (max)} \times RCCR[REF_CNT]\right) - CMU_FC_{VAR-}$$

    - LTCR[LFREF] = Floor(47.47 MHz ÷ 8.26 MHz × 80 – 3) = 456

Table 325. HTCR[HFREF] and LTCR[LFREF] calculation example

Property	Value	Unit
Bus clock frequency (f_{bus_clock})	16	MHz
Monitored clock frequency ($f_{monitored_clock}$ (ideal))	48	MHz
Specified variation of monitored clock	1.1	%
Monitored clock frequency (maximum after specified variation, $f_{monitored_clock}$ (max))	48.53	MHz

Table continues on the next page...

Table 325. HTCR[HFREF] and LTCR[LFREF] calculation example (continued)

Property	Value	Unit
Monitored clock frequency (minimum after specified variation, $f_{monitored_clock}$ (min))	47.47	MHz
Reference clock frequency ($f_{reference_clock}$ (ideal))	8	MHz
Specified variation of reference clock	3.3	%
Reference clock frequency (maximum after specified variation, $f_{reference_clock}$ (max))	8.26	MHz
Reference clock frequency (minimum after specified variation, $f_{reference_clock}$ (min))	7.74	MHz
Minimum reference count (RCCR[REF_CNT])	9	—
Programmed reference count (RCCR[REF_CNT]) (must be ≥ minimum RCCR[REF_CNT])	80	—
Measurement window	10000.00	ns
Calculated monitored clock count	480.00	—
Calculated monitored clock Count (rounded)	480	—
Module positive variation (CMU_FC_{VAR+})	3	—
Module negative variation (CMU_FC_{VAR-})	-3	—
Programmed higher threshold (HTCR[HFREF])	505	—
Programmed lower threshold (LTCR[LFREF])	456	—

56.5.3 Module programming sequence

56.5.3.1 Module programming sequence

The recommended programming sequence for CMU_FC frequency checking is:

1. In any order, complete the following:
   - Program RCCR[REF_CNT] to define the frequency measuring window duration in *reference_clock* cycles.
   - Program LTCR[LFREF] and HTCR[HFREF] to set the permissible frequency range of the *monitored_clock*.
   - Program IER to enable interrupt.
2. Write GCR[FCE] = 1 to start the frequency check operation.

   NOTE

   When GCR[FCE] = 1, you must not write to RCCR, HTCR, LTCR, or IER. Attempting to do so generates a bus transfer error.

3. To reconfigure RCCR, LTCR, HTCR , and IER or to stop frequency check operation, wait for SR[RS] = 1, then write GCR[FCE] = 0.

56.6 CMU_FC register descriptions

56.6.1 CMU_FC memory map

This section describes the address order of all the CMU_FC registers. Each description includes a standard register diagram and associated field descriptions.

CMU_0 base address: 402B_C000h

CMU_3 base address: 402B_C060h

CMU_4 base address: 402B_C080h

CMU_5 base address: 402B_C0A0h

CMU_6 base address: 402B_C0C0h

Offset	Register	Width (In bits)	Access	Reset value
0h	Global Configuration Register (GCR)	32	RW	0000_0000h
4h	Reference Count Configuration Register (RCCR)	32	RW	0000_0000h
8h	High Threshold Configuration Register (HTCR)	32	RW	00FF_FFFFh
Ch	Low Threshold Configuration Register (LTCR)	32	RW	0000_0000h
10h	Status Register (SR)	32	RW	0000_0000h
14h	Interrupt Enable Register (IER)	32	RW	0000_0000h

56.6.2 Global Configuration Register (GCR)

Offset

Register	Offset
GCR	0h

Function

Controls module level configurations such as enabling frequency check.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															FCE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-1 —	Reserved
0 FCE	Frequency Check Enable Starts or stops frequency checking. FCE is disabled by default. Software can enable FCE at any time. To stop the ongoing operation, write 0 to FCE only when SR[RS] = 1. 0b - Stops frequency checking 1b - Starts frequency checking

56.6.3 Reference Count Configuration Register (RCCR)

Offset

Register	Offset
RCCR	4h

Function

Programs reference count duration of the frequency check window.

NOTE

Write to RCCR only when GCR[FCE] = 0. A bus transfer error results if software writes RCCR when GCR[FCE] = 1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	REF_CNT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-0 REF_CNT	Reference clock count Total number of counts of *reference_clock* for which frequency check runs. This field defines the duration of one frequency check window. See Programming RCCR[REF_CNT] for RCCR calculation.

56.6.4 High Threshold Configuration Register (HTCR)

Offset

Register	Offset
HTCR	8h

Function

Determines the high threshold limit of the monitored clock counter.

NOTE

Write HTCR only when GCR[FCE] = 0. A bus transfer error results if software writes HTCR when GCR[FCE] = 1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								HFREF							
W																
Reset	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	HFREF															
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-24 —	Reserved
23-0 HFREF	High frequency reference threshold HFREF determines the high reference value for the monitored clock frequency. See Programming HFREF and LFREF for HFREF calculation.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE Do not program HFREF to a value greater than 0x00FFFFFC.

56.6.5 Low Threshold Configuration Register (LTCR)

Offset

Register	Offset
LTCR	Ch

Function

Determines the low threshold limit of the monitored clock counter.

NOTE

Write LTCR only when GCR[FCE] = 0. A bus transfer error results if software writes LTCR when GCR[FCE] = 1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								LFREF							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LFREF															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-0 LFREF	Low Frequency Reference Threshold LFREF determines the low reference value for the monitored clock frequency. See Programming HFREF and LFREF for LFREF calculation. **NOTE** Do not program LFREF to a value less than 0x00000003.

56.6.6 Status Register (SR)

Offset

Register	Offset
SR	10h

Function

Provides the internal status of the module.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0											RS	Reserved		FHH	FLL
W															W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-5 —	Reserved
4 RS	Run Status Shows the running status of module internal operation. After enabling the frequency check operation (GCR[FCE] = 1), there is a fixed delay before this field shows running status. If the system wants to disable the frequency check operation, then the software should read this register and write GCR[FCE] = 0 only when this field is 1. 0b - Frequency check stopped 1b - Frequency check running
3-2 —	Reserved
1 FHH	Frequency higher than high frequency reference threshold event status Hardware writes 1 to FHH when the monitored clock frequency becomes higher than the high threshold value in HTCR[HFREF]. FHH clears when software writes 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No FHH event 1b - FHH event occurred
0 FLL	Frequency lower than low frequency reference threshold event status Hardware writes 1 to FLL when the monitored clock frequency becomes lower than the low threshold value in LTCR[LFREF]. FLL clears when software writes 1. 0b - No FLL event 1b - FLL event occurred

56.6.7 Interrupt Enable Register (IER)

Offset

Register	Offset
IER	14h

Function

Enables CMU_FC interrupts.

NOTE

Write IER only when GCR[FCE] = 0. A bus transfer error results if software writes IER when GCR[FCE] = 1.

Enable only either asynchronous FHH event interrupt or synchronous FHH event interrupt at a time.

For example:

- If IER[FHHIE] is set to 1 then set IER[FHHAIE] to 0.
- If IER[FHHAIE] is set to 1 then set IER[FHHIE] to 0.

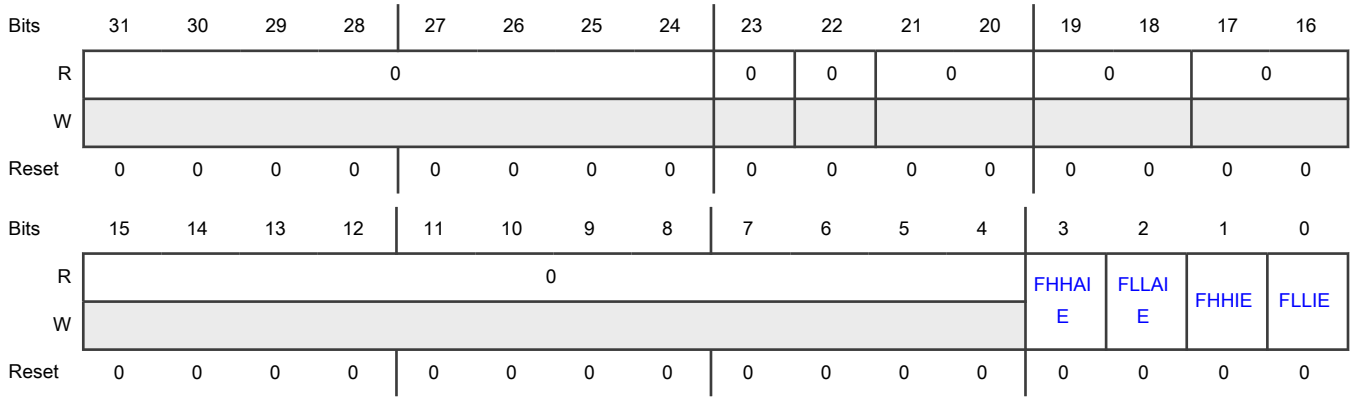
Enable only either asynchronous FLL event interrupt or synchronous FLL event interrupt at a time.

For example:

- If IER[FLLIE] is set to 1 then set IER[FLLAIE] to 0.
- If IER[FLLAIE] is set to 1 then set IER[FLLIE] to 0.

See Clocking chapter for interrupt usage information based on CMU_FC instances.

Diagram



Fields

Field	Function
31-24 —	Reserved
23 —	Reserved
22 —	Reserved
21-20 —	Reserved
19-18 —	Reserved
17-16 —	Reserved
15-4 —	Reserved
3 FHHAIE	Frequency Higher than High Frequency Reference Threshold Asynchronous Interrupt Enable FHHAIE enables FHH asynchronous interrupt at the module boundary. 0b - Asynchronous FHH event interrupt disabled 1b - Asynchronous FHH event interrupt enabled
2 FLLAIE	Frequency Lower than Low Frequency Reference Threshold Asynchronous Interrupt Enable FLLAIE enables FLL asynchronous interrupt at the module boundary.

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>0b - Asynchronous FLL event interrupt disabled
1b - Asynchronous FLL event interrupt enabled</td></tr>
<tr><td>1
FHHIE</td><td>Frequency Higher than High Frequency Reference Threshold Synchronous Interrupt Enable
FHHIE enables an FHH synchronous interrupt at the module boundary.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>CMU_0</td><td>IER</td><td>—</td></tr><tr><td>CMU_3</td><td>—</td><td>IER</td></tr><tr><td>CMU_4</td><td>—</td><td>IER</td></tr><tr><td>CMU_5</td><td>—</td><td>IER</td></tr><tr><td>CMU_6</td><td>—</td><td>IER</td></tr></table>0b - Synchronous FHH event interrupt disabled
1b - Synchronous FHH event interrupt enabled</td></tr>
<tr><td>0
FLLIE</td><td>Frequency Lower than Low Frequency Reference Threshold Synchronous Interrupt Enable
FLLIE enables an FLL synchronous interrupt at the module boundary.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>CMU_0</td><td>IER</td><td>—</td></tr><tr><td>CMU_3</td><td>—</td><td>IER</td></tr><tr><td>CMU_4</td><td>—</td><td>IER</td></tr><tr><td>CMU_5</td><td>—</td><td>IER</td></tr><tr><td>CMU_6</td><td>—</td><td>IER</td></tr></table></td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Synchronous FLL event interrupt disabled 1b - Synchronous FLL event interrupt enabled

56.7 Glossary

FHH The state that the monitored clock frequency is higher than the high frequency reference. FHH is also the name of the register field that indicates this state.

FHH event The event that the module triggers when it detects FHH.

FLL The state that the monitored clock frequency is lower than the lower frequency reference. FLL is also the name of the register field that indicates this state.

FLL event The event that the module triggers when it detects FLL.

Chapter 57
Clock Monitoring Unit – Frequency Meter (CMU_FM)

57.1 Chip-specific CMU_FM information

57.1.1 CMU_FM configuration

The device consists of a robust clock monitoring architecture targeted to monitor various clock nodes to ensure device clock robustness. There are upto seven clock monitoring units present in the device out of which 2 are of type CMU_FM. See 'Clock monitoring' section in Clocking chapter for details on device clock monitoring architecture. Register interface clock is referred as 'bus_clock' in the below sections. See Clocking chapter for details.

NOTE

CMU metering interrupt is not generated if the clock which is being metered is lost. A timeout status flag, CMU.SR[FMTO] is updated in this case.

NOTE

Software must ensure that CMU1 interrupt is serviced within POR_WDG timeout. In absence of CMU1 interrupt servicing, the POR_WDG detects this as an FIRC failure and initiates a POR recovery.

57.2 Overview

CMU_FM measures the frequency of the metered clock (*metered_clock*) using another clock as reference. This reference clock (*reference_clock*) must be within documented limits for correct CMU_FM operation. CMU_FM operation is only guaranteed if a reference clock is within normal operating parameters.

57.2.1 Block diagram

The CMU_FM counts the clock cycles of a metered clock in the user programmable time duration *n* number of clock cycles of the reference clock. After completion of a metered clock operation, a register field sets, indicating the operation has completed. The count of the metered clock also updates in the Status Register (SR) after completion of a metered clock operation.

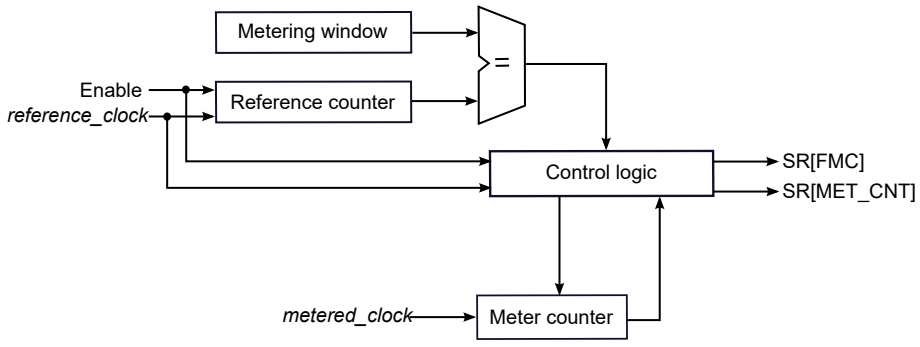


Figure 228. Block diagram

57.2.2 Features

Features of the CMU_FM are:

- Programmable duration of *reference_clock* cycles
- Maskable interrupt generation when frequency metering completes
- Timeout feature indicates loss of *metered_clock*

57.3 Functional description

The frequency meter operation initiates when software writes GCR[FME] = 1. When the operation starts, hardware writes SR[RS] = 1. Safety critical applications must poll SR[RS] to determine when metering operation starts.

Frequency metering window involves these operations:

1. Stage 1 — The reference clock counter runs for RCCR[REF_CNT] cycles of *reference_clock*. The metered clock counter runs in parallel for the same time duration as the *reference_clock* counter.
2. Stage 2 — At the end of stage 1, SR[FMC] = 1, SR[MET_CNT] updates with the *metered_clock* count, and CMU_FM writes GCR[FME] = 0.

57.3.1 Clocking

The following table lists the clock sources for this module.

Table 326. Clock description

Clock	I/O	Description
reference_clock	I	Clock signal that the module uses as a reference to evaluate *metered_clock*
metered_clock	I	Clock signal on which frequency metering is performed
bus_clock	I	Clock signal on which register read/write operation is performed

NOTE

See the Clocking chapter for details of metered and reference clocks used on this chip.

57.3.2 Reset

This module has the following reset sources.

- Global hard reset input
- Reset port synchronized to reference clock domain input
- Reset port synchronized to monitored clock domain input

57.4 Programming guidelines

57.4.1 Programming RCCR[REF_CNT]

RCCR[REF_CNT] defines the duration of the metering operation in number of *reference_clock* cycles. The minimum RCCR[REF_CNT] value can be calculated using the following formula:

$$\textbf{Minimum RCCR [REF_CNT]} = \text{CEILING value of MAX}\left(3 \text{ X } \left(\frac{f_{reference_clock}}{f_{bus_clock}}\right), 8 + 5 \text{ X } \left(\frac{f_{reference_clock}}{f_{monitored_clock}}\right)\right)$$

...where:

- $f_{reference_clock}$ is the frequency of the reference clock
- f_{bus_clock} is the frequency of the bus clock
- $f_{metering_clock}$ is the frequency of the metered clock

Example to determine RCCR[REF_CNT]:

- f_{bus_clock} = 16 MHz
- $f_{reference_clock}$ = 8 MHz
- $f_{metered_clock}$ = 48 MHz

Formula #1:

$$3 \times \left(\frac{f_{reference_clock}}{f_{bus_clock}} \right)$$

Inserting the values: 3 × (8 / 16) = 1.5

Formula #2:

$$8 + 5 \times \left(\frac{f_{reference_clock}}{f_{monitored_clock}} \right)$$

Inserting the values: 8 + 5 × (8 / 48) ≈ 8.83

MAX (1.5, 8.83): 8.83

CEILING value of MAX: 9

Therefore, minimum RCCR[REF_CNT]: 9

Programmed RCCR[REF_CNT] value must be greater than or equal to the calculated minimum value and should be decided after considering application requirements of frequency metering completion response time and overall accuracy required from CMU_FM.

- Higher values of RCCR[REF_CNT] results in longer metering window, leading to better accuracy in metered clock measurement.
- Lower values of RCCR[REF_CNT] results in shorter metering window, leading to faster FMC response, but higher inaccuracy in reported result.

57.4.2 Module programming sequence

The recommended programming sequence of CMU_FM is:

1. In any order, complete the following:
   - Program RCCR[REF_CNT] to define the frequency metering window duration in *reference_clock* cycles.
   - Program IER[FMCIE] to enable interrupt.
2. Write GCR[FME] = 1 to start frequency metering operation. Writes to RCCR and IER are disabled after GCR[FME] = 1. Attempting a write to any of them after GCR[FME] = 1 generates a bus transfer error.
3. To reconfigure RCCR and IER or to stop frequency metering operation, wait for SR[RS] = 1, then write GCR[FME] = 0.

Upon completion of metering operations:

- Hardware writes SR[FMC] = 1
- SR[MET_CNT] updates with metered count value
- Hardware writes GCR[FME] = 0

57.5 External signals

This module has no external signals.

57.6 CMU_FM register descriptions

57.6.1 CMU_FM memory map

This section describes the address order of all the CMU_FM registers. Each description includes a standard register diagram and associated field descriptions.

CMU_1 base address: 402B_C020h

CMU_2 base address: 402B_C040h

Offset	Register	Width (In bits)	Access	Reset value
0h	Global Configuration Register (GCR)	32	RW	0000_0000h
4h	Reference Count Configuration Register (RCCR)	32	RW	0000_0000h
8h	Status Register (SR)	32	RW	0000_0000h
Ch	Interrupt Enable Register (IER)	32	RW	0000_0000h

57.6.2 Global Configuration Register (GCR)

Offset

Register	Offset
GCR	0h

Function

Controls module level configurations such as enabling frequency metering.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															FME
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-1 —	Reserved
0 FME	Frequency Meter Enable Starts or stops frequency metering. FME is disabled by default. Software can enable FME at any time. To stop the ongoing operation, write 0 to FME only when SR[RS] = 1. FME automatically clears upon frequency metering completion or timeout event. 0b - Stops frequency metering 1b - Starts frequency metering

57.6.3 Reference Count Configuration Register (RCCR)

Offset

Register	Offset
RCCR	4h

Function

Programs reference count duration of the frequency meter window.

NOTE

Write to RCCR only when GCR[FME] = 0. A bus transfer error results if software writes RCCR when GCR[FME] = 1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	REF_CNT															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
15-0 REF_CNT	Reference Clock Count Total number of counts of *reference_clock* for which frequency metering runs. This field defines the duration of one frequency metering window. See Programming RCCR[REF_CNT] for RCCR calculation.

57.6.4 Status Register (SR)

Offset

Register	Offset
SR	8h

Function
Provides the internal status of the module and metered clock count.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MET_CNT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MET_CNT								0			RS	Reserved		FMTO	FMC
W															W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-8 MET_CNT	Meter Clock Count This field stores the count value of the metered clock cycles when frequency metering operation is complete, in other words, SR[FMC] = 1. CMU_FM has a maximum deviation of ± 3 *metered_clock* cycles over the ideal *metered_clock* cycles expected by the user.
7-5 —	Reserved
4	Run Status

Table continues on the next page...

Table continued from the previous page...

Field	Function
RS	Shows the running status of module internal operation. After enabling the frequency metering operation (GCR[FME] = 1), there is a fixed delay before this field shows running status. If the system wants to disable the frequency metering operation, then the software should read this register and write GCR[FME] = 0 only when this field is 1. 0b - Frequency meter stopped 1b - Frequency meter running
3-2 —	Reserved
1 FMTO	Frequency Meter Time Out Sets if there is a loss of metered clock during the metering operation. After metering window, the module waits for a timeout duration of RCCR[REF_CNT] cycles before writing one to this field. GCR[FME] = 0 and SR[FMC] = 0 when FMTO = 1. Writing one clears this field.
0 FMC	Frequency Meter Operation Complete On completion of frequency metering operation, FMC = 1, SR[MET_CNT] field updates, and GCR[FME] clears. Writing one clears this field.

57.6.5 Interrupt Enable Register (IER)

Offset

Register	Offset
IER	Ch

Function

Enables CMU_FM interrupts.

NOTE

Write IER only when GCR[FME] = 0. A bus transfer error results if software writes IER when GCR[FME] = 1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															FMCIE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-1 —	Reserved
0 FMCIE	Frequency Meter Complete Interrupt Enable This bit is used to enable/disable Frequency Meter complete interrupt at the module boundary. 0b - Frequency Meter complete interrupt is disabled 1b - Frequency Meter complete interrupt is enabled

Chapter 58
Cyclic Redundancy Check (CRC)

58.1 Chip-specific CRC information

This chip supports one instance of CRC module.

58.2 Overview

CRC generates 16-bit or 32-bit CRC codes output for error detection. You can calculate these codes for up to 32 bits input data at a time.

CRC provides a programmable polynomial and other parameters to meet 16-bit or 32-bit CRC standards.

58.2.1 Block diagram

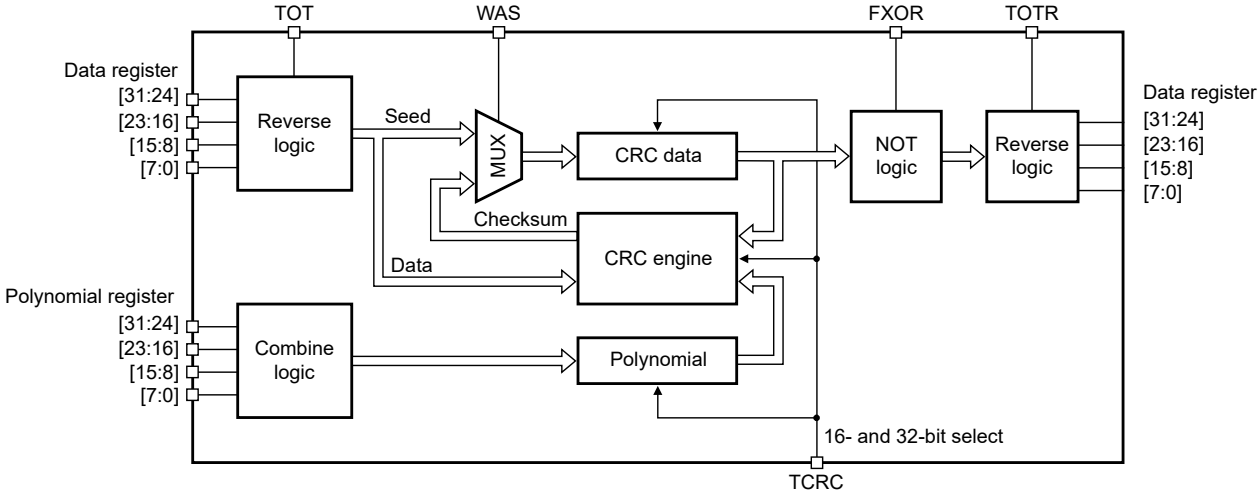


Figure 229. Block diagram

58.2.2 Features

CRC has the following features:

- Hardware CRC generator circuit using 16-bit or 32-bit programmable shift registers
- Programmable initial seed value and polynomial
- Transpose of input or output data (CRC result) in bitwise or bytewise (this option is required for certain CRC standards. You cannot perform a bytewise transpose operation when accessing DATA via 8-bit access.)
- Invert final CRC result
- 32-bit CPU register programming interface

58.3 Functional description

58.3.1 Modes of operation

The following sections describe various modes of operation that affect the functionality of CRC: Run mode and Low power mode.

58.3.1.1 Run mode

Run mode is the basic mode of operation.

58.3.1.2 Low-power modes

When the chip enters the low-power mode, the CRC module clock (ipg_clk and ipg_clk_s) is disabled and the in-progress CRC calculation stops. The calculation resumes after the CRC module clock is enabled or the chip exits low power mode via system reset.

NOTE

See the chip-specific power management chapter to know the specific low-power modes supported on your device.

58.3.2 CRC calculations

In 16-bit and 32-bit CRC modes, you can program data values as 8 bits, 16 bits, or 32 bits at a time, provided all bytes are contiguous. Noncontiguous bytes lead to an incorrect CRC calculation.

58.3.2.1 Calculating a 16-bit CRC

Perform these steps to calculate a 16-bit CRC:

1. Write 0 to CTRL[TCRC] to enable 16-bit CRC mode.
2. Program the transpose and complement options in Control (CTRL) as required for the CRC calculation.
3. Write a 16-bit polynomial to GPOLY[LOW].

   GPOLY[HIGH] is not usable in 16-bit CRC mode.
4. Write 1 to CTRL[WAS] to program the seed value.
5. Write a 16-bit seed to DATA[LU] and DATA[LL].

   DATA[HU] and DATA[HL] are not used.
6. Write 0 to CTRL[WAS] to start writing data values.
7. Write data values into DATA[LU], and DATA[LL]

   CRC is calculated on every data value write and the intermediate CRC result is stored back into DATA[LU] and DATA[LL]
8. After writing all the data values, read the final CRC result from DATA[LU] and DATA[LL].

CRC is calculated bytewise and two clocks are required to complete one CRC calculation.

The transpose and complement operations are performed on-the-fly when reading or writing values. See Transpose feature and Result complement for details.

58.3.2.2 Calculating a 32-bit CRC

Perform these steps to calculate a 32-bit CRC:

1. Write 1 to CTRL[TCRC] to enable 32-bit CRC mode.
2. Program the transpose and complement options in Control (CTRL) as required for CRC calculation. See Transpose feature and Result complement for details.
3. Write a 32-bit polynomial to GPOLY[HIGH] and GPOLY[LOW] .
4. Write 1 to CTRL[WAS] to program the seed value.
5. Write a 32-bit seed to DATA[HU], DATA[HL], DATA[LU], and DATA[LL].
6. Write 0 to CTRL[WAS] to start writing data values.
7. Write data values into DATA[HU], DATA[HL], DATA[LU], and DATA[LL]

CRC is computed on every data value write and the intermediate CRC result is stored back into DATA .

8. After writing all the values, read the final CRC result from DATA[HU], DATA[HL], DATA[LU], and DATA[LL].

   CRC is calculated bytewise and two clocks are required to complete one CRC calculation.

The transpose and complement operations are performed on-the-fly when reading or writing values. See Transpose feature and Result complement for details.

58.3.3 Transpose feature

Transpose is not enabled by default. However, CRC requires input data and/or final checksum to be transposed. You have an option to configure each transpose operation separately to meet CRC standards. The data is transposed on-the-fly while being read or written.

Some protocols use the little-endian format for data stream to calculateCRC. In this case, transpose flips bits.

58.3.3.1 Types of transpose

CRC provides several types of transpose to flip bits and/or bytes for both writing input data and reading result separately using the CTRL[TOT] and CTRL[TOTR] according to the CRC calculation being used.

The following types of transpose are available for writing to and reading from DATA.

1. CTRL[TOT] or CTRL[TOTR] is 0.

   No transposition occurs.

2. CTRL[TOT] or CTRL[TOTR] is 1.

   Bits in a byte are transposed when bytes are not transposed.

   reg[31:0] becomes {reg[24:31], reg[16:23], reg[8:15], reg[0:7]}.

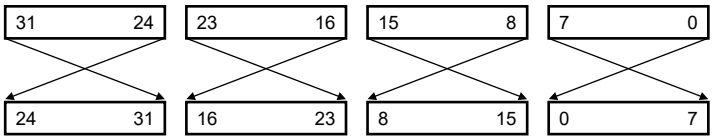


Figure 230. Transpose type 1b

3. CTRL[TOT] or CTRL[TOTR] is 10b.

   Both bits in bytes and bytes are transposed.

   reg[31:0] becomes {reg[0:7], reg[8:15], reg[16:23], reg[24:31]}.

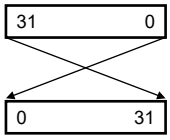


Figure 231. Transpose type 10b

4. CTRL[TOT] or CTRL[TOTR] is 11b.

   Bytes are transposed but bits are not transposed.

   reg[31:0] becomes {reg[7:0], reg[15:8], reg[23:16], reg[31:24]}.

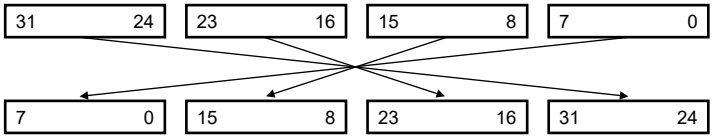


Figure 232. Transpose type 11b

NOTE

For 8-bit and 16-bit write accesses to Data (DATA), the data is transposed with 0s on the unused byte or bytes (taking 32 bits as a whole), but CRC is calculated on the valid byte(s) only. When reading the Data (DATA) for a 16-bit CRC result and using transpose options 10 and 11, the resulting value after transposition resides in DATA[HU] and DATA[HL]. You must account for this situation when reading the 16-bit CRC result, so reading 32 bits is preferred.

58.3.4 Result complement

When CTRL[FXOR] = 1, the checksum is complemented. The CRC result complement function outputs the complement of the checksum value stored in Data (DATA) every time Data (DATA) is read. When CTRL[FXOR] = 0,, reading Data (DATA) accesses the raw checksum value.

58.3.5 Clocking

Table 327. CRC clocks

Type of clock	Description
Bus clock (ipg_clk/ipg_clk_s)	ipg_clk_s controls the access to the CRC registers. ipg_clk and ipg_clk_s function the CRC module.

NOTE

See the chip-specific clocking chapter, to know the specific CRC clocks supported on your device.

58.3.6 Interrupts

This module has no interrupts.

58.4 External signals

There is no CRC signal that connects off chip.

58.5 Initialization

To enable CRC calculation, you must program:

- CTRL[WAS] .
- Polynomial (GPOLY).
- Parameters for transposition and CRC result inversion in the applicable registers.

Writing 1 to CTRL[WAS] enables you to program the seed value into CRC Data registers.

After writing all the data, you must wait for at least two clock cycles to read the data from CRC Data (DATA) register.

After a CRC calculation completes, you can reinitialize the module for a new CRC computation by again writing 1 to CTRL[WAS] and programming a new, or previously used, seed value. You must set all other parameters before programming the seed value and subsequent data values.

58.6 Use cases

The following tables use the little-endian format.

58.6.1 CTRL programming

The following table shows Control (CTRL) programming for 16-bit CRC.

Table 328. CTRL programming for 16-bit CRC

Algorithm	Polynomial	Seed	Ref in	Ref out	XOR out	CTRL[TOT]	CTRL[TOTR]	CTRL[FXOR]
CRC-16_CCITT_FALSE	1021h	FFFFh	0	0	0000h	0h	0h	0h
CRC-16_ARC	8005h	0000h	1	1	0000h	1h	2h	0h
CRC-16_AUG_CCITT	1021h	1D0Fh	0	0	0000h	0h	0h	0h
CRC-16_BUYPASS	8005h	0000h	0	0	0000h	0h	0h	0h
CRC-16_CCITT_ZERO	1021h	0000h	0	0	0000h	0h	0h	0h
CRC-16_CDMA2000	C867h	FFFFh	0	0	0000h	0h	0h	0h
CRC-16_DDS_110	8005h	800Dh	0	0	0000h	0h	0h	0h
CRC-16_DECT_X	589h	0000h	0	0	0000h	0h	0h	0h
CRC-16_DNP	3D65h	0000h	1	1	FFFFh	1h	2h	1h
CRC-16_EN_13757	3D65h	0000h	0	0	FFFFh	0h	0h	1h
CRC-16_GENIBUS	1021h	FFFFh	0	0	FFFFh	0h	0h	1h
CRC-16_MAXIM	8005h	0000h	1	1	FFFFh	1h	2h	1h
CRC-16_MCRF4XX	1021h	FFFFh	1	1	0000h	1h	2h	0h
CRC-16_RIELLO	1021h	B2AAh	1	1	0000h	1h	2h	0h
CRC-16_T10_DIF	8BB7h	0000h	0	0	0000h	0h	0h	0h
CRC-16_TELEDISK	A097h	0000h	0	0	0000h	0h	0h	0h
CRC-16_TMS37157	1021h	89ECh	1	1	0000h	1h	2h	0h
CRC-16_USB	8005h	FFFFh	1	1	FFFFh	1h	2h	1h
CRC-16_A	1021h	C6C6h	1	1	0000h	1h	2h	0h
CRC-16_KERMIT	1021h	0000h	1	1	0000h	1h	2h	0h
CRC-16_MODBUS	8005h	FFFFh	1	1	0000h	1h	2h	0h
CRC-16_X_25	1021h	FFFFh	1	1	FFFFh	1h	2h	1h
CRC-16_XMODEM	1021h	0000h	0	0	0000h	0h	0h	0h

The following table shows Control (CTRL) programming for 32-bit CRC.

Table 329. CTRL programming for 32-bit CRC

Algorithm	Polynomial	Seed	Ref in	Ref out	XOR out	CTRL[TOT]	CTRL[TOTR]	CTRL[FXOR]
CRC-32	04C11DB7h	FFFFFFFFh	1	1	FFFF_FFFFh	1h	2h	1h
CRC-32_BZIP2	04C11DB7h	FFFFFFFFh	0	0	FFFF_FFFFh	0h	0h	1h
CRC-32C	1EDC6F41h	FFFFFFFFh	1	1	FFFF_FFFFh	1h	2h	1h
CRC-32D	A833982Bh	FFFFFFFFh	1	1	FFFF_FFFFh	1h	2h	1h
CRC-32_MPEG-2	04C11DB7h	FFFFFFFFh	0	0	0000_0000h	0h	0h	0h
CRC-32_POSIX	04C11DB7h	00000000h	0	0	FFFF_FFFFh	0h	0h	1h
CRC-32Q	814141ABh	00000000h	0	0	0000_0000h	0h	0h	0h
CRC-32_JAMCRC	04C11DB7h	FFFFFFFFh	1	1	0000_0000h	1h	2h	0h
CRC-32_XFER	000000AFh	00000000h	0	0	0000_0000h	0h	0h	0h

58.6.2 Expected read data fields

The following table shows the expected read data fields for 16-bit CRC.

Table 330. Expected read data fields for 16-bit CRC

Algorithm	Data (DATA)
CRC16_CCITT_FALSE	[31:16] = Unknown[15:0] = Valid data
CRC16_ARC	[31:16] = Valid data [15:0] = Unknown
CRC16_AUG_CCITT	[31:16] = Unknown [15:0] = Valid data
CRC16_BUYPASS	[31:16] = Unknown [15:0] = Valid data
CRC16_CCITT_ZERO	[31:16] = Unknown [15:0] = Valid data
CRC16_CDMA2000	[31:16] = Unknown [15:0] = Valid data
CRC16_DDS_110	[31:16] = Unknown [15:0] = Valid data
CRC16_DECT_X	[31:16] = Unknown [15:0] = Valid data
CRC16_DNP	[31:16] = Valid data [15:0] = Unknown
CRC-16_EN_13757	[31:16] = Unknown [15:0] = Valid data
CRC-16_GENIBUS	[31:16] = Unknown [15:0] = Valid data
CRC-16_MAXIM	[31:16] = Valid data [15:0] = Unknown
CRC-16_MCRF4XX	[31:16] = Valid data [15:0] = Unknown
CRC-16_RIELLO	[31:16] = Valid data [15:0] = Unknown
CRC-16_T10_DIF	[31:16] = Unknown [15:0] = Valid data
CRC-16_TELEDISK	[31:16] = Unknown [15:0] = Valid data
CRC-16_TMS37157	[31:16] = Valid data [15:0] = Unknown
CRC-16_USB	[31:16] = Valid data [15:0] = Unknown

Table continues on the next page...

Table 330. Expected read data fields for 16-bit CRC (continued)

Algorithm	Data (DATA)
CRC-16_A	[31:16] = Valid data [15:0] = Unknown
CRC-16_KERMIT	[31:16] = Valid data [15:0] = Unknown
CRC-16_MODBUS	[31:16] = Valid data [15:0] = Unknown
CRC-16_X_25	[31:16] = Valid data [15:0] = Unknown
CRC-16_XMODEM	[31:16] = Unknown [15:0] = Valid data

The following table shows the expected read data fields for 32-bit CRC.

Table 331. Expected read data fields for 32-bit CRC

Algorithm	Data (DATA)
CRC-32	[31:0] = Valid data
CRC-32_BZIP2	[31:0] = Valid data
CRC-32C	[31:0] = Valid data
CRC-32D	[31:0] = Valid data
CRC-32_MPEG-2	[31:0] = Valid data
CRC-32_POSIX	[31:0] = Valid data
CRC-32Q	[31:0] = Valid data
CRC-32_JAMCRC	[31:0] = Valid data
CRC-32_XFER	[31:0] = Valid data

58.7 Memory map and register descriptions

NOTE

You must reconfigure CRC engine in case an IPS transfer error occurs (ips_xfr_err).

The CRC module generates a transfer error in the following cases:

- Write accesses to the register addresses that are not mapped to the peripherals but included in the address spaces of the peripherals.

58.7.1 CRC register descriptions

58.7.1.1 CRC memory map

CRC base address: 4038_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Data (DATA)	32	RW	FFFF_FFFFh
4h	Polynomial (GPOLY)	32	RW	0000_1021h
8h	Control (CTRL)	32	RW	0000_0000h

58.7.1.2 Data (DATA)

Offset

Register	Offset
DATA	0h

Function

Configures the value of seed, data, and checksum. When CTRL[WAS] = 1, any write to this register is regarded as the seed value. When CTRL[WAS] becomes 0, any write to this register is regarded as data for general CRC calculation.

In 16-bit CRC mode, DATA[HU] and DATA[HL] are not used for programming the seed value, and reads of these fields return an indeterminate value. In 32-bit CRC mode, all fields are used for programming the seed value.

When programming data values, you can program to write 8 bits, 16 bits, or 32 bits in big endian order, provided all bytes are contiguous, with the MSB of data value written first.

After writing all data values, you can read the CRC result from DATA register. In 16-bit CRC mode, the CRC result is available in DATA[LU] and DATA[LL]. In 32-bit CRC mode, all fields contain the result. After writing all data, you must wait for at least two clock cycles to read the data from CRC data (DATA) register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	HU								HL							
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	LU								LL							
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-24 HU	Upper Part of High Byte Generates CRC checksum in both 16-bit and 32-bit CRC modes if CTRL[WAS] = 0. • In 16-bit CRC mode (CTRL[TCRC] = 0), this field is not used for programming a seed value. • In 32-bit CRC mode (CTRL[TCRC] = 1), the values written to this field are part of the seed value when CTRL[WAS] = 1.
23-16 HL	Lower Part of High Byte Generates CRC checksum in both 16-bit and 32-bit CRC modes if CTRL[WAS] = 0. • In 16-bit CRC mode (CTRL[TCRC] = 0), this field is not used for programming a seed value.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• In 32-bit CRC mode (CTRL[TCRC] = 1), the values written to this field are part of the seed value when CTRL[WAS] = 1.
15-8 LU	Upper Part of Low Byte Generates CRC checksum when CTRL[WAS] = 0. When CTRL[WAS] = 1,the values written to this field are part of the seed value.
7-0 LL	Lower Part of Low Byte Generates CRC checksum when CTRL[WAS] = 0. When CTRL[WAS] = 1, the values written to this field are part of the seed value.

58.7.1.3 Polynomial (GPOLY)

Offset

Register	Offset
GPOLY	4h

Function

Configures polynomial value for CRC calculation.

- Sets the upper 16 bits of polynomial that are used only in 32-bit CRC mode. Writes to this field are ignored in 16-bit CRC mode.
- Sets the lower 16 bits of polynomial that are used in both 16-bit and 32-bit CRC modes.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	HIGH															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	LOW															
Reset	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	1

Fields

Field	Function
31-16 HIGH	High Half-Word Writable and readable in 32-bit CRC mode (CTRL[TCRC] = 1). You cannot write to this field in 16-bit CRC mode (CTRL[TCRC] = 0).
15-0 LOW	Low Half-Word Writable and readable in both 16-bit and 32-bit CRC modes.

58.7.1.4 Control (CTRL)

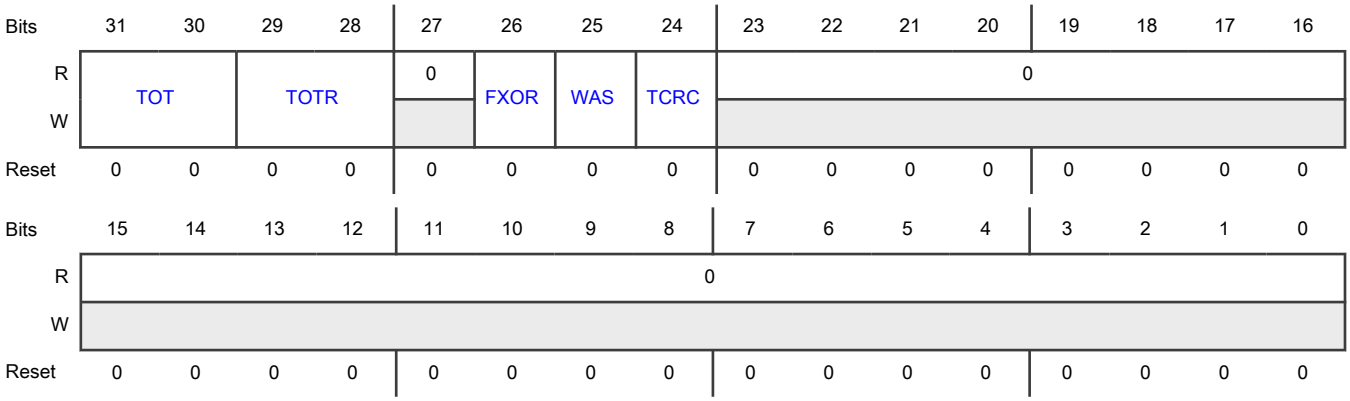
Offset

Register	Offset
CTRL	8h

Function

Sets control for CRC. You must write 1 to the appropriate fields of this register before starting a new CRC calculation, which you can initialize by writing 1 to CTRL[WAS] and then writing the seed into DATA.

Diagram



Fields

Field	Function
31-30 TOT	Transpose Type for Write Sets transpose type for the values written to DATA. See Transpose feature for the available transpose options. 00b - No transposition 01b - Bits in bytes are transposed, but bytes are not transposed.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10b - Both bits in bytes and bytes are transposed. 11b - Only bytes are transposed, no bits in a byte are transposed.
29-28 TOTR	Transpose Type for Read Sets transpose type for the values read from DATA . See Transpose feature for the available transpose options. 00b - No transposition 01b - Bits in bytes are transposed, but bytes are not transposed. 10b - Both bits in bytes and bytes are transposed. 11b - Only bytes are transposed, no bits in a byte are transposed.
27 —	Reserved
26 FXOR	Complement Read of CRC Data Register Enables on-the-fly complementing of read data. Some CRC protocols require the final checksum to be XORed with FFFFFFFFh or FFFFh. 0b - Disables XOR on reading data. 1b - Inverts or complements the read value of the CRC Data.
25 WAS	Write as Seed Specifies whether writes to DATA are data values or seed values. When this field = 1, the value that you write to is considered as seed value. When this field = 0, the value that you write to is considered as data for CRC calculation. 0b - Data values 1b - Seed values
24 TCRC	TCRC Defines the width of CRC. 0b - 16 bits 1b - 32 bits
23-0 —	Reserved

58.8 Glossary

Transpose	Flipping input or output data bits (or bytes)
Seed value	Initial value of CRC calculation

Chapter 59
Power Conversion and Motor Control (PCMC)

59.1 Introduction

The PCMC subsystem is a group of modules that can be interconnected to perform a variety of real-time tasks, such as:

- Motor control
- Power conversion
- Advanced PWM generation
- Lighting control

The modules that make up PCMC are:

- ADC
- BCTU
- eMIOS
- LCU
- LPCMP
- TRGMUX

See Feature comparison for the number of instances of each module in your chip.

59.2 Block diagram

The figure and table below show a typical configuration.

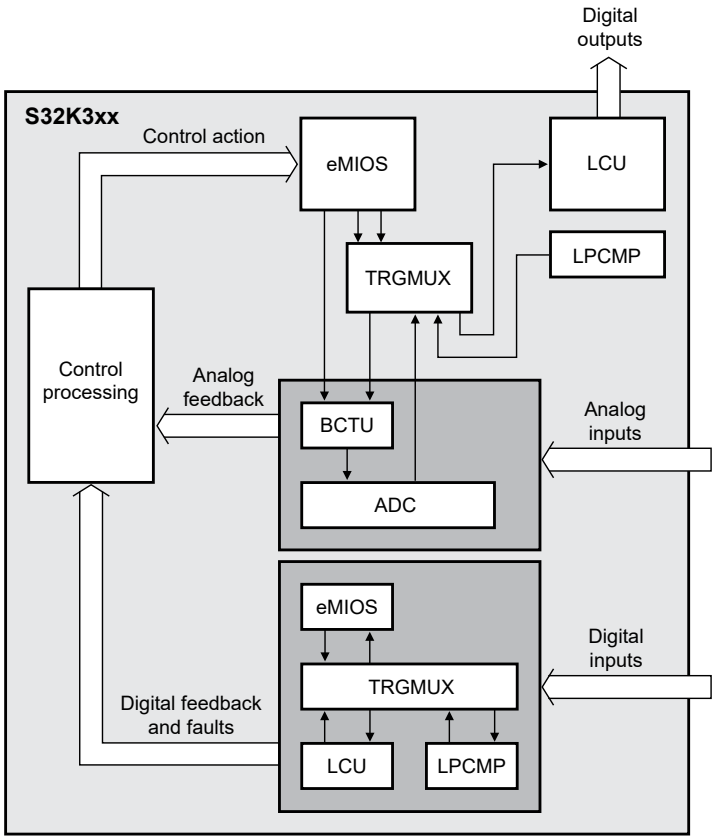


Figure 233. Block diagram

Table 332. Block diagram components

Component	Description
ADC	Measures external analog signals
BCTU	Performs advanced ADC triggering
Control processing	Arm Cortex-M7 core: • Processes digital and analog inputs to generate control outputs • Writes control outputs to peripherals
eMIOS	• Generates high-resolution PWM outputs • Measures time-based digital inputs
LCU	Performs programmable logic functions and fault control
LPCMP	Compares analog values to detect faults
TRGMUX	• Routes signals between internal peripherals • Routes external signals to internal peripherals

59.3 Functional overview

59.3.1 Analog signal measurement

ADC measures the voltage of external analog signals and converts that voltage to a digital value. Each ADC instance operates independently of other instances. Software or hardware can trigger conversions. Together with BCTU and eMIOS, ADC can preempt normal measurements by injecting high-priority measurements when needed.

See Analog-to-Digital Converter.

59.3.2 Advanced ADC triggering

BCTU works with ADC to initiate conversions triggered by outputs from other modules in the PCMC subsystem, such as PWM signals from eMIOS or LCU sequential logic outputs.

With BCTU, you can group measurements into configuration lists containing up to 32 items. Each item contains information such as:

- ADC instance to use for the conversion
- The ADC channel
- The trigger event for initiating the next conversion
- The conversion list address

BCTU can store conversion results by routing them through DMA or putting them in FIFO queues. The FIFO queues enable conversions to occur seamlessly by storing conversion results without software intervention.

BCTU has 72 trigger inputs, 69 of which come from the eMIOS channel outputs. The three remaining trigger inputs come from other on-chip modules via TRGMUX. You can also trigger conversions by software through the BCTU register interface.

BCTU can issue a single conversion or a list of conversions based on a single trigger occurrence. BCTU signals the eMIOS timers when ADC receives a single command or a list of commands. ADC sends conversion data back to BCTU, which loads it in the ADC*n* Result Data (ADC*n*DR) register for the CPU to read.

BCTU-to-ADC interface illustrates the connections between BCTU and an ADC.

See also:

- Body cross-triggering unit (BCTU)
- PCDR*n*, ICDR*n*, and ECDR*n* register descriptions in ADC register descriptions

59.3.3 BCTU-to-ADC interface

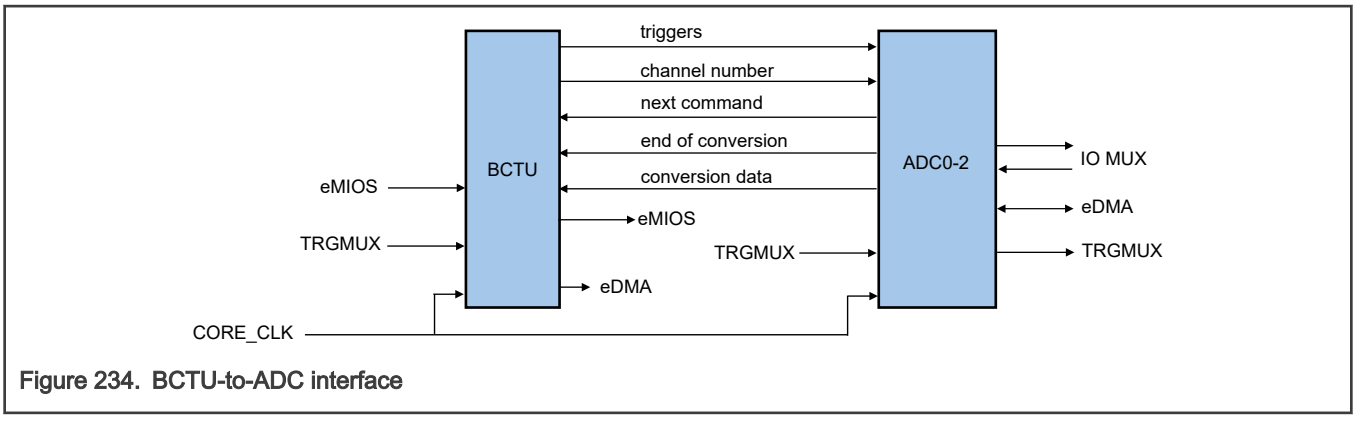


Figure 234. BCTU-to-ADC interface

59.3.4 Generate PWM outputs and measure digital inputs

eMIOS generates high-resolution PWM outputs. Each eMIOS instance has 24 dedicated channel outputs. You can configure each channel to operate in a different mode. Depending on the mode, eMIOS channels can:

- Measure the period of an input signal
- Measure the width of an input pulse
- Function as general-purpose I/O
- Count pulses or edges
- Capture input values
- Compare outputs

For example, eMIOS can generate periodic ADC triggers. Using TRGMUX, you can route eMIOS triggers to other on-chip peripherals such as LCU, another eMIOS instance, external chip I/O, and so on.

eMIOS connections illustrates the input and output connections for eMIOS.

See also:

- Enhanced Modular IO Subsystem (eMIOS)
- Body cross-triggering unit (BCTU)
- Trigger Mux (TRGMUX)

59.3.5 eMIOS connections

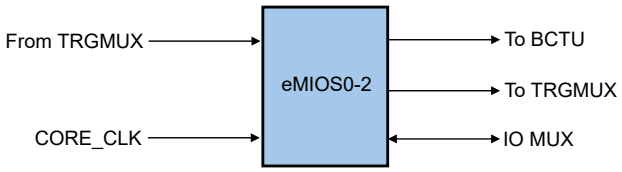


Figure 235. eMIOS connections

59.3.6 Create combinatorial and sequential logic functions

LCU enables you to implement hardware-based combinatorial and sequential logic on the chip. This capability reduces cost and decreases application size.

Each LCU instance has 12 dedicated inputs and 12 outputs. You can use LCU to implement a variety of combinatorial and sequential logic functions with no CPU intervention, including a simple OR gate, flip flops, and motor control logic. As detailed in Fault detection, LCU also plays a significant role in fault management.

LCU connections shows the input and output connections for LCU.

See also:

- Logic Control Unit (LCU)
- LCU use case examples
- Trigger MUX (TRGMUX)
- TRGMUX connectivity file attached to this document

59.3.7 LCU connections

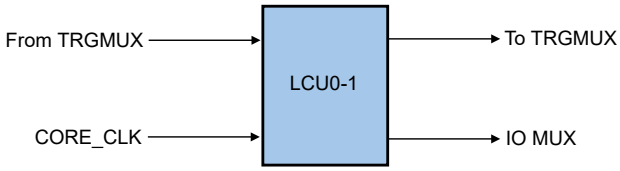


Figure 236. LCU connections

59.3.8 Fault detection

The PCMC subsystem uses LPCMP and LCU to implement fault detection.

LPCMP can detect fault conditions (such as short circuits, over-voltages, and excessive temperature) by comparing an external input voltage to a programmable reference voltage. It produces a digital output that indicates whether the input voltage is higher or lower than the reference voltage.

The chip can detect faults via external digital or analog signals. TRGMUX routes digital signals from SIUL to a peripheral to trigger the appropriate behavior.

LCU has three force inputs that you can use to manage fault assertion via its own internal logic mechanism.

The typical reaction to the presence of fault is to put the system in a safe state by:

- Deasserting output signals such as PWM outputs controlling a power stage.
- Disconnecting an entire circuit by switching off a relay.

See also Low Power Comparator (LPCMP).

59.3.9 Programmable module interconnections

TRGMUX enables you to interconnect on-chip peripherals to create application-specific configurations. TRGMUX has 128 inputs and 111 outputs, which you can interconnect in hundreds of combinations through software during runtime.

You can also route on-chip peripheral outputs to chip I/O pins to enable signal debugging. Similarly, you can route external signals to on-chip peripheral inputs.

NOTE

There is a two-cycle AIPS_SLOW_CLK delay through TRGMUX.

For details of PCMC interconnections, see PCMC connections diagram.

The attached TRGMUX connectivity file, a portion of which appears below, shows the interconnections possible between module instances. Column B in the spreadsheet lists TRGMUX inputs. Row 3 lists TRGMUX outputs. A connection between an input and an output is possible if the intersecting cell of the input row and the output column is blank. "NA" indicates that a connection is not possible.

															Output no.	0	1	2	4	5	6
															Output register no.	0	0	0	1	1	1
															S32K310	Y	Y	Y	Y	Y	Y
															S32K311	Y	Y	Y	Y	Y	Y
															S32K312	Y	Y	Y	Y	Y	Y
															S32K322	Y	Y	Y	Y	Y	Y
															S32K341	Y	Y	Y	Y	Y	Y
															S32K342	Y	Y	Y	Y	Y	Y
															S32K314	Y	Y	Y	Y	Y	Y
															S32K324	Y	Y	Y	Y	Y	Y
															S32K344	Y	Y	Y	Y	Y	Y
															S32K328	Y	Y	Y	Y	Y	Y
															S32K338	Y	Y	Y	Y	Y	Y
															S32K348	Y	Y	Y	Y	Y	Y
															S32K356	Y	Y	Y	Y	Y	Y
															S32K358	Y	Y	Y	Y	Y	Y
															S32K388	Y	Y	Y	Y	Y	Y
Input no.	S32K310	S32K311	S32K312	S32K322	S32K341	S32K342	S32K314	S32K324	S32K344	S32K328	S32K338	S32K348	S32K356	S32K358	S32K388	TRGMUX output	EXTRG for normal conversion	IRG for injected conversion	2_0 EXTRG to sync the start pulses	EXTRG for normal conversion	IRG for injected conversion	2_1 EXTRG to sync the start

Figure 237. Portion of TRGMUX connectivity

See also:

- Trigger Mux (TRGMUX)
- TRGMUX connectivity file attached to this document

59.3.9.1 PCMC connections diagram

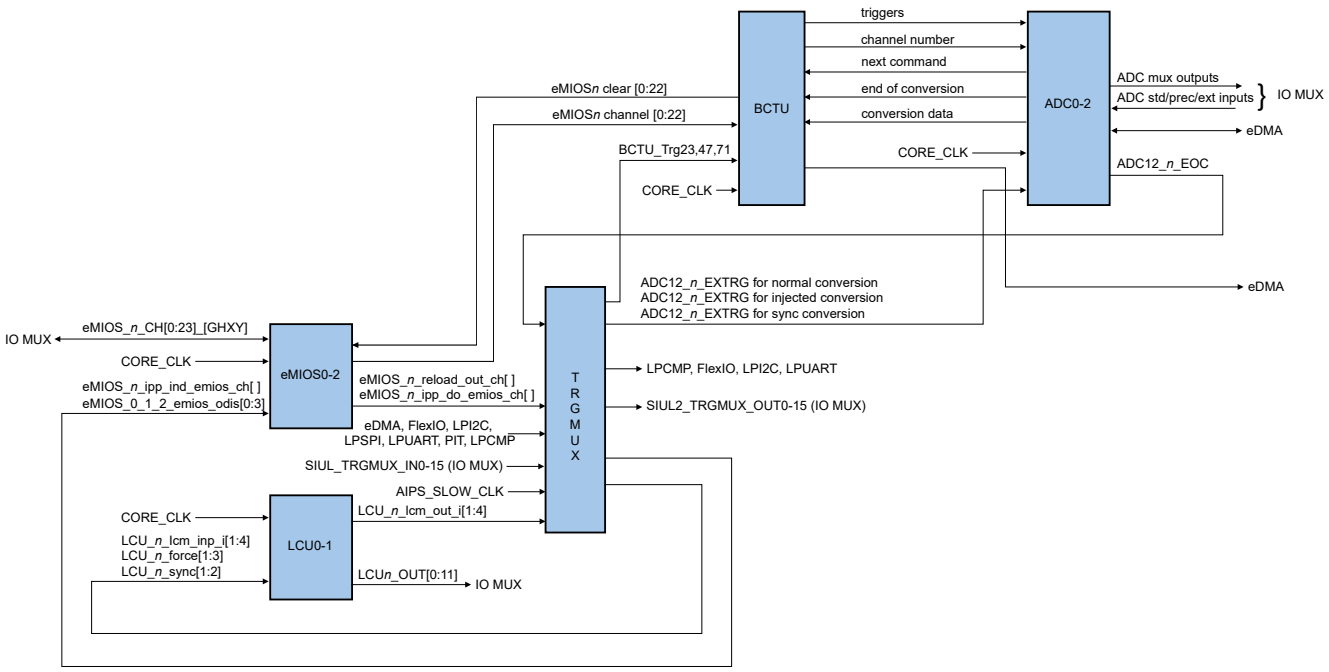


Figure 238. PCMC connections diagram

59.4 Reset, clocking, and other considerations

59.4.1 Reset

ADC, BCTU, eMIOS, LCU, and TRGMUX fully reset during a POR, destructive reset, or functional reset.

See also:

- Module reset status
- Software reset

59.4.2 Clocking

Detailed clocking information is given in the links below. In summary:

- CORE_CLK clocks ADC, BCTU, eMIOS, and LCU.
- AIPS_SLOW_CLK clocks TRGMUX.
- In general, you must set AIPS_SLOW_CLK to one-fourth or one-half the frequency of CORE_CLK.

See also:

- ADC and motor control modules
- System clocking configurations
- System clock frequency limitations
- Clock frequency

59.4.3 Power modes

All ADC, BCTU, eMIOS, LCU, and TRGMUX instances operate only in Run mode. None of the instances operate in Standby mode.

See also:

- Module reset status
- ADC modes of operation
- LCU modes of operation

59.5 Use case examples

59.5.1 Motor control use case

The following sections show:

- A PMSM control system
- A BLDC control system

Both PMSM and BLDC use a rotating magnetic stator field to turn the stator consisting of permanent magnets. The stator is driven by 3-phase AC voltage. The difference between PMSM and BLDC is in how stator position is calculated.

- PMSM calculates stator position by measuring back EMF from the motor.
- BLDC calculates stator position based on feedback from motor sensors.

59.5.1.1 PMSM motor control use case

59.5.1.1.1 Generic PMSM motor control

The following figure and table illustrate a generic PMSM motor control system. The feedback loops in the system create a motor control loop.

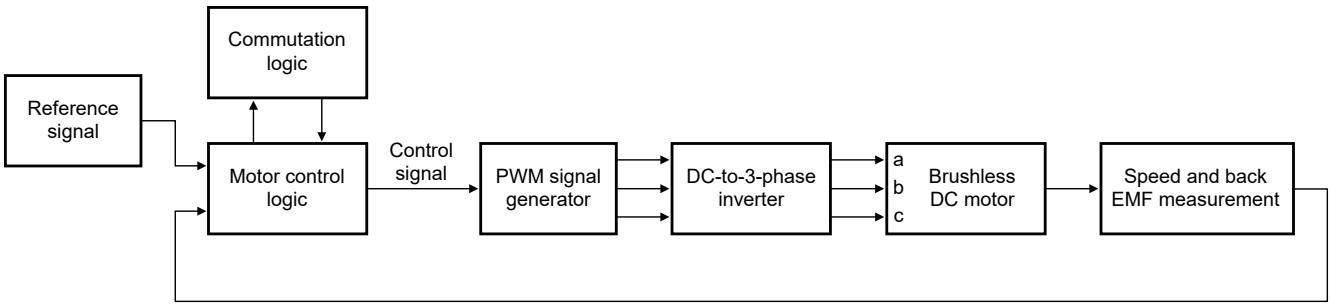


Figure 239. Generic PMSM motor control

Table 333. Block diagram components

Block	Description
Angular position sensor	Generates data that enables commutation logic to calculate rotor position.
Brushless DC motor	Electronically commutated electric motor.
Commutation logic	Compares values from the rotor angular position sensors to entries in a lookup table, and applies voltage to stator winding corresponding to same set of values in lookup table.

Table continues on the next page...

Table 333. Block diagram components (continued)

Block	Description
DC-to-3-phase inverter	Switching network that transmits power to the motor stator windings according to a control signal.
Motor control logic	• Compares speed measurement to required speed and adjusts the PWM signal generator. • Generates the assertions for the PWM signal generator according to commutation logic input.
PWM signal generator	Generates PWM signal voltage source.
Reference signal	Signal that indicates required motor speed.
Speed and back EMF measurement	An ADC is linked to a shunt resistor to perform measurements and detect over-current conditions. If one is found, then hardware generates a fault interrupt.

59.5.1.1.2 PMSM motor control

The PCMC subsystem consists of a group of modules that enable you to implement PMSM motor control loops in a variety of ways. The following figure and table illustrate one possible implementation.

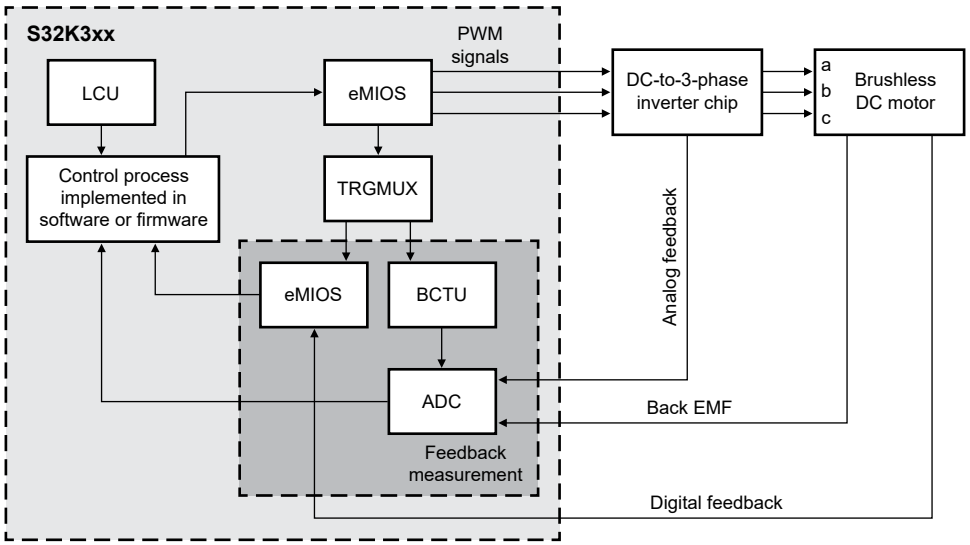


Figure 240. PMSM motor control

Table 334. Block diagram components

Block	Description
ADC	Measures analog inputs (currents and/or voltages) and converts to digital values. Typically, it measures currents and can be configured to generate a fault interrupt when a measured input is outside a pre-defined acceptable range.
BCTU	Triggers conversions by ADC.
Control process	Provides top-level system control functions for FOC or another feasible control algorithm:

Table continues on the next page...

Table 334. Block diagram components (continued)

Block	Description
	• Compares speed measurement to required speed and adjusts the PWM signal generator. • Generates the assertions for the PWM signal generator according to commutation logic input.
DC-to-3-phase inverter	Switching network that transmits power to the motor stator windings according to a control signal.
eMIOS	• Generates PWM input for DC to 3-phase inverter. • Measures digital feedback inputs—BLDC position sensors, for example.
LCU	Implements commutation lookup table. Control Process compares values from the angular position sensors to entries in the lookup table. It then applies voltage to stator winding corresponding to same set of values in lookup table.
TRGMUX	• Enables interconnect to the required set of on-chip peripherals. • Indirectly triggers ADC sampling via BCTU. • Triggers eMIOS to measure digital feedback signals.

59.5.1.2 Sensored BLDC motor control use case

59.5.1.2.1 Generic sensored BLDC motor control

The following figure and table illustrate a generic sensored BLDC motor control system. The feedback loops in the system create a motor control loop.

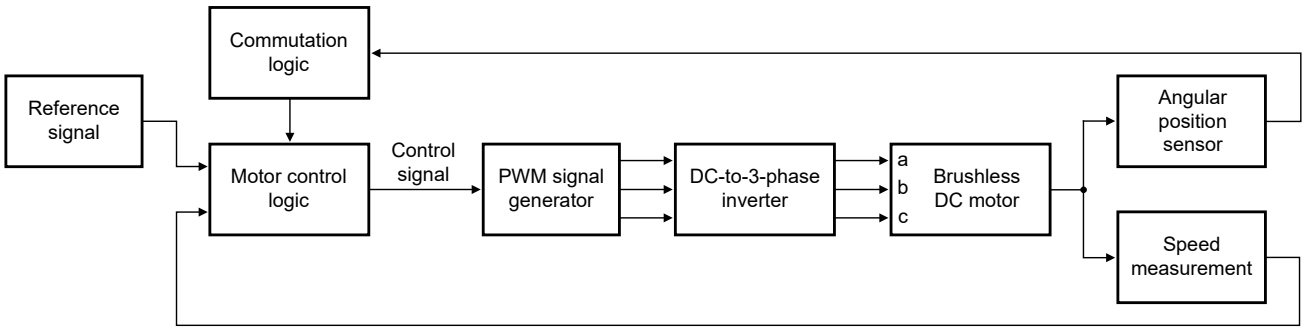


Figure 241. Generic sensored BLDC motor control

Table 335. Block diagram components

Block	Description
Angular position sensor	Hall effect sensors or optical decoders detect rotor position
Commutation logic	Compares values from the angular position sensors to entries in a lookup table and applies voltage to stator winding corresponding to same set of values in lookup table
Reference signal	Signal that indicates required motor speed

Table continues on the next page...

Table 335. Block diagram components (continued)

Block	Description
DC-to-3-phase inverter	Switching network that transmits power to the stator windings according to a control signal
Motor control logic	• Compares speed measurement to required speed and adjusts the PWM signal generator • Generates the assertions for the PWM signal generator according to commutation logic input
Speed measurement	Uses an ADC linked to a shunt resistor to perform measurements and detect over-current conditions, in which case hardware generates an interrupt.

59.5.1.2.2 PCMC sensored field-oriented control (FOC) of 3-phase BLDC motor

FOC is a VFD control method in which the stator currents of a 3-phase AC electric motor (such as BLDC, PMCM, reluctance motor, or other spinning electrical machine) is identified as a vector.

The PCMC subsystem consists of a group of modules that enable you to implement sensored FOC BLDC motor control loops in a variety of ways. The following figure and table show one possible implementation.

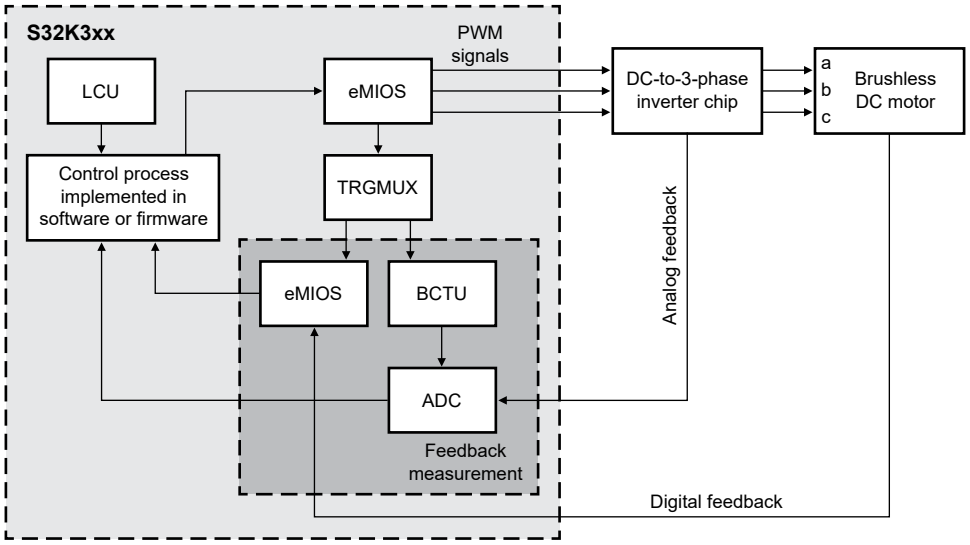


Figure 242. PCMC sensored BLDC motor control

Table 336. Block diagram components

Block	Description
ADC	Measures analog inputs (currents and/or voltages) and converts to digital values. Typically, it measures currents, and can be configured to generate a fault interrupt when a measured input is outside a pre-defined acceptable range.
BCTU	Triggers conversions by ADC.
Control process	Provides top-level system control functions for FOC control:

Table continues on the next page...

Table 336. Block diagram components (continued)

Block	Description
	• Compares speed measurement to required speed and adjusts the PWM signal generator. • Generates the assertions for the PWM signal generator according to commutation logic input.
DC-to-3-phase inverter	Switching network that transmits power to the motor stator windings according to a control signal.
eMIOS	• Generates PWM input for DC to 3-phase inverter. • Measures digital feedback inputs—BLDC position sensors, for example.
LCU	Implements commutation lookup table. Control process compares values from the angular position sensors to entries in the lookup table. It then applies voltage to stator winding corresponding to same set of values in lookup table.
TRGMUX	• Enables interconnect of the required set on on-chip peripherals. • Indirectly triggers ADC sampling via BCTU. • Triggers eMIOS to measure digital feedback signals.

59.6 Design considerations

59.6.1 ADC design considerations

The following topics discuss factors that affect ADC function in applications.

59.6.1.1 ADC measurement accuracy

The FOC motor control algorithm requires a high degree of confidence in the reliability of conversion data. Here are some best practices that contribute to ADC precision.

- Use precision channels (see ADC precision channels).
- Minimize input noise (see ADC input noise).
- Use BCTU triggering (see ADC triggering).

59.6.1.1.1 ADC precision channels

See the chip-specific ADC configuration information for the number of precision channels for each ADC instance.

See the channel mapping table for the channel numbers for the precision channels.

59.6.1.1.2 ADC input noise

Regardless of the application, ADCs are inherently sensitive to input noise. The following best practices help minimize input noise.

- Avoid simultaneous conversions.
- Consider using a low-pass filter on input pins.

NOTE

When possible, use staggered ADC conversions instead of simultaneous conversions.

All variants of this chip include multiple, independent ADC instances, so it is easily possible to have simultaneous sampling by different ADC instances. This should be avoided, though, because ADC sampling can cause noise on adjacent device ADC

inputs. Avoidance of simultaneous sampling should be ensured by software design. This is important for implementations of the FOC algorithm.

Additionally, an RC low-pass filter on ADC input pins can help reduce noise and enhance accuracy.

59.6.1.1.3 ADC triggering

Some applications, including FOC motor control, must use BCTU for precise triggering of ADC measurement. This is because the FOC algorithm requires sampling at specific times to implement the “observer/predictor”. BCTU makes it possible to perform precisely-timed ADC measurement.

For less exigent algorithms or non-closed-loop controlled motors, BCTU triggering and precision ADC channels are not necessarily required.

59.6.1.2 Example ADC configurations

Following are some ADC configurations that might be used for FOC motor control.

Table 337. Example ADC configurations

Description	Settings
Single-shot mode	• Enable Overwrite (OWREN=1) • Disable Write Left-Aligned (WLSIDE=0) • Select single shot (MODE=0) • Disable External Trigger (TRGEN=0) • Disable Injection Trigger (JTRGEN=0) • Disable BCTU triggering (BCTUEN=0) • Disable averaging (AVGEN=0) • Select conversion clock = 1/2 module clock (ADCCLKSEL=01)
Normal conversion scan mode	• Enable Overwrite (OWREN=1) • Disable Write Left-Aligned (WLSIDE=0) • Select looped conversions (MODE=1) • Disable External Trigger (TRGEN=0) • Disable Injection Trigger (JTRGEN=0) • Disable BCTU triggering (BCTUEN=0) • Disable averaging (AVGEN=0) • Select conversion clock = 1/2 module clock (ADCCLKSEL=01)
BCTU control mode	• Enable Overwrite (OWREN=1) • Disable Write Left-Aligned (WLSIDE=0) • Select single shot (MODE=0) • Disable External Trigger (TRGEN=0) • Disable Injection Trigger (JTRGEN=0) • Enable BCTU triggering (BCTUEN=1)

Table continues on the next page...

Table 337. Example ADC configurations (continued)

Description	Settings
	• Disable averaging (AVGEN=0) • Select conversion clock = 1/2 module clock (ADCCLKSEL=01)
BCTU trigger mode	• Enable Overwrite (OWREN=1) • Disable Write Left-Aligned (WLSIDE=0) • Select single shot (MODE=0) • Disable External Trigger (TRGEN=0) • Disable Injection Trigger (JTRGEN=0) • Enable BCTU triggering (BCTUEN=1) • Enable all trigger sources (BCTU_MODE=1) • Disable averaging (AVGEN=0) • Select conversion clock = 1/2 module clock (ADCCLKSEL=01)

59.6.2 BCTU design considerations

For reasons discussed in ADC triggering, some applications require precise ADC triggering and BCTU provides that capability. The following topics discuss some design considerations when using BCTU.

59.6.2.1 BCTU or ADC data retrieval

Although the primary function of BCTU is to provide precisely timed triggers for ADC conversions, there is some overlap in BCTU and ADC function in that conversion data can be retrieved from either module. For applications in which frequent conversions are performed, such as real-time motor control, BCTU offers a distinct advantage over ADC, because BCTU trigger sources can be configured to route conversion results to a BCTU internal FIFO. Doing so prevents subsequent conversion data from overwriting previous data, making it available for either direct access or DMA transfer to system RAM.

59.6.2.2 Disabling BCTU triggers

There are fault scenarios in which an application must disable BCTU triggers as part of putting the system into a safe state. Examples include:

- Over-voltage condition
- Under-voltage condition
- Calibration parameter mismatch between the embedded motor control system and the motor

Application fault reaction design must include the disabling of BCTU triggers in scenarios in which either system conditions indicate an impending failure, or ADC conversion data is outside control boundaries.

59.6.3 eMIOS design considerations

In a BLDC motor control application, eMIOS plays a central role, providing the driving PWM signal as well as performing feedback signal measurements. The following topics discuss eMIOS considerations in the application design.

59.6.3.1 PWM considerations

In a BLDC motor control application, PWM/power signal requirements are affected by a number of parameters, including:

- Displacement Power Factor
- DF

- THD
- DVA
- CF
- FF

Ensure that the above parameters are included when choosing the timebase and prescaler combination for eMIOS channels producing PWM signals.

59.6.3.2 Input measurement

In a BLDC motor control application, eMIOS measures feedback signals for the motor and provides critical information to the control system. Selecting a timebase and prescaler for eMIOS channels performing input measurement tasks is usually a tradeoff between application requirements for precision and the computational cost in relation to the selected clock speed.

For example, some applications like electric steering demand an extremely precise control for position and torque. This is different from a compressor e-motor, where speed and torque do not demand rigid precision requirements.

Other considerations are related to control of:

- THD
- FF
- GM
- pf correction

The above considerations are significant from the perspective of the PWM reaction speed for the output signal for BLDC applications. These parameters must be considered in the design.

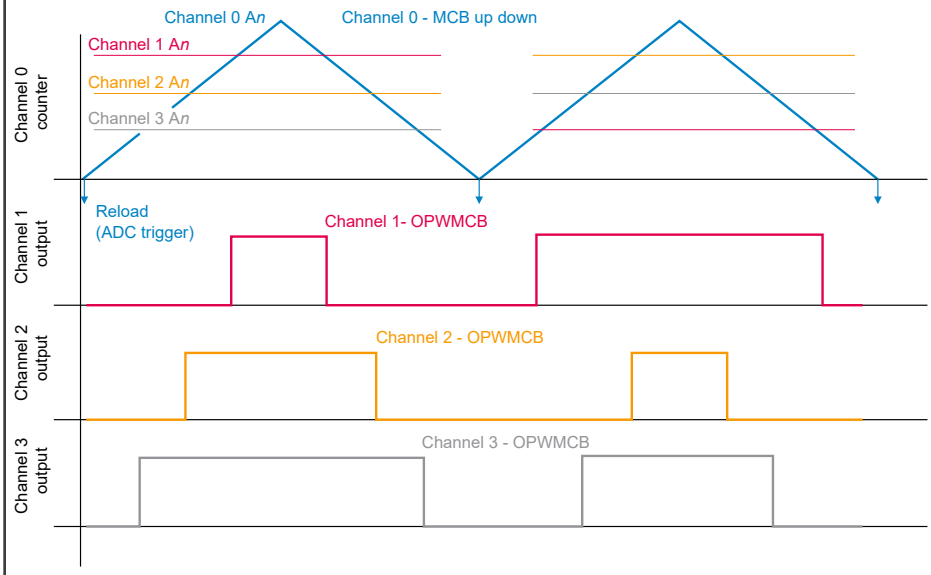
59.6.3.3 eMIOS channel modes useful in motor control

Table 338. eMIOS channel modes useful in motor control

Mode	Description	Use in motor control
MCB	Modulus Counter Buffered (Up/Down)	Time base • PWM frequency • Reload
IPWM	Input Pulse Width Measurement	Capture modes: • Speed measurement • Position measurement • Voltage (duty cycle)
IPM	Input Period Measurement	Speed measurement
PEC	Pulse Edge Counting	Position measurement
OPWMCB	Center aligned Output PWM Buffered with dead time	PWM duty cycle
OPWMB	Output Pulse Width Modulation Buffered	Trigger placement within the period
OPWMT	Output Pulse Width Modulation Trigger	Trigger placement within the period

59.6.3.3.1 OPWMCB PWM and trigger generation

The following figure shows one option for OPWMCB PWM and trigger generation.



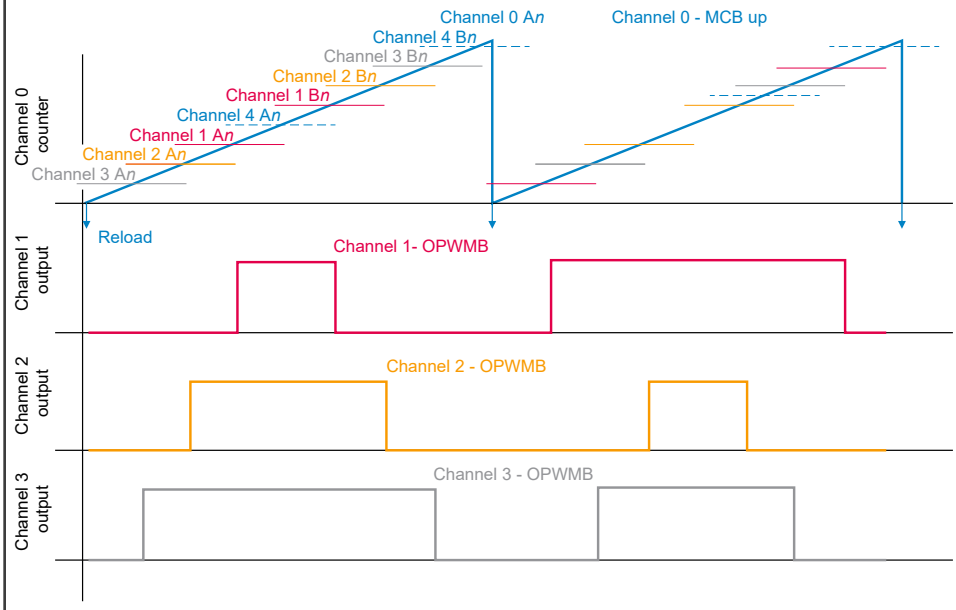
Modulo (channel 0 A*n* register) is double-buffered and updated after a reload signal.

Duty cycle is double-buffered (A*n* registers for channels 1, 2, and 3) and updated after a reload signal.

Figure 243. OPWMCB PWM trigger generation

59.6.3.3.2 OPWMB PWM and trigger generation

The following figure shows another option for OPWMB PWM and trigger generation.



Modulo (channel 0 A*n* register) is double-buffered and updated after a reload signal.

Duty cycle is double-buffered (A*n* and B*n* registers of channels 1-4) and updated after a reload signal.

Figure 244. OPWMB PWM trigger generation

59.7 Glossary

BLDC	Brushless DC. An electronically commutated DC motor.
CF	Crest Factor. A waveform parameter that is the ratio of peak values to its RMS value.
DF	Distortion Factor.

Displacement Power Factor	In power electronics, the phase relationship between voltage and current: $\cos(\theta_v - \theta_i)$, where θ_v is the phase of the voltage and θ_i is the phase of the current.
DVA	Distortion Volt-Amps.
FF	Form Factor.
FOC	Field-Oriented Control.
GM	Gain Margin.
pf	Power factor. In power electronics, it is the ratio of average power to RMS power.
PMSM	Permanent Magnet Synchronous Motor.
THD	Total Harmonic Distortion.
VFD	Variable-Frequency Drive.

Chapter 60
Analog-to-Digital Converter (ADC)

60.1 Chip-specific ADC information

60.1.1 ADC Configuration

This chip has up to 3 instances of ADC. ADC channels are divided into 3 groups - Precision, Standard and External (each with independent configuration settings and different accuracy/performance level). The following table shows the ADC configuration:

Table 339. ADC instances

Instance	S32K388/S32K389/S32K358/S32K356/ S32K348/S32K338/S32K328/S32K344/ S32K324/S32K314	S32K342/S32K322/S32K341/S32K312/ S32K311/S32K310
ADC_0	Yes	Yes
ADC_1	Yes	Yes
ADC_2	Yes	No

Table 340. ADC configuration

Feature	ADC_0	ADC_1	ADC_2
No. of precision channels	8	8	8
No. of standard channels	16	16	16
No. of special internal channels	1[1]	0	0
No. of external channels	32	32	0
BCTU trigger support	Yes	Yes	Yes
DMA support	Yes	Yes	Yes
Hardware interleaving	Yes	Yes	Yes
No. of watchdogs	4	4	4
No. of Hardware trigger	3	3	3

1. CHAN_INT0 - channel 50(S)

NOTE

- No ADC fault is routed to FCCU.
- All supplies are monitored internally for functionality and safety requirements and are to be used by the ADC in the application for debug or informational purposes only.
- See the device IOMUX file attached to this document for the availability of ADC channels for different devices and packages.

Table 341. ADC maximum clock frequency

Feature	S32K389/S32K388	S32K358/ S32K356/ S32K348/ S32K338/ S32K328	S32K344/ S32K324/ S32K314/ S32K342/ S32K322/ S32K341	S32K312	S32K311/ S32K310
ADC module clock frequency (f_mc)	160 MHz	240 MHz	160/120 MHz	120 MHz	120 MHz

NOTE

For register configurations need for different frequencies, see 'Functional description' section in this chapter.

ADC Channel mapping Table

Table 342. ADC channel mapping

Channel Number	ADC_0	ADC_1	ADC_2
0	ADC0_P0	ADC1_P0	ADC2_P0
1	ADC0_P1	ADC1_P1	ADC2_P1
2	ADC0_P2	ADC1_P2	ADC2_P2
3	ADC0_P3	ADC1_P3	ADC2_P3
4	ADC0_P4	ADC1_P4	ADC2_P4
5	ADC0_P5	ADC1_P5	ADC2_P5
6	ADC0_P6	ADC1_P6	ADC2_P6
7	ADC0_P7	ADC1_P7	ADC2_P7
8	—[1]	—	—
9	—	—	—
10	—	—	—
11	—	—	—
12	—	—	—
13	—	—	—
14	—	—	—
15	—	—	—
16	—	—	—
17	—	—	—
18	—	—	—
19	—	—	—
20	—	—	—
21	—	—	—

Table continues on the next page...

Table 342. ADC channel mapping (continued)

Channel Number	ADC_0	ADC_1	ADC_2
22	—	—	—
23	—	—	—
24	—	—	—
25	—	—	—
26	—	—	—
27	—	—	—
28	—	—	—
29	—	—	—
30	—	—	—
31	—	—	—
32	ADC0_S[8]	ADC1_S[8]	ADC2_S[8]
33	ADC0_S[9]	ADC1_S[9]	ADC2_S[9]
34	ADC0_S[10]	ADC1_S[10]	ADC2_S[10]
35	ADC0_S[11]	ADC1_S[11]	ADC2_S[11]
36	ADC0_S[12]	ADC1_S[12]	ADC2_S[12]
37	ADC0_S[13]	ADC1_S[13]	ADC2_S[13]
38	ADC0_S[14]	ADC1_S[14]	ADC2_S[14]
39	ADC0_S[15]	ADC1_S[15]	ADC2_S[15]
40	ADC0_S[16]	ADC1_S[16]	ADC2_S[16]
41	ADC0_S[17]	ADC1_S[17]	ADC2_S[17]
42	ADC0_S[18]	ADC1_S[18]	ADC2_S[18]
43	ADC0_S[19]	ADC1_S[19]	ADC2_S[19]
44	ADC0_S[20]	ADC1_S[20]	ADC2_S[20]
45	ADC0_S[21]	ADC1_S[21]	ADC2_S[21]
46	ADC0_S[22]	ADC1_S[22]	ADC2_S[22]
47	ADC0_S[23]	ADC1_S[23]	ADC2_S[23]
48	Bandgap	Bandgap	Bandgap
49	Tempsensor output	Tempsensor output	Tempsensor output
50	ANAMUX_OUT	—	—
51	—	—	—
52	—	—	—
53	—	—	—

Table continues on the next page...

Table 342. ADC channel mapping (continued)

Channel Number	ADC_0	ADC_1	ADC_2
54	VREFL	VREFL	VREFL
55	VREFH	VREFH	VREFH
56	—	—	—
57	—	—	—
58	—	—	—
59	—	—	—
60	—	—	—
61	—	—	—
62	—	—	—
63	—	—	—
64	ADC0_X[0] ADC0_MA[2:0]=3'h0	ADC1_X[0] ADC1_MA[2:0]=3'h0	—
65	ADC0_X[0] ADC0_MA[2:0]=3'h1	ADC1_X[0] ADC1_MA[2:0]=3'h1	—
66	ADC0_X[0] ADC0_MA[2:0]=3'h3	ADC1_X[0] ADC1_MA[2:0]=3'h3	—
67	ADC0_X[0] ADC0_MA[2:0]=3'h2	ADC1_X[0] ADC1_MA[2:0]=3'h2	—
68	ADC0_X[0] ADC0_MA[2:0]=3'h6	ADC1_X[0] ADC1_MA[2:0]=3'h6	—
69	ADC0_X[0] ADC0_MA[2:0]=3'h7	ADC1_X[0] ADC1_MA[2:0]=3'h7	—
70	ADC0_X[0] ADC0_MA[2:0]=3'h5	ADC1_X[0] ADC1_MA[2:0]=3'h5	—
71	ADC0_X[0] ADC0_MA[2:0]=3'h4	ADC1_X[0] ADC1_MA[2:0]=3'h4	—
72	ADC0_X[1] ADC0_MA[2:0]=3'h0	ADC1_X[1] ADC1_MA[2:0]=3'h0	—
73	ADC0_X[1] ADC0_MA[2:0]=3'h1	ADC1_X[1] ADC1_MA[2:0]=3'h1	—

Table continues on the next page...

Table 342. ADC channel mapping (continued)

Channel Number	ADC_0	ADC_1	ADC_2
74	ADC0_X[1] ADC0_MA[2:0]=3'h3	ADC1_X[1] ADC1_MA[2:0]=3'h3	—
75	ADC0_X[1] ADC0_MA[2:0]=3'h2	ADC1_X[1] ADC1_MA[2:0]=3'h2	—
76	ADC0_X[1] ADC0_MA[2:0]=3'h6	ADC1_X[1] ADC1_MA[2:0]=3'h6	—
77	ADC0_X[1] ADC0_MA[2:0]=3'h7	ADC1_X[1] ADC1_MA[2:0]=3'h7	—
78	ADC0_X[1] ADC0_MA[2:0]=3'h5	ADC1_X[1]	—
79	ADC0_X[1] ADC0_MA[2:0]=3'h4	ADC1_X[1] ADC1_MA[2:0]=3'h4	—
80	ADC0_X[2] ADC0_MA[2:0]=3'h0	ADC1_X[2] ADC1_MA[2:0]=3'h0	—
81	ADC0_X[2] ADC0_MA[2:0]=3'h1	ADC1_X[2] ADC1_MA[2:0]=3'h1	—
82	ADC0_X[2] ADC0_MA[2:0]=3'h3	ADC1_X[2] ADC1_MA[2:0]=3'h3	—
83	ADC0_X[2] ADC0_MA[2:0]=3'h2	ADC1_X[2] ADC1_MA[2:0]=3'h2	—
84	ADC0_X[2] ADC0_MA[2:0]=3'h6	ADC1_X[2] ADC1_MA[2:0]=3'h6	—
85	ADC0_X[2] ADC0_MA[2:0]=3'h7	ADC1_X[2] ADC1_MA[2:0]=3'h7	—
86	ADC0_X[2] ADC0_MA[2:0]=3'h5	ADC1_X[2] ADC1_MA[2:0]=3'h5	—
87	ADC0_X[2] ADC0_MA[2:0]=3'h4	ADC1_X[2] ADC1_MA[2:0]=3'h4	—
88	ADC0_X[3]	ADC1_X[3]	—

Table continues on the next page...

Table 342. ADC channel mapping (continued)

Channel Number	ADC_0	ADC_1	ADC_2
	ADC0_MA[2:0]=3'h0	ADC1_MA[2:0]=3'h0	
89	ADC0_X[3] ADC0_MA[2:0]=3'h1	ADC1_X[3] ADC1_MA[2:0]=3'h1	—
90	ADC0_X[3] ADC0_MA[2:0]=3'h3	ADC1_X[3] ADC1_MA[2:0]=3'h3	—
91	ADC0_X[3] ADC0_MA[2:0]=3'h2	ADC1_X[3] ADC1_MA[2:0]=3'h2	—
92	ADC0_X[3] ADC0_MA[2:0]=3'h6	ADC1_X[3] ADC1_MA[2:0]=3'h6	—
93	ADC0_X[3] ADC0_MA[2:0]=3'h7	ADC1_X[3] ADC1_MA[2:0]=3'h7	—
94	ADC0_X[3] ADC0_MA[2:0]=3'h5	ADC1_X[3] ADC1_MA[2:0]=3'h5	—
95	ADC0_X[3] ADC0_MA[2:0]=3'h4	ADC1_X[3] ADC1_MA[2:0]=3'h4	—

1. Channels marked with “-“ means they are reserved and access to any reserved channel may result in inappropriate data.

NOTE

The decoding signals for the selection of external ADC channels (ADC_X) are gray-encoded.

NOTE

Value of ANAMUX_OUT is selected by DCMRWF1[SUPPLY_MON_SEL] field.

NOTE

The module clock mentioned in this chapter is CORE_CLK for this chip.

60.1.2 ANAMUX for internal supply monitoring

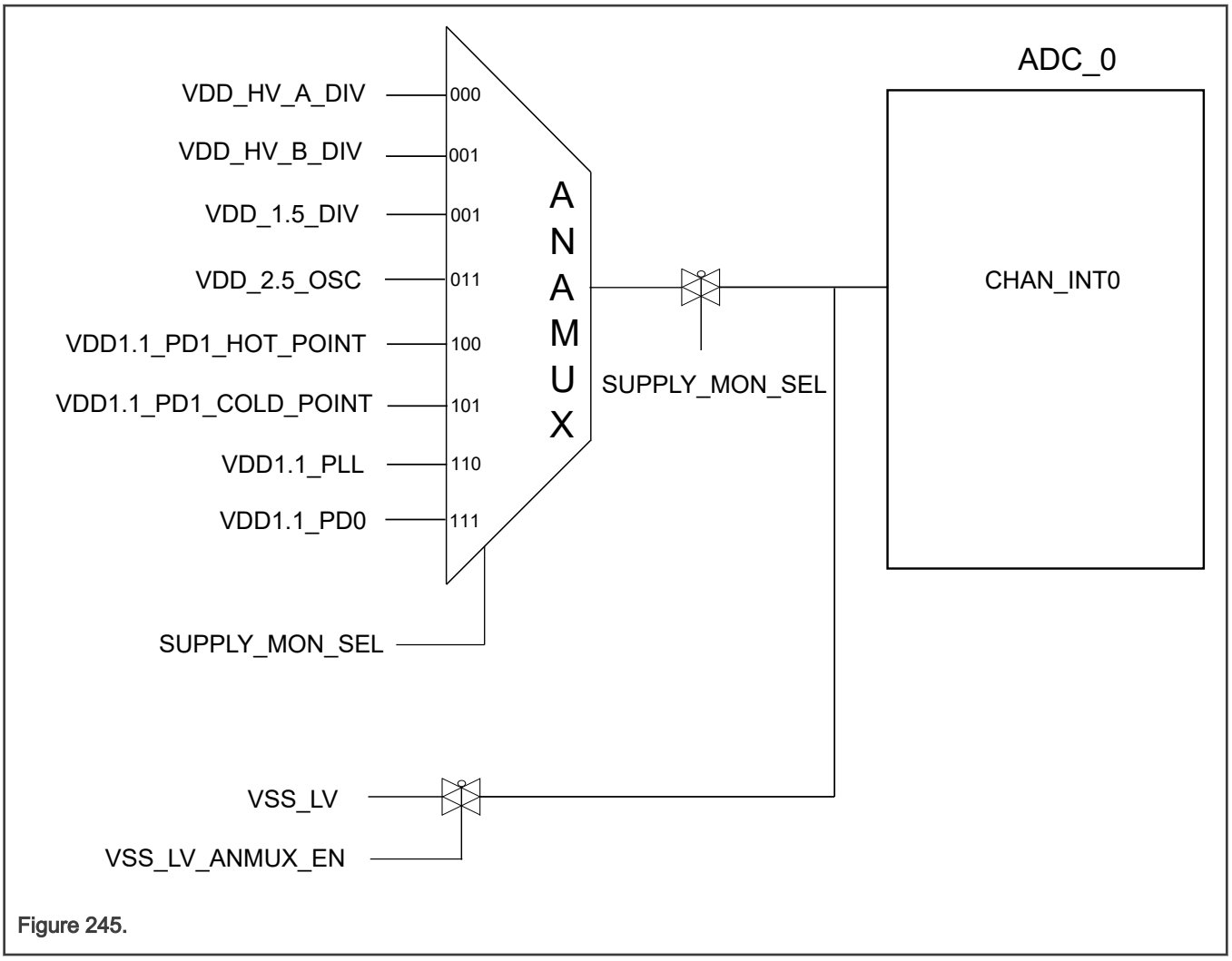


Figure 245.

NOTE

VDD_HV_B_DIV, VDD_1.5_DIV are not available in S32K312 and S32K311.

60.1.3 BCTU Interface

All 3 instances of ADC are triggered from the BCTU. The BCTU provides channel conversion commands to the ADC (includes channel number information). In addition, the ADC provides the conversion result back to the BCTU.

NOTE

Self-test conversion in BCTU mode is not supported in S32K3xx devices.

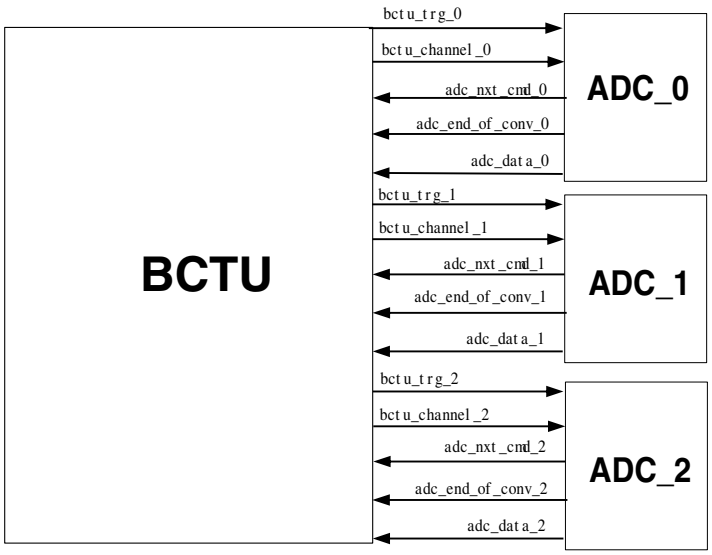


Figure 246. ADC Cross-triggering Unit

BCTU gives trigger pulse to all the 72 channels of ADC to initiate a conversion based on BCTU channel number.

BCTU-ADC result register

ADC result registers (ADCx_PCDRn[CDATA], ADCx_ICDRn[CDATA] and ADCx_ECDRn[CDATA]) store 15 bit conversion data. The BCTU ADC registers store 15-bit conversion data (BCTU_ADC0DR[ADC0_DATA], BCTU_ADC1DR[ADC1_DATA] and BCTU_ADC2DR[ADC2_DATA]).

60.1.4 DMA interface

Each ADC supports a single DMA request that can be requested after the conversion of every channel. The conversion result is stored into a register DMAR and is transferred via DMA or host access.

NOTE

The last ongoing conversion will be aborted as soon as ADC gets STOP request from the software and it will move to STOP mode.

60.1.5 Hardware triggers

In this chip, hardware trigger signals can be provided from TRGMUX outputs. This feature enables synchronous conversion of two independent ADC instances in parallel. If ADC is in Idle state (that is, no conversion phase ongoing and the MCR[PWDN] and MCR[ACKO] fields are 0) and the MCR[XSTRTEN] is set, an event on the external start signal causes ADC to start either normal or injected conversion operation. The MCR[NSTART]/MCR[JSTART] field is automatically set respectively. The normal conversion sync pulse is used with normal conversion trigger to synchronize the start timing of normal conversion between multiple ADC instances.

Table 343. Hardware triggers

TRGMUX output number	Hardware trigger functions
0	ADC_0 Normal Conversion
1	ADC_0 Injected Conversion
2	ADC_0 Normal Conversion Sync Pulse

Table continues on the next page...

Table 343. Hardware triggers (continued)

TRGMUX output number	Hardware trigger functions
4	ADC_1 Normal Conversion
5	ADC_1 Injected Conversion
6	ADC_1 Normal Conversion Sync Pulse
8	ADC_2 Normal Conversion
9	ADC_2 Injected Conversion
10	ADC_2 Normal Conversion Sync Pulse

60.1.6 ADC Self-Test

It is important to check at regular intervals if the ADC is operating correctly. For this purpose a self test feature is provided. When self-test is enabled, the ADC automatically checks its components and flags errors.

60.1.7 ADC mux-mode channels

There are some channels in ADC which are driven by multiple pads. For this, GPR bits from DCM are used to define which pad is connected to a particular ADC channel.

For example as below, where ADC0 S8 is driven by GPIO_PAD0 (default) and GPIO_PAD45 (mux_mode). ADC2 S8 is driven by GPIO_PAD133 (default) and GPIO_PAD45 (mux_mode). ADC1 S8 has a dedicated pad GPIO45, so its TGATE doesn't need GPR bit gating. Similar implementation is for ADC0_S9 and ADC2_S9. Following DCM GPR bits have been used:

Table 344. ADC mux_mode GPR

ADC channel	GPR_bit	Connected pad
ADC1 Standard channel 18 (ADC1 S18)[1]	DCM.DCMRWF4[0]	0: GPIO PAD70 (default) 1: GPIO PAD6
ADC0 Standard channel 8 (ADC0 S8)	DCM.DCMRWF4[1]	0: GPIO PAD0 (default) 1: GPIO PAD45
ADC0 Standard channel 9 (ADC0 S9)	DCM.DCMRWF4[2]	0: GPIO PAD1 (default) 1: GPIO PAD46
ADC 1 standard channel 14 (ADC1 S14)	DCM.DCMRWF4[3]	0: GPIO PAD69 (default) 1: GPIO PAD32
ADC 1 standard channel 15 (ADC1 S15)	DCM.DCMRWF4[4]	0: GPIO PAD4 (default) 1: GPIO PAD33
ADC1 Standard channel 22 (ADC1 S22)	DCM.DCMRWF4[5]	0: GPIO PAD124 (default) 1: GPIO PAD145
ADC1 Standard channel 23 (ADC1 S23)	DCM.DCMRWF4[6]	0: GPIO PAD125 (default) 1: GPIO PAD146
ADC0 Standard channel 12 (ADC0 S12)	DCM.DCMRWF4[7]	0: GPIO PAD72(default)

Table continues on the next page...

Table 344. ADC mux_mode GPR (continued)

ADC channel	GPR_bit	Connected pad
		1: GPIO PAD78
ADC0 Standard channel 13 (ADC0 S13)[1]	DCM.DCMRWF4[8]	0: GPIO PAD73 (default) 1: GPIO PAD79
ADC2 Standard channel 8 (ADC2 S8)	DCM.DCMRWF4[9]	0: GPIO PAD133 (default) 1: GPIO PAD45
ADC2 Standard channel 9 (ADC2 S9)	DCM.DCMRWF4[10]	0: GPIO PAD132 (default) 1: GPIO PAD46
ADC0 Standard channel 14 (ADC0 S14)[1]	DCM.DCMRWF4[11]	0: GPIO PAD32 (default) 1: GPIO PAD80
ADC0 Standard channel 17 (ADC0 S14)[1]	DCM.DCMRWF4[12]	0: GPIO PAD75 (default) 1: GPIO PAD44
ADC0 Precision channel 2 (ADC0 P2)[1]	DCM.DCMRWF4[28]	0: GPIO PAD8 (default) 1: GPIO PAD10

1. Applicable for S32K311 and S32K310 only.

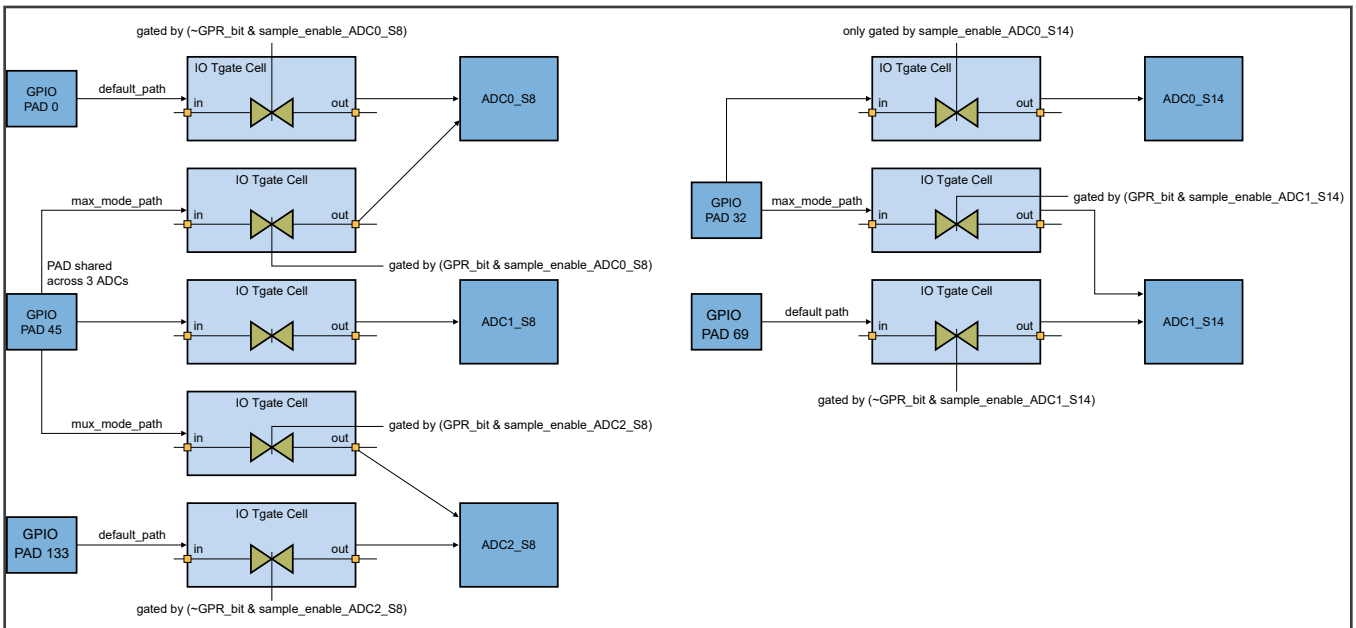


Figure 247. ADC channel interleaving

60.2 Overview

ADC produces a digital value from an input analog voltage using the SAR algorithm. It features various methods to trigger a conversion and offers flexibility in the selection of input channels. ADC is useful for a wide range of applications.

A configurable self-test capability supports use cases with increased requirements for functional safety.

60.2.1 Block diagram

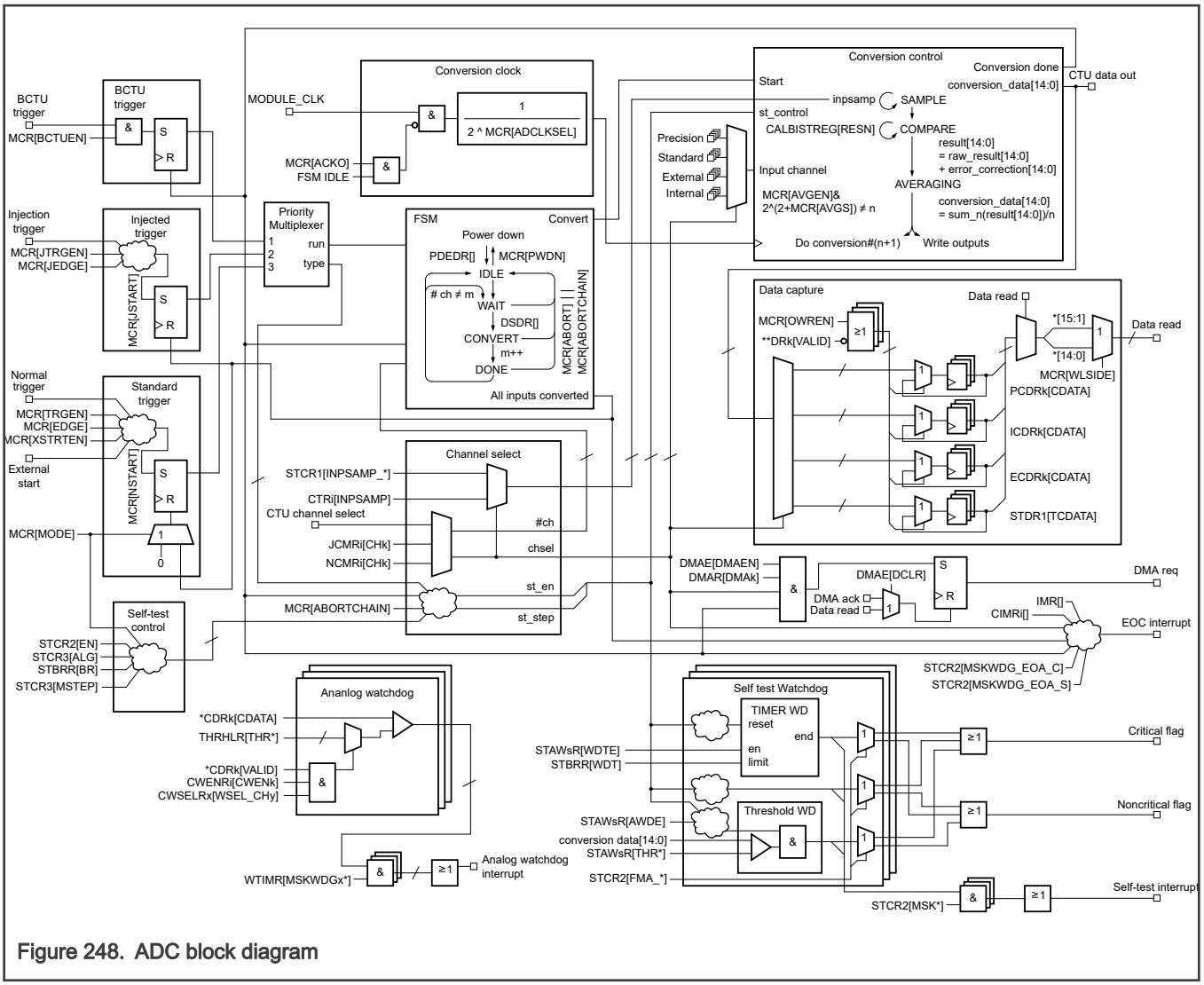


Figure 248. ADC block diagram

Guide to the diagram:

- The signal flow through the diagram is from left to right: inputs to blocks are on the left side, outputs are on the right.
- Block names have the first letters in uppercase and the other letters in lowercase—"Conversion Control," for example).
- Register names are in uppercase followed by a field name in brackets ("MCR[NSTART]," for example). Letters in lowercase are indexes. Asterisks are placeholders for letters (wildcards). Register names in the block diagram map to the names in ADC register descriptions.
- Signal names are in lowercase ("conversion done," for example).
- All digital signals shown are synchronous signals. Asynchronous inputs to the ADC are internally synchronized (synchronizers not shown).

The block diagram shows the signal flow through ADC from the left to the right and from the top to the bottom:

1. When ADC receives one of the three different types of triggers (BCTU, injected, or normal), the priority multiplexer initiates the conversion.
2. The input to be converted is selected in the Channel Select subblock, based on the type of trigger and on the self-test configuration.

3. The state machine (FSM) transitions from Idle state to Convert state. In case the channel to convert changed since the last conversion and this new channel is an external channel, the conversion starts after a delay configured by DSDR[DSD].
4. After a conversion begins, the Conversion Control block holds the internal capacitance network CDAC in the sample phase for a time defined by CTRi[INPSAMP].
5. During the compare phase, CALBISTREG[RESN] defines the number of steps used to execute the successive approximation algorithm. The error_correction values, determined during calibration, are added to the raw result. If averaging is enabled (MCR[AVGEN] = 1), up to 32 (specified via MCR[AVGS]) conversion results are averaged to obtain the result.
6. When the conversion completes and all averaging steps have completed:
    a. The conversion done signal indicates completion.
    b. Conversion_data[14:0] is written to the result registers in the Data Capture subblock.
    c. The FSM transitions into Done state.
7. The next input in the Channel Select block is selected and the next conversion is started by the transition of the FSM to Convert state. This continues until all selected inputs have been converted.
8. If an analog watchdog is enabled, the conversion result is compared to the configured threshold. If the limits are exceeded, an interrupt can be triggered.
9. Self-test can check the integrity of ADC, either interleaved with normal conversions or as a standalone check.
10. If an ongoing conversion is aborted (by writing 1 to MCR[ABORT]) or an ongoing conversion of a set of input channels is aborted (by writing 1 to MCR[ABORTCHAIN]), the FSM transitions into Idle state. The Conversion Control block stops the current conversion if MCR[ABORT] was written with 1, or finishes the current conversion if MCR[ABORTCHAIN] was written with 1. In both cases, the conversion done output signal of the Conversion Control block is set to 1.

Some details are not shown in the block diagram:

- Calibration must be performed to determine the error_correction values (see Calibration) before ADC can do meaningful conversions.
- An input can presample the internal low or high reference voltages to avoid a memory effect that may be present in the comparison due to a previous conversion (see Presampling).
- Monitor ADC status via status flags.
    - See Main Status (MSR) for the FSM state.
    - See Interrupt Status (ISR) to monitor interrupts.
    - See Channel End Of Conversion Flag For Precision Inputs (CEOCFR0) for conversion phase per input.
    - See the following registers to monitor analog watchdogs:
        - Analog Watchdog Out Of Range For Precision Inputs (AWORR0)
        - Analog Watchdog Out Of Range For Standard Inputs (AWORR1)
        - Analog Watchdog Out Of Range For External Inputs (AWORR2)

      See Analog Watchdog Threshold Interrupt Status (WTISR) to monitor analog watchdog interrupts.
    - See Self-Test Status 1 (STSR1) – Self-Test Status 4 (STSR4) to monitor self-test.

60.2.2 Features

- Selectable resolution (8-, 10-, 12-, 14-bit). Note that the conversion result is always 15 bits wide, even though the selected resolution is smaller (see CALBISTREG[RESN])
- Conversion data captured in a separate register for each input channel.
- Option to improve accuracy via averaging, which calculates conversion data by averaging the data of up to 32 conversions.

- Conversion triggers:
  - Normal conversion trigger converts a number of input channels, either once per trigger or continuously.
  - Injected conversion trigger interrupts an ongoing normal conversion and converts another set of input channels.
  - BCTU conversion trigger interrupts an ongoing conversion and converts an input channel, in which the input is selected and the conversion is started via the BCTU.
- An analog watchdog optionally monitors conversion data for each input channel and issues an interrupt if the converted data is below or above configurable limits.
- DMA functionality transfers conversion data to other modules.
- Programmable interrupts optionally issue an interrupt when conversion of one or of a set of input channels is finished.
- Self-test functions validate ADC structural integrity during functional operation and generate events with different severities on any finding.
- Conversion clock (AD_clk) control enables the use of ADC in systems with a higher clock frequency by using internal clock dividers.
- Auto turn-off of the conversion clock when ADC is idle.

60.3 Functional description

60.3.1 Clock

ADC is controlled by one clock signal, the module clock. Internally, the conversion circuit is controlled by the conversion clock, which is derived from the module clock.

The frequency of the conversion clock has to be within the limits defined in the data sheet. If the module clock frequency is higher than the maximum frequency of the conversion clock allowed during the functional conversion or during the calibration (see the chip data sheet), then you must configure the ADC conversion clock divider (MCR[ADCLKSEL]) so that the frequency of the conversion clock is within allowed limits. See Table 345.

Table 345. Clock configuration for highest conversion speeds

Module Clock Frequency f_mc[1]	Configuration during calibration	Configuration during functional conversion
40 MHz < f_mc ≤ 80 MHz	MCR[ADCLKSEL] = 1h AMSIO[HSEN] = 0h AMSIO[CMPCTRL0] = 0b	MCR[ADCLKSEL] = 0h AMSIO[HSEN] = 0h AMSIO[CMPCTRL0] = 0b
80 MHz < f_mc ≤ 120 MHz	MCR[ADCLKSEL] = 1h AMSIO[HSEN] = 1h AMSIO[CMPCTRL0] = 1b	MCR[ADCLKSEL] = 0h AMSIO[HSEN] = 1h AMSIO[CMPCTRL0] = 1b
120 MHz < f_mc ≤ 160 MHz	MCR[ADCLKSEL] = 2h AMSIO[HSEN] = 0h AMSIO[CMPCTRL0] = 0b	MCR[ADCLKSEL] = 1h AMSIO[HSEN] = 0h AMSIO[CMPCTRL0] = 0b
160 MHz < f_mc ≤ 240 MHz	MCR[ADCLKSEL] = 2h AMSIO[HSEN] = 1h AMSIO[CMPCTRL0] = 1b	MCR[ADCLKSEL] = 1h AMSIO[HSEN] = 1h AMSIO[CMPCTRL0] = 1b

Table continues on the next page...

Table 345. Clock configuration for highest conversion speeds (continued)

Module Clock Frequency f_mc[1]	Configuration during calibration	Configuration during functional conversion
240 MHz < f_mc ≤ 320 MHz	MCR[ADCLKSEL] = 3h AMSIO[HSEN] = 0h AMSIO[CMPCTRL0] = 0b	MCR[ADCLKSEL] = 2h AMSIO[HSEN] = 0h AMSIO[CMPCTRL0] = 0b
320 MHz < f_mc ≤ 480 MHz	MCR[ADCLKSEL] = 3h AMSIO[HSEN] = 1h AMSIO[CMPCTRL0] = 1b	MCR[ADCLKSEL] = 2h AMSIO[HSEN] = 1h AMSIO[CMPCTRL0] = 1b

1. Lower resulting frequencies are allowed—see the chip data sheet.

When ADC is not converting (state machine is in Idle state), the conversion clock can be turned off (MCR[ACKO]).

60.3.2 Modes of operation

ADC is always in Functional mode. No other mode selection exists.

- Put ADC into Power Down state by writing 1 to MCR[PWDN] to reduce power consumption.
- Gate the clock signal by writing 1 to MCR[ACKO] when ADC is in Idle state.

60.3.3 Conversion

ADC measures the voltage of a signal—it converts an analog input value into a digital representation. When you read the conversion result, you must divide this conversion result by the number of codes (defined by selected conversion resolution in CALBISTREG[RESN]), and multiply it with the reference voltage value. This calculation provides the voltage value of the converted signal.

To convert an analog input voltage into its digital representation, ADC:

1. Receives a signal, referred to as a trigger, indicating it is to perform a conversion.
2. Captures the voltage of a signal via internal capacitances in a process referred to as sampling.
3. Compares the voltage to the reference voltage, using the SAR algorithm.

Before a conversion can be performed, you must select:

- The trigger source(s) to use (on the chip level)
- The input channel(s) to convert (NCMR0–2, JCMR0–2)

Additionally, you can tailor the conversion to meet your needs by changing the default configuration. For example, you can:

- Configure ADC to convert the selected input channel(s) only once or continuously in a loop (MCR[MODE]).
- Specify the duration that ADC samples the input signal (CTR0/1/2).
- Specify the resolution of the conversion result by selecting the number of bits (CALBISTREG[RESN]).
- Enable an analog watchdog to generate an interrupt when a conversion result falls outside a specified range (see Analog watchdog functions).
- Enable the DMA interface to send requests and receive acknowledgments (see DMA functionality).
- Check the operational integrity of ADC by running a self-test (see Self-test).

After power-up or a functional reset, ADC remains in Power Down state until enabled by writing 0 to MCR[PWDN]. After the system is powered on, you must run the calibration (see Calibration), before you can do a meaningful conversion with ADC.

Some configuration fields (listed below) are writeable only in Power Down state. If you intend to write to them, you must do so before exiting Power Down state.

- Select the conversion clock frequency to be within the allowed limits for a conversion.

60.3.4 Normal trigger

Select the input channels to convert by configuring the fields in NCMR0, NCMR1, and NCMR2.

These registers must be programmed before the start of conversion. You cannot reconfigure them until the conversion of all selected channels is complete. The sequence is always to convert the selected input channels in this order:

1. Start from the lowest input channel of the precision input channels.
2. Proceed through the standard input channels in ascending order.
3. End with the highest input channel of the external input channels.

60.3.4.1 Starting conversions in normal conversion mode

After programming the required fields of the following registers, you can start a conversion by writing 1 to MCR[NSTART]:

- Normal Conversion Enable For Precision Inputs (NCMR0)
- Normal Conversion Enable For Standard Inputs (NCMR1)
- Normal Conversion Enable For External Inputs (NCMR2)
- Main Configuration (MCR)

A hardware trigger can also start a normal conversion.

- If the external trigger is enabled (MCR[TRGEN] = 1), an external trigger enable is detected to start the conversion. The enable is checked only during start of conversion.
- Detection of an active edge defined by MCR[EDGE] (rising = 1, falling = 0) on an external trigger transitions MCR[NSTART] to 1 and starts the normal conversion.
- When MCR[TRGEN] = 1 (for example, when external trigger is enabled), conversion can be started by software.
- Any external trigger or software initiated conversion in normal conversion mode is ignored during any ongoing conversion chain.

NOTE

To have reliable operation there must be a gap of 3 clock cycles between configuration of NCMRx and the start of a conversion (by writing 1 to MCR[NSTART], or arrival of an external or internal hardware start trigger). ADC calculates the number of channels to be converted during this period.

60.3.4.2 Operation modes in normal conversions

Two operational modes are available during normal conversion:

- One-Shot mode
- Scan mode

For normal conversion, select the mode via MCR[MODE]. The first phase of the conversion process involves sampling the analog channel. The next phase is the conversion phase, curing which the sampled analog value is converted to digital as shown in the following figure.

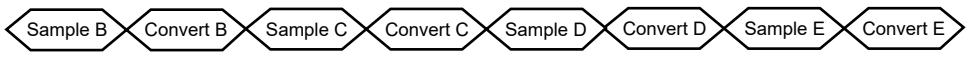


Figure 249. Normal conversion flow

In One-Shot mode (MCR[MODE] = 0), a sequential conversion specified in the NCMR*n* mask registers is performed only once. At the end of each conversion, the digital result of the conversion is stored in the corresponding data register (CDR*n*).

For example: Channels A→B→C→D→E→F→G→H are present in a device where channels B→D→E are to be converted in One-Shot mode. Conversion begins with channel B, followed by conversion of channels D and E. At the end of conversion of channel E, the scanning of channels stops.

MSR[NSTART] automatically transitions to 1 when a normal conversion starts. At the same time, MCR[NSTART] is reset to zero by hardware, enabling software to program a new start of conversion in advance. In this case, the new requested conversion starts after completion of the running/current conversion. However, for correct functioning of the device, you should program MCR[NSTART] for the new conversion only after completion of the current conversion. The application can wait for an ECH interrupt prior to writing 1 to MCR[NSTART] again.

If an external trigger starts the conversion chain, ADC does not observe the trigger input until the conversion chain is finished. After the chain is finished, the next chain can be triggered by another hardware trigger edge.

In Scan mode operation (MCR[MODE]=1), a sequential conversion of N channels specified in the NCMR*n* registers is continuously performed. At the end of each conversion, the result is stored in the corresponding data register, as in One-Shot mode.

MSR[NSTART] automatically transitions to 1 when a normal conversion starts. Unlike One-Shot mode, MCR[NSTART] is not reset to 0 in Scan mode. It can be reset by software when you must exit Scan mode. In that case, ADC completes the current chain and after the last conversion, it resets MCR[NSTART] to 0.

For example: Channels A→B→C→D→E→F→G→H are present in the device where channels B→D→E are to be converted in Scan mode. MCR[MODE] = 1 selects Scan mode operation. Conversion starts from channel B followed by conversion of channels D→E. After conversion of channel E, the scanning of channel B starts followed by conversion of the channels D→E. This sequence repeats itself until MCR[NSTART] is reset to 0 by software.

If an external trigger starts a conversion, then MCR[NSTART] does not transition to 1. This scan chain can be stopped by writing 0 to MCR[NSTART]. The conversion stops when the ongoing chain is finished. Consequently, after it starts, the only way to stop Scan mode conversion is to write 0 to MCR[MODE].

End of conversion

In both modes, at the end of each conversion, an End of Conversion (EOC) interrupt is issued if enabled by the corresponding mask fields in CIMR*n* and IMR).

After conversion of all selected channels in NCMR*n* is complete, the conversion operation is considered finished. Then ISR[ECH] transitions to 1 and the End of Chain (ECH) interrupt is issued (if enabled in IMR[MSKECH]).

The corresponding channel field in the CEOCFR*n* register(s) is updated to indicate that data is available on data register CDR*n* of the respective channel.

If there is no channel selected in NCMR*n* and there is a start-of-conversion trigger, then ISR[ECH] is set to 1 and the End of Chain (ECH) interrupt is immediately issued (if enabled).

60.3.5 Injected trigger

An injected trigger enables you to convert a set of input channels although the standard trigger has already started conversion of another set of input channels. Each channel can be individually enabled by writing 1 to the corresponding field in JCMR0, JCMR1, or JCMR2.

The set of input channels of the injected trigger is converted only once. A continuous conversion of a set of input channels in a loop is only possible when started by a standard trigger. When an injected trigger is received, an ongoing conversion started by a standard trigger is interrupted. After the set of input channels of the injected trigger is converted once, the interrupted conversion of the set of input channels of the standard trigger resumes from the input channel that was interrupted, as shown in the following figure.

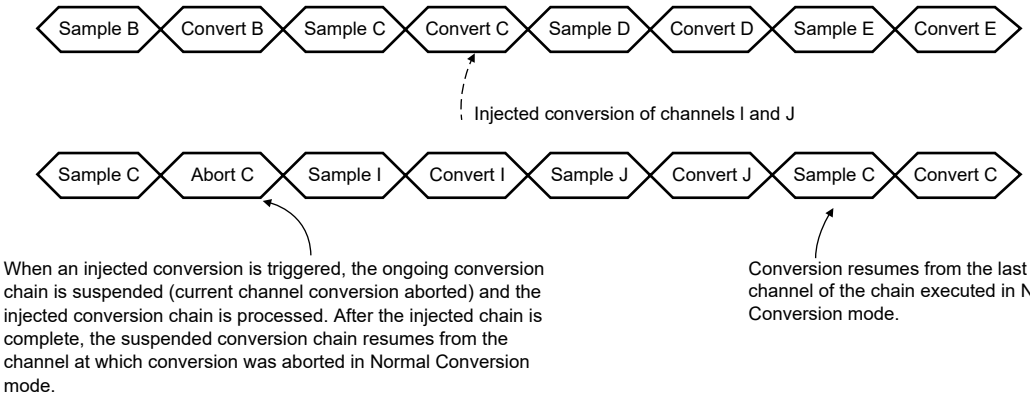


Figure 250. Injected trigger conversion sequence

60.3.5.1 Starting conversions in injected conversion mode

An injected conversion chain can be started:

- By software: When external triggering is disabled (MCR[JTRGEN] = 0), you can write 1 to MCR[JSTART], which causes the current conversion chain processed in Normal Conversion mode to be suspended and the injected chain to be processed.
- By external trigger: When external triggering is enabled (MCR[JTRGEN] = 1), a programmed event (rising/falling edge depending on the value of MCR[JEDGE]) on the injection external input starts the injected conversion.

MCR[JSTART] automatically transitions to 1 when the injected conversion chain starts. At the same time MCR[JSTART] is reset to 0, enabling software to program a new start of conversion in advance. In that case the new requested conversion starts after the running conversion completes.

At the end of each conversion, an End Of Injected Conversion (JEOC) interrupt is issued, if enabled by the corresponding mask field in Interrupt Mask (IMR). At the end of a chain, an End Of Injected Chain (JECH) interrupt is issued, if enabled by the corresponding mask field. Additionally, ADC writes 1 to ISR[JECH].

NOTE

If the content of all the Injected Conversion mask registers (JCMR*n*) is zero, that is, no channel is selected, the JECH interrupt is immediately issued after the start of conversion.

To have reliable ADC operation, there must be a gap of 3 cycles between configuration of JCMRx and starting of conversion (writing 1 to MCR[JSTART] or arrival of either an external or internal hardware start trigger). ADC calculates the total channel numbers to be converted during this period.

The corresponding channel bit in the CEOCFR*n* register is updated to indicate data is available in the channel's conversion data register, CDR*n*.

After starting, an injected chain conversion cannot be interrupted.

60.3.6 BCTU interface

The BCTU interface enhances ADC's injected conversion capability. It contains control inputs to select the channels to be converted from the appropriate event configuration register. Figure 251 shows the interface.

The BCTU generates a trigger (bctu_trigger) and a channel number (bctu_numchannel) to be converted. A single channel is converted for each request. After performing the conversion, ADC returns the result on the bctu_dataout bus together with two output signals named bctu_nextcmd and bctu_push. The assertion of signal bctu_nextcmd means ADC is ready to accept the next trigger from BCTU. The bctu_push signal is asserted at the end of conversion, meaning that conversion is finished and the conversion result available at output bctu_dataout is valid.

The conversion result is also saved in the corresponding channel's data register and is compared with analog watchdog thresholds if requested.

The signals bctu_trigger, bctu_nextcmd and bctu_push are all of type single-cycle active-high-pulse in the ADC clock domain. The channel number provided from BCTU must be valid when bctu trigger is active-high. The result data from ADC is valid with bctu_push high.

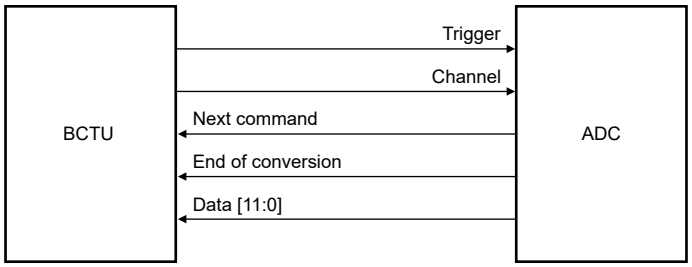


Figure 251. BCTU interface

The BCTU interface has two modes of operation:

- Trigger
- Control

To enable the BCTU interface, write 1 to MCR[BCTUEN]. The operating mode (Trigger or Control) can be fixed or programmable.

60.3.6.1 BCTU Trigger mode

In Trigger mode, normal and injected conversions can also be performed. All types of conversions can be initiated in this mode. The priorities among the three types of conversions are discussed below.

When a trigger is received, the channel number is taken as an injected channel value and the triggered injected conversion starts. MSR[BCTUSTART] transitions to 1 automatically at this point and it is also automatically reset to 0 when the triggered injected conversion is completed.

BCTU conversions must be requested only after successful ADC calibration. The application must prevent any BCTU trigger during calibration.

If a BCTU trigger is received during an ongoing injected conversion, the injected chain is aborted immediately and only the BCTU-triggered injected conversion proceeds. Additionally, MSR[JSTART] is reset to zero. The abort of an injected conversion is indicated by MSR[JABORT].

If a BCTU trigger is received during an ongoing normal conversion, the ongoing normal channel conversion is suspended and the BCTU-triggered injected conversion is processed. Normal conversion resumes from the suspended channel after completion of the BCTU-triggered conversion.

If a normal conversion is requested during BCTU conversion (MSR[BCTUSTART] = 1), the normal conversion starts after the BCTU conversion completes (MSR[BCTUSTART] is reset to zero).

Any injected conversion is discarded if requested during BCTU conversion and MCR[JSTART] is immediately reset to zero.

60.3.6.2 BCTU Control mode

In BCTU control mode (MCR[BCTU_MODE] = 0), only the BCTU can start a conversion. All other trigger sources are ignored.

Along with BCTU trigger, the information provided with signal bctu_numchannel is taken as the channel number for injected channel and BCTU triggered conversion starts. MSR[BCTUSTART] transitions to 1 automatically at the start of the conversion and it remains 1 unless BCTU is disabled by writing 0 to MCR[BCTUEN].

The conversion must be requested (generating bctu_trigger) when calibration has finished successfully. If a BCTU trigger is received during calibration execution, calibration is stopped immediately in order to satisfy the BCTU request. Calibration fails in this case.

60.3.7 Aborting a conversion

Two abort functions are provided:

- Abort of a single channel
- Abort of a chain

To abort an ongoing conversion, write 1 to MCR[ABORT]. The current conversion aborts and conversion of the next channel of the chain begins immediately.

Depending on the current state of a conversion, an Abort action may take 1 to 4 cycles of the bus clock. At the start of any conversion, and at the end of a particular conversion when internal state counters are changing, an abort action is delayed by a maximum of 3 cycles to put all states in a stable condition.

To ensure that the abort action completes, do not change MCR[ABORT] for the next 3 cycles.

Avoid writing 1 to MCR[ABORT] along with, and within 3 cycles of, writing 1 to MCR[NSTART] or MCR[JSTART].

During an abort:

- MCR[NSTART] and MCR[JSTART] remain 1, if not in the last channel of the chain.
- MCR[ABORT] is reset to 0 when the channel is aborted.
- The EOC interrupt corresponding to the aborted channel is not generated. This behavior applies to normal and injected conversions. If the last channel of a chain is aborted, the end of chain is reported by generating an ECH interrupt.

It is possible to abort the current chain of conversions by writing 1 to MCR[ABORTCHAIN]. In this case, the behavior of ADC depends on the value of MCR[MODE] (One-Shot/Scan operation modes). If Scan operation mode is disabled, MCR[NSTART] is automatically reset to 0 along with MCR[ABORTCHAIN]. Otherwise, in Scan Operation mode, a new chain is started. The EOC interrupt of the current aborted conversion is not generated but an ECH interrupt is generated to signal the end of the chain.

NOTE

For a single channel chain, an ECH interrupt is not generated for an abort or abort chain.

When an abort chain is requested while an injected chain is running over a suspended normal chain, both the injected and normal chains are aborted, and both MCR[NSTART] and MCR[JSTART] are reset to 0.

60.3.8 Configuring ADC clock divider and sample time settings

You can scale the AD_clk frequency via MCR[ADCLKSEL]. ADCLKSEL can only be written in Power Down state (MCR[PWDN] = 1). Depending on the frequency of the module clock, there might be different settings necessary for functional conversion and for calibration. See the chip data sheet for ADC electrical characteristics.

Three conversion timing registers are used (CTR*n*) to support different sampling times for the different types of channels. An exception exists for the temperature sensor channel, which always uses the Conversion Timing For Standard Inputs (CTR1) value. See the register description for details.

60.3.9 Presampling

The presampling feature enables pre-charge or discharge of the ADC internal sample capacitor node to a defined level before sampling of the selected analog input channel starts. This is useful for resetting information (history effect/offset) from the most recent conversion. During presampling, ADC samples the internally generated voltage. During the sampling phase, ADC samples analog input coming from pads.

Enable presampling on a per-channel basis by setting the corresponding fields in the PSR*n* registers to one.

After enabling presampling for a channel, the normal sequence of operation for the channel is:

1. Presampling
2. Sampling
3. Evaluation

You can bypass sampling of all channels by writing 1 to PSCR[PRECONV]. When sampling of a channel is bypassed, the sampled and stored internal voltage applied during the presampling phase is converted (Figure 252). See Conversion time for the timing equation for conversion.

Presampling does not apply for self-test channels.

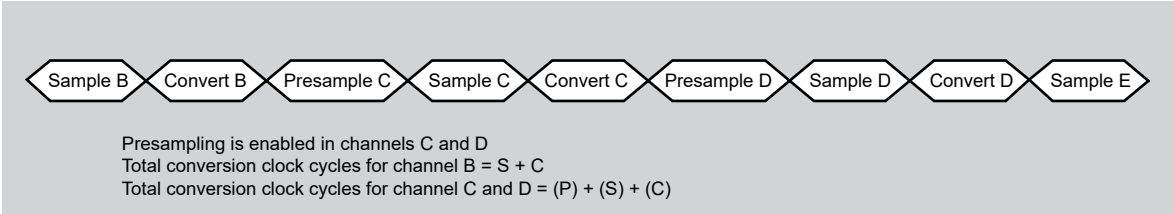


Figure 252. Presampling sequence

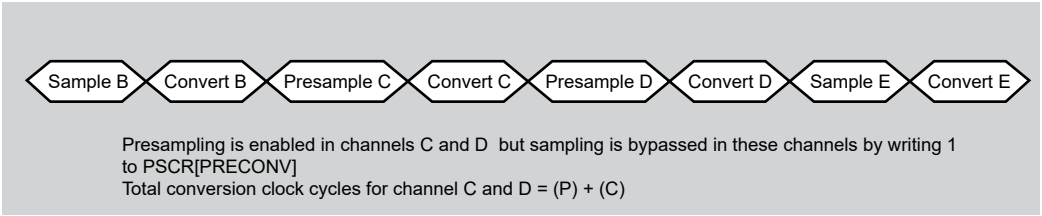


Figure 253. Presampling sequence with PRECONV=1

60.3.9.1 Enabling presampling per channel

Enable presampling for specific channels by programming Presampling Control (PSCR). You can select between two internally generated voltages by configuring the PREVAL*n* fields in Presampling Control (PSCR) to the appropriate values, as given in the following table.

Table 346. Presampling voltage selection via PREVAL*n*

PREVAL*n*	Presampling voltage
0	Presample voltage – 0: VREFL
1	Presample voltage – 1: VREFH

The presampling configuration field(s) (PSCR[PREVAL0], PSCR[PREVAL1], PSCR[PREVAL2]) are located in Presampling Control (PSCR). These fields enable selection of the presampling channel.

The temperature sensor channel has an exception. This channel is mapped with a fixed channel depending on device configuration, but it uses the presampling configuration defined for channel group 2. So, when the temperature sensor channel is selected, the presampling voltage defined by PSCR[PREVAL1] is selected. This voltage might be different from the one selected for channel group 1 via PSCR[PREVAL0].

60.3.10 Analog watchdog functions

When enabled, an analog watchdog monitors conversion results for an associated channel and reports an issue if it determines a result is outside customer-defined limits (as shown in Analog watchdog configuration (up to 16 watchdogs)). These limits are specified by an upper and a lower threshold value named THRH and THRL respectively.

After the conversion of the selected channel a comparison is performed between the converted value and the threshold values. If the converted value is outside that threshold value, then ADC generates a corresponding threshold violation interrupt.

The comparison result is stored as HAWIFx and LAWIFx fields in Analog Watchdog Threshold Interrupt Status (WTISR), as explained in the following table. Depending on the values of WTIMR[LAWIFENx] and WTIMR[HDWIFENx] mask fields, an interrupt is generated on a threshold violation.

Table 347. Values of HAWIFx and LAWIFx

HAWIFx	LAWIFx	Conversion result
1	0	Conversion result > THRH
0	1	Conversion result < THRL
0	0	THRH ≥ conversion result ≥ THRL

Depending on the chip's factory settings, up to 16 analog watchdogs are available. See the chip-specific configuration information for availability.

There are two types of mutually exclusive settings for analog watchdogs. Both types are described in the following sections. The availability depends on device configuration. See the chip-specific configuration information for availability.

Different capabilities are available with the two settings, as described below.

60.3.10.1 Channel analog watchdog select registers

ADC configuration includes selecting the analog watchdog threshold register (THRHLR*n*) that provides limits to monitor inputs of a given type. You select the THRHLR*n* using CWSELR*a*l*n* registers.

Table 348. CWSELR*a*l*n* register naming

Index	Description
a	ADC input type associated with a CWSELR*a*l*n* register. • P = Precision inputs • S = Standard inputs • E = External inputs
n	0-indexed register number.

60.3.10.2 Analog watchdog configuration (up to 16 watchdogs)

ADC can have up to 16 watchdogs in the factory configuration. (See the chip-specific configuration information for availability.) For each watchdog there is one threshold value register, THRHLR*n*, that contains the high and low thresholds.

You can independently enable the analog watchdog for each channel by programming the appropriate CWENR*n* register.

The threshold values for each channel can be selected independently from a maximum of 16 threshold registers (THRHLR*n*) using the corresponding WSEL_CH*n* field in the appropriate CWSELR*a*l*n*, where *a* indicates the input type:

- P = precision
- S = standard
- E = external

Each CWSELR*a*l*n* register is associated with 8 consecutive channels. CWSELR*a*l0 holds 8 WSEL_CH fields for channels 0 to 7. CWSELR*a*l1 is for channels 8 to 15 and so on. The availability of fields and registers depends on device configuration. See the chip-specific configuration information for availability.

If the conversion result of a selected channel is outside the range specified by threshold values, then the corresponding field in the Analog Watchdog Out of Range register (AWORR*n*) transitions to 1. For example, if channel 7 is to be monitored with the threshold values in THRHLR3, then CWSELR*a*l0[WSEL_CH7] must be programmed with 3. Enabling the watchdog is done by writing 1 to the corresponding field for channel 7 in CWENR0.

In this configuration, a set of threshold values (THRHLR*n*) can be linked to several ADC channels.

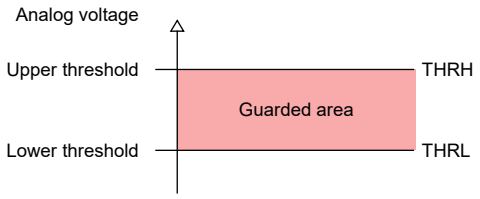


Figure 254. Guarded area

NOTE

If the higher threshold of the analog watchdog is programmed below the lower threshold and the conversion result is below the lower threshold, then the WDG*n*L interrupt for the low threshold violation transitions to 1. If the conversion result is greater than the lower threshold (and therefore also greater than the higher threshold), then the interrupt WDG*n*H for high threshold violation is generated. Thus you should take care to avoid that situation, as it could lead to misinterpretation of the watchdog interrupts.

60.3.11 DMA functionality

Conversion result data from any channel can be transferred from a register to system memory via Direct Memory Access (DMA). DMA transfers are enabled by writing 1 to DMAE[DMAEN]. After being enabled, the on-chip DMA controller can receive a DMA request after the conversion of each channel by writing 1 to the respective masking field in:

- DMA Request Enable For Precision Inputs (DMAR0)
- DMA Request Enable For Standard Inputs (DMAR1)
- DMA Request Enable For External Inputs (DMAR2)

DMA masking registers must be programmed before starting any conversion.

A DMA request to the DMA controller can be cleared at different times in two modes:

- Mode-1: Clearing of DMA request on acknowledgment from DMA controller (DMAE[DCLR] = 0)
- Mode-2: Clearing of DMA request on read to data registers (DMAE[DCLR] = 1)

The figures below show the operation of DMA in two modes (cycle counts are typical values).

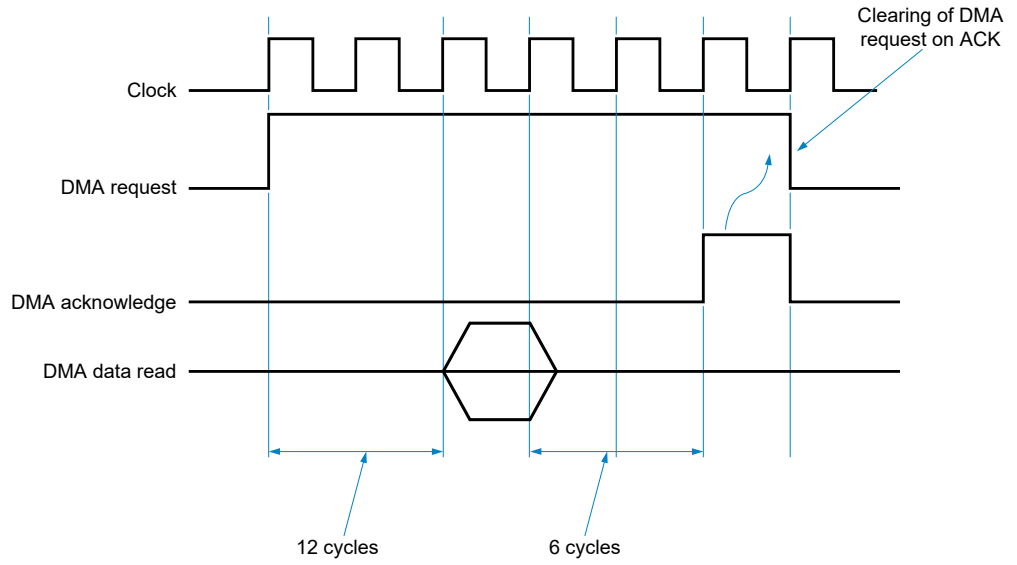


Figure 255. DMA operation Mode-1 (DMAE[DCLR] = 0)

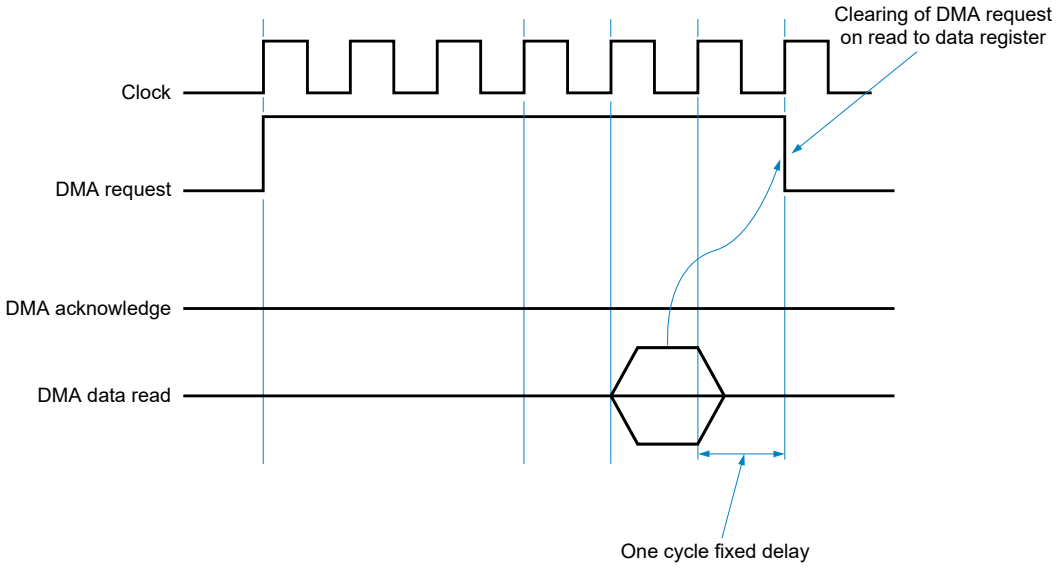


Figure 256. DMA operation Mode-2 (DMAE[DCLR] = 1)

60.3.12 Interrupts

ADC generates the following maskable interrupts:

- EOC (End of Conversion)
- ECH (End of Chain)
- JEOC (End of Injected Conversion)
- JECH (End of Injected Chain)
- EOBCTU (End of BCTU-triggered Conversion)
- LAWIFx and HDWIFx (Watchdog threshold)

- Self-testing interrupts

Interrupts are generated during the conversion process to signal events such as End Of Conversion (EOC), and so on, as described in Interrupt Status (ISR). ISR and another register, Interrupt Mask (IMR), are provided to check and enable the interrupt request to an external interrupt controller.

Interrupts can be individually enabled on a per-channel basis by programming EOC Interrupt Enable For Precision Inputs (CIMR0).

CEOCFR*n* are used for pending End of Conversion interrupts, with one field per channel to indicate completed conversions on a per-channel basis.

Interrupts generated due to the analog watchdogs are managed by two 32-bit registers, Analog Watchdog Threshold Interrupt Status (WTISR) and Analog Watchdog Threshold Interrupt Enable (WTIMR), to check and enable the interrupt request. A watchdog interrupt causes two corresponding fields to transition to 1 for each channel monitored. One field is in WTISR[WDG*n*H] and another in WTISR[WDG*n*L].

Generated events are logged into status registers and used to generate interrupts if enabled in corresponding Mask registers.

There are three different interrupt outputs. The interrupt structure is shown in Figure 257.

Each interrupt is generated by combining a group of events.

The following interrupts are combined (logic OR) into one interrupt output.

- EOC
- ECH
- EOBCTU
- JEOC
- JECH
- End of self-test algorithm (WDG_EOA_S and WDG_EOA_C)

Interrupts of all HDWIFx and LAWIFx are combined on the second interrupt output.

Interrupts of all self-test errors (watchdog threshold, sequence, and timer violations) in Self-Test Status 1 (STSR1) are combined on the third interrupt output. The errors are:

- WDSERR
- WDTERR
- ERR_S0
- ERR_S1
- ERR_S2
- ERR_C

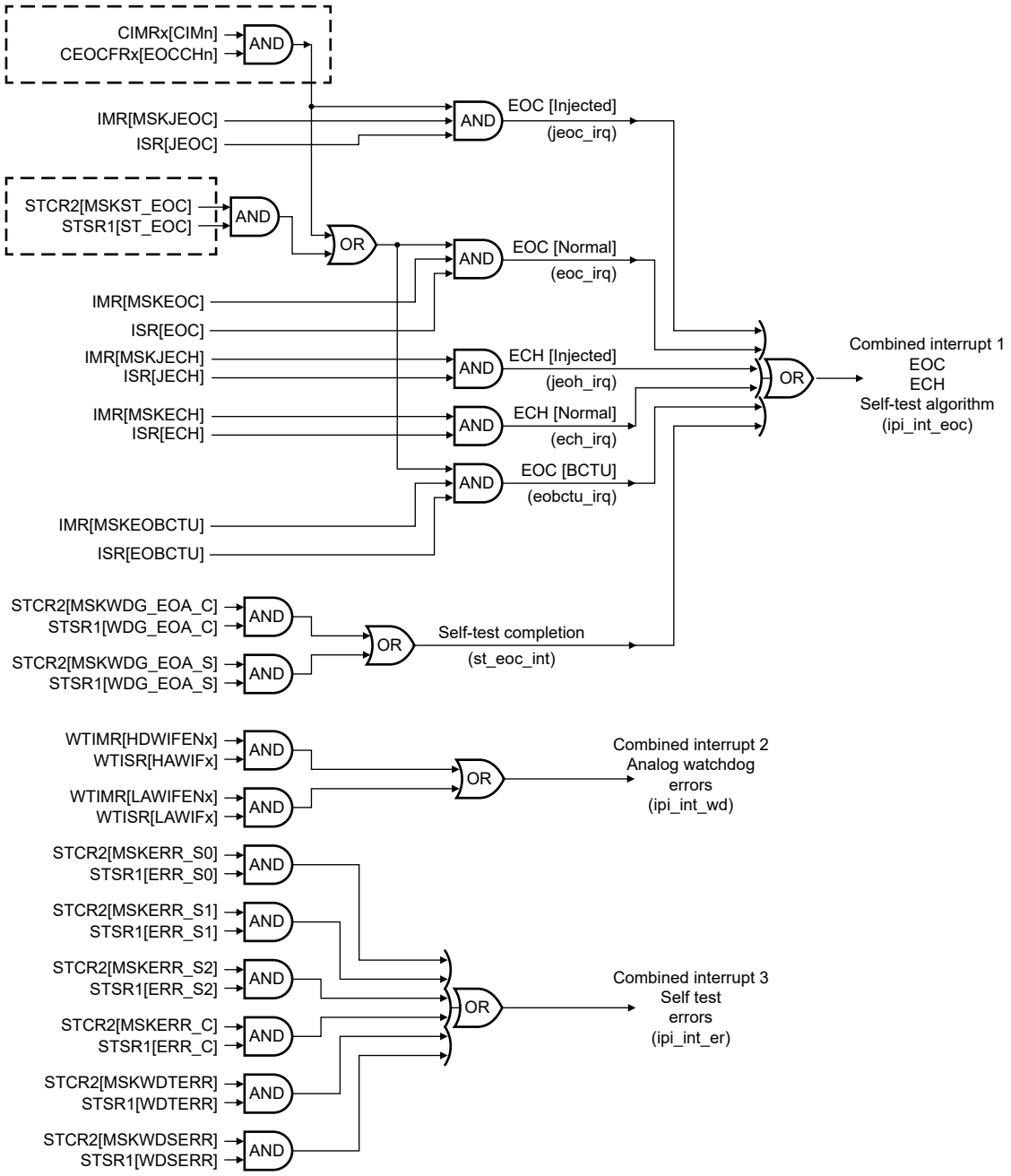


Figure 257. Interrupt structure of ADC

Interrupt Status (ISR) Indicates the interrupt pending request status. Write 1 to the corresponding status field to clear the pending interrupt flag (at this write operation all the other fields in Interrupt Status (ISR) must be maintained at 0).

ISR checks for three types of interrupts.

The first type, Interrupt 1, relates to the following:

- End of normal or injected conversion (EOC or JEOC)
- End of normal or injected conversion chain (ECH or JECH)
- End of algorithm C or algorithm S self-test (WD_EOA_C or WD_EOA_S)

Perform the following actions for this type of interrupt:

1. Read Interrupt Status (ISR) to determine whether the event is an EOC, JEOC, ECH, JECH, or EOBCTU event.
2. Read Self-Test Status 1 (STSR1) to identify the end of self-test algorithm events (WDG_EOA_S or WDG_EOA_C), if present.
3. If the interrupt is an EOC or JEOC event:
   - Clear the respective channel pending field in the appropriate CEOCFR*n* register for an EOC by writing a 1 to it.
   - Clear ISR[EOC] or ISRJEOC] by writing a 1 to it.
4. If the interrupt is an ECH or JECH, clear ISR[ECH] or ISR[JECH] by writing 1 to it.
5. If the interrupt is EOBCTU, clear ISR[EOBCTU] by writing 1 to it.
6. If the interrupt is for self-test end of algorithm, clear STSR1[WDG_EOA_S] or STSR1[WDG_EOA_C] by writing 1 to it.

The second type, Interrupt 2, relates to analog watchdog errors. Perform the following actions for this type of interrupt:

1. Read Analog Watchdog Threshold Interrupt Status (WTISR) to determine which watchdog triggered the interrupt.
2. Read the AWORRx register(s) to determine which channel's data caused the interrupt (if present).
3. Clear the respective field of the WTISR[WDG*n*H/WDG*n*L] register, where *n* = watchdog number, by writing 1 to it.
4. Clear the respective field of the AWORR*n*[AWOR_CH*n*] register, where x = register, n = channel, by writing 1 to it.

The third type, Interrupt 3, relates to self-test errors. Perform the following actions for this type of interrupt:

1. Read Self-Test Status 1 (STSR1) to determine the type of error encountered in self-test (WDSERR, WDTERR, ERR_S0, ERR_S1, ERR_S2, ERR_C, and so on) if present.
2. Clear the respective field in Self-Test Status 1 (STSR1) by writing 1 to it.

The connectivity of interrupt lines at the chip level (external to ADC) determines whether software must read all the registers for all three interrupts or any one or combination of any two.

60.3.13 External decode signals delay

The ADC provides three external decode signals which it uses to select one channel out of eight in the external analog multiplexers. There can be a maximum of four such multiplexers used to connect the 32 external channels. The ADC automatically sets the decode signals to control these external analog multiplexers, based on the current channel selected for conversion. Delay Start Of Data Conversion (DSDR) is provided to take the switching time of the external analog multiplexer into account. It enables you to program the delay between the decoding signal selection and the actual start of conversion.

If presampling is enabled, the DSD delay starts after the presample phase (presampling time is the same duration as sample time).

60.3.14 Power Down state

The analog part, along with the ADC controller, can be put in Power Down state by writing 1 to MCR[PWDN]. This shuts down ADC and stops the clock to the ADC controller. After release of reset, MCR[PWDN] remains 1, so the ADC analog module is kept in Power Down state by default. You must exit this state before starting any operation, by writing 0 to MCR[PWDN]. In Power Down state, no conversion can be started. If a BCTU trigger pulse is received during this state, it is discarded. You must ensure that a BCTU trigger is not initiated to ADC before writing 0 to MCR[PWDN], as it may cause the BCTU module to be stuck in a wait state.

You can set MCR[PWDN] by writing 1 to it at any time. If a conversion is ongoing, ADC does not move into Power Down state immediately. ADC enters Power Down state only after completion of the ongoing conversion. During scan operation, the ongoing operation should be aborted manually by writing 0 to MCR[NSTART] before or after writing 1 to MCR[PWDN].

ADC Power Down state status is indicated by MSR[ADCSTATUS] when ADC enters into Power Down state.

If the BCTU is enabled and MSR[BCTUSTART] = 1, then you cannot write 1 to PWDN. When BCTU Trigger mode is enabled, the application must wait for the end of conversion (MSR[BCTUSTART] is automatically reset to 0). When BCTU Control mode is enabled, before entering Power Down state the application must write 0 to BCTUEN also.

If Power Down state is entered by writing 1 to MCR[PWDN], the process running before entry into Power Down state must be restarted manually (by writing 1 to the appropriate START field in Main Configuration (MCR)) after exiting Power Down state.

NOTE

Writing 0 to MCR[PWDN] and writing 1 to MCR[NSTART] or MCR[JSTART] during the same cycle is not supported.

60.3.15 Low Power mode support

For low-power operation, ADC must be in Idle state and then put into Power Down state to ensure proper signal condition at the analog boundary.

From any of the states listed below, follow the sequence of steps shown to put ADC into Power Down state.

1. Idle condition
    a. Verify that ADC is in Idle state (MSR[ADCSTATUS] = 000b).
    b. Write 1 to MCR[PWDN], to shut off the clock to the hard macro with the proper state of signals required for Low Power mode.
    c. Verify that ADC is in Power Down state (MSR[ADCSTATUS] = 001b).
2. During calibration

    NOTE

    Writing 1 to MCR[PWDN] during calibration is prohibited. After calibration is started it should be allowed to finish normally. Alternatively, it can be terminated by starting a normal conversion before writing 1 to MCR[PWDN].

    a. Check the current ADC state either via MSR[ADCSTATUS] (000b = Idle, 011b = Calibration), or via CALBISTREG[C_T_BUSY] (1 = calibration is in progress).
    b. When ADC is in Idle state, write 1 to MCR[PWDN] to shut off the clock to the hard macro with the proper state of signals required for Low Power mode.
    c. Verify that ADC is in Power Down state (MSR[ADCSTATUS] = 001b).
3. During conversion (normal [one-shot]/Injected/BCTU):
    a. Write 1 to MCR[PWDN].
    b. ADC enters Power Down mode after completing the current conversion chain (for normal and injected conversions) or the conversion for conversion triggered by BCTU.
    c. Check the ADC state via MSR[ADCSTATUS] (000b = Idle state, 001b = Power Down state).
4. During Scan mode operation of normal conversion

    NOTE

    Writing 1 to MCR[PWDN] during scan mode operation has no effect other than to prevent software from starting a new normal or injected conversion. It does not stop a conversion in progress and cannot put ADC into Power Down state.

    a. Software must write 0 to MCR[NSTART] to stop the ongoing scan conversion before (or after) writing 1 to MCR[PWDN]. This stops conversion at the current chain boundary.
    b. ADC enters Power Down mode after coming to an Idle state after MCR[PWDN] is written with 1. This shuts the clock off to the hard macro and puts analog inputs in the proper state required for Low Power mode.
    c. Verify that ADC is in Power Down state (MSR[ADCSTATUS] = 001b).

60.3.16 Auto Clock Off mode

To reduce power consumption during Idle state (without going into Power Down state), an auto-clock-off feature can be enabled by writing 1 to MCR[ACKO]. When enabled, the internal ADC operating clock (AD_Clk) is automatically switched off during an idle period (for example, no conversion is programmed).

60.3.17 Calibration and self-test

60.3.17.1 Calibration

ADC must be calibrated before it can do meaningful conversions. The application must prevent any start of conversion before the ongoing calibration has finished successfully. No conversion trigger may be received during calibration.

During calibration, a known reference voltage is sampled and converted under controlled conditions to determine the correction values (calibration values) for offset, gain, and capacitor mismatch.

These calibration values (except gain calibration) are used in a post-processing step after conversion, reducing or eliminating the various error contribution effects. The gain calibration is used during sample phase to define the additional charge to be loaded to compensate for gain failure.

You must run calibration after every power-up reset (check device-specific reset connectivity for appropriate reset applied). You must also run calibration if it is indicated by the self-test (see Self-test).

Configuration and start of calibration is done by programming the Control And Calibration Status (CALBISTREG). During calibration, internal hardware averaging function should be enabled. Maximum averaging is recommended, and 16 should be the minimum.

Calibration status is captured by a status flag (CALBISTREG[TEST_FAIL]) to indicate any fault detected during calibration execution or if calibration was prematurely aborted. Calibration is aborted if any normal conversion is initiated during the calibration execution, which is called premature termination. (Flag TEST_FAIL = 1 indicates a calibration fail, for example when a calibration value is out of internally defined limits.)

To execute the calibration algorithm, perform the following steps:

1. Select the conversion clock frequency to be within the allowed limits for the calibration.
2. Bring ADC from Power Down state to active conversion (program MCR[PWDN] = 0b).
3. Configure the Control And Calibration Status (CALBISTREG). The default values are set for maximum accuracy (recommended).
4. Disable the error correction for the smaller capacitances in the CDAC of the ADC (write 0h to CAL2[ENX]).
5. Start calibration (program CALBISTREG[TEST_EN] = 1b), and calibration starts immediately.
6. Poll the status of CALBISTREG[C_T_BUSY] for 0 (wait until it transitions to 0).
7. Check CALBISTREG[TEST_FAIL] to determine the final status. If 1, then calibration failed.
8. Check the status of MSR[CALIBRTD]. If calibration was successful this field is 1.

60.3.17.2 Self-test

For safety applications, it is important to verify correct operation at regular intervals. ADC has a self-test feature for this purpose. When self-test is enabled, ADC automatically checks its components and flags errors, if any are found.

Tests can be enabled to check the reference (VrefH) and calibrated values.

The following test algorithms have been implemented:

- Supply self-test (algorithm S): It includes the conversion of the band gap and VREF voltages. It includes a sequence of num_supply_steps test conversions (steps). The supply test conversions must be an atomic operation (that is, all supply algorithm conversions must be performed one after another with no functional conversions in between).

- Capacitive self-test (algorithm C): It includes a sequence of test steps per definition of algorithm C (see Table 349) which executes the capacitive matrix of the CDAC used for sampling and conversion.

Individual steps can take up to 1 µs at 80 MHz ADC clock frequency.

Table 349. Self-test steps

STCR3[ALG]	STCR3 [step](per algorithm definition)	Description	Outcome	Comment
00 (algorithm S)	0	Supply self-test (band gap voltage)	Measures the internal band gap voltage	
	1	Supply self-test (reference voltage high)	Measures ADC high reference voltage	
	2	Supply self-test (reference voltage high)	Measures ADC high reference voltage	
01 (Reserved)	Reserved			
10 (algorithm C)	0 – 11	Capacitive self-test. One of the calibration steps is being run.	The difference/error of the individual offset value from the previously calibrated value is being returned as an ADC result that is compared with the programmed value in the self-test analog watchdog register(s) to detect any fault.	The ideal result value is expected to be 0. If the returned value is higher, it indicates a runtime fault or accuracy shift. The error is indicated in the status register and can be used to generate an interrupt also. It that indicates ADC should be recalibrated to check whether the reported error can be managed by calibration or that permanent damage occurred to ADC.
11 (algorithm S + C)	0 – 2 (S), 0 – 11 (C)	Supply for One-Shot mode and (supply + capacitive) for Scan mode only.	Same as above.	

ADC:

- Schedules self-testing algorithms using configuration registers.
- Monitors the converted data using analog watchdog registers.
- Flags any errors to a fault control unit (FCCU), if one exists.

Self-test steps can be activated from software (CPU) or BCTU.

ADC mode	TEST algorithm (CPU)	TEST algorithm/step (BCTU)
CPU mode (MCR[BCTUEN] = 0)	Yes	No

Table continues on the next page...

Table continued from the previous page...

ADC mode	TEST algorithm (CPU)	TEST algorithm/step (BCTU)
	• One-Shot operation mode • Scan operation mode	
BCTU mode	No	Yes

60.3.17.2.1 CPU mode

In this mode, self-test conversion is similar to normal conversion. You enable self-test by writing 1 to STCR2[EN].

Self-test conversions execute along with the functional conversions. The sequencing of steps of the selected algorithm depends on the operating mode of normal conversions (selected by MCR[MODE]).

In One-Shot operation mode, if self-test is enabled, only one step of the selected self-testing algorithm is executed at the end of the chain. The step number and algorithm to be executed are programmed in Self-Test Status 3 (STSR3). In One-Shot operation mode the sequence is:

1. Program NCMR*n* to select channels to be converted for normal conversion.
2. Write 0 to MCR[MODE] to select one-shot operation.
3. Configure the self-test algorithm threshold values (see Table 350).
4. Program sampling duration values in STCR1[INPSAMP*m*] (*m* = S, C).
5. Select the self-testing algorithm (in STCR3[ALG]), and step (in STCR3[MSTEP]). The default is algorithm S, step 0.
6. Enable self-testing by writing 1 to STCR2[EN].
7. Start the normal conversion by writing 1 to MCR[NSTART].
8. All normal conversions are executed as usual.
9. At the end of all normal conversions in the chain, the step number (programmed in STCR3[MSTEP]) of the self-testing algorithm (selected by STCR3[ALG]), is executed similar to a normal functional channel.
10. At the end of the conversion for the self-test channel:
    - The result is written to STDR1[TCDATA].
    - STDR1[VALID] transitions to 1.
    - ISR[EOC], ISR[ECH], and STSR1[ST_EOC] transition to 1.
11. ADC returns to Idle state.

For example: Channels A→B→C→D→E→F→G→H are present in the device where channels B→D→E are to be converted in One-Shot operation mode. At the end of conversion for channels B→D→E, self-test conversion is performed and ISR[ECH] and ISR[EOC] transition to 1. The sequence is B→D→E→self-test step.

MSR[NSTART] transitions to 1 when the normal conversion starts and transitions to 0 at the end of conversion for the test channel.

In Scan operation mode, consecutive steps of the selected self-test algorithm are converted continuously at the end of each chain of normal conversions. The number of channels converted at the end of each chain is 1 (except for algorithm S, in which all the steps are performed sequentially without any functional conversion interleaved). So, in Scan operation mode the sequence is:

1. Program NCMR*n* to select the channels to be converted for normal conversion.
2. Write 1 to MCR[MODE] to select Scan operation mode.
3. Configure the self-test algorithm threshold values (see Table 350).
4. Program sampling duration values in STCR1[INPSAMP*m*] field (*m* = S, C).

5. Select the self-testing algorithm in STCR3[ALG]. By default, all algorithms (supply and capacitive) are selected, that is, all algorithms are executed step-by-step, one after the other.
6. Enable self-testing by writing 1 to STCR2[EN].
7. Start the normal conversion by writing 1 to MCR[NSTART].
8. All normal conversions are executed as usual.
9. At the end of the chain of normal conversions (assuming the default value of STCR3[ALG]), all steps of algorithm S are performed (as algorithm S is always atomic). MSR[SELF_TEST_S] transitions to 1.
10. At the end of the conversion of the self-test channel for the last step of algorithm S, the result is written to STDR1[TCDATA] and STDR1[VALID] transitions to 1. At the same time, MSR[SELF_TEST_S] transitions to 0. The following registers also transition to 1:
    - ISR[EOC]
    - ISR[ECH]
    - STSR1[ST_EOC]
11. The next chain of normal conversions then starts.
12. At end of the chain of normal conversions, ADC executes step 0 of algorithm C.
13. At the end of the conversion of the test channel for step 0 of algorithm C, the result is stored in STDR1[TCDATA] and STDR1[VALID] transitions to 1 (if STSR1[OVERWR] is 1). Also, ISR[EOC] and ISR[ECH] transition to 1. STSR1[ST_EOC] also transitions to 1.
14. The next chain of normal conversion then starts.
15. At the end of the normal conversion chain, ADC executes step1 of algorithm C.

This process continues for all steps of all three algorithms. The state machine returns to Idle state when MCR[NSTART] transitions to 0.

For example: channels A→B→C→D→E→F→G→H are present in the device where channels B→D→E are to be converted in Scan operation mode. At the end of every conversion chain for channels B→D→E, one step of the self-test conversion is performed and ISR[ECH] and ISR[EOC] transition to 1.

The sequence is:

B→D→E→ST-S0→ST-S1→ST-S2→B→D→E→ST-C0→B→D→E→ST-C1→B→D→E→ST-C2→ . . . B→D→E→ST-C11→B→D→E→ST-S0 . . .

ST-Sx - Self-test supply step x (x = 0, 1, 2)

ST-Cx - Self-test capacitive step x (x = 0 to 11)

MSR[NSTART] transitions to 1 when normal conversion starts.

NOTE

- If instead of starting normal conversion by software (by writing 1 to MCR[NSTART]), it is started by an external trigger, the self-test behavior remains the same.
- Self-test channel conversion is not performed for injected conversions. It is performed only during normal conversions.
- If, during a test channel conversion, an injected conversion arrives, the test conversion is aborted (just as a normal functional channel) and the injected conversion is performed. After the injected conversion is complete, the test conversion resumes from the step at which it was aborted. In this case, MSR[SELF_TEST_S] remains 1 during the injected conversion.
- For self-test, MCR[MODE] should be programmed (at least one cycle) before writing 1 MCR[NSTART] and should not be changed until the conversion is finished or terminated.

Table 350. Recommended self-test threshold values

Register field	Hex value	Decimal value	Comment
STAW0R[THRH]	3669h	13929	The threshold values depend on the stability of the supply and the reference voltage. When you see voltage variations, widen the limits.
STAW0R[THRL]	5 V reference: 1A7Fh 3 V reference: 2B27h	5 V reference: 6784 3 V reference: 11047	
STAW1R[THRL]	3FF9h	16377	
STAW2R[THRL]	3FF9h	16377	
STAW4R[THRH]	20h	32	
STAW4R[THRL]	7FE0h	–32	
STAW5R[THRH]	20h	32	
STAW5R[THRL]	7FE0h	–32	

60.3.17.2.2 BCTU mode

BCTU mode is enabled by writing 1 to MCR[BCTUEN].

Self-test conversion behaves in the same way for all BCTU operating modes (Trigger mode and Control mode). In both cases, if MCR[BCTUEN] = 1, the self-test conversion can be started only by BCTU—software cannot start it. MCR[BCTUEN] must not be changed during normal conversion. The interface between BCTU and ADC (for self-test) is shown in the following figure.

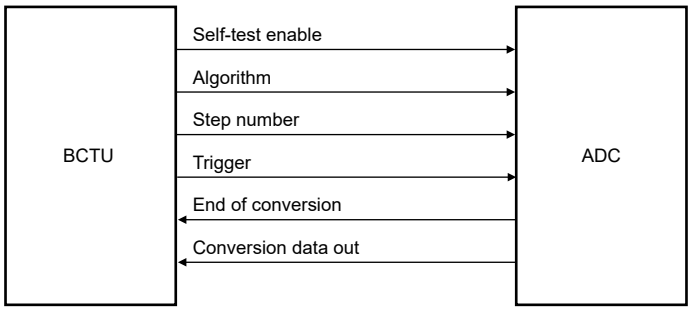


Figure 258. Self-test interface between ADC and BCTU

For self-test conversions in BCTU mode, BCTU asserts self-test enable along with the Trigger command. The algorithm and the step number to be executed is put on algorithm and step number, respectively. The other commands (Trigger, End Of Conversion, and Conversion Data Out) have the same meaning, like normal BCTU functional conversion.

As already mentioned for algorithm S, the three steps must be atomic. In BCTU mode, this is managed by BCTU, that is, BCTU sends three triggers (one for each step) asserting self-test enable and updating test_step for each step.

60.3.17.2.3 Abort and abort chain for self-testing channel

Writing 1 to MCR[ABORT] during self-test channel conversion has no effect.

In One-Shot operation mode, if you write 1 to MCR[ABORTCHAIN] during the self-test channel conversion, the self-test channel is aborted and ISR[ECH] transitions to 1. In this case, ISR[EOC] for the self-test channel is not generated.

In Scan operation mode, if MCR[ABORTCHAIN] = 1 when self-test channel step *n* is ongoing, self-test channel step *n* is aborted and the next chain conversion begins. At the end of the chain, the step *n* conversion is performed again. (For algorithm S, the full algorithm is executed again).

60.3.17.2.4 Self-test analog watchdog

ADC provides monitoring options for conversion data generated by self-test algorithms. Analog watchdogs determine whether the conversion results for self-test algorithms are in the range of a particular guard area. ADC provides separate self-test analog watchdog registers for each algorithm.

After the conversion of each step of a self-test algorithm, ADC compares the converted value and the threshold values if the analog watchdog feature has been enabled by writing 1 to STAW*n*R[AWDE] (for example, STAW0R[AWDE]). If the converted value is not between the upper and lower threshold values specified by the Self-Test Analog Watchdog Register for the particular algorithm, the corresponding error flag, STSR1[ERR_x], transitions to 1 and the step number in which error occurred is captured in STSR1[STEP_x] (for algorithm C). Erroneous data is written to STSR4[DATA*n*]. The STSR1[ERR_x] flags generate an interrupt if enabled by the corresponding mask bit in Self-Test Configuration 2 (STCR2). The fault indication is also forwarded to the fault control unit, if present, so that necessary action can be taken at the chip level. The fault type (critical or noncritical) is specified by the configuration in Self-Test Configuration 2 (STCR2).

The analog watchdog feature works differently for algorithm S. Algorithm S is always an atomic operation. Self-Test Status 1 (STSR1) has a separate error field for each step of algorithm S to avoid overwrite in case an error occurs in more than one step. Therefore, there are separate mask bits for each step in STCR1. For the same reason, the status registers (Self-Test Status 2 (STSR2) and Self-Test Status 3 (STSR3)) have separate fields for each step to store erroneous data.

In step 0 of the supply algorithm, ADC measures the band gap voltage (1.2 V), which is assumed to be stable. The conversion result of step 0 is compared against high (THRH) and low (THRL) thresholds as defined in Self-Test Analog Watchdog S0 (STAW0R), if enabled (STAW0R[AWDE] = 1). The STSR1[ERR_S0] flag transitions to 1 if any of the thresholds are violated. STSR1[ERR_S0] is cleared by writing 1 to it. The conversion result of ADC can be calculated with the following formula:

- ADC data (decimal) = (Vin × 2^{15} ÷ (Vrh - Vrl))

Where:

- Vin: input voltage (band gap voltage in this case)
- Vrh: ADC reference high voltage
- Vrl: ADC reference low voltage

The upper and lower threshold value can be calculated with the above formula using maximum and minimum values of Vin, Vrh, Vrl (Vin is the band gap supply voltage in this case):

- Upper threshold (THRH, decimal) = (Vin[max] × 2^{15}) ÷ (Vrh[min] - Vrl[max])
- Lower threshold (THRL, decimal) = (Vin[min] × 2^{15}) ÷ (Vrh[max] - Vrl[min])

The minimum and maximum voltages can be obtained from the chip data sheet. The band gap voltage tolerance, as well as the actual reference voltage used and its tolerances, must be considered when calculating high and low thresholds.

For algorithm S steps 1 and 2, (VREF/VREF) is measured in order to check the integrity of the sampling signal. For these particular conversions, no higher threshold value is required as the ideal value is FFFh. Only the lower threshold value is programmed in Self-Test Analog Watchdog S1 (STAW1R) and Self-Test Analog Watchdog S2 (STAW2R).

For algorithm C, a separate register is provided for step 0. In step 0 an offset for other steps is measured. The converted data is compared with the threshold values in Self-Test Analog Watchdog C (STAW5R) if STAW4R[AWDE] = 1. For other steps, this offset is subtracted from converted data before performing watchdog checks.

60.3.17.2.5 Watchdog timer

The watchdog timer is an additional check that monitors the sequence of the self-testing algorithm that has been implemented, and also checks whether the algorithm is completed within a safe time period. The watchdog timers are enabled for CPU as well as BCTU conversions. Each algorithm has an independent watchdog timer. The watchdog timer for a particular algorithm is enabled by writing 1 to STAWxR[WDTE]. The safe time value is programmed in STBRR[WDT] (the default value is 10 ms, assuming an 80 MHz clock).

The safe time is measured starting from step 0 of the algorithm (including all normal chain conversions in between) to the point where step 0 of the same algorithm starts again.

The programming sequence is:

1. Program NCMR*n* to select the channels for normal conversion in Scan operation mode (MCR[MODE] = 1).
2. Select the self-test algorithm in STCR3[ALG]. By default, all algorithms (supply and capacitive) are selected (all algorithms are executed step-by-step, one after the other).
3. Enable the self-test channel by writing 1 to STCR2[EN].
4. Program the safe period value in STBRR[WDT].
5. Enable the watchdog timer by writing 1 to STAWxR[WDTE]. Assume writing 1 to STAWxR[WDTE] occurs at time *t*0. It is important to do all configuration programming before writing 1 to STAWxR[WDTE], because the safe time check is also performed between writing 1 to STAWxR[WDTE] and the start of step 0. This is to verify that the algorithm has started within the safe time.
6. Start the normal conversion by writing 1 to MCR[NSTART].
7. At the end of the first conversion chain, three steps (steps 0, 1, and 2) of algorithm S are executed in sequence. Assume the start of Step 0 occurs at time *t*1.
8. After completion of algorithm S, ADC performs conversion of the next chain.
9. At the end of the conversion chain, the first step (step 0) of algorithm C is performed.
10. In this way, one step of algorithm C is performed after completion of one conversion chain until all steps are finished. Steps are executed in sequence (steps 0–11)
11. After the last step of algorithm C, another chain conversion is executed. At the end of this chain conversion, step 0 of algorithm S is started again, repeating the whole sequence. Assume this time (starting of step 0 of the supply algorithm) to be at time *t*2.
12. For algorithm S, if (*t*1 - *t*0) > safe period or (*t*2 - *t*1) > safe period, the watchdog timer flags an error and STSR1[WDTERR] transitions to 1. ADC asserts a fault signal per the criticality configuration (critical or noncritical) and an interrupt is generated, if enabled (STCR2[MSKWDTERR] = 1). Otherwise, the watchdog timer counter is reset and starts again to monitor the next sequence.
13. A similar sequence is followed for watchdog timers for algorithm C.

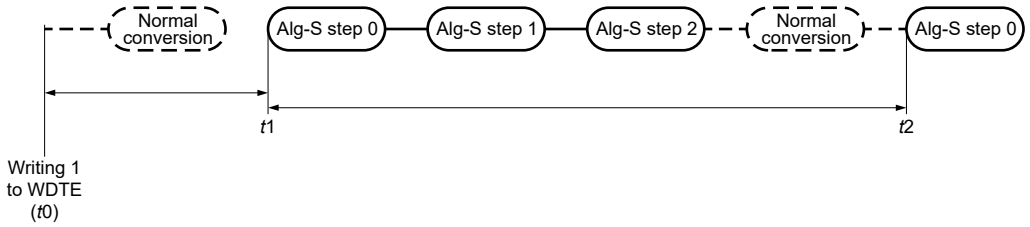


Figure 259. Watchdog timer monitor for algorithm S

NOTE

1. Because BCTU may not incorporate any safe period checking mechanism, the watchdog timers can also be enabled for BCTU conversions.
2. You must disable watchdog timers for self-test algorithms that are not executed.

60.3.17.2.5.1 Watchdog sequence checking

The self-test watchdog timer incorporates sequence checking features to verify that the steps of each algorithm are executed in the correct order. If the steps are not in the correct order, STSR1[WDSERR] transitions to 1 to indicate the error. A fault is indicated by an interrupt flag if STCR2[MSKWDSERR] = 1. A fault is also asserted as per criticality configuration (critical or noncritical) and an interrupt generated if STCR2[MSKWDSERR] = 1.

A watchdog sequence error is flagged in the following cases:

- The steps of any algorithm are not executed in proper order.
- The step numbers provided by BCTU for a BCTU conversion are not in order. Watchdog sequence checking is significant only for BCTU burst mode.
- When an abort chain occurs during a self-test channel conversion, that step must be repeated at the end of the next chain. This generates a sequence error as soon as the self-test channel conversion resumes. There is an exception, however. If an abort chain occurs during the last step of algorithm S, the sequence error is not flagged because the whole algorithm must be repeated.

If an injected conversion occurs during self-test channel conversion, a watchdog sequence error is not flagged, although the ongoing step number is aborted and is repeated.

NOTE

The watchdog timer feature is applicable for Scan operation mode but not for One-Shot operation mode.

60.3.17.2.6 Baud rate control for test channel

The baud rate control feature controls the scheduling of self-test channel conversions between normal conversion chains. The scheduling rate is specified by STBRR[BR].

By default, if the self-test channel is enabled, one step of the selected algorithm is executed after each normal conversion chain. The bandwidth consumed by the self-test channel depends on the number of channels in the normal chain. For example, if you have 50 normal conversions in a chain, then the self-test channel consumes only 2% of the total bandwidth, which is very small. If the number decreases to just 3 channels, the bandwidth consumed by the self-test channel is 25%, which is significant (and may not be desirable because it slows down the normal conversion rate).

STBRR[BR] provides flexibility by scheduling the self-test channel conversion to be performed not at the end of every chain, but at the end of BR + 1 chains. For example, if STBRR[BR] = 5, a single step of the selected algorithm for the test channel is performed after 6 chain conversions. The next step is performed at the end of the next six chain conversions, and so on. By default, STBRR[BR] = 0.

NOTE

This feature is applicable only for Scan operation mode and not for One-Shot operation mode. STBRR[BR] should be written with 0 for One-Shot operation mode.

To use the baud rate control feature in Scan operation mode, NCMR*n* should have a nonzero value.

60.3.17.2.6.1 Abort chain when baud rate is nonzero

As already described, for a nonzero value of STBRR[BR], the self-test channel conversion is performed at the end of BR + 1 chains. If an abort chain occurs during the chain in which a self-test channel conversion is scheduled to be performed, the self-test channel conversion is performed after the next BR + 1 chains.

For example, if STBRR[BR] = 2, the sequence is two normal chains (without any self-test channel conversion), followed by a chain with the self-test channel converted at the end. If an abort chain occurs during the first two chains, it is treated as a normal chain abort and the self-test channel is converted at the end of the third chain only (as in the case without any abort chain). However, if an abort chain occurs during the third chain in which the self-test channel is scheduled to be converted, the self-test channel is converted after the next three chains (for example, at the end of the sixth chain, counting from the beginning).

60.3.18 Conversion time

Total conversion time depends on the conversion clock frequency, which is configured by programming MCR[ADCLKSEL].

The components of conversion time and affecting configurations are:

- PST
- ST
- CT

- DP
- TPT

Presample phase time is equal to the sample time with a one AD_clk cycle delay for phase transition from presample phase to sample phase. CTR*n*[INPSAMP] controls sample time for different types of channels and is specified in terms of AD_clk cycles, so the presampling time is sample time plus one cycle of AD_clk. The minimum value of sample time is 8. If the value programmed is less than 8, it has no effect on sample time and the sample time is 8 AD_clk cycles.

Compare phase time is controlled by:

- The evaluation time of a single bit
- The number of bits converted

For *n*-bit conversions, the value of CT is *n* multiplied by the evaluation time of a single bit in terms of AD_clk cycles.

The evaluation time of a single bit is 4 AD_clk cycles if AMSIO[HSEN] = 0h and AMSIO[CMPCTRL0] = 0b. The evaluation time of a single bit is 6 AD_clk cycles if AMSIO[HSEN] = 1h and AMSIO[CMPCTRL0] = 1b.

The data processing time is 2 cycles of AD_clk. During these cycles raw converted data is corrected for offset, gain, capacitor mismatch, and so on.

Trigger processing time consists of:

- One module_clock cycle, which is required to prepare the channel and calculate the initial gain value used by ADC for the first conversion.
- BCTU triggering time:
  - Triggers from the synchronous BCTU interface require 1 cycle of the module clock for processing. One more cycle is required to register the BCTU trigger in synchronous mode.

The total conversion time, in terms of the module clock cycles, is calculated using the following equation for normal and injected conversions:

Total_conversion_time = ([(PST + ST + CT + DP) × chain_length] + TPT) × TAD_clk

Example:

The ADC controller clock is equal to the module clock (80 MHz, clock cycle = 12.5 ns)

- ADC resolution 12 bit + 1 bit for a special capacitor (CS)
- Three channels are programmed in NCMR*n*, so a chain of three is to be converted
- Default sample time (22 cycles) is specified
- No presampling
- Conversion time (4 cycles per bit)

The total time for the three conversions = [(0 + 22 + (4×13) + 2) × 3] + 1 = 229 cycles ~= 2.862 µs

Conversion timing in Calibration for full test:

- Single conversion time = [sample time (CALBISTREG[TSAMP]) + compare time (11 × 4) + data proc(2)]
- Inter-conversion gap = 1 cycle
- Total number of tests = 12
- Averaging = 32

One test duration = (single conversion time + inter-conversion gap) × averaging samples

Total test duration = ((one test duration) × total number of tests).

Example: sample time = 22

Total test time = ((((22 + (11×4) +2)+1) × 32) × 12) = 26496 cycles × 12.5 ns (80 MHz) = 331200 ns

Conversion time in self-test:

Self-test conversion time equals one test period multiplied by the number of consecutive (atomic) steps. For supply self-test, the atomic step is 3 and for other tests it is 1.

One test period for self-test is as below.

- For supply tests: ([sample_time + (13 × 4) + 2])
- For other algorithms: ([sample_time + (11 × 4) + 2])

NOTE

Algorithm S steps (conversions) always run in atomic operation, required for an algorithmic procedure. In Scan operation mode, when all algorithms are selected, three steps of algorithm S run together after the end of the conversion chain. algorithm C steps are not required to be run together, so after each conversion chain one step is executed serially. Conversion time with self-test must also be calculated accordingly.

Example.

- Channels in chain A→B→C.
- Scan operation mode.
- All self-test algorithms are selected.

So, the resulting conversion order is:

[A→B→C→AS_step0→AS_step1→AS_step2→A→B→C→AC_step0→A→B→C→AC_step1→A→B→C→AC_step2→.....→A→B→C→AC_step11]→[same sequence repeats]

NOTE

- AS = Algorithm S (supply self-test)
- AC = Algorithm C (capacitive self-test)

60.3.19 Conversion data processing

Raw converted data contains many types of errors, such as offset, gain, and so on. ADC processes raw conversion data before writing the result to a data register, to reduce or eliminate error contributions. The process of error correction happens in parallel with bit-by-bit evaluation, using the correction values generated during the offset determination and calibration process.

The error correction value is subtracted from the raw conversion result whenever the comparator output for one of the calibrated capacitors evaluates high during the conversion period.

To reduce data processing time, error correction is done in parallel during each bit evaluation of a conversion. Lower-weighted capacitors (smaller than C9) are not calibrated, so no error correction is performed for these capacitors. However, they are self-tested to guarantee that the error contribution is below specified accuracy limits.

The final result is checked for any overflow or underflow. When the processed data is above the maximum value that can be represented by ADC resolution, the output data is forced to all 1s (FFFh for 12-bit resolution). Similarly, if the processed data is negative then output data is forced to all 0s (000h for 12-bit resolution).

60.3.20 User-defined offset and gain values

In addition to the ADC-calculated offset value and gain value, ADC enables you to specify another set of offset and gain values in Offset And Gain User (OFSGNUSR). These values must be stored in two's complement format, where the leftmost bit (the most significant bit) is the sign bit.

ADC subtracts these values from the final processed conversion result. The values are used to removed fixed DC bias or any other known errors—for instance those caused by board design, sensors, and so on.

60.4 External signals

This module has no external signals.

60.5 Clock frequency

ADC's internal blocks are designed for an input clock frequency up to the Fin (Input Signal Bandwidth) maximum (see the ADC electrical specification section in the chip data sheet). The specified speed can be achieved with this clock frequency. If the input clock frequency exceeds the specified maximum frequency, ADC accuracy degrades and, in the worst case scenario, ADC malfunctions. ADC accuracy also degrades if the input clock frequency is below the specified minimum frequency. Sampling time has an absolute maximum value to guarantee the specified parameters and accuracy. For slower clocks, the sampling clock count must be programmed accordingly.

60.6 Memory map and register definition

ADC has a wide range of configurations to tailor its operation to application requirements. This comes with the drawback of possible corner cases. To avoid unintended behavior, configure ADC before starting a conversion and change any configuration only when ADC is idle, in other words when MSR[ADCSTATUS] = 0.

60.6.1 Transfer error description

The following register accesses cause a transfer error and do not change register content:

- Any access to an unused address.
- Any write access to a read-only register.

60.6.2 ADC register descriptions

60.6.2.1 ADC memory map

ADC_0 base address: 400A_0000h

ADC_1 base address: 400A_4000h

ADC_2 base address: 400A_8000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Main Configuration (MCR)	32	RW	0000_0001h
4h	Main Status (MSR)	32	R	0000_0001h
10h	Interrupt Status (ISR)	32	RW	0000_0000h
14h	Channel End Of Conversion Flag For Precision Inputs (CEOCFR0)	32	RW	0000_0000h
18h	Channel End Of Conversion Flag For Standard Inputs (CEOCFR1)	32	RW	0000_0000h
1Ch	Channel End Of Conversion Flag For External Inputs (CEOCFR2)	32	RW	0000_0000h
20h	Interrupt Mask (IMR)	32	RW	0000_0000h
24h	EOC Interrupt Enable For Precision Inputs (CIMR0)	32	RW	0000_0000h
28h	EOC Interrupt Enable For Standard Inputs (CIMR1)	32	RW	0000_0000h
2Ch	EOC Interrupt Enable For External Inputs (CIMR2)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
30h	Analog Watchdog Threshold Interrupt Status (WTISR)	32	RW	0000_0000h
34h	Analog Watchdog Threshold Interrupt Enable (WTIMR)	32	RW	0000_0000h
40h	Direct Memory Access Configuration (DMAE)	32	RW	0000_0000h
44h	DMA Request Enable For Precision Inputs (DMAR0)	32	RW	0000_0000h
48h	DMA Request Enable For Standard Inputs (DMAR1)	32	RW	0000_0000h
4Ch	DMA Request Enable For External Inputs (DMAR2)	32	RW	0000_0000h
60h - 6Ch	Analog Watchdog Threshold Values (THRHLR0 - THRHLR3)	32	RW	7FFF_0000h
80h	Presampling Control (PSCR)	32	RW	0000_0000h
84h	Presampling Enable For Precision Inputs (PSR0)	32	RW	0000_0000h
88h	Presampling Enable For Standard Inputs (PSR1)	32	RW	0000_0000h
8Ch	Presampling Enable For External Inputs (PSR2)	32	RW	0000_0000h
94h	Conversion Timing For Precision Inputs (CTR0)	32	RW	0000_0016h
98h	Conversion Timing For Standard Inputs (CTR1)	32	RW	0000_0016h
9Ch	Conversion Timing For External Inputs (CTR2)	32	RW	0000_0016h
A4h	Normal Conversion Enable For Precision Inputs (NCMR0)	32	RW	0000_0000h
A8h	Normal Conversion Enable For Standard Inputs (NCMR1)	32	RW	0000_0000h
ACh	Normal Conversion Enable For External Inputs (NCMR2)	32	RW	0000_0000h
B4h	Injected Conversion Enable For Precision Inputs (JCMR0)	32	RW	0000_0000h
B8h	Injected Conversion Enable For Standard Inputs (JCMR1)	32	RW	0000_0000h
BCh	Injected Conversion Enable For External Inputs (JCMR2)	32	RW	0000_0000h
C4h	Delay Start Of Data Conversion (DSDR)	32	RW	0000_0000h
C8h	Power Down Exit Delay (PDEDR)	32	RW	0000_0000h
100h - 11Ch	Precision Input n Conversion Data (PCDR0 - PCDR7)	32	R	0000_0000h
180h - 1DCh	Standard Input n Conversion Data (ICDR0 - ICDR23)	32	R	0000_0000h
200h - 27Ch	External Input n Conversion Data (ECDR0 - ECDR31)	32	R	0000_0000h
2B0h	Channel Analog Watchdog Select For Precision Inputs (CWSELRPI0)	32	RW	0000_0000h
2B4h	Channel Analog Watchdog Select For Precision Inputs (CWSELRPI1)	32	R	0000_0000h
2C0h	Channel Analog Watchdog Select For Standard Inputs (CWSELRSI0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
2C4h	Channel Analog Watchdog Select For Standard Inputs (CWSELRSI1)	32	RW	0000_0000h
2C8h	Channel Analog Watchdog Select For Standard Inputs (CWSELRSI2)	32	RW	0000_0000h
2D0h	Channel Analog Watchdog Select For External inputs (CWSELREI0)	32	RW	0000_0000h
2D4h	Channel Analog Watchdog Select For External inputs (CWSELREI1)	32	RW	0000_0000h
2D8h	Channel Analog Watchdog Select For External inputs (CWSELREI2)	32	RW	0000_0000h
2DCh	Channel Analog Watchdog Select For External inputs (CWSELREI3)	32	RW	0000_0000h
2E0h	Channel Watchdog Enable For Precision Inputs (CWENR0)	32	RW	0000_0000h
2E4h	Channel Watchdog Enable For Standard Inputs (CWENR1)	32	RW	0000_0000h
2E8h	Channel Watchdog Enable For External Inputs (CWENR2)	32	RW	0000_0000h
2F0h	Analog Watchdog Out Of Range For Precision Inputs (AWORR0)	32	RW	0000_0000h
2F4h	Analog Watchdog Out Of Range For Standard Inputs (AWORR1)	32	RW	0000_0000h
2F8h	Analog Watchdog Out Of Range For External Inputs (AWORR2)	32	RW	0000_0000h
340h	Self-Test Configuration 1 (STCR1)	32	RW	1818_2507h
344h	Self-Test Configuration 2 (STCR2)	32	RW	0000_0005h
348h	Self-Test Configuration 3 (STCR3)	32	RW	0000_0300h
34Ch	Self-Test Baud Rate (STBRR)	32	RW	0005_0000h
350h	Self-Test Status 1 (STSR1)	32	RW	0000_0000h
354h	Self-Test Status 2 (STSR2)	32	R	0000_0000h
358h	Self-Test Status 3 (STSR3)	32	R	0000_0000h
35Ch	Self-Test Status 4 (STSR4)	32	R	0000_0000h
370h	Self-Test Conversion Data 1 (STDR1)	32	R	0000_0000h
380h	Self-Test Analog Watchdog S0 (STAW0R)	32	RW	0727_04C5h
388h	Self-Test Analog Watchdog S1 (STAW1R)	32	RW	0000_3FF9h
38Ch	Self-Test Analog Watchdog S2 (STAW2R)	32	RW	0000_3FF9h
394h	Self-Test Analog Watchdog C0 (STAW4R)	32	RW	0010_3FF0h
398h	Self-Test Analog Watchdog C (STAW5R)	32	RW	0010_3FF0h
39Ch	Analog Miscellaneous In/Out register (AMSIO)	32	RW	0000_0811h
3A0h	Control And Calibration Status (CALBISTREG)	32	RW	See section
3A8h	Offset And Gain User (OFSGNUSR)	32	RW	0004_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
3B4h	Calibration Value 2 (CAL2)	32	RW	4300_8243h

60.6.2.2 Main Configuration (MCR)

Offset

Register	Offset
MCR	0h

Function

Configures most ADC features. You must change field values only when ADC is in Idle state, except for the ABORTCHAIN and ABORT fields.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	OWREN	WLSIDE	MODE	0	TRGEN	EDGE	XSTRTEN	NSTART	0	JTRGEN	JEDGE	JSTART	0		BCTUEN	BCTU_MO...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R		0			AVGEN	AVGS		Reserved	ABORTCH...	ABORT	ACKO	0	ADCLKSEL			PWDN
W	STCL								W1C	W1C						
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Fields

Field	Function
31 OWREN	Overwrite Enable Specifies whether a conversion data register accepts new data before the current conversion data has been read. When enabled, the new conversion result overwrites the older data, regardless of the validity of the older conversion data. When PCDR*n*[VALID] = 1, the conversion data has not been read. 0b - Disable 1b - Enable
30 WLSIDE	Write Left-Aligned

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Specifies whether conversion data is right-aligned or left-aligned when written to one of the conversion data registers: • PCDR*n* • ICDR0*n* • ECDR*n* Right-aligned data occupies *CDR[14:0]. Left-aligned data occupies *CDR[15:1], 0b - Right aligned 1b - Left-aligned
29 MODE	Normal Conversion Mode Specifies whether the set of input channels selected for a normal conversion are converted only once per start event or continuously after a start event. If a conversion is done only once, ADC enters an idle state after conversion is complete. For continuous conversion, all input channels selected for the normal conversion are converted continuously in a loop. Stop the loop by writing 0 to NSTART. 0b - Single conversion 1b - Continuous conversion
28 —	Reserved
27 TRGEN	External Trigger Enable Specifies whether the normal trigger input starts a conversion. 0b - Normal trigger input does not start a conversion 1b - Normal trigger input starts a conversion
26 EDGE	External Trigger Edge Selection Selects which edge of the normal trigger input starts a conversion. 0b - Falling edge 1b - Rising edge
25 XSTRTEN	Auxiliary External Start Enable Enables the auxiliary normal trigger source to start a conversion. You can use this field to synchronize the start of a conversion of two ADC instances. 0b - Disable 1b - Enable
24 NSTART	Start Normal Conversion

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Starts a normal conversion. If the continuous mode is selected (MODE = 1), the value of this field remains 1. Write 0 to this field to stop the in-progress conversion loop after conversion of the last input channel of the selected set of input channels is complete. If continuous mode is not selected (MODE = 0), this field automatically resets to 0 after writing 1. 0b - No effect 1b - Starts conversion
23 —	Reserved
22 JTRGEN	Injection Trigger Enable Enables the injected trigger input as a source to start a conversion. 0b - Disable 1b - Enable
21 JEDGE	Injected Trigger Edge Selection Selects which edge of the injected trigger input starts an injected conversion. 0b - Falling edge 1b - Rising edge
20 JSTART	Injected Start Interrupts any ongoing normal conversion and starts an injected conversion. This field automatically resets to 0. If an injected conversion is already ongoing, the field value remains 1 until the next injected conversion starts. This field can only be written with 1. 0b - Injected conversion can be started 1b - Starts an injected conversion
19-18 —	Reserved
17 BCTUEN	Body Cross Trigger Unit Enable Enables BCTU as a trigger source. 0b - Disable 1b - Enable
16 BCTU_MODE	Body Cross Trigger Unit Mode Select Specifies whether sources other than BCTU can start a conversion when the BCTU is enabled as a trigger source (BCTUEN = 1). This field is writeable only when ADC is in Power Down state (PWDN = 1). 0b - Only BCTU can trigger conversion 1b - All trigger sources can trigger conversion

Table continues on the next page...

Table continued from the previous page...

Field	Function
15 STCL	Self-Test Configuration Lock Protects the following registers from writing: • STCR1 • STCR2 • STCR3 • STBRR • STAW0R • STAW1R • STAW2R • STAW4R • STAW5R 0b - Registers are writeable 1b - Registers are read-only
14-12 —	Reserved
11 AVGEN	Averaging Enable Enables conversion averaging. 0b - Disable 1b - Enable
10-9 AVGS	Averaging Select Specifies the number of conversions ADC uses to calculate the conversion result. 00b - 4 conversions 01b - 8 conversions 10b - 16 conversions 11b - 32 conversions
8 —	Reserved
7 ABORTCHAIN	Abort Chain Aborts the conversion of the selected set of input channels (chain of input channels). The currently ongoing conversion completes. This field always reads 0. This field cannot be programmed during BCTU conversion.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field behaves differently for register reads and writes. When reading 0b - Undefined When writing 0b - Conversion continues 1b - Conversion aborted
6 ABORT	Abort Conversion Aborts an ongoing conversion. This field always reads 0 and cannot be written during BCTU conversion. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Undefined When writing 0b - Conversion continues 1b - Conversion aborted
5 ACKO	Auto Clock Off Reduces power consumption by turning off the clock of the analog part of ADC when it is in Idle state. 0b - Clock always active 1b - Clock gated
4-3 —	Reserved
2-1 ADCLKSEL	Conversion Clock (AD_clk) Frequency Selection Selects the frequency for the clock signal of the conversion circuit (AD_clk = module clock ÷ ($2^{ADCLKSEL}$)). This field can be written only when ADC is in Power Down state (PWDN = 1). 00b - Module clock frequency 01b - Module clock frequency ÷ 2 10b - Module clock frequency ÷ 4 11b - Module clock frequency ÷ 8
0 PWDN	Power Down Reduces power consumption by turning off power to the analog portion of ADC.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - ADC enters a functional state 1b - ADC enters Power Down state

60.6.2.3 Main Status (MSR)

Offset

Register	Offset
MSR	4h

Function

Contains flags that indicate the current ADC state.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CALIBRTD	0						NSTART	JABORT	0		JSTART	0	SELF_TE...	Reserved	BCTUSTA...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CHADDR							0			ACKO	0		ADCSTATUS		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Fields

Field	Function
31 CALIBRTD	Calibration Status Indicates ADC calibration status. 0b - Uncalibrated or calibration unsuccessful 1b - Calibrated
30-25 —	Reserved
24 NSTART	Normal Conversion Started Shows whether a normal conversion is in progress.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Not in progress 1b - In progress
23 JABORT	Injected Conversion Aborted Indicates whether the conversion of a set of inputs selected for injected conversion has been aborted. This field resets to 0 when a new injected conversion is started. 0b - Not aborted 1b - Aborted
22-21 —	Reserved
20 JSTART	Injected Conversion Started Indicates whether an ongoing conversion was started by the injection trigger. 0b - Not an injected conversion 1b - Injected conversion
19 —	Reserved
18 SELF_TEST_S	Indicates whether an ongoing conversion is for self-test. 0b - Not self-test 1b - Self-test
17 —	Reserved
16 BCTUSTART	BCTU Conversion Started Indicates whether a BCTU conversion is ongoing. This field is 1 when a BCTU trigger event is received and the BCTU conversion starts. When the BCTU trigger mode is selected, this field is automatically reset to 0 when conversion is completed. Otherwise, if BCTU control mode is selected, this field resets to 0 when the BCTU is disabled (BCTUEN = 0). 0b - Conversion was not triggered by BCTU 1b - Ongoing conversion was triggered by BCTU
15-9 CHADDR	Input Under Measure Contains the number of the input that is currently converted. 000_0000b-011_1111b - Input number
8-6 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
5 ACKO	Auto Clock-Off On Indicates whether the auto clock-off feature is active, and the conversion circuit is not receiving a clock signal. When auto clock-off is inactive, the conversion circuit is clocked. 0b - Inactive 1b - Active
4-3 —	Reserved
2-0 ADCSTATUS	ADC State Indicates the current state of the ADC Finite State Machine (FSM). 000b - Idle 001b - Power Down 010b - Wait 011b - Calibrate 100b - Convert 110b - Done

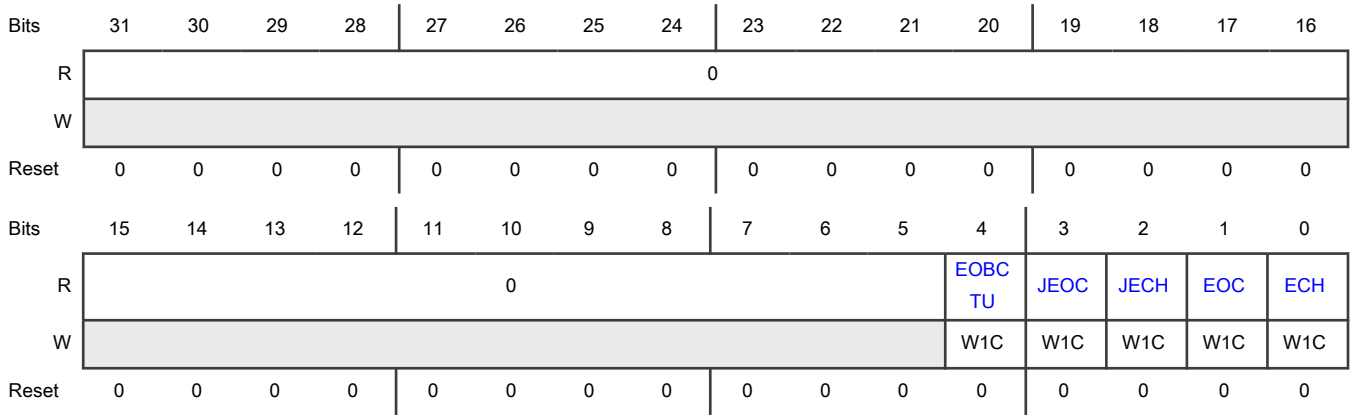
60.6.2.4 Interrupt Status (ISR)

Offset

Register	Offset
ISR	10h

Function
Contains flags that indicate whether an interrupt has been generated.

Diagram



Fields

Field	Function
31-5 —	Reserved
4 EOBCTU	End Of BCTU Conversion Indicates the status of the End of BCTU Conversion (EOBCTU) interrupt, which is generated after a BCTU conversion completes. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No EOBCTU interrupt generated 1b - EOBCTU interrupt generated When writing 0b - No effect 1b - Clears flag
3 JEOC	End Of Injected Conversion Indicates the status of the End of Injected Conversion (JEOC) interrupt, which is generated after an injected conversion completes. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No JEOC interrupt generated 1b - JEOC interrupt generated When writing

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No effect 1b - Clears flag
2 JECH	End Of Injected Chain Conversion Indicates the status of the End of Injected Chain (JECH) interrupt, which is generated after injected conversion of a set of inputs completes. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No JECH interrupt generated 1b - JECH interrupt generated When writing 0b - No effect 1b - Clears flag
1 EOC	End Of Conversion Indicates the status of the End of Conversion (EOC) interrupt, which is generated after a normal conversion completes. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No EOC interrupt generated 1b - Interrupt generated When writing 0b - No effect 1b - Clears flag
0 ECH	End Of Chain Conversion Indicates the status of the End of Chain (ECH) conversion interrupt, which is generated after a set of inputs has been converted normally. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Indicates no ECH interrupt generated 1b - Indicates an ECH interrupt has been generated When writing

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No effect 1b - Clears flag

60.6.2.5 Channel End Of Conversion Flag For Precision Inputs (CEOCFR0)

Offset

Register	Offset
CEOCFR0	14h

Function

Contains flags that indicate whether conversion is complete for a precision input.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved						Reserved	Reserved	PIEOCF7	PIEOCF6	PIEOCF5	PIEOCF4	PIEOCF3	PIEOCF2	PIEOCF1	PIEOCF0
W									W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-10 —	Reserved
9 —	Reserved
8 —	Reserved
7	Precision Input End Of Conversion Flag 7

Table continues on the next page...

Table continued from the previous page...

Field	Function
PIEOCF7	Indicates whether conversion of a precision input 7 has completed. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion not complete 1b - Conversion complete When writing 0b - No effect 1b - Clears flag
6 PIEOCF6	Precision Input End Of Conversion Flag 6 Indicates whether conversion of a precision input 6 has completed. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion not complete 1b - Conversion complete When writing 0b - No effect 1b - Clears flag
5 PIEOCF5	Precision Input End Of Conversion Flag 5 Indicates whether conversion of a precision input 5 has completed. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion not complete 1b - Conversion complete When writing 0b - No effect 1b - Clears flag
4 PIEOCF4	Precision Input End Of Conversion Flag 4 Indicates whether conversion of a precision input 4 has completed.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field behaves differently for register reads and writes. When reading 0b - Conversion not complete 1b - Conversion complete When writing 0b - No effect 1b - Clears flag
3 PIEOCF3	Precision Input End Of Conversion Flag 3 Indicates whether conversion of a precision input 3 has completed. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion not complete 1b - Conversion complete When writing 0b - No effect 1b - Clears flag
2 PIEOCF2	Precision Input End Of Conversion Flag 2 Indicates whether conversion of a precision input 2 has completed. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion not complete 1b - Conversion complete When writing 0b - No effect 1b - Clears flag
1 PIEOCF1	Precision Input End Of Conversion Flag 1 Indicates whether conversion of a precision input 1 has completed. **NOTE** This field behaves differently for register reads and writes.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When reading 0b - Conversion not complete 1b - Conversion complete When writing 0b - No effect 1b - Clears flag
0 PIEOCF0	Precision Input End Of Conversion Flag 0 Indicates whether conversion of a precision input 0 has completed. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion not complete 1b - Conversion complete When writing 0b - No effect 1b - Clears flag

60.6.2.6 Channel End Of Conversion Flag For Standard Inputs (CEOCFR1)

Offset

Register	Offset
CEOCFR1	18h

Function
Contains flags that indicate whether conversion is complete for a standard input.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								SIEOCF23	SIEOCF22	SIEOCF21	SIEOCF20	SIEOCF19	SIEOCF18	SIEOCF17	SIEOCF16
W									W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SIEOCF15	SIEOCF14	SIEOCF13	SIEOCF12	SIEOCF11	SIEOCF10	SIEOCF9	SIEOCF8	SIEOCF7	SIEOCF6	SIEOCF5	SIEOCF4	SIEOCF3	SIEOCF2	SIEOCF1	SIEOCF0
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-0 SIEOCFn	Standard Input End Of Conversion Flag Indicates whether conversion on standard input *n* is complete. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion not complete 1b - Conversion complete When writing 0b - No effect 1b - Clears flag

60.6.2.7 Channel End Of Conversion Flag For External Inputs (CEOCFR2)

Offset

Register	Offset
CEOCFR2	1Ch

Function

Contains flags that indicate whether conversion is complete for an external input.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ADC_0	CEOCFR2	—
ADC_1	CEOCFR2	—
ADC_2	—	CEOCFR2

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	EIEOCF31	EIEOCF30	EIEOCF29	EIEOCF28	EIEOCF27	EIEOCF26	EIEOCF25	EIEOCF24	EIEOCF23	EIEOCF22	EIEOCF21	EIEOCF20	EIEOCF19	EIEOCF18	EIEOCF17	EIEOCF16
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EIEOCF15	EIEOCF14	EIEOCF13	EIEOCF12	EIEOCF11	EIEOCF10	EIEOCF9	EIEOCF8	EIEOCF7	EIEOCF6	EIEOCF5	EIEOCF4	EIEOCF3	EIEOCF2	EIEOCF1	EIEOCF0
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 EIEOCFn	External Input End Of Conversion Flag Indicates whether conversion on external input *n* is complete. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion not complete 1b - Conversion complete When writing 0b - No effect 1b - Clears flag

60.6.2.8 Interrupt Mask (IMR)

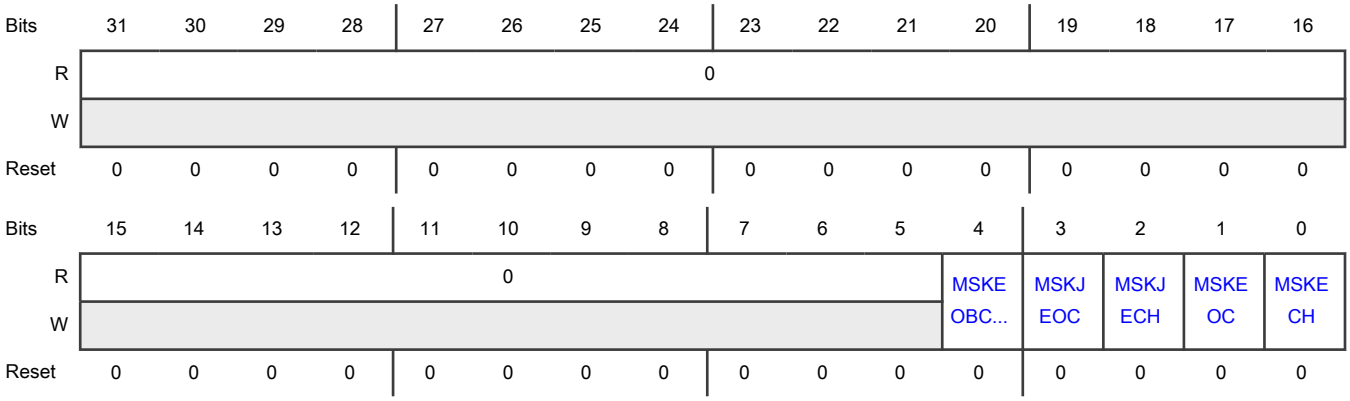
Offset

Register	Offset
IMR	20h

Function

Enables flagging of interrupts.

Diagram



Fields

Field	Function
31-5 —	Reserved
4 MSKEOBCTU	EOBCTU Interrupt Flag Enable Specifies whether a completed BCTU conversion flags the EOBCTU interrupt. 0b - Interrupt is not flagged 1b - Interrupt is flagged
3 MSKJEOC	JEOC Interrupt Flag Enable Specifies whether completion of an injected conversion flags the JEOC interrupt. 0b - Interrupt is not flagged 1b - Interrupt is flagged
2 MSKJECH	JECH Interrupt Flag Enable Specifies whether completion of an injected conversion of a set of inputs flags the JECH interrupt. 0b - Interrupt is not flagged 1b - Interrupt is flagged
1 MSKEOC	EOC Interrupt Flag Enable Specifies whether completion of a normal conversion flags the EOC interrupt. 0b - Interrupt is not flagged 1b - Interrupt is flagged
0	ECH Interrupt Flag Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
MSKECH	Specifies whether completion of a normal conversion of a set of inputs flags the ECH interrupt. 0b - Interrupt is not flagged 1b - Interrupt is flagged

60.6.2.9 EOC Interrupt Enable For Precision Inputs (CIMR0)

Offset

Register	Offset
CIMR0	24h

Function

Specifies whether a completed conversion of a precision input flags an EOC interrupt.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved						Reserved	Reserved	PIEOCIE...	PIEOCIE...	PIEOCIE...	PIEOCIE...	PIEOCIE...	PIEOCIE...	PIEOCIE...	PIEOCIE...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-10 —	Reserved
9 —	Reserved
8 —	Reserved
7	Precision Input EOC Interrupt Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
PIEOCIEN7	Specifies whether a completed conversion of precision input *n* flags an EOC interrupt. 0b - Interrupt is not flagged 1b - Interrupt is flagged
6 PIEOCIEN6	Precision Input EOC Interrupt Enable Specifies whether a completed conversion of precision input *n* flags an EOC interrupt. 0b - Interrupt is not flagged 1b - Interrupt is flagged
5 PIEOCIEN5	Precision Input EOC Interrupt Enable Specifies whether a completed conversion of precision input *n* flags an EOC interrupt. 0b - Interrupt is not flagged 1b - Interrupt is flagged
4 PIEOCIEN4	Precision Input EOC Interrupt Enable Specifies whether a completed conversion of precision input *n* flags an EOC interrupt. 0b - Interrupt is not flagged 1b - Interrupt is flagged
3 PIEOCIEN3	Precision Input EOC Interrupt Enable Specifies whether a completed conversion of precision input *n* flags an EOC interrupt. 0b - Interrupt is not flagged 1b - Interrupt is flagged
2 PIEOCIEN2	Precision Input EOC Interrupt Enable Specifies whether a completed conversion of precision input *n* flags an EOC interrupt. 0b - Interrupt is not flagged 1b - Interrupt is flagged
1 PIEOCIEN1	Precision Input EOC Interrupt Enable Specifies whether a completed conversion of precision input *n* flags an EOC interrupt. 0b - Interrupt is not flagged 1b - Interrupt is flagged
0 PIEOCIEN0	Precision Input EOC Interrupt Enable Specifies whether a completed conversion of precision input *n* flags an EOC interrupt. 0b - Interrupt is not flagged 1b - Interrupt is flagged

60.6.2.10 EOC Interrupt Enable For Standard Inputs (CIMR1)

Offset

Register	Offset
CIMR1	28h

Function
Specifies whether a completed conversion of a standard input flags an EOC interrupt.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								SIEOCIE...	SIEOCIE...	SIEOCIE...	SIEOCIE...	SIEOCIE...	SIEOCIE...	SIEOCIE...	SIEOCIE...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SIEOCIE...	SIEOCIE...	SIEOCIE...	SIEOCIE...	SIEOCIE...	SIEOCIE...	SIEOCIE...	SIEOCIE...	SIEOCIE...	SIEOCIE...	SIEOCIE...	SIEOCIE...	SIEOCIE...	SIEOCIE...	SIEOCIE...	SIEOCIE...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-0 SIEOCIENn	Standard Input EOC Interrupt Enable Specifies whether a completed conversion of standard input *n* flags an EOC interrupt. 0b - Interrupt is not flagged 1b - Interrupt is flagged

60.6.2.11 EOC Interrupt Enable For External Inputs (CIMR2)

Offset

Register	Offset
CIMR2	2Ch

Function
Specifies whether a completed conversion of an external input flags an EOC interrupt.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ADC_0	CIMR2	—
ADC_1	CIMR2	—
ADC_2	—	CIMR2

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	EIEOCIE...	EIEOCIE...	EIEOCIE...	EIEOCIE...	EIEOCIE...	EIEOCIE...	EIEOCIE...	EIEOCIE...	EIEOCIE...	EIEOCIE...	EIEOCIE...	EIEOCIE...	EIEOCIE...	EIEOCIE...	EIEOCIE...	EIEOCIE...
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	EIEOCIE...	EIEOCIE...	EIEOCIE...	EIEOCIE...	EIEOCIE...	EIEOCIE...	EIEOCIE...	EIEOCIE...	EIEOCIE...	EIEOCIE...	EIEOCIE...	EIEOCIE...	EIEOCIE...	EIEOCIE...	EIEOCIE...	EIEOCIE...
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 EIEOCIENn	External Input EOC Interrupt Enable Specifies whether a completed conversion of external input *n* flags an EOC interrupt. 0b - Interrupt is not flagged 1b - Interrupt is flagged

60.6.2.12 Analog Watchdog Threshold Interrupt Status (WTISR)

Offset

Register	Offset
WTISR	30h

Function

Contains flags that indicate the result of a comparison between the conversion result and the threshold value defined in the analog watchdog.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	HAWIF16	LAWIF16	HAWIF15	LAWIF15	HAWIF14	LAWIF14	HAWIF13	LAWIF13	HAWIF12	LAWIF12	HAWIF11	LAWIF11	HAWIF10	LAWIF10	HAWIF9	LAWIF9
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	HAWIF8	LAWIF8	HAWIF7	LAWIF7	HAWIF6	LAWIF6	HAWIF5	LAWIF5	HAWIF4	LAWIF4	HAWIF3	LAWIF3	HAWIF2	LAWIF2	HAWIF1	LAWIF1
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 HAWIF16	High Analog Watchdog Interrupt Flag 16 Indicates whether the conversion result is higher than the high threshold value of the analog watchdog 16. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion result is lower than the specified high threshold 1b - Conversion result is higher than the specified high threshold When writing 0b - No effect 1b - Clears flag
30 LAWIF16	Low Analog Watchdog Interrupt Flag 16 Indicates whether the conversion result is lower than the low threshold value of the analog watchdog 16. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion result is greater than the specified low threshold 1b - Conversion result is lower than the specified low threshold When writing 0b - No effect 1b - Clears flag
29	High Analog Watchdog Interrupt Flag 15

Table continues on the next page...

Table continued from the previous page...

Field	Function
HAWIF15	Indicates whether the conversion result is higher than the high threshold value of the analog watchdog 15. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion result is lower than the specified high threshold 1b - Conversion result is higher than the specified high threshold When writing 0b - No effect 1b - Clears flag
28 LAWIF15	Low Analog Watchdog Interrupt Flag 15 Indicates whether the conversion result is lower than the low threshold value of the analog watchdog 15. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion result is greater than the specified low threshold 1b - Conversion result is lower than the specified low threshold When writing 0b - No effect 1b - Clears flag
27 HAWIF14	High Analog Watchdog Interrupt Flag 14 Indicates whether the conversion result is higher than the high threshold value of the analog watchdog 14. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion result is lower than the specified high threshold 1b - Conversion result is higher than the specified high threshold When writing 0b - No effect 1b - Clears flag
26 LAWIF14	Low Analog Watchdog Interrupt Flag 14 Indicates whether the conversion result is lower than the low threshold value of the analog watchdog 14.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field behaves differently for register reads and writes. When reading 0b - Conversion result is greater than the specified low threshold 1b - Conversion result is lower than the specified low threshold When writing 0b - No effect 1b - Clears flag
25 HAWIF13	High Analog Watchdog Interrupt Flag 13 Indicates whether the conversion result is higher than the high threshold value of the analog watchdog 13. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion result is lower than the specified high threshold 1b - Conversion result is higher than the specified high threshold When writing 0b - No effect 1b - Clears flag
24 LAWIF13	Low Analog Watchdog Interrupt Flag 13 Indicates whether the conversion result is lower than the low threshold value of the analog watchdog 13. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion result is greater than the specified low threshold 1b - Conversion result is lower than the specified low threshold When writing 0b - No effect 1b - Clears flag
23 HAWIF12	High Analog Watchdog Interrupt Flag 12 Indicates whether the conversion result is higher than the high threshold value of the analog watchdog 12.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field behaves differently for register reads and writes. When reading 0b - Conversion result is lower than the specified high threshold 1b - Conversion result is higher than the specified high threshold When writing 0b - No effect 1b - Clears flag
22 LAWIF12	Low Analog Watchdog Interrupt Flag 12 Indicates whether the conversion result is lower than the low threshold value of the analog watchdog 12. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion result is greater than the specified low threshold 1b - Conversion result is lower than the specified low threshold When writing 0b - No effect 1b - Clears flag
21 HAWIF11	High Analog Watchdog Interrupt Flag 11 Indicates whether the conversion result is higher than the high threshold value of the analog watchdog 11. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion result is lower than the specified high threshold 1b - Conversion result is higher than the specified high threshold When writing 0b - No effect 1b - Clears flag
20 LAWIF11	Low Analog Watchdog Interrupt Flag 11 Indicates whether the conversion result is lower than the low threshold value of the analog watchdog 11.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field behaves differently for register reads and writes. When reading 0b - Conversion result is greater than the specified low threshold 1b - Conversion result is lower than the specified low threshold When writing 0b - No effect 1b - Clears flag
19 HAWIF10	High Analog Watchdog Interrupt Flag 10 Indicates whether the conversion result is higher than the high threshold value of the analog watchdog 10. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion result is lower than the specified high threshold 1b - Conversion result is higher than the specified high threshold When writing 0b - No effect 1b - Clears flag
18 LAWIF10	Low Analog Watchdog Interrupt Flag 10 Indicates whether the conversion result is lower than the low threshold value of the analog watchdog 10. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion result is greater than the specified low threshold 1b - Conversion result is lower than the specified low threshold When writing 0b - No effect 1b - Clears flag
17 HAWIF9	High Analog Watchdog Interrupt Flag 9 Indicates whether the conversion result is higher than the high threshold value of the analog watchdog 9.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field behaves differently for register reads and writes. When reading 0b - Conversion result is lower than the specified high threshold 1b - Conversion result is higher than the specified high threshold When writing 0b - No effect 1b - Clears flag
16 LAWIF9	Low Analog Watchdog Interrupt Flag 9 Indicates whether the conversion result is lower than the low threshold value of the analog watchdog 9. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion result is greater than the specified low threshold 1b - Conversion result is lower than the specified low threshold When writing 0b - No effect 1b - Clears flag
15 HAWIF8	High Analog Watchdog Interrupt Flag 8 Indicates whether the conversion result is higher than the high threshold value of the analog watchdog 8. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion result is lower than the specified high threshold 1b - Conversion result is higher than the specified high threshold When writing 0b - No effect 1b - Clears flag
14 LAWIF8	Low Analog Watchdog Interrupt Flag 8 Indicates whether the conversion result is lower than the low threshold value of the analog watchdog 8. **NOTE** This field behaves differently for register reads and writes.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When reading 0b - Conversion result is greater than the specified low threshold 1b - Conversion result is lower than the specified low threshold When writing 0b - No effect 1b - Clears flag
13 HAWIF7	High Analog Watchdog Interrupt Flag 7 Indicates whether the conversion result is higher than the high threshold value of the analog watchdog 7. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion result is lower than the specified high threshold 1b - Conversion result is higher than the specified high threshold When writing 0b - No effect 1b - Clears flag
12 LAWIF7	Low Analog Watchdog Interrupt Flag 7 Indicates whether the conversion result is lower than the low threshold value of the analog watchdog 7. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion result is greater than the specified low threshold 1b - Conversion result is lower than the specified low threshold When writing 0b - No effect 1b - Clears flag
11 HAWIF6	High Analog Watchdog Interrupt Flag 6 Indicates whether the conversion result is higher than the high threshold value of the analog watchdog 6. **NOTE** This field behaves differently for register reads and writes. When reading

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Conversion result is lower than the specified high threshold 1b - Conversion result is higher than the specified high threshold When writing 0b - No effect 1b - Clears flag
10 LAWIF6	Low Analog Watchdog Interrupt Flag 6 Indicates whether the conversion result is lower than the low threshold value of the analog watchdog 6. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion result is greater than the specified low threshold 1b - Conversion result is lower than the specified low threshold When writing 0b - No effect 1b - Clears flag
9 HAWIF5	High Analog Watchdog Interrupt Flag 5 Indicates whether the conversion result is higher than the high threshold value of the analog watchdog 5. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion result is lower than the specified high threshold 1b - Conversion result is higher than the specified high threshold When writing 0b - No effect 1b - Clears flag
8 LAWIF5	Low Analog Watchdog Interrupt Flag 5 Indicates whether the conversion result is lower than the low threshold value of the analog watchdog 5. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion result is greater than the specified low threshold

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Conversion result is lower than the specified low threshold When writing 0b - No effect 1b - Clears flag
7 HAWIF4	High Analog Watchdog Interrupt Flag 4 Indicates whether the conversion result is higher than the high threshold value of the analog watchdog 4. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion result is lower than the specified high threshold 1b - Conversion result is higher than the specified high threshold When writing 0b - No effect 1b - Clears flag
6 LAWIF4	Low Analog Watchdog Interrupt Flag 4 Indicates whether the conversion result is lower than the low threshold value of the analog watchdog 4. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion result is greater than the specified low threshold 1b - Conversion result is lower than the specified low threshold When writing 0b - No effect 1b - Clears flag
5 HAWIF3	High Analog Watchdog Interrupt Flag 3 Indicates whether the conversion result is higher than the high threshold value of the analog watchdog 3. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion result is lower than the specified high threshold 1b - Conversion result is higher than the specified high threshold When writing

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No effect 1b - Clears flag
4 LAWIF3	Low Analog Watchdog Interrupt Flag 3 Indicates whether the conversion result is lower than the low threshold value of the analog watchdog 3. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion result is greater than the specified low threshold 1b - Conversion result is lower than the specified low threshold When writing 0b - No effect 1b - Clears flag
3 HAWIF2	High Analog Watchdog Interrupt Flag 2 Indicates whether the conversion result is higher than the high threshold value of the analog watchdog 2. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion result is lower than the specified high threshold 1b - Conversion result is higher than the specified high threshold When writing 0b - No effect 1b - Clears flag
2 LAWIF2	Low Analog Watchdog Interrupt Flag 2 Indicates whether the conversion result is lower than the low threshold value of the analog watchdog 2. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion result is greater than the specified low threshold 1b - Conversion result is lower than the specified low threshold When writing 0b - No effect 1b - Clears flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 HAWIF1	High Analog Watchdog Interrupt Flag 1 Indicates whether the conversion result is higher than the high threshold value of the analog watchdog 1. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion result is lower than the specified high threshold 1b - Conversion result is higher than the specified high threshold When writing 0b - No effect 1b - Clears flag
0 LAWIF1	Low Analog Watchdog Interrupt Flag 1 Indicates whether the conversion result is lower than the low threshold value of the analog watchdog 1. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion result is greater than the specified low threshold 1b - Conversion result is lower than the specified low threshold When writing 0b - No effect 1b - Clears flag

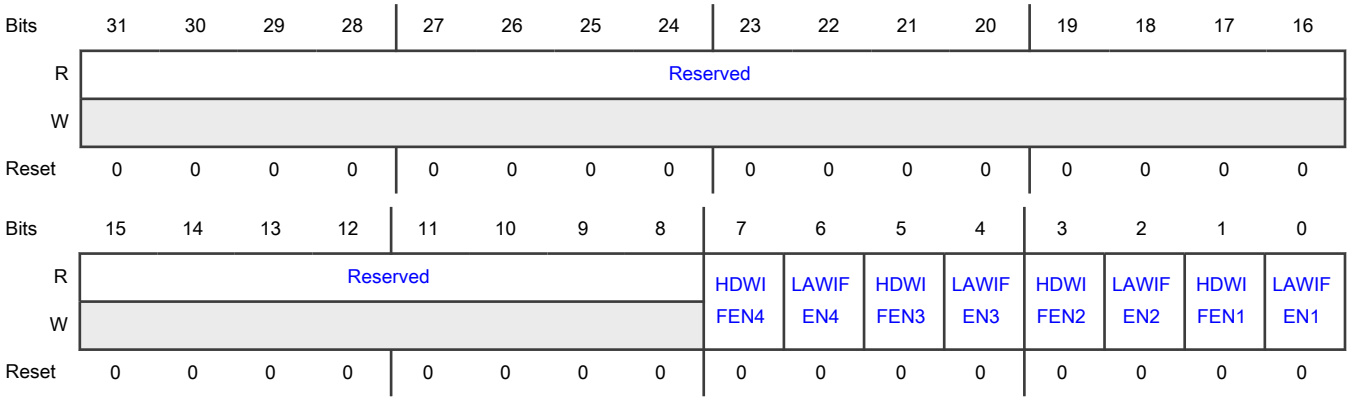
60.6.2.13 Analog Watchdog Threshold Interrupt Enable (WTIMR)

Offset

Register	Offset
WTIMR	34h

Function
Enables the interrupt for each analog watchdog threshold value.

Diagram



Fields

Field	Function
31-8 —	Reserved
7 HDWIFEN4	High Data Watchdog Interrupt Flag Enable n Enables flagging of an interrupt when a conversion result is higher than the high threshold value of the analog watchdog *n*. 0b - Interrupt is not flagged 1b - Interrupt is flagged
6 LAWIFEN4	Low Analog Watchdog Interrupt Flag Enable n Enables flagging of an interrupt when a conversion result is lower than the low threshold value of analog watchdog *n*. 0b - Interrupt is not flagged 1b - Interrupt is flagged
5 HDWIFEN3	High Data Watchdog Interrupt Flag Enable n Enables flagging of an interrupt when a conversion result is higher than the high threshold value of the analog watchdog *n*. 0b - Interrupt is not flagged 1b - Interrupt is flagged
4 LAWIFEN3	Low Analog Watchdog Interrupt Flag Enable n Enables flagging of an interrupt when a conversion result is lower than the low threshold value of analog watchdog *n*. 0b - Interrupt is not flagged 1b - Interrupt is flagged
3	High Data Watchdog Interrupt Flag Enable n

Table continues on the next page...

Table continued from the previous page...

Field	Function
HDWIFEN2	Enables flagging of an interrupt when a conversion result is higher than the high threshold value of the analog watchdog *n*. 0b - Interrupt is not flagged 1b - Interrupt is flagged
2 LAWIFEN2	Low Analog Watchdog Interrupt Flag Enable n Enables flagging of an interrupt when a conversion result is lower than the low threshold value of analog watchdog *n*. 0b - Interrupt is not flagged 1b - Interrupt is flagged
1 HDWIFEN1	High Data Watchdog Interrupt Flag Enable n Enables flagging of an interrupt when a conversion result is higher than the high threshold value of the analog watchdog *n*. 0b - Interrupt is not flagged 1b - Interrupt is flagged
0 LAWIFEN1	Low Analog Watchdog Interrupt Flag Enable n Enables flagging of an interrupt when a conversion result is lower than the low threshold value of analog watchdog *n*. 0b - Interrupt is not flagged 1b - Interrupt is flagged

60.6.2.14 Direct Memory Access Configuration (DMAE)

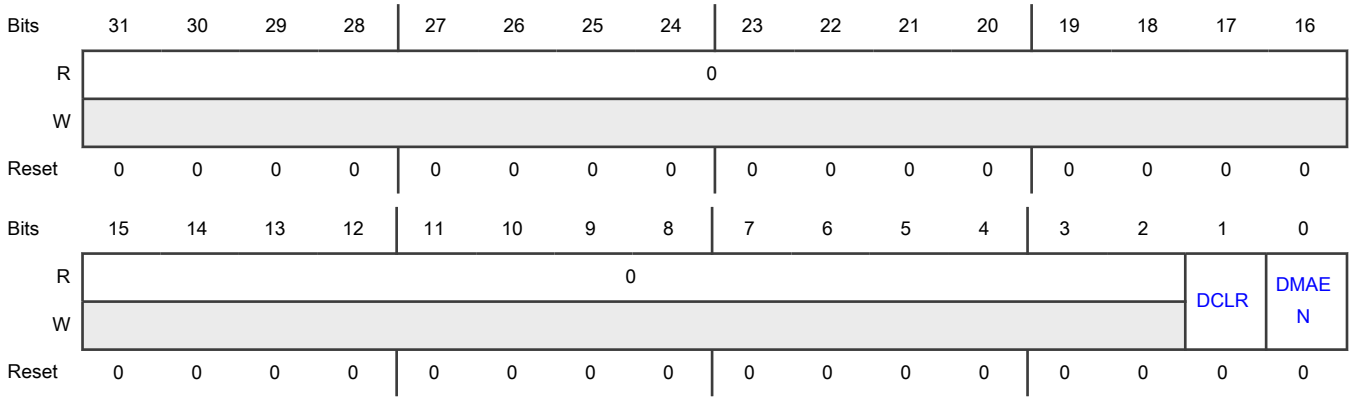
Offset

Register	Offset
DMAE	40h

Function

Configures the DMA feature.

Diagram



Fields

Field	Function
31-2 —	Reserved
1 DCLR	DMA Clear Request Selects one of the following events to clear a DMA request: • DMA controller acknowledges the request. • Application reads the conversion data register. 0b - DMA controller acknowledges the request 1b - Conversion data register is read
0 DMAEN	DMA Enable Enables DMA. 0b - Disable 1b - Enable

60.6.2.15 DMA Request Enable For Precision Inputs (DMAR0)

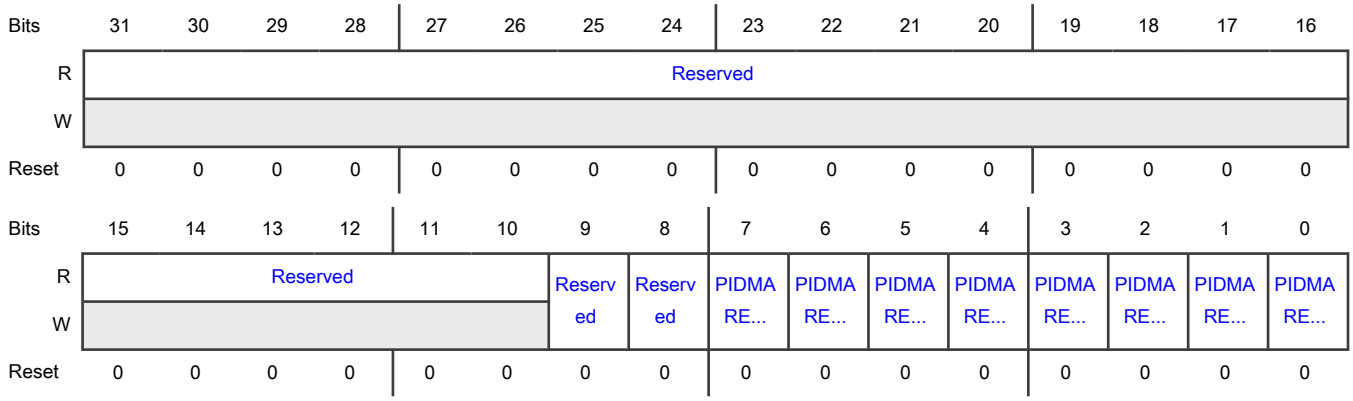
Offset

Register	Offset
DMAR0	44h

Function

Selects the precision inputs that trigger a DMA request after conversion is complete.

Diagram



Fields

Field	Function
31-10 —	Reserved
9 —	Reserved
8 —	Reserved
7 PIDMAREN7	Precision Input DMA Request Enable 7 Specifies whether a DMA request is triggered after conversion is complete for precision input 7. 0b - Not triggered 1b - Triggered
6 PIDMAREN6	Precision Input DMA Request Enable 6 Specifies whether a DMA request is triggered after conversion is complete for precision input 6. 0b - Not triggered 1b - Triggered
5 PIDMAREN5	Precision Input DMA Request Enable 5 Specifies whether a DMA request is triggered after conversion is complete for precision input 5. 0b - Not triggered 1b - Triggered
4 PIDMAREN4	Precision Input DMA Request Enable 4 Specifies whether a DMA request is triggered after conversion is complete for precision input 4. 0b - Not triggered

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Triggered
3 PIDMAREN3	Precision Input DMA Request Enable 3 Specifies whether a DMA request is triggered after conversion is complete for precision input 3. 0b - Not triggered 1b - Triggered
2 PIDMAREN2	Precision Input DMA Request Enable 2 Specifies whether a DMA request is triggered after conversion is complete for precision input 2. 0b - Not triggered 1b - Triggered
1 PIDMAREN1	Precision Input DMA Request Enable 1 Specifies whether a DMA request is triggered after conversion is complete for precision input 1. 0b - Not triggered 1b - Triggered
0 PIDMAREN0	Precision Input DMA Request Enable 0 Specifies whether a DMA request is triggered after conversion is complete for precision input 0. 0b - Not triggered 1b - Triggered

60.6.2.16 DMA Request Enable For Standard Inputs (DMAR1)

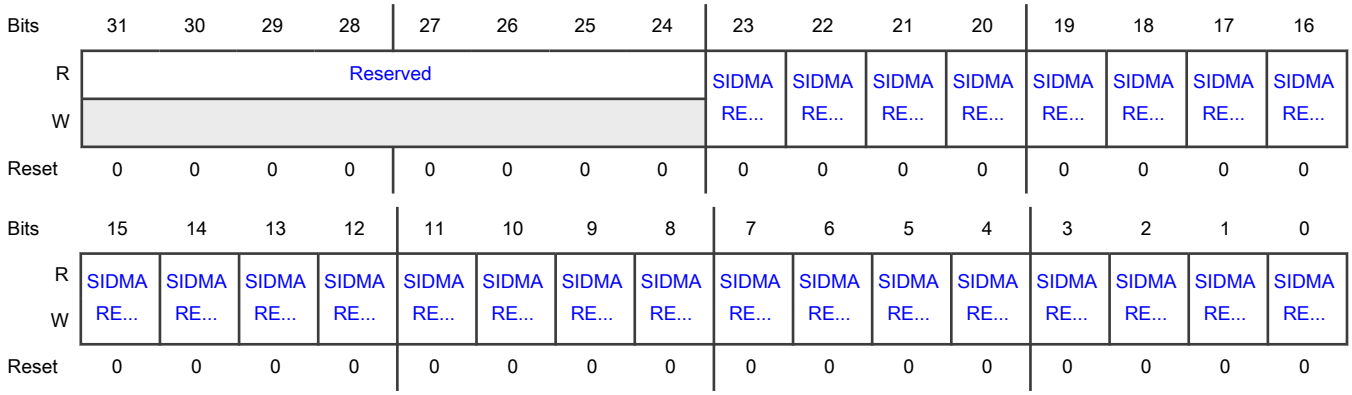
Offset

Register	Offset
DMAR1	48h

Function

Selects the standard inputs that trigger a DMA request after conversion is complete.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-0 SIDMARENn	Standard Input DMA Request Enable n Specifies whether a DMA request is triggered after conversion is complete for standard input *n*. 0b - DMA request is not triggered 1b - DMA is request triggered

60.6.2.17 DMA Request Enable For External Inputs (DMAR2)

Offset

Register	Offset
DMAR2	4Ch

Function

Selects the external inputs that trigger a DMA request after conversion is complete.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ADC_0	DMAR2	—
ADC_1	DMAR2	—
ADC_2	—	DMAR2

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	EIDMARE...	EIDMARE...	EIDMARE...	EIDMARE...	EIDMARE...	EIDMARE...	EIDMARE...	EIDMARE...	EIDMARE...	EIDMARE...	EIDMARE...	EIDMARE...	EIDMARE...	EIDMARE...	EIDMARE...	EIDMARE...
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	EIDMARE...	EIDMARE...	EIDMARE...	EIDMARE...	EIDMARE...	EIDMARE...	EIDMARE...	EIDMARE...	EIDMARE...	EIDMARE...	EIDMARE...	EIDMARE...	EIDMARE...	EIDMARE...	EIDMARE...	EIDMARE...
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 EIDMARENn	External Input DMA Request Enable n Specifies whether a DMA request is triggered after external input *n* is converted. 0b - DMA request is not triggered 1b - DMA request is triggered

60.6.2.18 Analog Watchdog Threshold Values (THRHLR0 - THRHLR3)

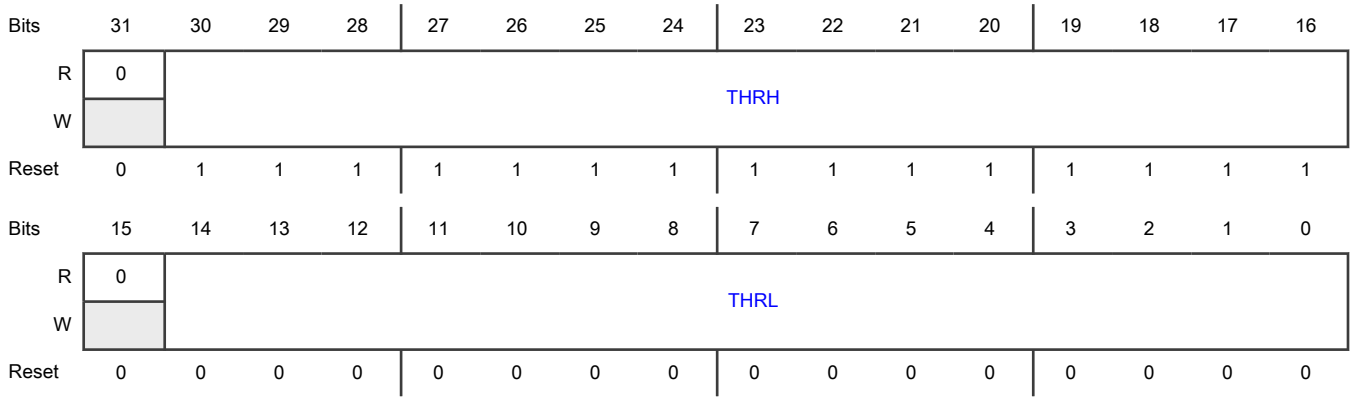
Offset

Register	Offset
THRHLR0	60h
THRHLR1	64h
THRHLR2	68h
THRHLR3	6Ch

Function

Specifies limits for the valid operating range of a monitored input.

Diagram



Fields

Field	Function
31 —	Reserved
30-16 THRH	High Threshold Value If enabled, flags an interrupt when the converted data of the input is higher than this value.
15 —	Reserved
14-0 THRL	Low Threshold Value If enabled, flags an interrupt when the converted data of the input is lower than this value.

60.6.2.19 Presampling Control (PSCR)

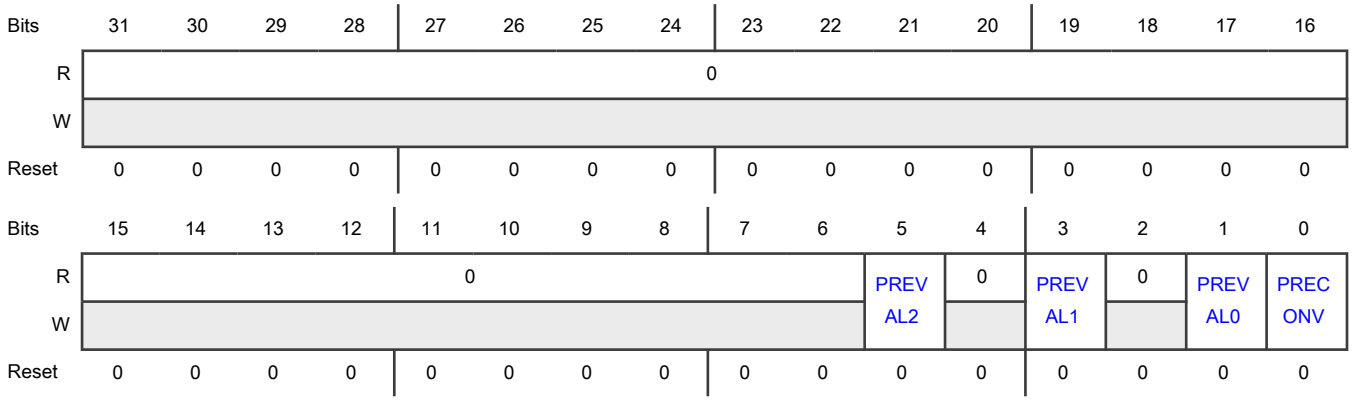
Offset

Register	Offset
PSCR	80h

Function

Configures ADC to presample an internal voltage before the actual input is converted.

Diagram



Fields

Field	Function
31-6 —	Reserved
5 PREVAL2	Presampling Voltage Select For External Inputs Selects the internal voltage that is presampled for external inputs. **NOTE** This field is not supported in every instance. The following table includes only supported registers. 0b - VREFL 1b - VREFH
4 —	Reserved
3 PREVAL1	Presampling Voltage Select For Standard Inputs Selects the internal voltage that is presampled for standard inputs and for the temperature sensor. 0b - VREFL 1b - VREFH
2	Reserved

Instance	Field supported in	Field not supported in
ADC_0	PSCR	—
ADC_1	PSCR	—
ADC_2	—	PSCR

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
1 PREVAL0	Presampling Voltage Select For Precision Inputs Selects the internal voltage that is presampled for precision inputs. 0b - VREFL 1b - VREFH
0 PRECONV	Convert Presampled Value Specifies whether presampling is followed by the comparison. If enabled, presampling is followed by conversion and the result is written to the conversion data register of the selected input. 0b - No conversion after presampling 1b - Presampling is followed by conversion

60.6.2.20 Presampling Enable For Precision Inputs (PSR0)

Offset

Register	Offset
PSR0	84h

Function

Enables presampling for each precision input.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved						Reserved	Reserved	PRES7	PRES6	PRES5	PRES4	PRES3	PRES2	PRES1	PRES0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-10 —	Reserved
9 —	Reserved
8 —	Reserved
7 PRES7	Presampling Enable n Enables presampling for precision input *n*. 0b - Disable 1b - Enable
6 PRES6	Presampling Enable n Enables presampling for precision input *n*. 0b - Disable 1b - Enable
5 PRES5	Presampling Enable n Enables presampling for precision input *n*. 0b - Disable 1b - Enable
4 PRES4	Presampling Enable n Enables presampling for precision input *n*. 0b - Disable 1b - Enable
3 PRES3	Presampling Enable n Enables presampling for precision input *n*. 0b - Disable 1b - Enable
2 PRES2	Presampling Enable n Enables presampling for precision input *n*. 0b - Disable 1b - Enable
1	Presampling Enable n

Table continues on the next page...

Table continued from the previous page...

Field	Function
PRES1	Enables presampling for precision input *n*. 0b - Disable 1b - Enable
0 PRES0	Presampling Enable n Enables presampling for precision input *n*. 0b - Disable 1b - Enable

60.6.2.21 Presampling Enable For Standard Inputs (PSR1)

Offset

Register	Offset
PSR1	88h

Function
Enables presampling for each standard input.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								PRES 23	PRES 22	PRES 21	PRES 20	PRES 19	PRES 18	PRES 17	PRES 16
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PRES 15	PRES 14	PRES 13	PRES 12	PRES 11	PRES 10	PRES 9	PRES 8	PRES 7	PRES 6	PRES 5	PRES 4	PRES 3	PRES 2	PRES 1	PRES 0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-0	Presampling Enable n

Table continues on the next page...

Table continued from the previous page...

Field	Function
PRESn	Enables presampling for standard input *n*. 0b - Disable 1b - Enable

60.6.2.22 Presampling Enable For External Inputs (PSR2)

Offset

Register	Offset
PSR2	8Ch

Function

Enables presampling for each external input.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ADC_0	PSR2	—
ADC_1	PSR2	—
ADC_2	—	PSR2

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PRES 31	PRES 30	PRES 29	PRES 28	PRES 27	PRES 26	PRES 25	PRES 24	PRES 23	PRES 22	PRES 21	PRES 20	PRES 19	PRES 18	PRES 17	PRES 16
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PRES 15	PRES 14	PRES 13	PRES 12	PRES 11	PRES 10	PRES 9	PRES 8	PRES 7	PRES 6	PRES 5	PRES 4	PRES 3	PRES 2	PRES 1	PRES 0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PRESn	Presampling Enable n Enables presampling for external input *n*. 0b - Disable 1b - Enable

60.6.2.23 Conversion Timing For Precision Inputs (CTR0)

Offset

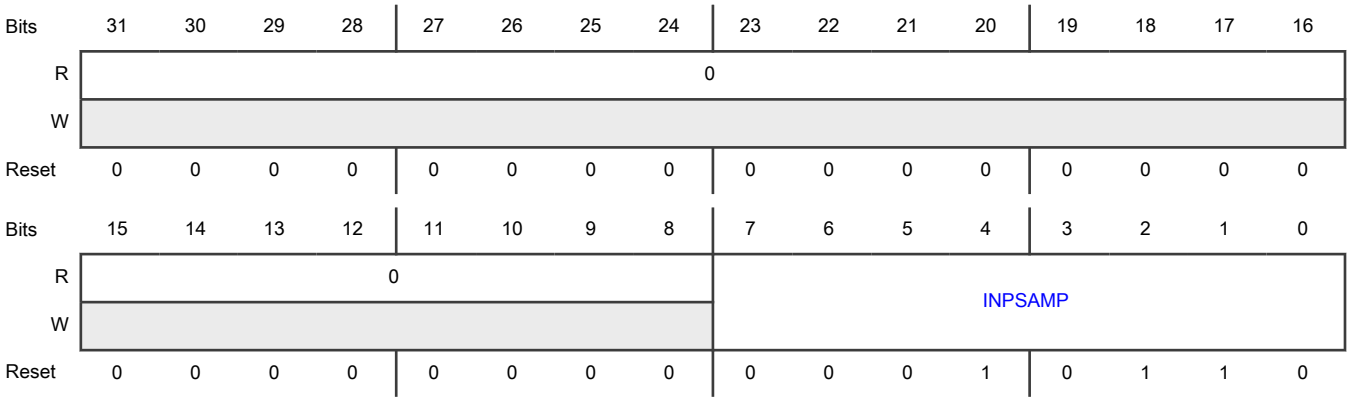
Register	Offset
CTR0	94h

Function

Specifies duration, in terms of the number of conversion clock cycles, of the sampling of precision inputs.

The conversion clock frequency depends on the configuration of MCR[ADCLKSEL].

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 INPSAMP	Input Sample Cycles Specifies the sample duration in terms of conversion clock cycles. The minimum value is 8. Specifying a lower value automatically forces a value of 8.

60.6.2.24 Conversion Timing For Standard Inputs (CTR1)

Offset

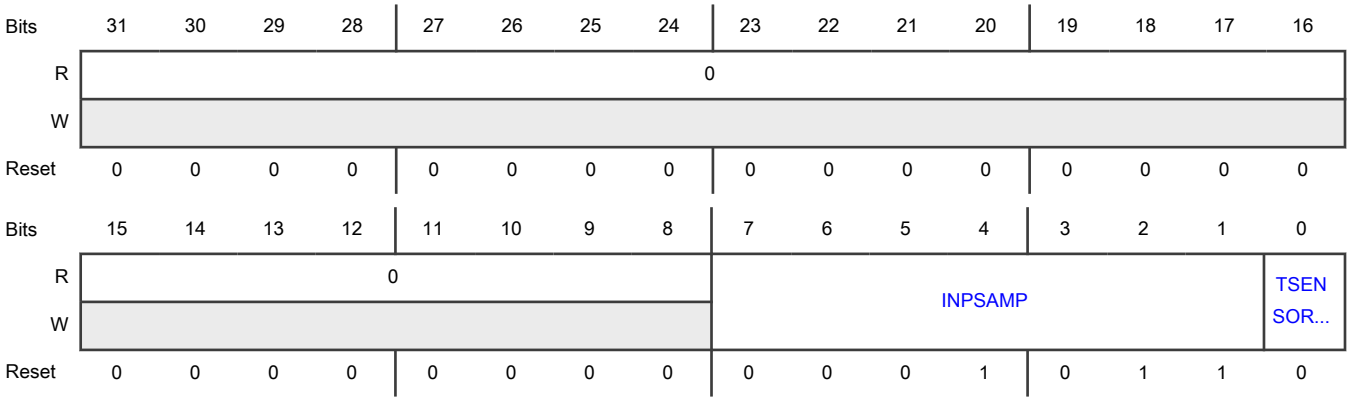
Register	Offset
CTR1	98h

Function

Specifies the duration, in terms of the number of conversion clock cycles, of the sampling of standard inputs or the temperature sensor.

The conversion clock frequency depends on the configuration of MCR[ADCLKSEL].

Diagram



Fields

Field	Function
31-8 —	Reserved
7-1 INPSAMP	Specifies the sample duration in terms of conversion clock cycles. The minimum value is 8. Specifying a lower value automatically forces a value of 8. This field uses the TSENSOR_SEL bit as its low-order bit. In other words, INPSAMP is effectively an 8-bit field, but the low-order bit has a dual function.
0 TSENSOR_SEL	Temperature Sensor Voltage Select Selects the voltage for external temperature sensors (if available). If a temperature sensor is available, this field may be used as the selector for the mux between two temperature sensors. **NOTE** This field is applicable only when there is more than one temperature sensor. 0b - Selects temperature sensor source 0 1b - Selects temperature sensor source 1

60.6.2.25 Conversion Timing For External Inputs (CTR2)

Offset

Register	Offset
CTR2	9Ch

Function

Specifies the number of conversion clock cycles when the external inputs are sampled.

The conversion clock frequency depends on the configuration of MCR[ADCLKSEL].

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ADC_0	CTR2	—
ADC_1	CTR2	—
ADC_2	—	CTR2

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								INPSAMP							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1	0

Fields

Field	Function
31-8 —	Reserved
7-0 INPSAMP	Input Sample Cycles Specifies the sample duration in terms of conversion clock cycles. The minimum acceptable value is 8. Specifying a lower value automatically forces a value of 8.

60.6.2.26 Normal Conversion Enable For Precision Inputs (NCMR0)

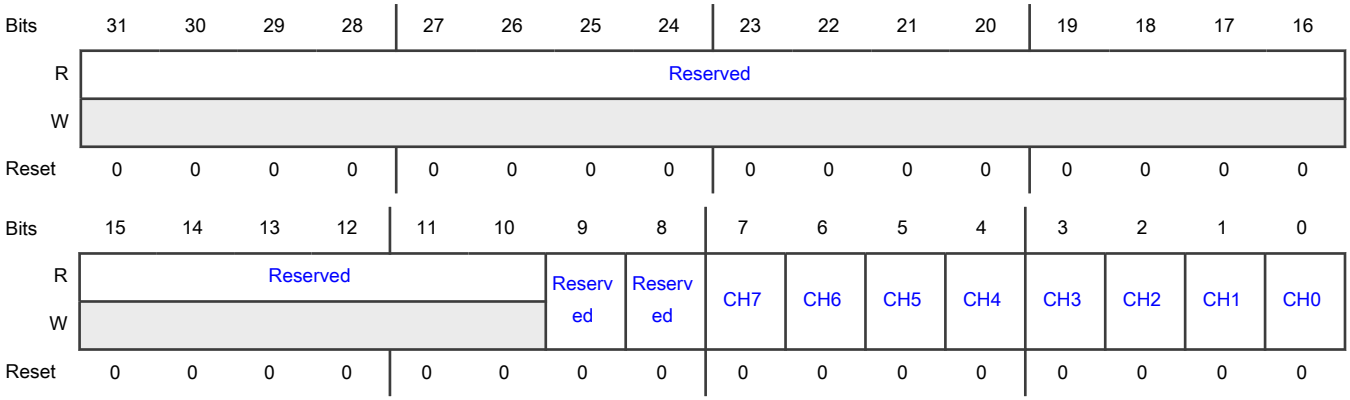
Offset

Register	Offset
NCMR0	A4h

Function

Selects the precision inputs to be converted during a normal conversion.

Diagram



Fields

Field	Function
31-10 —	Reserved
9 —	Reserved
8 —	Reserved
7 CH7	Precision Input To Be Converted Selects precision input *n* for conversion. 0b - Input is not selected 1b - Input is selected
6 CH6	Precision Input To Be Converted Selects precision input *n* for conversion. 0b - Input is not selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Input is selected
5 CH5	Precision Input To Be Converted Selects precision input *n* for conversion. 0b - Input is not selected 1b - Input is selected
4 CH4	Precision Input To Be Converted Selects precision input *n* for conversion. 0b - Input is not selected 1b - Input is selected
3 CH3	Precision Input To Be Converted Selects precision input *n* for conversion. 0b - Input is not selected 1b - Input is selected
2 CH2	Precision Input To Be Converted Selects precision input *n* for conversion. 0b - Input is not selected 1b - Input is selected
1 CH1	Precision Input To Be Converted Selects precision input *n* for conversion. 0b - Input is not selected 1b - Input is selected
0 CH0	Precision Input To Be Converted Selects precision input *n* for conversion. 0b - Input is not selected 1b - Input is selected

60.6.2.27 Normal Conversion Enable For Standard Inputs (NCMR1)

Offset

Register	Offset
NCMR1	A8h

Function

Selects the standard inputs to be converted during a normal conversion.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								CH55	CH54	CH53	CH52	CH51	CH50	CH49	CH48
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CH47	CH46	CH45	CH44	CH43	CH42	CH41	CH40	CH39	CH38	CH37	CH36	CH35	CH34	CH33	CH32
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-0 CHn	Standard Input To Be Converted Selects standard input *n* for conversion. 0b - Input *n* is not selected 1b - Input *n* is selected

60.6.2.28 Normal Conversion Enable For External Inputs (NCMR2)

Offset

Register	Offset
NCMR2	ACh

Function

Selects the external inputs to be converted during a normal conversion.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ADC_0	NCMR2	—

Table continues on the next page...

Table continued from the previous page...

Instance	Register supported	Register not supported
ADC_1	NCMR2	—
ADC_2	—	NCMR2

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	CH95	CH94	CH93	CH92	CH91	CH90	CH89	CH88	CH87	CH86	CH85	CH84	CH83	CH82	CH81	CH80
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	CH79	CH78	CH77	CH76	CH75	CH74	CH73	CH72	CH71	CH70	CH69	CH68	CH67	CH66	CH65	CH64
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 CHn	External Input To Be Converted Selects external input *n* for conversion. 0b - Input *n* is not selected 1b - Input *n* is selected

60.6.2.29 Injected Conversion Enable For Precision Inputs (JCMR0)

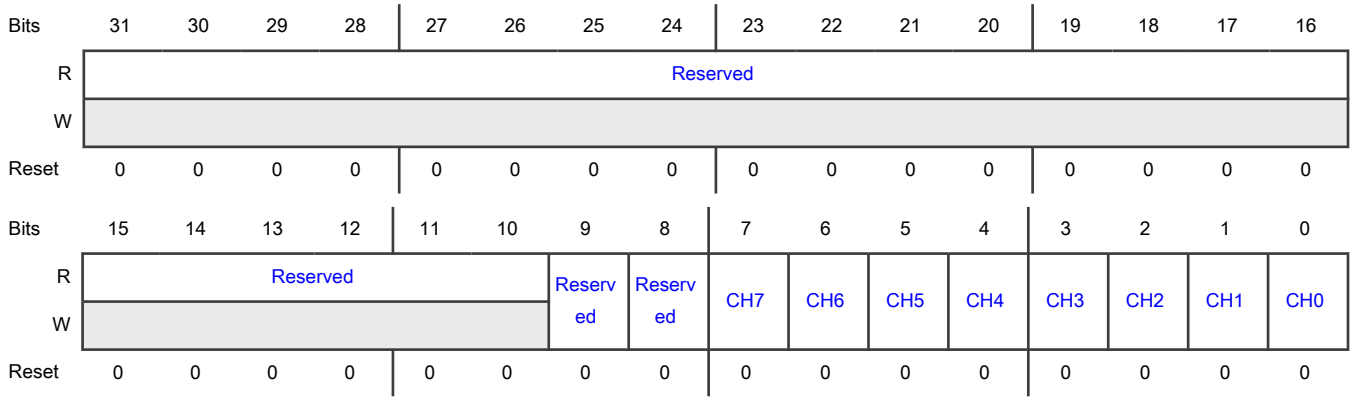
Offset

Register	Offset
JCMR0	B4h

Function

Selects the precision inputs to be converted during an injected conversion.

Diagram



Fields

Field	Function
31-10 —	Reserved
9 —	Reserved
8 —	Reserved
7 CH7	Precision Input To Be Converted Selects precision input 7 for conversion. 0b - Input 7 is not selected 1b - Input 7 is selected
6 CH6	Precision Input To Be Converted Selects precision input 6 for conversion. 0b - Input 6 is not selected 1b - Input 6 is selected
5 CH5	Precision Input To Be Converted Selects precision input 5 for conversion. 0b - Input 5 is not selected 1b - Input 5 is selected
4 CH4	Precision Input To Be Converted Selects precision input 4 for conversion. 0b - Input 4 is not selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Input 4 is selected
3 CH3	Precision Input To Be Converted Selects precision input 3 for conversion. 0b - Input 3 is not selected 1b - Input 3 is selected
2 CH2	Precision Input To Be Converted Selects precision input 2 for conversion. 0b - Input 2 is not selected 1b - Input 2 is selected
1 CH1	Precision Input To Be Converted Selects precision input 1 for conversion. 0b - Input 1 is not selected 1b - Input 1 is selected
0 CH0	Precision Input To Be Converted Selects precision input 0 for conversion. 0b - Input 0 is not selected 1b - Input 0 is selected

60.6.2.30 Injected Conversion Enable For Standard Inputs (JCMR1)

Offset

Register	Offset
JCMR1	B8h

Function
Selects the standard inputs to be converted during an injected conversion.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								CH55	CH54	CH53	CH52	CH51	CH50	CH49	CH48
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CH47	CH46	CH45	CH44	CH43	CH42	CH41	CH40	CH39	CH38	CH37	CH36	CH35	CH34	CH33	CH32
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-0 CHn	Standard Input To Be Converted Selects standard input *n* for conversion. 0b - Input *n* is not selected 1b - Input *n* is selected

60.6.2.31 Injected Conversion Enable For External Inputs (JCMR2)

Offset

Register	Offset
JCMR2	BCh

Function

Selects the external inputs to be converted during an injected conversion.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ADC_0	JCMR2	—
ADC_1	JCMR2	—
ADC_2	—	JCMR2

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	CH95	CH94	CH93	CH92	CH91	CH90	CH89	CH88	CH87	CH86	CH85	CH84	CH83	CH82	CH81	CH80
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	CH79	CH78	CH77	CH76	CH75	CH74	CH73	CH72	CH71	CH70	CH69	CH68	CH67	CH66	CH65	CH64
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 CHn	External Input To Be Converted Selects external input *n* for conversion. 0b - Input *n* is not selected 1b - Input *n* is selected

60.6.2.32 Delay Start Of Data Conversion (DSDR)

Offset

Register	Offset
DSDR	C4h

Function

Delays the start of the conversion when another input channel is selected. The necessary delay depends on the device characteristics of, for example, the input channel multiplexer.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ADC_0	DSDR	—
ADC_1	DSDR	—
ADC_2	—	DSDR

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DSD															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-0 DSD	Delay Specifies the delay in terms of the number of module clock cycles.

60.6.2.33 Power Down Exit Delay (PDEDR)

Offset

Register	Offset
PDEDR	C8h

Function

Delays the transition out of Power Down into Idle state to wait for necessary settling times. See the device data sheet for the minimal time between the power-down exit request and when ADC can start a conversion.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								PDED							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-8 —	Reserved
7-0 PDED	Delay Specifies a delay in terms of the number of conversion clock cycles.

60.6.2.34 Precision Input n Conversion Data (PCDR0 - PCDR7)

Offset

For n = 0 to 7:

Register	Offset
PCDRn	100h + (n × 4h)

Function

Contains conversion data from precision input *n*.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												VALID	OVERW	RESULT	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CDATA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-20 —	Reserved
19 VALID	Conversion Data Available Indicates whether new conversion data is available. This field is automatically reset to 0 when the data is read.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No unread conversion data 1b - Unread conversion data is available
18 OVERW	Overwrite Status Flag Indicates whether the previous conversion data was overwritten without having been read, in which case the overwritten data is lost. The ability to overwrite conversion data is controlled by MCR[OWREN]. 0b - No unread data is overwritten 1b - Unread data is overwritten
17-16 RESULT	Conversion Data Type Indicates the type of trigger that started the conversion for this conversion data. 00b - Normal trigger 01b - Injected trigger 10b - BCTU trigger
15-0 CDATA	Conversion Data Contains conversion data from precision input *n*, determined by the successive approximation algorithm. The conversion data is always 15 bits wide, regardless of the conversion resolution selected (CALBISTREG[RESN]). Depending on the value of MCR[WLSIDE], the conversion data can be in bits [14:0] (MCR[WLSIDE] = 0), or in bits [15:1] (MCR[WLSIDE] = 1).

60.6.2.35 Standard Input n Conversion Data (ICDR0 - ICDR23)

Offset

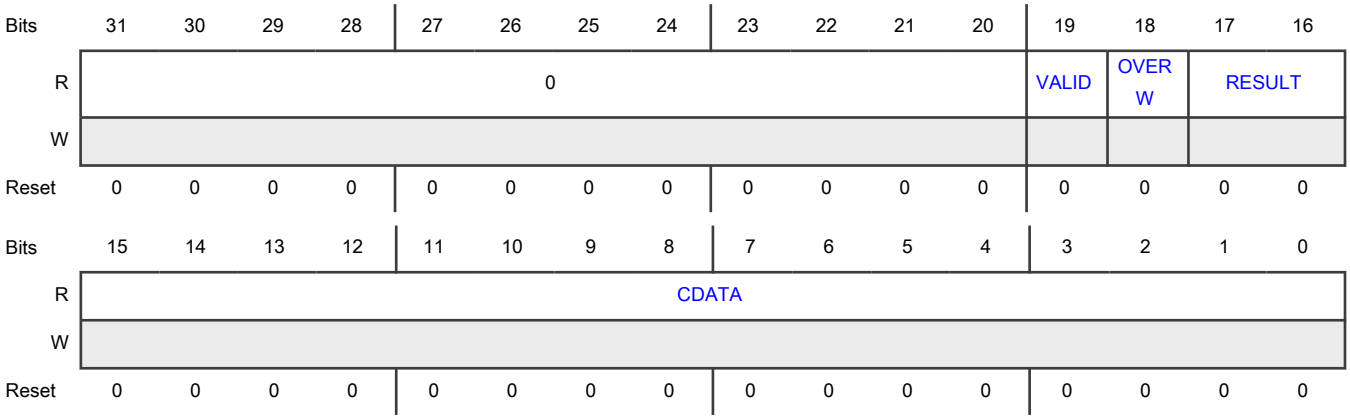
For a = 0 to 23:

Register	Offset
ICDRa	180h + (a × 4h)

Function

Contains conversion data from standard input *n*.

Diagram



Fields

Field	Function
31-20 —	Reserved
19 VALID	Conversion Data Available Indicates whether new conversion data is available. This field is automatically reset to 0 when the data is read. 0b - No unread conversion data 1b - Unread conversion data is available
18 OVERW	Overwrite Status Flag Indicates whether the previous conversion data was overwritten without having been read, in which case the overwritten data is lost. The ability to overwrite conversion data is controlled by MCR[OWREN]. 0b - No unread data is overwritten 1b - Unread data is overwritten
17-16 RESULT	Conversion Data Type Indicates the type of trigger that started the conversion for this conversion data. 00b - Normal trigger 01b - Injected trigger 10b - BCTU trigger
15-0 CDATA	Conversion Data Contains conversion data from standard input *n*, determined by the SAR algorithm. The conversion data bit count is dependent on the conversion resolution selected (CALBISTREG[RESN]). Depending on the value of MCR[WLSIDE], the conversion data MSB bits start from 14 (MCR[WLSIDE] = 0), or 15 (MCR[WLSIDE] = 1).

60.6.2.36 External Input n Conversion Data (ECDR0 - ECDR31)

Offset

For a = 0 to 31:

Register	Offset
ECDRa	200h + (a × 4h)

Function

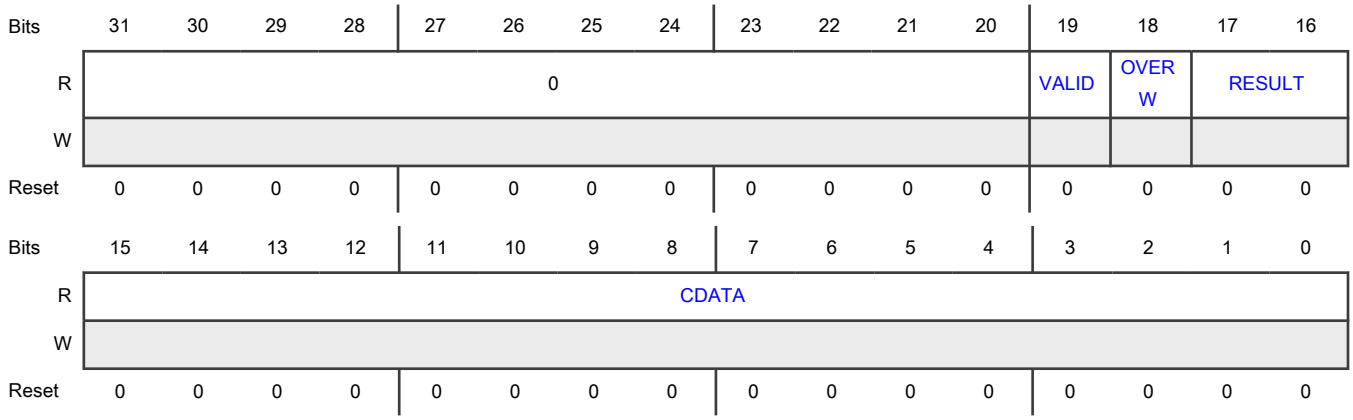
Contains conversion data from external input *n*.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ADC_0	ECDR0–ECDR31	—
ADC_1	ECDR0–ECDR31	—
ADC_2	—	ECDR0–ECDR31

Diagram



Fields

Field	Function
31-20 —	Reserved
19 VALID	Conversion Data Available Indicates whether new conversion data is available. This field is automatically reset to 0 when the data is read.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No unread conversion data 1b - Unread conversion data is available
18 OVERW	Overwrite Status Flag Indicates whether the previous conversion data was overwritten without having been read, in which case the overwritten data is lost. The ability to overwrite conversion data is controlled by MCR[OWREN]. 0b - No unread data is overwritten 1b - Unread data is overwritten
17-16 RESULT	Conversion Data Type Indicates the type of trigger that started the conversion for this conversion data. 00b - Normal trigger 01b - Injected trigger 10b - BCTU trigger
15-0 CDATA	Conversion Data Contains conversion data from external input *n*, determined by the SAR algorithm. The conversion data is always 15 bits wide, regardless of the conversion resolution selected (CALBISTREG[RESN]). Depending on the value of MCR[WLSIDE], the conversion data can be in bits [14:0] (MCR[WLSIDE] = 0), or in bits [15:1] (MCR[WLSIDE] = MCR[WLSIDE]).

60.6.2.37 Channel Analog Watchdog Select For Precision Inputs (CWSELRPI0)

Offset

Register	Offset
CWSELRPI0	2B0h

Function

Selects the analog watchdog threshold register (THRHLR) that provides limits to monitor precision inputs.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		WSEL_SI0_7		0		WSEL_SI0_6		0		WSEL_SI0_5		0		WSEL_SI0_4	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		WSEL_SI0_3		0		WSEL_SI0_2		0		WSEL_SI0_1		0		WSEL_SI0_0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29-28 WSEL_SI0_7	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
27-26 —	Reserved
25-24 WSEL_SI0_6	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
23-22 —	Reserved
21-20 WSEL_SI0_5	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
19-18 —	Reserved
17-16 WSEL_SI0_4	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
15-14 —	Reserved
13-12 WSEL_SI0_3	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
11-10 —	Reserved
9-8 WSEL_SI0_2	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
7-6 —	Reserved
5-4 WSEL_SI0_1	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
3-2 —	Reserved
1-0 WSEL_SI0_0	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3

60.6.2.38 Channel Analog Watchdog Select For Precision Inputs (CWSELRPI1)

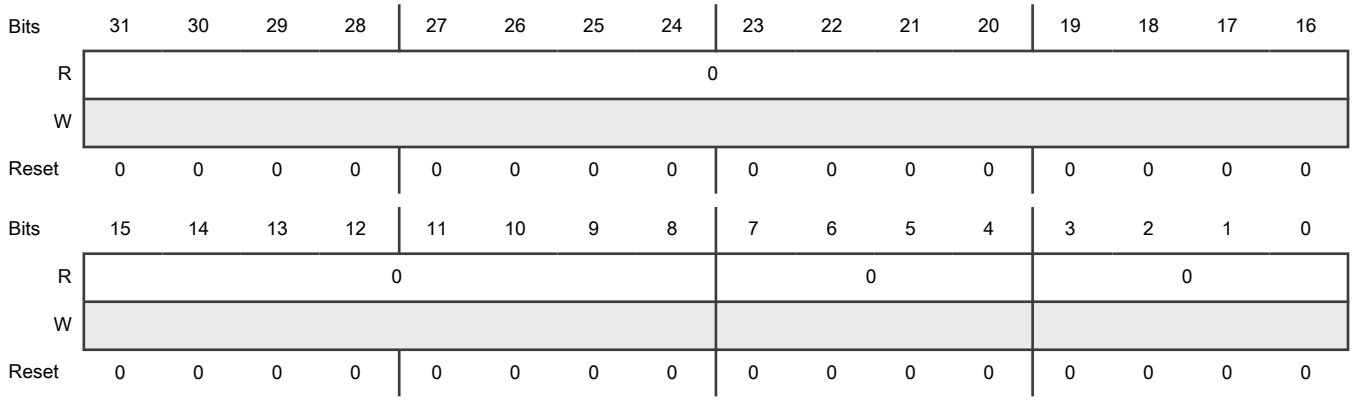
Offset

Register	Offset
CWSELRPI1	2B4h

Function

Selects the analog watchdog threshold register (THRHLR) that provides limits to monitor precision inputs.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-4 —	Reserved
3-0 —	Reserved

60.6.2.39 Channel Analog Watchdog Select For Standard Inputs (CWSELRSI0)

Offset

Register	Offset
CWSELRSI0	2C0h

Function
Selects the analog watchdog threshold register (THRHLR) that provides limits to monitor the standard inputs.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		WSEL_SI7_0		0		WSEL_SI6_0		0		WSEL_SI5_0		0		WSEL_SI4_0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		WSEL_SI3_0		0		WSEL_SI2_0		0		WSEL_SI1_0		0		WSEL_SI0_0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29-28 WSEL_SI7_0	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
27-26 —	Reserved
25-24 WSEL_SI6_0	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
23-22 —	Reserved
21-20 WSEL_SI5_0	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
19-18 —	Reserved
17-16 WSEL_SI4_0	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
15-14 —	Reserved
13-12	Analog Watchdog Selection

Table continues on the next page...

Table continued from the previous page...

Field	Function
WSEL_SI3_0	Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
11-10 —	Reserved
9-8 WSEL_SI2_0	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
7-6 —	Reserved
5-4 WSEL_SI1_0	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
3-2 —	Reserved
1-0 WSEL_SI0_0	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3

60.6.2.40 Channel Analog Watchdog Select For Standard Inputs (CWSELRSI1)

Offset

Register	Offset
CWSELRSI1	2C4h

Function

Selects the analog watchdog threshold register (THRHLR) that provides limits to monitor the standard inputs.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		WSEL_SI7_1		0		WSEL_SI6_1		0		WSEL_SI5_1		0		WSEL_SI4_1	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		WSEL_SI3_1		0		WSEL_SI2_1		0		WSEL_SI1_1		0		WSEL_SI0_1	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29-28 WSEL_SI7_1	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
27-26 —	Reserved
25-24 WSEL_SI6_1	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
23-22 —	Reserved
21-20 WSEL_SI5_1	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
19-18 —	Reserved
17-16 WSEL_SI4_1	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
15-14 —	Reserved
13-12 WSEL_SI3_1	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
11-10 —	Reserved
9-8 WSEL_SI2_1	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
7-6 —	Reserved
5-4 WSEL_SI1_1	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
3-2 —	Reserved
1-0 WSEL_SI0_1	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3

60.6.2.41 Channel Analog Watchdog Select For Standard Inputs (CWSELRSI2)

Offset

Register	Offset
CWSELRSI2	2C8h

Function

Selects the analog watchdog threshold register (THRHLR) that provides limits to monitor the standard inputs.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		WSEL_SI7_2		0		WSEL_SI6_2		0		WSEL_SI5_2		0		WSEL_SI4_2	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		WSEL_SI3_2		0		WSEL_SI2_2		0		WSEL_SI1_2		0		WSEL_SI0_2	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29-28 WSEL_SI7_2	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
27-26 —	Reserved
25-24 WSEL_SI6_2	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
23-22 —	Reserved
21-20 WSEL_SI5_2	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
19-18 —	Reserved
17-16 WSEL_SI4_2	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
15-14 —	Reserved
13-12 WSEL_SI3_2	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
11-10 —	Reserved
9-8 WSEL_SI2_2	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
7-6 —	Reserved
5-4 WSEL_SI1_2	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
3-2 —	Reserved
1-0 WSEL_SI0_2	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3

60.6.2.42 Channel Analog Watchdog Select For External inputs (CWSELREI0)

Offset

Register	Offset
CWSELREI0	2D0h

Function

Selects the analog watchdog threshold register (THRHLR) that provides limits to monitor external inputs.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ADC_0	CWSELREI0	—
ADC_1	CWSELREI0	—
ADC_2	—	CWSELREI0

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		WSEL_SI0_7		0		WSEL_SI0_6		0		WSEL_SI0_5		0		WSEL_SI0_4	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		WSEL_SI0_3		0		WSEL_SI0_2		0		WSEL_SI0_1		0		WSEL_SI0_0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29-28 WSEL_SI0_7	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
27-26 —	Reserved
25-24 WSEL_SI0_6	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
23-22 —	Reserved
21-20 WSEL_SI0_5	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
19-18 —	Reserved
17-16 WSEL_SI0_4	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
15-14 —	Reserved
13-12 WSEL_SI0_3	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
11-10 —	Reserved
9-8 WSEL_SI0_2	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
7-6 —	Reserved
5-4 WSEL_SI0_1	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
3-2 —	Reserved
1-0 WSEL_SI0_0	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3

60.6.2.43 Channel Analog Watchdog Select For External inputs (CWSELREI1)

Offset

Register	Offset
CWSELREI1	2D4h

Function

Selects the analog watchdog threshold register (THRHLR) that provides limits to monitor external inputs.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ADC_0	CWSELREI1	—
ADC_1	CWSELREI1	—
ADC_2	—	CWSELREI1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		WSEL_SI1_15		0		WSEL_SI1_14		0		WSEL_SI1_13		0		WSEL_SI1_12	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		WSEL_SI1_11		0		WSEL_SI1_10		0		WSEL_SI1_9		0		WSEL_SI1_8	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29-28 WSEL_SI1_15	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
27-26 —	Reserved
25-24 WSEL_SI1_14	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
23-22 —	Reserved
21-20 WSEL_SI1_13	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
19-18 —	Reserved
17-16 WSEL_SI1_12	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
15-14 —	Reserved
13-12 WSEL_SI1_11	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
11-10 —	Reserved
9-8 WSEL_SI1_10	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
7-6 —	Reserved
5-4 WSEL_SI1_9	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
3-2 —	Reserved
1-0 WSEL_SI1_8	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3

60.6.2.44 Channel Analog Watchdog Select For External inputs (CWSELREI2)

Offset

Register	Offset
CWSELREI2	2D8h

Function
Selects the analog watchdog threshold register (THRHLR) that provides limits to monitor external inputs.

NOTE
Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ADC_0	CWSELREI2	—
ADC_1	CWSELREI2	—
ADC_2	—	CWSELREI2

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		WSEL_SI2_23		0		WSEL_SI2_22		0		WSEL_SI2_21		0		WSEL_SI2_20	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		WSEL_SI2_19		0		WSEL_SI2_18		0		WSEL_SI2_17		0		WSEL_SI2_16	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29-28 WSEL_SI2_23	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
27-26 —	Reserved
25-24 WSEL_SI2_22	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
23-22 —	Reserved
21-20 WSEL_SI2_21	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
19-18 —	Reserved
17-16 WSEL_SI2_20	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
15-14 —	Reserved
13-12 WSEL_SI2_19	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
11-10 —	Reserved
9-8 WSEL_SI2_18	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
7-6 —	Reserved
5-4 WSEL_SI2_17	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
3-2 —	Reserved
1-0 WSEL_SI2_16	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3

60.6.2.45 Channel Analog Watchdog Select For External inputs (CWSELREI3)

Offset

Register	Offset
CWSELREI3	2DCh

Function

Selects the analog watchdog threshold register (THRHLR) that provides limits to monitor external inputs.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ADC_0	CWSELREI3	—
ADC_1	CWSELREI3	—
ADC_2	—	CWSELREI3

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		WSEL_SI3_31		0		WSEL_SI3_30		0		WSEL_SI3_29		0		WSEL_SI3_28	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		WSEL_SI3_27		0		WSEL_SI3_26		0		WSEL_SI3_25		0		WSEL_SI3_24	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29-28 WSEL_SI3_31	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
27-26 —	Reserved
25-24 WSEL_SI3_30	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
23-22 —	Reserved
21-20 WSEL_SI3_29	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
19-18 —	Reserved
17-16 WSEL_SI3_28	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
15-14 —	Reserved
13-12 WSEL_SI3_27	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
11-10 —	Reserved
9-8 WSEL_SI3_26	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
7-6 —	Reserved
5-4 WSEL_SI3_25	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3
3-2 —	Reserved
1-0 WSEL_SI3_24	Analog Watchdog Selection Selects the analog watchdog thresholds that the input is to be compared to. 00b - Analog watchdog THRHLR0 01b - Analog watchdog THRHLR1 10b - Analog watchdog THRHLR2 11b - Analog watchdog THRHLR3

60.6.2.46 Channel Watchdog Enable For Precision Inputs (CWENR0)

Offset

Register	Offset
CWENR0	2E0h

Function

Enables the analog watchdog per precision input.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved						Reserved	Reserved	CWEN7	CWEN6	CWEN5	CWEN4	CWEN3	CWEN2	CWEN1	CWEN0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-10 —	Reserved
9 —	Reserved
8 —	Reserved
7 CWEN7	Channel Analog Watchdog Enable 7 Enables the analog watchdog for precision input 7. When enabled, conversion data on the input is compared to the selected data watchdog threshold value. 0b - Disable 1b - Enable
6 CWEN6	Channel Analog Watchdog Enable 6 Enables the analog watchdog for precision input 6. When enabled, conversion data on the input is compared to the selected data watchdog threshold value. 0b - Disable 1b - Enable
5 CWEN5	Channel Analog Watchdog Enable 5 Enables the analog watchdog for precision input 5. When enabled, conversion data on the input is compared to the selected data watchdog threshold value. 0b - Disable 1b - Enable
4 CWEN4	Channel Analog Watchdog Enable 4 Enables the analog watchdog for precision input 4. When enabled, conversion data on the input is compared to the selected data watchdog threshold value. 0b - Disable 1b - Enable
3 CWEN3	Channel Analog Watchdog Enable 3 Enables the analog watchdog for precision input 3. When enabled, conversion data on the input is compared to the selected data watchdog threshold value. 0b - Disable 1b - Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
2 CWEN2	Channel Analog Watchdog Enable 2 Enables the analog watchdog for precision input 2. When enabled, conversion data on the input is compared to the selected data watchdog threshold value. 0b - Disable 1b - Enable
1 CWEN1	Channel Analog Watchdog Enable 1 Enables the analog watchdog for precision input 1. When enabled, conversion data on the input is compared to the selected data watchdog threshold value. 0b - Disable 1b - Enable
0 CWEN0	Channel Analog Watchdog Enable 0 Enables the analog watchdog for precision input 0. When enabled, conversion data on the input is compared to the selected data watchdog threshold value. 0b - Disable 1b - Enable

60.6.2.47 Channel Watchdog Enable For Standard Inputs (CWENR1)

Offset

Register	Offset
CWENR1	2E4h

Function

Enables the analog watchdog for standard input *n*.

When enabled, conversion data on the input is compared to the selected data watchdog threshold value.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								CWEN 55	CWEN 54	CWEN 53	CWEN 52	CWEN 51	CWEN 50	CWEN 49	CWEN 48
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CWEN 47	CWEN 46	CWEN 45	CWEN 44	CWEN 43	CWEN 42	CWEN 41	CWEN 40	CWEN 39	CWEN 38	CWEN 37	CWEN 36	CWEN 35	CWEN 34	CWEN 33	CWEN 32
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-0 CWENn	Channel Analog Watchdog Enable For Standard Inputs Enables the analog watchdog for standard input *n*. 0b - Disable 1b - Enable

60.6.2.48 Channel Watchdog Enable For External Inputs (CWENR2)

Offset

Register	Offset
CWENR2	2E8h

Function

Enables the analog watchdog per external input.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ADC_0	CWENR2	—
ADC_1	CWENR2	—
ADC_2	—	CWENR2

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	CWEN 95	CWEN 94	CWEN 93	CWEN 92	CWEN 91	CWEN 90	CWEN 89	CWEN 88	CWEN 87	CWEN 86	CWEN 85	CWEN 84	CWEN 83	CWEN 82	CWEN 81	CWEN 80
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	CWEN 79	CWEN 78	CWEN 77	CWEN 76	CWEN 75	CWEN 74	CWEN 73	CWEN 72	CWEN 71	CWEN 70	CWEN 69	CWEN 68	CWEN 67	CWEN 66	CWEN 65	CWEN 64
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 CWENn	Channel Analog Watchdog Enable For External Inputs Enables the analog watchdog for external input *n*. 0b - Disable 1b - Enable

60.6.2.49 Analog Watchdog Out Of Range For Precision Inputs (AWORR0)

Offset

Register	Offset
AWORR0	2F0h

Function

Indicates the status of analog watchdog comparisons on precision inputs.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0						Reserved	Reserved	AWOR_CH7	AWOR_CH6	AWOR_CH5	AWOR_CH4	AWOR_CH3	AWOR_CH2	AWOR_CH1	AWOR_CH0
W									W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-10 —	Reserved
9 —	Reserved
8 —	Reserved
7 AWOR_CH7	Analog Watchdog Out Of Range For Precision Inputs Indicates whether a data conversion on precision input 7 is out of the limits defined by the selected analog watchdog threshold value. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion is within limits 1b - Conversion is not within limits When writing 0b - No effect 1b - Clears flag
6 AWOR_CH6	Analog Watchdog Out Of Range For Precision Inputs Indicates whether a data conversion on precision input 6 is out of the limits defined by the selected analog watchdog threshold value. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion is within limits 1b - Conversion is not within limits When writing 0b - No effect 1b - Clears flag
5 AWOR_CH5	Analog Watchdog Out Of Range For Precision Inputs Indicates whether a data conversion on precision input 5 is out of the limits defined by the selected analog watchdog threshold value.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field behaves differently for register reads and writes. When reading 0b - Conversion is within limits 1b - Conversion is not within limits When writing 0b - No effect 1b - Clears flag
4 AWOR_CH4	Analog Watchdog Out Of Range For Precision Inputs Indicates whether a data conversion on precision input 4 is out of the limits defined by the selected analog watchdog threshold value. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion is within limits 1b - Conversion is not within limits When writing 0b - No effect 1b - Clears flag
3 AWOR_CH3	Analog Watchdog Out Of Range For Precision Inputs Indicates whether a data conversion on precision input 3 is out of the limits defined by the selected analog watchdog threshold value. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion is within limits 1b - Conversion is not within limits When writing 0b - No effect 1b - Clears flag
2 AWOR_CH2	Analog Watchdog Out Of Range For Precision Inputs Indicates whether a data conversion on precision input 2 is out of the limits defined by the selected analog watchdog threshold value.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field behaves differently for register reads and writes. When reading 0b - Conversion is within limits 1b - Conversion is not within limits When writing 0b - No effect 1b - Clears flag
1 AWOR_CH1	Analog Watchdog Out Of Range For Precision Inputs Indicates whether a data conversion on precision input 1 is out of the limits defined by the selected analog watchdog threshold value. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion is within limits 1b - Conversion is not within limits When writing 0b - No effect 1b - Clears flag
0 AWOR_CH0	Analog Watchdog Out Of Range For Precision Inputs Indicates whether a data conversion on precision input 0 is out of the limits defined by the selected analog watchdog threshold value. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion is within limits 1b - Conversion is not within limits When writing 0b - No effect 1b - Clears flag

60.6.2.50 Analog Watchdog Out Of Range For Standard Inputs (AWORR1)

Offset

Register	Offset
AWORR1	2F4h

Function

Indicates the status of analog watchdog comparisons on standard inputs.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								AWOR_CH...	AWOR_CH...	AWOR_CH...	AWOR_CH...	AWOR_CH...	AWOR_CH...	AWOR_CH...	AWOR_CH...
W									W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	AWOR_CH...	AWOR_CH...	AWOR_CH...	AWOR_CH...	AWOR_CH...	AWOR_CH...	AWOR_CH9	AWOR_CH8	AWOR_CH7	AWOR_CH6	AWOR_CH5	AWOR_CH4	AWOR_CH3	AWOR_CH2	AWOR_CH1	AWOR_CH0
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-0 AWOR_CHn	Analog Watchdog Out Of Range For Standard Inputs Indicates whether a data conversion on a standard input *n* is out of the limits defined by the selected analog watchdog threshold value. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Conversion is within limits 1b - Conversion is not within limits When writing 0b - No effect 1b - Clears flag

60.6.2.51 Analog Watchdog Out Of Range For External Inputs (AWORR2)

Offset

Register	Offset
AWORR2	2F8h

Function

Indicates the status of analog watchdog comparisons on external inputs.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ADC_0	AWORR2	—
ADC_1	AWORR2	—
ADC_2	—	AWORR2

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	AWOR_CH...	AWOR_CH...	AWOR_CH...	AWOR_CH...	AWOR_CH...	AWOR_CH...	AWOR_CH...	AWOR_CH...	AWOR_CH...	AWOR_CH...	AWOR_CH...	AWOR_CH...	AWOR_CH...	AWOR_CH...	AWOR_CH...	AWOR_CH...
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	AWOR_CH...	AWOR_CH...	AWOR_CH...	AWOR_CH...	AWOR_CH...	AWOR_CH...	AWOR_CH9	AWOR_CH8	AWOR_CH7	AWOR_CH6	AWOR_CH5	AWOR_CH4	AWOR_CH3	AWOR_CH2	AWOR_CH1	AWOR_CH0
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 AWOR_CHn	Analog Watchdog Out Of Range For External Inputs Indicates whether a data conversion on external input *n* is out of the limits defined by the selected analog watchdog threshold value. **NOTE** This field behaves differently for register reads and writes. When reading

Table continues on the next page...

Field	Function
	0b - Conversion is within limits 1b - Conversion is not within limits When writing 0b - No effect 1b - Clears flag

60.6.2.52 Self-Test Configuration 1 (STCR1)

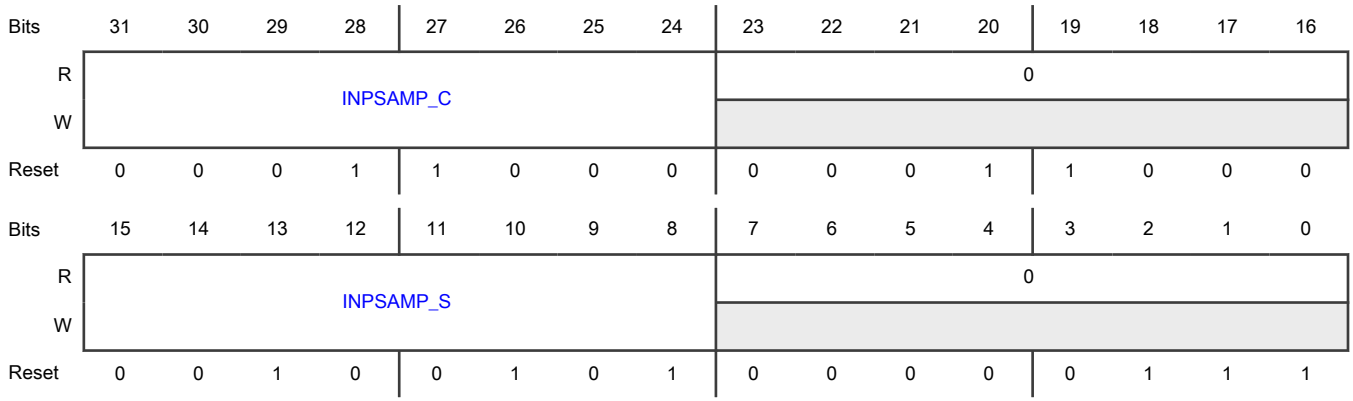
Offset

Register	Offset
STCR1	340h

Function
Specifies the input sampling duration in terms of conversion clock cycles.

Conversion clock frequency depends on the configuration of MCR[ADCLKSEL].

Diagram



Fields

Field	Function
31-24 INPSAMP_C	Input Sampling Time Algorithm C Specifies the sample duration for test conversions related to algorithm C. The minimum acceptable value is 8. Specifying a lower value automatically forces a value of 8.
23-16 —	Reserved
15-8	Input Sampling Time Algorithm S

Table continues on the next page...

Table continued from the previous page...

Field	Function
INPSAMP_S	Specifies sample duration for test conversions related to algorithm S. The minimum value is 8. Specifying a lower value automatically forces a value of 8.
7-0 —	Reserved

60.6.2.53 Self-Test Configuration 2 (STCR2)

Offset

Register	Offset
STCR2	344h

Function

Configures ADC self-test functions.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				MSKWDSE...	SERR	MSKWDTE...	0	MSKST_E...	0				MSKWDG_...	0	MSKWDG_...
W						W1C										
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MSKERR_C	0	MSKERR_...	MSKERR_...	MSKERR_...	0			EN	0		FMA_WDS...	FMA_WDT...	FMA_C	0	FMA_S
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-28 —	Reserved
27 MSKWDSERR	Mask Interrupt Self-Test Watchdog Sequence Error Generates an interrupt when the WDSERR flag transitions to 1. 0b - No interrupt is generated 1b - Interrupt is generated

Table continues on the next page...

Table continued from the previous page...

Field	Function
26 SERR	Self-Test Error Injection Writes 1 to the STSR1[ERR_C], STSR1[ERR_S0], STSR1[ERR_S1], and STSR1[ERR_S2] status fields, causing an error. This field is reset to 0 immediately after writing. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Error can be injected 1b - Error is being injected When writing 0b - No effect 1b - Injects a self-test error
25 MSKWDTERR	Mask Interrupt Self-Test Watchdog Timer Error Generates an interrupt when the WDTERR flag is set. 0b - No interrupt is generated 1b - Interrupt is generated
24 —	Reserved
23 MSKST_EOC	Mask Interrupt Self-Test End Of Conversion Generates an interrupt when the ST_EOC flag is set. 0b - No interrupt is generated 1b - Interrupt is generated
22-19 —	Reserved
18 MSKWDG_EOA_C	Mask Error Interrupt End Of Algorithm C Generates an interrupt when the WDG_EOA_C flag is set. 0b - No interrupt is generated 1b - Interrupt is generated
17 —	Reserved
16 MSKWDG_EOA_S	Mask Error Interrupt End Of Algorithm S Generates an interrupt when the WDG_EOA_S flag is set.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No interrupt is generated 1b - Interrupt is generated
15 MSKERR_C	Mask Error Interrupt Algorithm C Generates an interrupt when the ERR_C flag is set. 0b - No interrupt is generated 1b - Interrupt is generated
14 —	Reserved
13 MSKERR_S2	Mask Error Interrupt Algorithm S2 Generates an interrupt when the ERR_S2 flag is set. 0b - No interrupt is generated 1b - Interrupt is generated
12 MSKERR_S1	Mask Error Interrupt Algorithm S1 Generates an interrupt when the ERR_S1 flag is set. 0b - No interrupt is generated 1b - Interrupt is generated
11 MSKERR_S0	Mask Error Interrupt Algorithm S0 Generates an interrupt when the ERR_S0 flag is set. 0b - No interrupt is generated 1b - Interrupt is generated
10-8 —	Reserved
7 EN	Self-Test Enable Enables ADC structural self-test. When BCTU is enabled, this field has no effect. This field must be 1 before starting normal conversion and must not be changed during conversion. This field must be reset to 0 only after end of conversion for the last self-test channel has been received. 0b - Disable 1b - Enable
6-5 —	Reserved
4	Fault Mapping Self-Test Watchdog Sequence Error

Table continues on the next page...

Table continued from the previous page...

Field	Function
FMA_WDSERR	Specifies whether a self-test watchdog sequence error sets the flag on the critical or noncritical fault line. 0b - Noncritical fault line 1b - Critical fault line
3 FMA_WDTERR	Fault Mapping Self-Test Watchdog Timer Error Specifies whether a self-test watchdog timer error sets the flag on the critical or noncritical fault line. 0b - Noncritical fault line 1b - Critical fault line
2 FMA_C	Fault Mapping Algorithm C Specifies whether a fault in algorithm C sets the flag on the critical or noncritical fault line. 0b - Noncritical fault line 1b - Critical fault line
1 —	Reserved
0 FMA_S	Fault Mapping Algorithm S Specifies whether a fault in algorithm S sets the flag on the critical or noncritical fault line. 0b - Noncritical fault line 1b - Critical fault line

60.6.2.54 Self-Test Configuration 3 (STCR3)

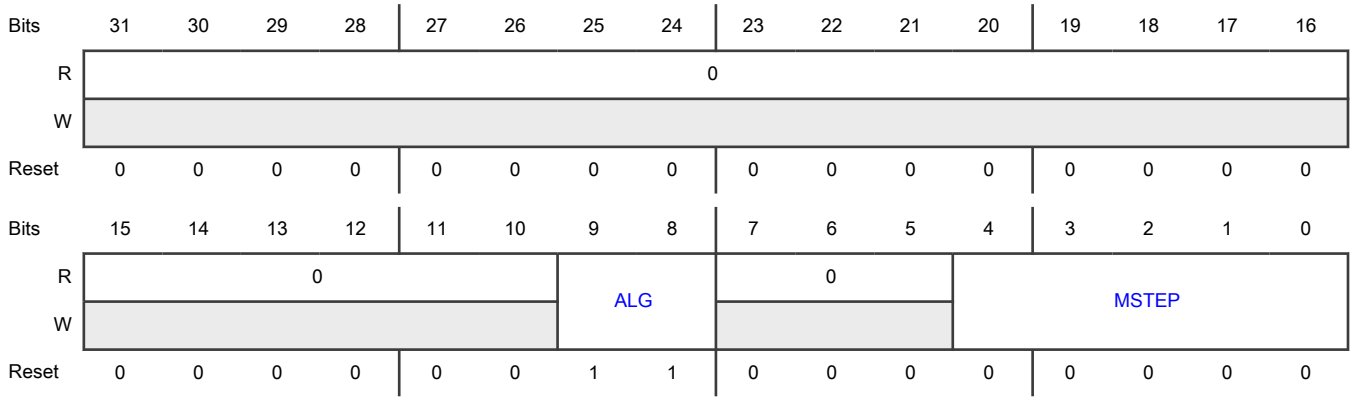
Offset

Register	Offset
STCR3	348h

Function

Configures ADC self-test functions. When BCTU is enabled, writes to the fields registers have no effect. This register must be programmed before starting a conversion and must not be changed when conversion is ongoing.

Diagram



Fields

Field	Function
31-10 —	Reserved
9-8 ALG	Algorithm Selection One-Shot operation mode: • 00 Algorithm S (single step = MSTEP) • 01 Reserved • 10 Algorithm C (single step = MSTEP) • 11 Algorithm S (default) (use for test/debug purposes) Continuous conversion mode: • 00 Algorithm S • 01 Reserved • 10 Algorithm C • 11 Algorithm S + algorithm C (default)
7-5 —	Reserved
4-0 MSTEP	Algorithm Step For One-Shot operation mode: • MSTEP = 0 to 2 for algorithm S • MSTEP = 0 to 11 for algorithm C Unused codes are reserved and must not be used. For Scan operation mode:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is not used in Scan operation mode because single-step execution (interleaved mode) is always performed. This field must be programmed to zero in Scan operation mode.

60.6.2.55 Self-Test Baud Rate (STBRR)

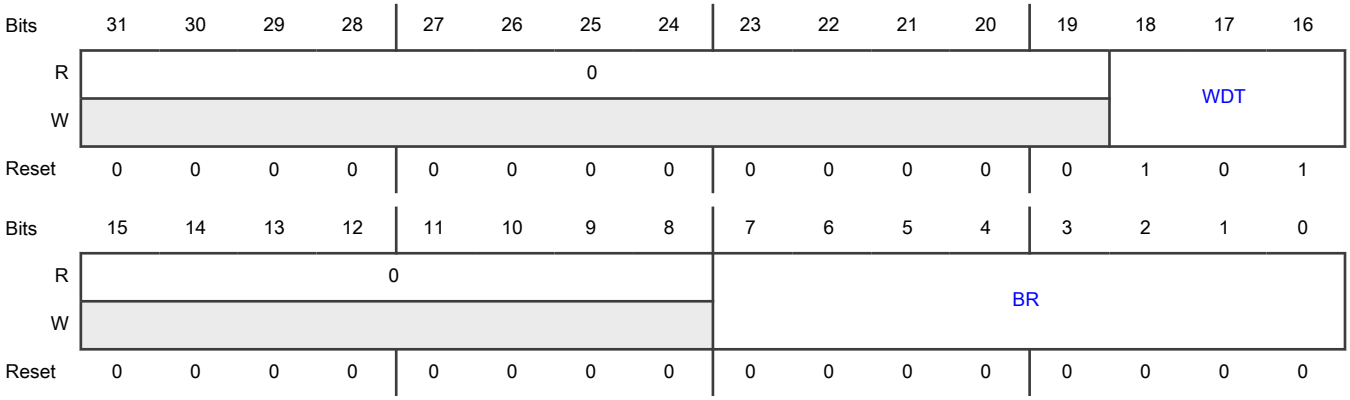
Offset

Register	Offset
STBRR	34Ch

Function

Specifies when the self-test algorithm steps are executed and the maximum time allowed for self-test to finish.

Diagram



Fields

Field	Function
31-19 —	Reserved
18-16 WDT	Self-Test Watchdog Timer Specifies the safe time period in terms of conversion clock cycles. The safe time period is the maximum duration until the running self-test is expected to finish. 000b - 8192 conversion clock cycles (~0.1 ms at 80 MHz) 001b - 39,936 conversion clock cycles (~0.5 ms at 80 MHz) 010b - 79,872 conversion clock cycles (~1 ms at 80 MHz)

Table continues on the next page...

Table continued from the previous page...

Field	Function
	011b - 159,744 conversion clock cycles (~2 ms at 80 MHz) 100b - 400,384 conversion clock cycles (~5 ms at 80 MHz) 101b - 799,744 conversion clock cycles (~10 ms at 80 MHz) 110b - 1,599,488 conversion clock cycles (~20 ms at 80 MHz) 111b - 3,999,744 conversion clock cycles (~50 ms at 80 MHz)
15-8 —	Reserved
7-0 BR	Baud Rate Selects the number of conversions of the selected inputs, after which the next self-test algorithm step executes. This field must be programmed when STCR2[EN] is zero, that is, before enabling self-test. 0000_0000b - A step of the selected self-test algorithm is executed every time after the set of selected inputs has been converted. 1111_1111b - A step of the selected self-test algorithm is executed after the set of selected inputs has been converted 256 times.

60.6.2.56 Self-Test Status 1 (STSR1)

Offset

Register	Offset
STSR1	350h

Function
Indicates self-test status.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				WDSERR	0	WDTERR	OVERWR	ST_EOC	0				WDG_EOA...	0	WDG_EOA...
W					W1C		W1C	W1C	W1C					W1C		W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ERR_C	0	ERR_S2	ERR_S1	ERR_S0	0	STEP_C						0			
W	W1C		W1C	W1C	W1C											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved
27 WDSERR	Self-Test Watchdog Sequence Error Indicates whether the selected self-test algorithm has executed in the correct sequence. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Algorithm executed in correct sequence 1b - Algorithm did not execute in correct sequence When writing 0b - No effect 1b - Clears flag
26 —	Reserved
25 WDTERR	Self-Test Watchdog Timer Error Indicates whether the self-test algorithm has finished within the defined safe time period. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Algorithm finished within the safe time period (or safe time period not yet elapsed). 1b - Algorithm did not finish within safe time period. When writing 0b - No effect 1b - Clears flag
24 OVERWR	Self-Test Error Status Overwrite Indicates whether an error occurred when the respective error status flag (ERR_S0, ERR_S1, ERR_S2, or ERR_C) was set. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No self-test error status flag overwritten

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Self-test error status flag overwritten When writing 0b - No effect 1b - Clears flag
23 ST_EOC	Self-Test End Of Conversion Indicates whether a self-test conversion has completed. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not complete 1b - Complete When writing 0b - No effect 1b - Clears flag
22-19 —	Reserved
18 WDG_EOA_C	Self-Test Watchdog End Of Algorithm C Indicates whether self-test algorithm C has completed. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not complete 1b - Complete When writing 0b - No effect 1b - Clears flag
17 —	Reserved
16 WDG_EOA_S	Self-Test Watchdog End Of Algorithm S Indicates whether self-test algorithm S has completed.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field behaves differently for register reads and writes. When reading 0b - Not complete 1b - Complete When writing 0b - No effect 1b - Clears flag
15 ERR_C	Error Algorithm C Indicates whether an error occurred during execution of algorithm C. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No error 1b - Error When writing 0b - No effect 1b - Clears flag
14 —	Reserved
13 ERR_S2	Error Algorithm S Step 2 Indicates whether an error occurred during execution of step 2 of algorithm S. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No error 1b - Error When writing 0b - No effect 1b - Clears flag
12 ERR_S1	Error Algorithm S Step 1 Indicates whether an error occurred during execution of step 1 of algorithm S.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field behaves differently for register reads and writes. When reading 0b - No error 1b - Error When writing 0b - No effect 1b - Clears flag
11 ERR_S0	Error Algorithm S Step 0 Indicates whether an error occurred during execution of step 0 of algorithm S. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No error 1b - Error When writing 0b - No effect 1b - Clears flag
10 —	Reserved
9-5 STEP_C	Step Of Algorithm C Indicates the step of self-test algorithm C in which an error occurred. Although this register field is read-only, no transfer error is generated when writing to it.
4-0 —	Reserved

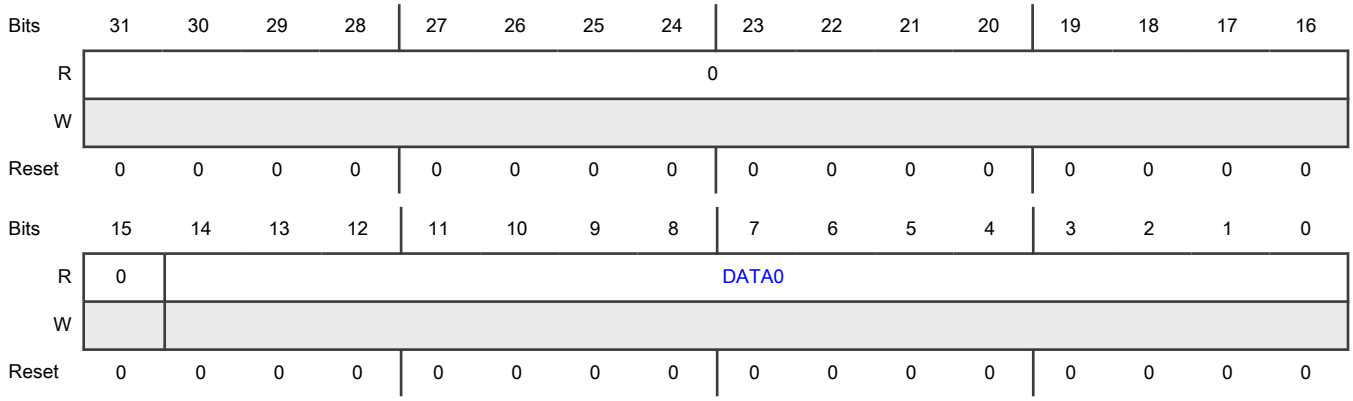
60.6.2.57 Self-Test Status 2 (STSR2)

Offset

Register	Offset
STSR2	354h

Function
Contains the conversion result when an error in step 1 of algorithm S has occurred.

Diagram



Fields

Field	Function
31-16 —	Reserved
15 —	Reserved
14-0 DATA0	Conversion Data ERR_S1 Contains conversion data from the moment when the error in step 1 of algorithm S has occurred.

60.6.2.58 Self-Test Status 3 (STSR3)

Offset

Register	Offset
STSR3	358h

Function

Contains the conversion result when an error has occurred in step 2 or step 0 of algorithm S.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	DATA1														
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	DATA0														
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30-16 DATA1	Conversion Data ERR_S2 Contains conversion data from the moment when the error in step 2 of algorithm S has occurred.
15 —	Reserved
14-0 DATA0	Conversion Data ERR_S0 Contains the conversion data from the moment when the error in step 0 of algorithm S has occurred.

60.6.2.59 Self-Test Status 4 (STSR4)

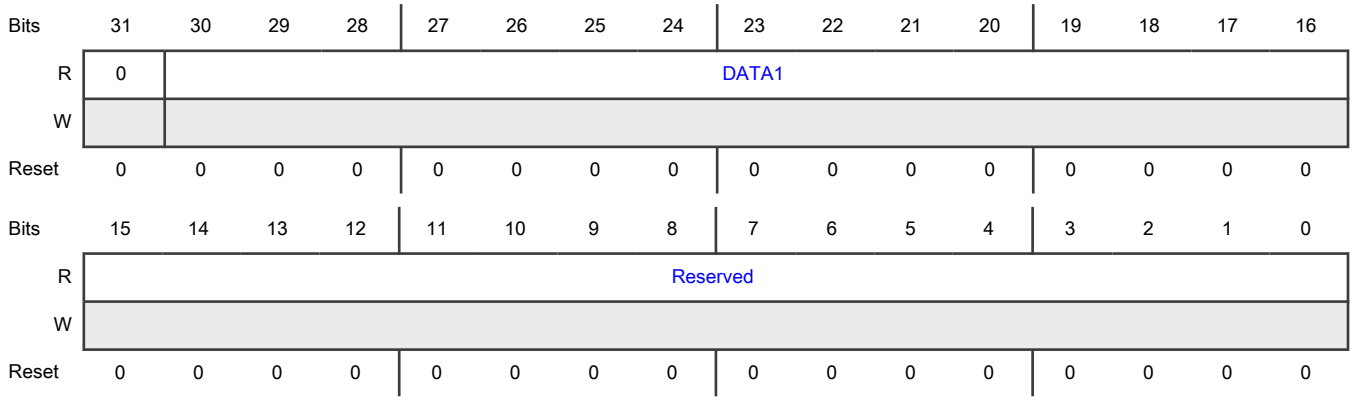
Offset

Register	Offset
STSR4	35Ch

Function

Contains the conversion result when an error has occurred in algorithm C.

Diagram



Fields

Field	Function
31 —	Reserved
30-16 DATA1	Conversion Data ERR_C Conversion data from the moment when the error in algorithm C has occurred.
15-0 —	Reserved

60.6.2.60 Self-Test Conversion Data 1 (STDR1)

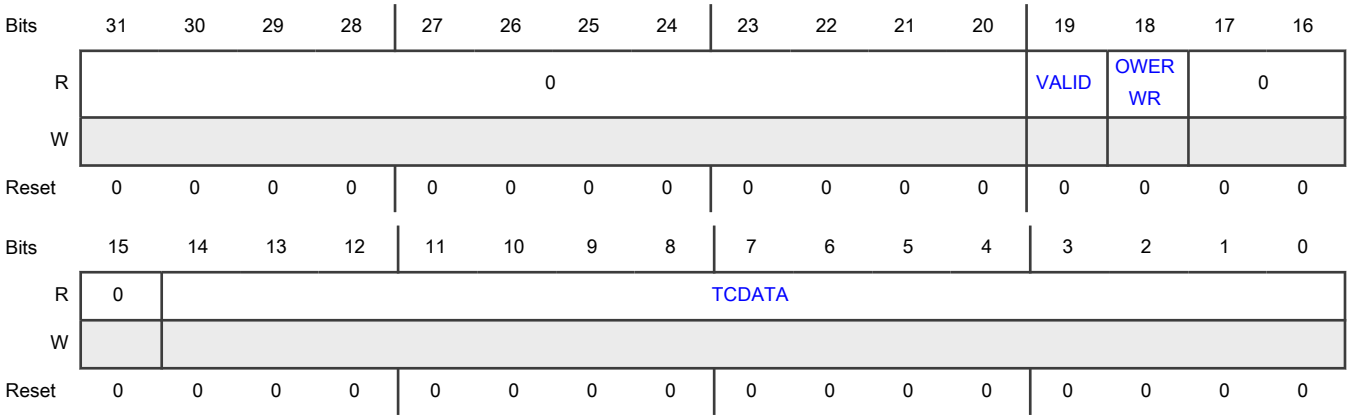
Offset

Register	Offset
STDR1	370h

Function

Contains the result of the self-test conversion.

Diagram



Fields

Field	Function
31-20 —	Reserved
19 VALID	Valid Conversion Data Indicates that conversion data is available in TCDATA. This field is automatically reset to 0 when the conversion data is read. 0b - No unread conversion data 1b - Unread conversion data is available
18 OWERWR	Conversion Data Overwrite Status Indicates whether the conversion data in TCDATA has been overwritten over previous data that had not been read. 0b - Current conversion data not overwritten 1b - Current conversion data was overwritten
17-16 —	Reserved
15 —	Reserved
14-0 TCDATA	Test Channel Conversion Data Contains the conversion data from self-test. If overwrite is enabled (MCR[OWREN]=1), the conversion data is from the last-executed self-test conversion. When overwrite is not enabled, conversion data is available (not overwritten) until it is read by software. The conversion data in this register is in two's complement format.

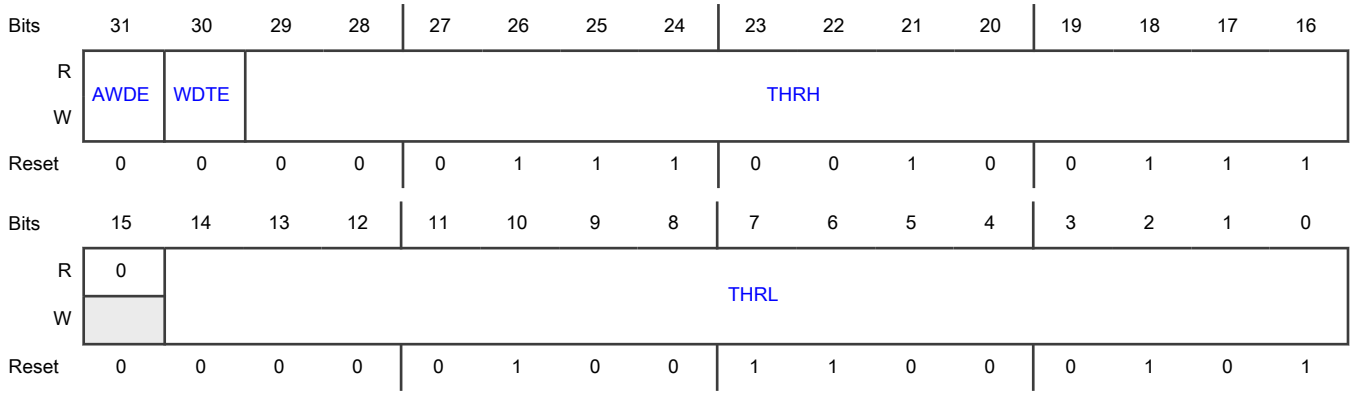
60.6.2.61 Self-Test Analog Watchdog S0 (STAW0R)

Offset

Register	Offset
STAW0R	380h

Function
Configures self-test for step 0 of algorithm S.

Diagram



Fields

Field	Function
31 AWDE	Self-Test Watchdog Enable Enables the self-test watchdog for step 0 of algorithm S. 0b - Disable 1b - Enable
30 WDTE	Self-Test Watchdog Timer Enable Enables the self-test watchdog timer for algorithm S. The self-test watchdog timer verifies: • Correct sequence of execution of the steps of the algorithm • Execution of the algorithm within the safe time period as defined by STBRR[WDT] This field must be 1 only in continuous conversion mode (MCR[MODE]=1). The safe time period starts when this field is written with 1. If the safe time period has elapsed before algorithm S starts, the WDTERR flag is set. The safe time period is restarted each time algorithm S starts. 0b - Disable 1b - Enable
29-16	Higher Threshold Value

Table continues on the next page...

Table continued from the previous page...

Field	Function
THRH	Sets the ERR_S0 flag when a self-test conversion value is greater than this value.
15 —	Reserved
14-0 THRL	Lower Threshold Value Sets the ERR_S0 flag when a self-test conversion value is less than this value.

60.6.2.62 Self-Test Analog Watchdog S1 (STAW1R)

Offset

Register	Offset
STAW1R	388h

Function
Configures self-test for step 1 of algorithm S.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	AWDE	0														
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	THRL														
W																
Reset	0	0	1	1	1	1	1	1	1	1	1	1	1	0	0	1

Fields

Field	Function
31 AWDE	Self-Test Watchdog Enable Enables the self-test watchdog for step 1 of algorithm S. 0b - Disable 1b - Enable
30-16	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
15 —	Reserved
14-0 THRL	Lower Threshold Value Sets the ERR_S1 flag when a self-test conversion value is less than this value.

60.6.2.63 Self-Test Analog Watchdog S2 (STAW2R)

Offset

Register	Offset
STAW2R	38Ch

Function

Configures self-test for step 2 of algorithm S.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	AWDE	0														
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	THRL														
W																
Reset	0	0	1	1	1	1	1	1	1	1	1	1	1	0	0	1

Fields

Field	Function
31 AWDE	Self-Test Watchdog Enable Enables the self-test watchdog for step 2 of algorithm S. 0b - Disable 1b - Enable
30-16	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
15 —	Reserved
14-0 THRL	Lower Threshold Value Sets the ERR_S2 flag when a self-test conversion value is less than this value.

60.6.2.64 Self-Test Analog Watchdog C0 (STAW4R)

Offset

Register	Offset
STAW4R	394h

Function

Configures self-test for step 0 of algorithm C.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	AWDE	WDTE	THRH													
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	THRL														
W																
Reset	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0

Fields

Field	Function
31 AWDE	Self-Test Watchdog Enable Enables the self-test watchdog for algorithm C. 0b - Disable 1b - Enable
30	Self-Test Watchdog Timer Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
WDTE	Enables the self-test watchdog timer for algorithm C. The self-test watchdog timer verifies: • Correct sequence of execution of algorithm steps • Execution of the algorithm within the safe time period as defined by STBRR[WDT] This field must be 1 only in continuous conversion mode (MODE 1). The safe time period starts when this field is written with 1. If the safe time period elapses before algorithm C starts, the WDTERR flag is set. The safe time period is restarted each time algorithm C starts. 0b - Disable 1b - Enable
29-16 THRH	Higher Threshold Value Sets the ERR_C flag when a self-test conversion value is greater than this value. This value must be in two's complement format and must be positive. This value is valid only for step 0 of algorithm C (see STAW5R for the values used for the other steps of algorithm C).
15 —	Reserved
14-0 THRL	Lower Threshold Value Sets the ERR_C flag when a self-test conversion value is less than this value. This value must be in two's complement format and must be negative. (With the reset value of this bit field, the test always fails. You must set at least STAW4R[14]=1, in order to have a negative value in this bit field.) This value is used only for step 0 of algorithm C (see STAW5R for the values used for the other steps of algorithm C).

60.6.2.65 Self-Test Analog Watchdog C (STAW5R)

Offset

Register	Offset
STAW5R	398h

Function
Configures self-test for steps 1–11 of algorithm C.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	THRH														
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	THRL														
W																
Reset	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0

Fields

Field	Function
31 —	Reserved
30-16 THRH	Higher Threshold Value Sets the ERR_C flag when a self-test conversion value is greater than this value. This value must be entered in two's complement format and must be positive. This value is valid only for steps 1–11 of algorithm C (see STAW4R for the value used for step 0 of algorithm C).
15 —	Reserved
14-0 THRL	Lower Threshold Value Sets the ERR_C flag when a self-test conversion value is less than this value. This value must be in two's complement format and must be negative. This value is used only for steps 1–11 of algorithm C (see STAW4R for the value used for step 0 of algorithm C).

60.6.2.66 Analog Miscellaneous In/Out register (AMSIO)

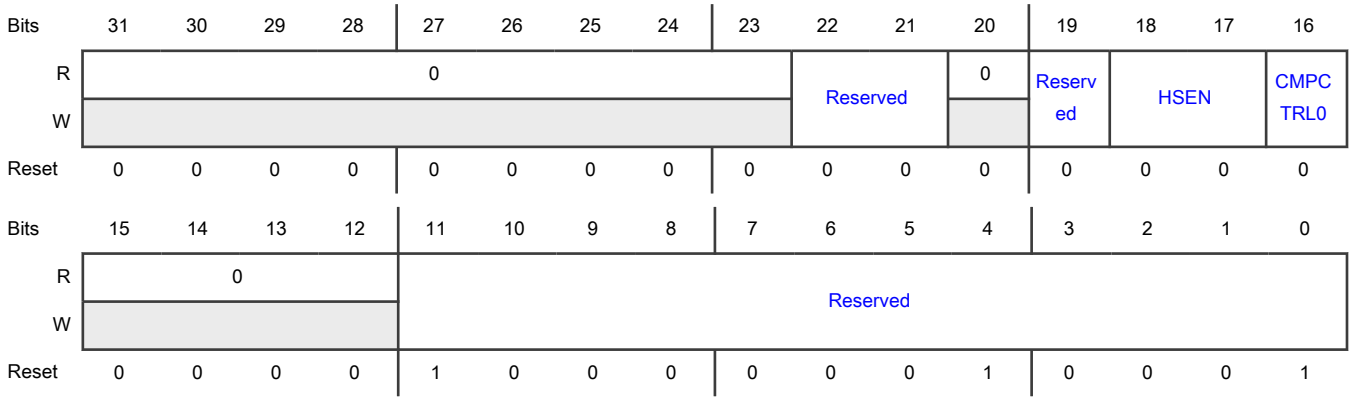
Offset

Register	Offset
AMSIO	39Ch

Function

Configures the SAR algorithm compare step. In case you want to do a high-speed calibration or a high-speed conversion see Table 345. All other values in this register must stay at their reset value and may not be written.

Diagram



Fields

Field	Function
31-23 —	Reserved
22-21 —	Reserved
20 —	Reserved
19 —	Reserved
18-17 HSEN	High-Speed Enable This setting must be adapted to the ADC clock frequency. See Table 345.
16 CMPCTRL0	Compare Control 0 This setting must be adapted to the ADC clock frequency. See Table 345.
15-12 —	Reserved
11-0 —	Reserved

60.6.2.67 Control And Calibration Status (CALBISTREG)

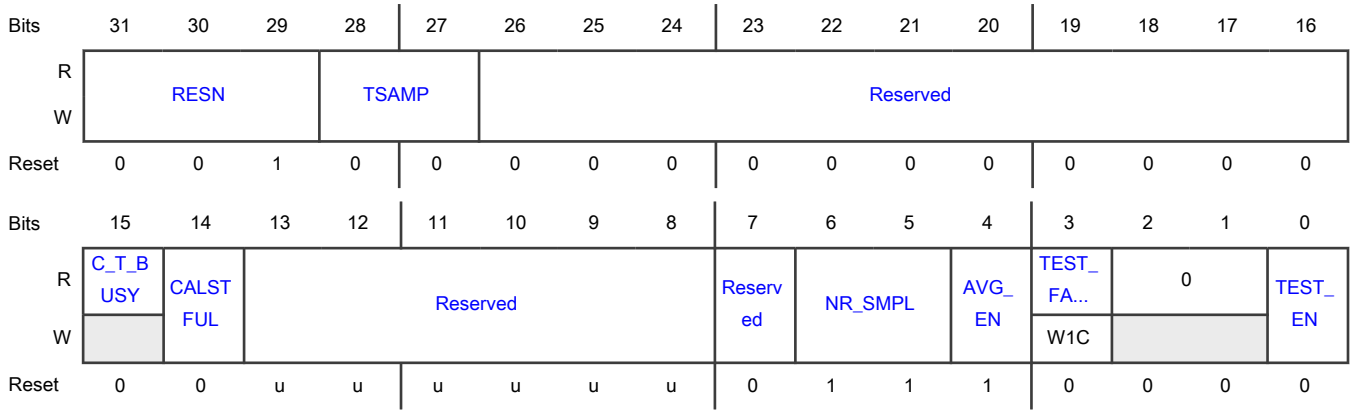
Offset

Register	Offset
CALBISTREG	3A0h

Function

Controls several ADC functions for data conversion, calibration, and calibration status.

Diagram



Fields

Field	Function
31-29 RESN	Conversion Resolution Specifies the number of significant bits per conversion data (functional conversion only, not for calibration or self-test). Reducing this number speeds up the conversion because the SAR algorithm executes fewer steps. This field must be modified only when ADC is idle. 000b - 14-bit resolution 001b - 12-bit resolution 010b - 10-bit resolution 011b - 8-bit resolution
28-27 TSAMP	Sample Period In Calibration Specifies the number of conversion clock cycles during which the reference voltage is sampled. 00b - 22 conversion clock cycles 01b - 8 conversion clock cycles 10b - 16 conversion clock cycles 11b - 32 conversion clock cycles

Table continues on the next page...

Table continued from the previous page...

Field	Function
26-16 —	Reserved The value of this field must not be modified.
15 C_T_BUSY	Calibration Busy Indicates whether calibration is running. 0b - Calibration can be started 1b - Calibration is in progress
14 CALSTFUL	Calibration And Self-Test Full Range Comparison Specifies the range of conversion data bits compared when ADC is calibrated or running self-test. Because the results from calibration and self-test are so low that the significant bits are in the lower 11 bits only, the comparison cycles for the upper 4 bits can be skipped to reduce the execution duration of the calibration and of the self-test. 0b - Lowest 11 bits are compared. 1b - All 15 bits are compared.
13-8 —	Reserved **CAUTION** This field contains a value that ADC uses internally. Writing to this field can cause ADC to operate unreliably.
7 —	Reserved This field is reserved and always has the value 0. Only the reset value can be written to this field.
6-5 NR_SMPL	Calibration Averaging Number Specifies the number of data conversions per result (averaging) during calibration. 00b - 4 samples 01b - 8 samples 10b - 16 samples 11b - 32 samples
4 AVG_EN	Calibration Averaging Enable Enables averaging of the calibration results over several conversion iterations to improve accuracy. This field only affects the calibration process. 0b - Disable 1b - Enable
3 TEST_FAIL	Calibration Status Indicates calibration status. This field reads 0 when calibration is in progress. This field is reset to 0 when calibration starts.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field behaves differently for register reads and writes. When reading 0b - Calibration finished successfully or has not been run since the last reset 1b - Calibration did not finish successfully When writing 0b - No effect 1b - Clears flag
2-1 —	Reserved
0 TEST_EN	Calibration Enable Calibrates ADC. This field automatically resets to 0 after the calibration is complete or is interrupted. 0b - Wait to start a calibration 1b - Start calibration

60.6.2.68 Offset And Gain User (OFSGNUSR)

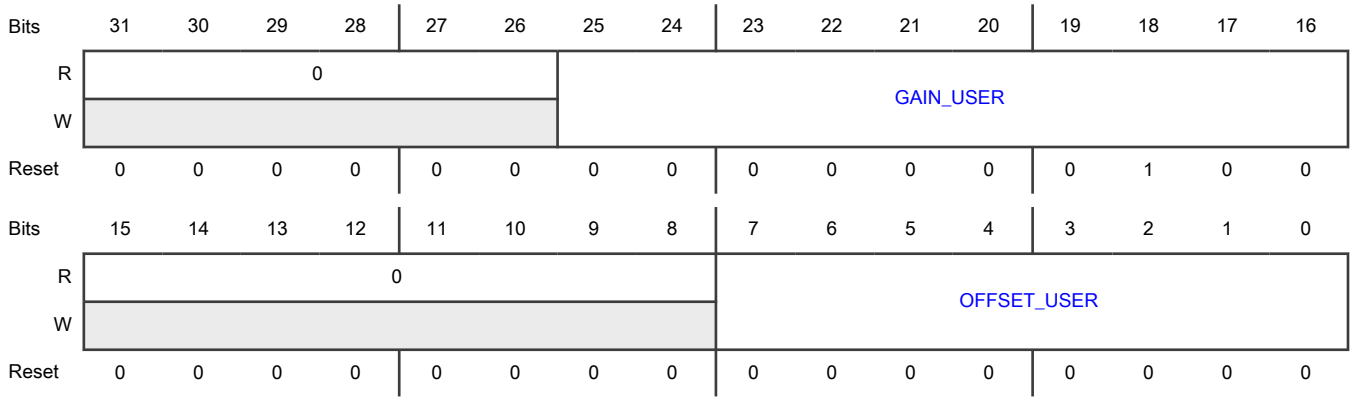
Offset

Register	Offset
OFSGNUSR	3A8h

Function

Specifies the user-configured offset and gain values used by the SAR algorithm. The values must be written in two's complement format and can be either positive or negative.

Diagram



Fields

Field	Function
31-26 —	Reserved
25-16 GAIN_USER	Gain User Adds extra gain to the gain calculated during calibration. The value must be in two's complement format.
15-8 —	Reserved
7-0 OFFSET_USER	Offset User Adds extra offset to the offset calculated by calibration. The value is must be in two's complement format.

60.6.2.69 Calibration Value 2 (CAL2)

Offset

Register	Offset
CAL2	3B4h

Function
Contains fields used during calibration.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved								Reserved	Reserved						
Reset	0	1	0	0	0	0	1	1	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ENX	0			Reserved											
Reset	1	0	0	0	0	0	1	0	0	1	0	0	0	0	1	1

Fields

Field	Function
31-24 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
23 —	Reserved
22-16 —	Reserved
15 ENX	Enable X Enables the inclusion of CLPX in error processing. 0b - Disable 1b - Enable
14-12 —	Reserved
11-0 —	Reserved

60.7 Glossary

CDAC	Capacitive digital to analog converter
CT	Compare phase time
DMA	Direct memory access
DP	Data processing time
PST	Presample phase time
ST	Sample phase time
SAR	Successive approximation
TPT	Trigger processing time
VrefH	Reference voltage high
VrefL	Reference voltage low

Chapter 61
Low Power Comparator (LPCMP)

61.1 Chip-specific LPCMP information

61.1.1 Instantiation information

Table 351. LPCMP instances

Chip	Instance	No. of external inputs
S32K388/S32K389/S32K358/S32K356/ S32K348/S32K338/S32K328/S32K344/ S32K324/S32K314	LPCMP_0	8
	LPCMP_1	4
	LPCMP_2	4
S32K312/S32K342/S32K322/S32K341	LPCMP_0	8
	LPCMP_1	4
S32K311/S32K310	LPCMP_0	8

61.1.2 LPCMP input output connections

See the IOMUX file attached to this document for pin/pad assignments corresponding to CMP pins.

The LPCMP channels can be used as a wakeup source in trigger mode to wakeup the device from standby mode.

61.1.3 LPCMP-DAC "vrefh0" and "vrefh1" references

The 8-bit DAC sub-block supports selection of "vrefh0" and "vrefh1" by CMPx_DCR[VRSEL]. For this device, the references are connected as follows:

- vrefh0 (External Reference): VDD_HV_A
- vrefh1 (Internal Reference): 1.2 V PMC bandgap reference

NOTE

1.2 V internal reference voltage is not available in Standby mode.

61.1.4 LPCMP window control

The window mode operation of all the comparator instances in the chip can be enabled/disabled by the TRGMUX. See the TRGMUX connectivity file attached to this document for details.

NOTE

Window signal must be of minimum 4 cycle pulse for window mode to function.

61.1.5 Comparator Trigger Mode

The comparator modules in the device support trigger mode operation as described in 'Trigger Mode' section. Device can operate in trigger mode in both standby and run mode to continuously scan the input channels. The main features of the device trigger mode operation are:

- Round robin clock: RTC_CLK
- Round robin trigger source: RTC_API

NOTE

- It must be ensured that the RTC_CLK period is greater than the comparison time corresponding to the value of C0[PMODE]. It is also required to **not** select the internal reserved channels, if available on the package, for trigger mode operation by INPSEL and INNSEL. See the IOMUX file attached to this document for the pins available in various packages supported for the device.
- Only SIRC and SXOSC are supported as RTC_CLK for trigger mode operation. The LPCMP initialization delay is not supported by RRCR0[RR_INITMOD] with FXOSC or FIRC as RTC_CLK.
- In run mode, the generated trigger is delayed by 3 RTC_CLK cycles before being given to comparator trigger input.
- When used in Round Robin mode, the LPCMP outputs the enable signal of the comparator on CMPx_RRT pin, on each cycle of comparator circuit. See the IOMUX file attached to this document for the pins on which CMPx_RRT functions are available.

61.1.6 Interaction with RTC API to cause wakeup

LPCMP can be used for waking up the chip from standby. For this, RTC-API and LPCMP must be configured before entering into standby mode as per below shown figure.

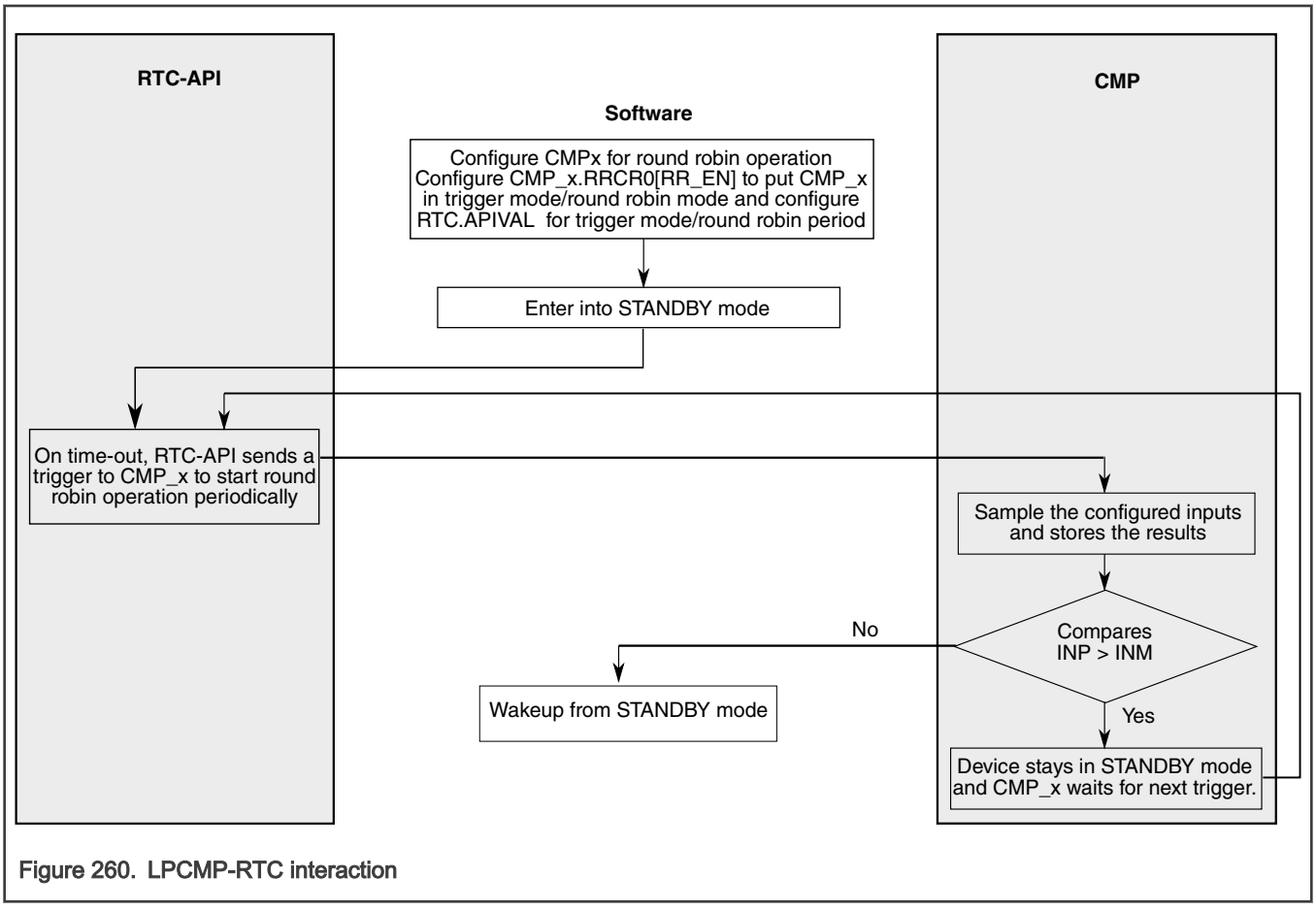


Figure 260. LPCMP-RTC interaction

In the trigger mode operation, only the continuous mode is supported. None of the window/filter/sample functions should be used. Refer to the "Functional Modes" section for details on comparator modes of operation.

Register configurations before entering Standby mode for LPCMP trigger mode operation:

1. Configure RTC.APIVAL to set the period of the round robin operation.
2. Execute standby mode entry.

61.2 Overview

The LPCMP module provides a circuit to compare two analog input voltages. It includes the following:

- A low power comparator (CMP)
- A DAC
- An analog mux (ANMUX)

See Block diagram for more information.

LPCMP can operate across the full range of the supply voltage, known as rail-to-rail operation.

DAC is a 256-tap resistor ladder network that provides a selectable voltage reference for applications requiring a voltage reference. DAC divides the supply reference V_{in} into 256 voltage levels. An 8-bit digital signal input selects the output voltage level, which varies from V_{in} to $V_{in}/256$.

You can select V_{in} from the following voltage sources:

- VREFH0
- VREFH1

See the Chip-specific LPCMP information for more information on source of VREFH0 and VREFH1.

NOTE

The LPCMP's internal DAC output is available as an on-chip internal signal only and is not available for an external chip pin.

ANMUX allows you to select an analog input signal from among eight channel options. One channel option is the DAC output. Other chip resources are connected to the other channels. See the Chip-specific LPCMP information section for more information. ANMUX can operate across the full range of the supply voltage.

61.2.1 Block diagram

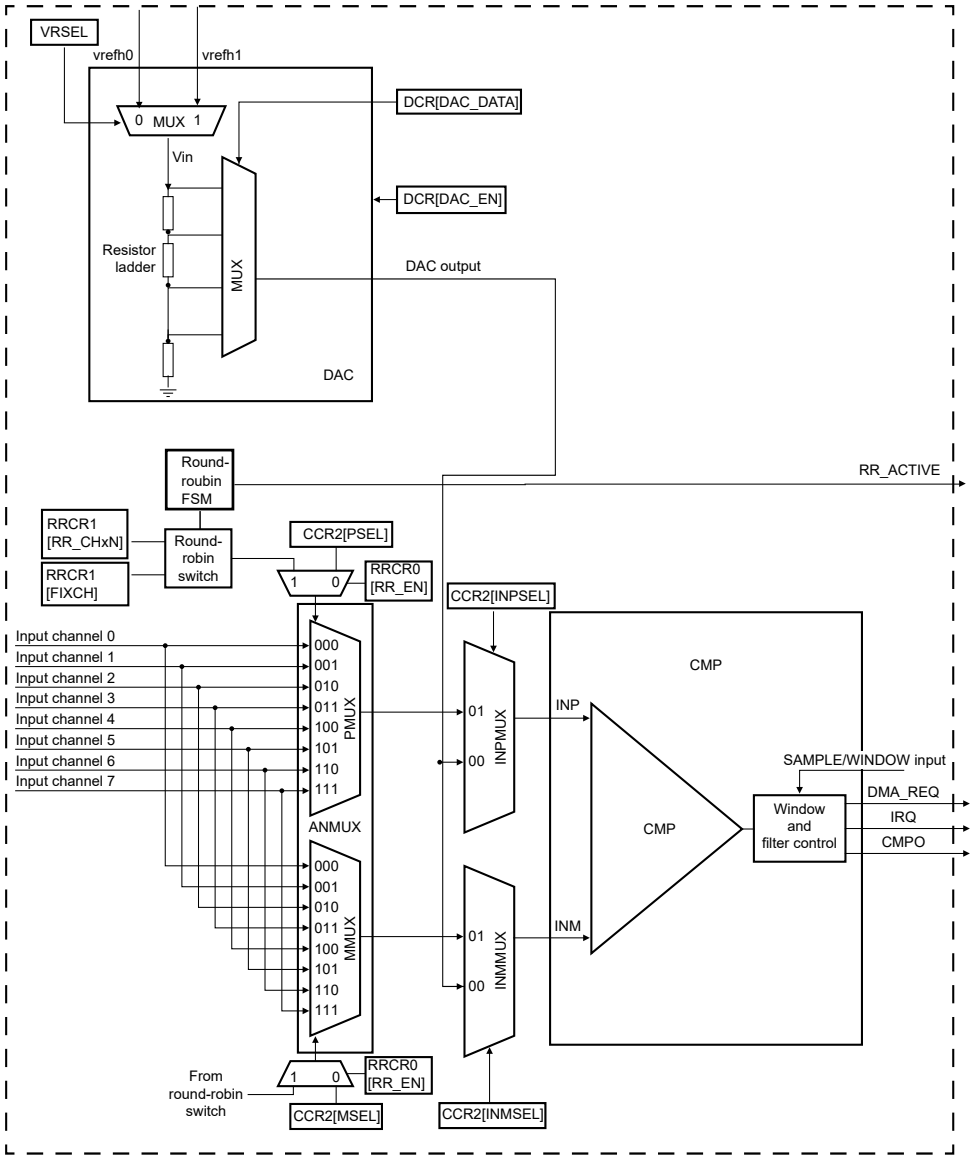


Figure 261. Block diagram

61.2.2 Features

The features of the LPCMP module include：

- Includes two 8-to-1 channel MUXes to select input signal from eight channels
- Supports multiple operation modes to produce a wide range of outputs such as:
  - Sampled
  - Windowed, which is ideal for certain PWM zero-crossing-detection applications
  - Digitally filtered
- Provides the following advance features for window and sample:

- Window and sample signals can be inverted.
- CMPO rising, falling or both edges closes the window.
- CMPO level can be defined when window is closed.

- Provides selectable performance levels:
  - Low-Power (speed) mode
  - High-Power (speed) mode
- Supports programmable hysteresis control
- Provides a selectable inversion on comparator output
- Uses an external hysteresis at the same time the output filter is used for internal functions
- Provides interrupt and DMA support
- Supports Round Robin Trigger mode
- Includes an 8-bit resolution DAC
- Provides a selectable supply reference source for DAC

61.3 Functional description

61.3.1 Functional block diagram

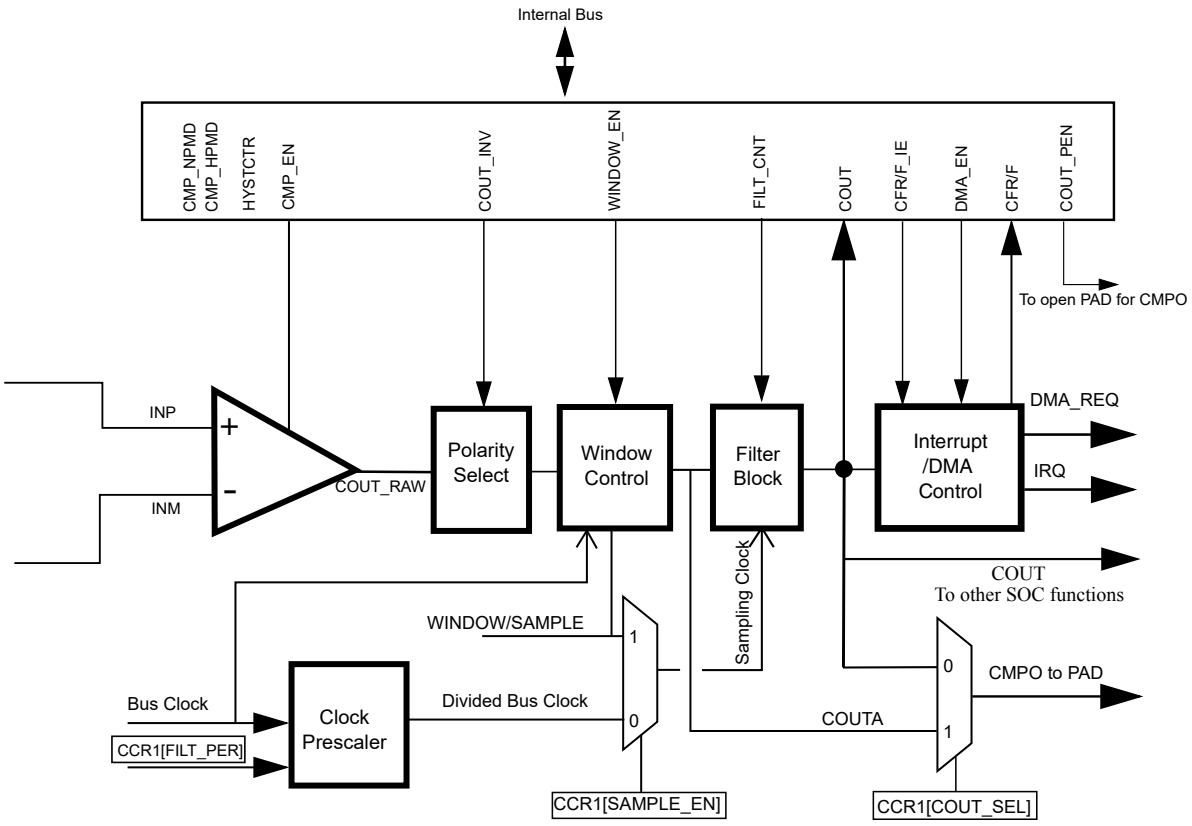


Figure 262. Functional block diagram

As shown in the block diagram, the functions are:

- Compared two analog input voltages applied to INP and INM, COUT_RAW is high when the INP input voltage is greater than the INM input voltage, and COUT_RAW is low when the INP input voltage is less than the INM input voltage.
- The COUT_RAW signal can be inverted by enabling CCR1[COUT_INV].
- The optionally inverted comparator output COUT_RAW is sampled on every bus clock when you enable the CCR1[WINDOW_EN] to generate COUTA. In this case, the comparator output is ignored during time periods when the input voltages are not valid. This is useful when you implement zero-crossing-detection for certain PWM applications.
- The window control block is bypassed when CCR1[WINDOW_EN] is disabled.
- The filter block acts as a simple sampler when CCR1[FILT_CNT] is set to 01h.
- The filter block acts as a filter based on multiple samples when CCR1[FILT_CNT] is set to be greater than 01h.
  - If CCR1[SAMPLE_EN] is set to 1, use the external SAMPLE input as the sampling clock.
  - If CCR1[SAMPLE_EN] is set to 0, use the divided bus clock as the sampling clock.
- Bypasses the filter block when it is not in use.

```
Bypass_Filter_Block = (FILT_CNT == 0x00) |  (~SAMPLE_EN & (FILT_PER == 0x00))
```

- Both COUTA and COUT can be configured as module output CMPO by configuring CCR1[COUT_SEL], and are used for different purposes within the system.
- The optionally filtered COUT can be read directly in CSR[COUT].
- The SAMPLE/WINDOW signal can be inverted by setting CCR1[WINDOW_INV].
- The SAMPLE/WINDOW signal can be closed by CMPO's falling edge and/or rising edge by setting CCR1[WINDOW_CLS] in Window mode.
- In Window mode, when window is closed, define the COUTA value as CCR1[COUTA_OW] by setting CCR1[COUTA_OWEN]. If CCR1[COUTA_OWEN] is not set, COUTA holds the last sampled value.

NOTE

See the chip configuration section for the source of SAMPLE/WINDOW input.

61.3.2 Round-robin trigger mode

You can enable Round-Robin Trigger mode by setting RRCR0[RR_EN] and CCR0[CMP_EN] to 1. A trigger event initiates a comparison sequence. The next trigger event should not occur before the current sequence completes.

RRCR1[FIXP] and RRCR1[FIXCH] select the reference channel for the plus side mux or the minus side mux. RRCR1[RR_CHnEN] selects active channels.

When a trigger comes, the analog comparator enables. After the comparison sequence completes, the analog comparator disables again. RRCR0[RR_INITMOD] controls the analog stabilization time.

NOTE

RR_INITMOD*round robin clock period must be longer than the initialization delay specified in the Comparator and 8-bit DAC electrical specifications section of LPCMP datasheet.

After the stabilization process completes, the round robin manner comparison sequence begins. Sample the comparison result for the selected active channel after RRCR0[RR_NSAM] defines the configurable number of operation clocks.

After all the active channels are sampled/compared, if the comparison result changes from its pre-programmed state, the corresponding flag in RRSR[RR_CHnF] is set. Write to RRCSR[RR_CHnOUT] to configure the pre-programmed state for each channel. Update RRCSR[RR_CHnOUT] to store the last comparison result for each channel. If any flag in RRSR[RR_CHnF] sets, CSR[RRF] also sets. If IER[RRF_IE] sets, an asynchronous interrupt asserts. Note that these flags do not support generating a DMA transfer event.

The following diagram shows the basic flow of this mode. In the diagram, RRCR1[RR_CH1EN], RRCR1[RR_CH3EN], and RRCR1[RR_CH4EN] are 1, so channels #1, #3, and #4 are selected for round-robin depending on their priority setting. RRCR0[RR_NSAM] sets to 2'b01, so you can sample one clock later the comparison result of the selected channel. After you compare the channel #4, the result is sampled, and round-robin ends. If any of the comparison results from channel #1, #3, or #4 changed from their programmed value (written to RRCSR[RR_CH1OUT], RRCSR[RR_CH3OUT], and RRCSR[RR_CH4OUT]), generates an interrupt. Software can then poll RRSR[RR_CHnF] to see which channel input(s) changed value.

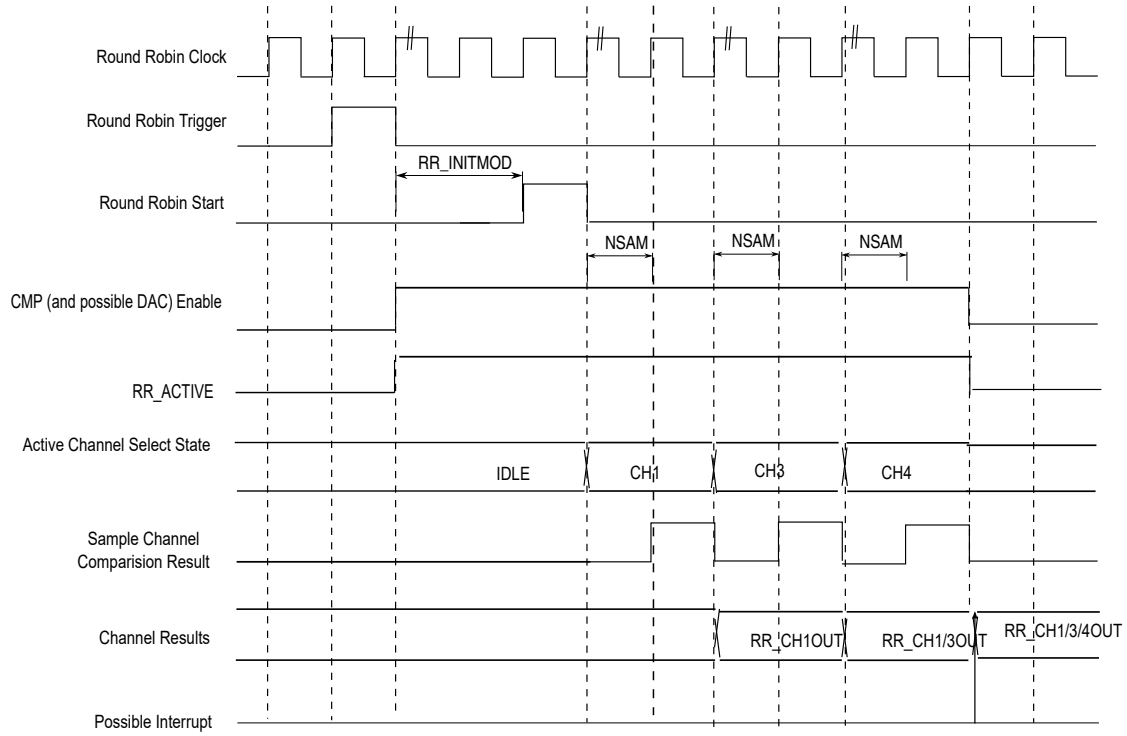


Figure 263. Trigger mode

The table below shows the channel decode in both Functional mode and Trigger mode. Other cases not in the table are illegal.

Table 352. CMP channel decode in functional mode and round-robin trigger mode

Mode	RR_EN	PSEL[2:0]	MSEL[2:0]	INPSEL[1:0]	INMSEL[1:0]	FIXP	FIXCH[2:0]	RR_CHxN	INP	INM	CMP Behavior
Functional mode	0	x[1]	0 to 7	0	1	x	x	x	DAC	Channel decoded from MSEL[2:0]	Channel 0 to 7 can be compared with DAC

Table continues on the next page...

Table 352. CMP channel decode in functional mode and round-robin trigger mode (continued)

Mode	RR_EN	PSEL[2:0]	MSEL[2:0]	INPSEL[1:0]	INMSEL[1:0]	FIXP	FIXCH[2:0]	RR_CHxN	INP	INM	CMP Behavior
		0 to 7	x	1	0	x	x	x	Channel decoded from PSEL[2:0]	DAC	Channel 0 to 7 can be compared with DAC
		0 to 7	0 to 7	1	1	x	x	x	Channel decoded from PSEL[2:0]	Channel decoded from MSEL[2:0]	Channel 0 to 7 can be compared with channel 0 to 7[2]
Trigger mode	1	x	x	0	1	0	x	0 to 7	DAC	Channel sweep (RR_CHxN)	Channel 0 to 7 can be swept with DAC
		x	x	1	0	1	x	0 to 7	Channel sweep (RR_CHxN)	DAC	Channel 0 to 7 can be swept with DAC
		x	x	1	1	0	0 to 7	0 to 7	Channel fixed by FIXCH[2:0]	Channel sweep (RR_CHxN)	Channel 0 to 7 can be swept with a fixed channel(0 to 7)[3]
		x	x	1	1	1	0 to 7	0 to 7	Channel sweep (RR_CHxN)	Channel fixed by FIXCH[2:0]	Channel 0 to 7 can be swept with a fixed channel(0 to 7)[3]

1. "x" means "don't care"
2. PSEL should not be same as MSEL.
3. Channel in the sweep side should not be same as the fixed side.

61.3.3 Low-pass filter mode

The low-pass filter mode operates on an unfiltered, optionally inverted comparator output COUTA, and generates the filtered and synchronized output COUT. You can configure both COUTA and COUT as module outputs and use for different purposes within the system.

Synchronization and edge detection determine the bit values of status register. They also apply to COUT for all sampling and windowed modes. You can perform filtering using an internal timebase defined by CCR1[FILT_PER], or use an external sample input to determine sample time.

The need for digital filtering and the amount of filtering depends on your requirements. Filtering can become more useful in the absence of an external hysteresis circuit. Without external hysteresis, generate a high-frequency oscillations at COUTA when the selected INM and INP input voltages differ by less than the offset voltage of the differential comparator.

61.3.3.1 Enabling low-pass filter mode

You can enable low-pass filter mode by setting the following:

- CCR1[FILT_CNT] > 01h
- CCR1[FILT_PER] to a nonzero value or writing 1 to CCR1[SAMPLE_EN].

If you use the divided bus clock to drive the low-pass filter, it samples COUTA every CCR1[FILT_PER] bus clock cycle.

If CCR1[SAMPLE_EN] is set to 1, the low-pass filter samples COUTA on each positive transition of the sample input. The output state of the filter changes when all the consecutive CCR1[FILT_CNT] samples agree that the output value has changed.

61.3.3.2 Latency issues

Program the value of CCR1[FILT_PER] or sample period such that the sampling period is longer than the period of the expected noise, ensuring that a given noise spike corrupts only one sample. You must choose the value of CCR1[FILT_CNT] to reduce the probability of noisy samples causing an incorrect transition to recognize. The probability of an incorrect transition is defined as the probability of an incorrect sample raised to the power of CCR1[FILT_CNT].

You must trade off the values of CCR1[FILT_PER] or sample period and CCR1[FILT_CNT] against the need for minimal latency in recognizing actual comparator output transitions. The probability of detecting an actual output change within the nominal latency is the probability of a correct sample raised to the power of CCR1[FILT_CNT].

Table 354 summarizes maximum latency values for the various modes of operation in the absence of noise. Filtering latency restarts each time the noise masks an actual output transition.

61.3.4 Low power mode operation

Below table introduces the mode of operation of lower power.

Table 353. Low power mode operation

Mode of operation	Description
STOP	LPCMP can operate only in Continuous mode or Round-robin trigger mode.

61.3.5 Functional modes

You can combine the comparator window and filter features as shown in the following table.

Table 354. Functional modes

Mode #	CMP_EN	WINDOW_EN	SAMPLE_EN	FILT_CNT	FILT_PER	Operation	Maximum latency[1]
1	0	X	X	X	X	See the Disabled mode (#1).	N/A
2A	1	0	X	0x00	X	See the Continuous mode (#2A and #2B).	T_{PD}
2B	1	0	0	X	0x00		
3A	1	0	1	0x01	X	See the Sampled, non-filtered mode (#3A and #3B).	$T_{PD} + T_{SAMPLE} + 3T_{per}$
3B	1	0	0	0x01	> 0x00		$T_{PD} + (FILT_PER * T_{per}) + 3T_{per}$

Table continues on the next page...

Table 354. Functional modes (continued)

Mode #	CMP_EN	WINDOW_EN	SAMPLE_EN	FILT_CNT	FILT_PER	Operation	Maximum latency[1]
4A	1	0	1	> 0x01	X	See the Sampled, filtered mode (#4A and #4B).	T_{PD} + (FILT_CNT * T_{SAMPLE}) + $3T_{per}$
4B	1	0	0	> 0x01	> 0x00		T_{PD} + (FILT_CNT * FILT_PER x T_{per}) + $3T_{per}$
5A	1	1	0	0x00	X	See the Windowed mode (#5A and #5B).	T_{PD} + $2T_{per}$
5B	1	1	0	X	0x00		
6	1	1	0	0x01	> 0x00	See the Windowed/ Resampled mode (#6).	T_{PD} + (FILT_PER * T_{per}) + $3T_{per}$
7	1	1	0	> 0x01	> 0x00	See the Windowed/ Filtered mode (#7).	T_{PD} + (FILT_CNT * FILT_PER x T_{per}) + $3T_{per}$
All other combinations of CMP_EN, WINDOW_EN, SAMPLE_EN, FILT_CNT, and FILT_PER are illegal.							

1. T_{PD} represents the intrinsic delay of the analog component plus the polarity select logic. T_{SAMPLE} is the clock period of the external sample clock. T_{per} is the period of the bus clock.

61.3.5.1 Disabled mode (#1)

In this mode:

- The analog comparator is non-functional and consumes no power.
- CSR[COUT] and CMPO are the same as CCR1[COUT_INV].

61.3.5.2 Continuous mode (#2A and #2B)

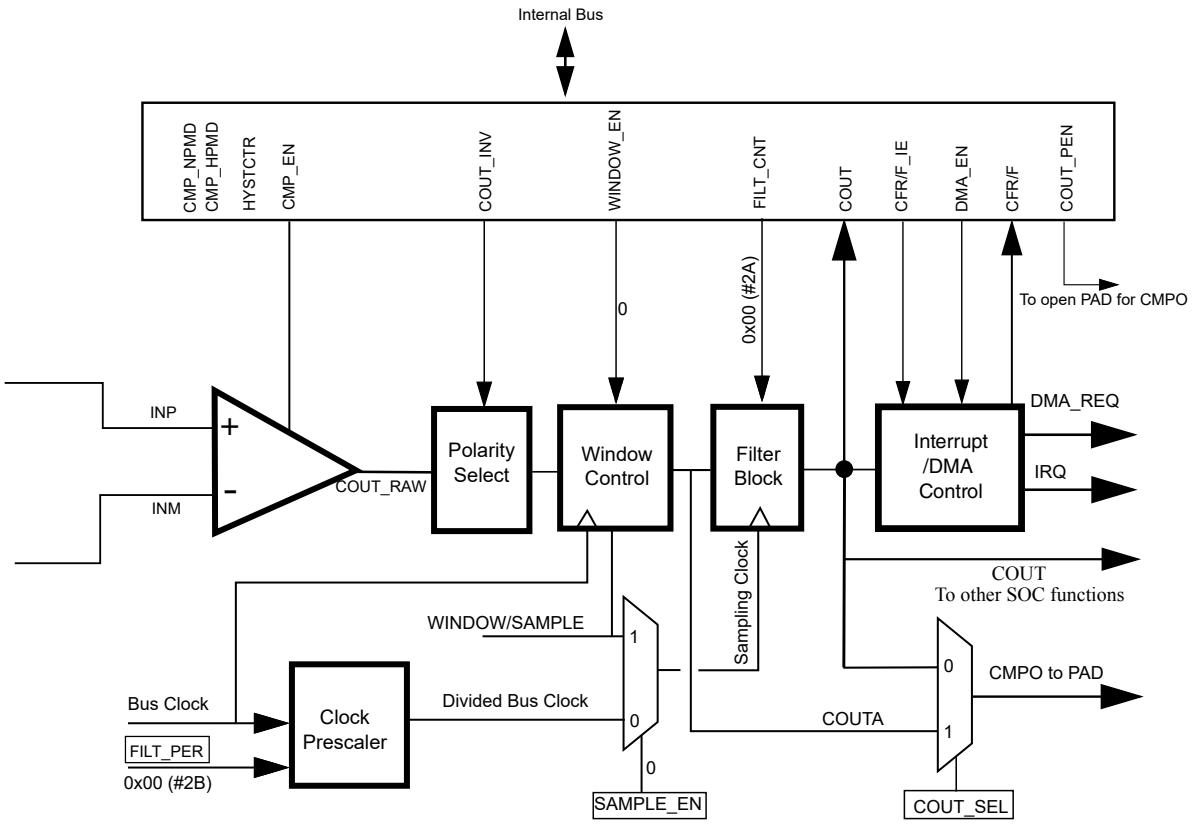


Figure 264. Comparator operation in continuous mode

COUT_RAW is optionally inverted in this mode but is not subject to external sampling or filtering. Both window control and filter blocks bypass completely, and CSR[COUT] updates continuously. The path from comparator input pins to output pins operates in a combinational (unclocked) mode. COUT and COUTA are identical in this mode.

For cases where a comparator drives a fault input, you must configure it to operate in Continuous mode so that an external fault can immediately pass to the target fault circuitry through the comparator.

61.3.5.3 Sampled, non-filtered mode (#3A and #3B)

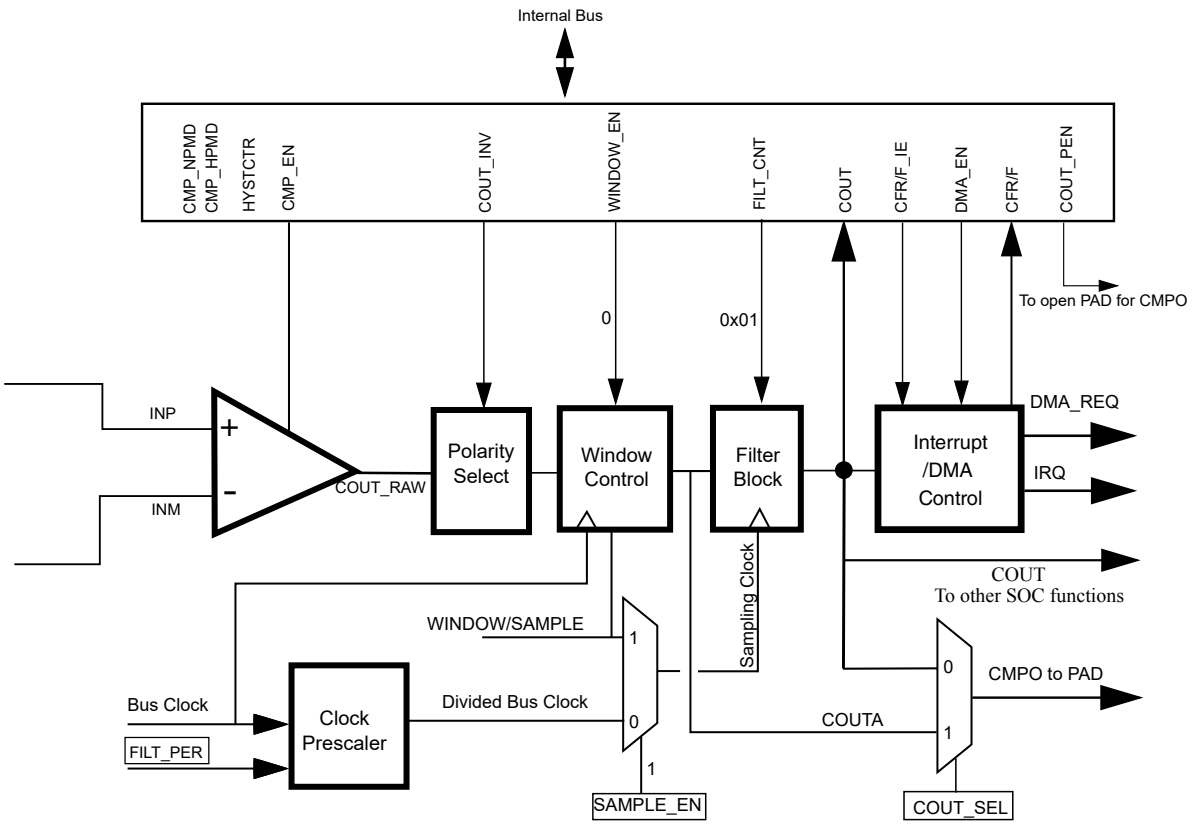


Figure 265. Sampled, non-filtered (#3A): sampling point externally derived

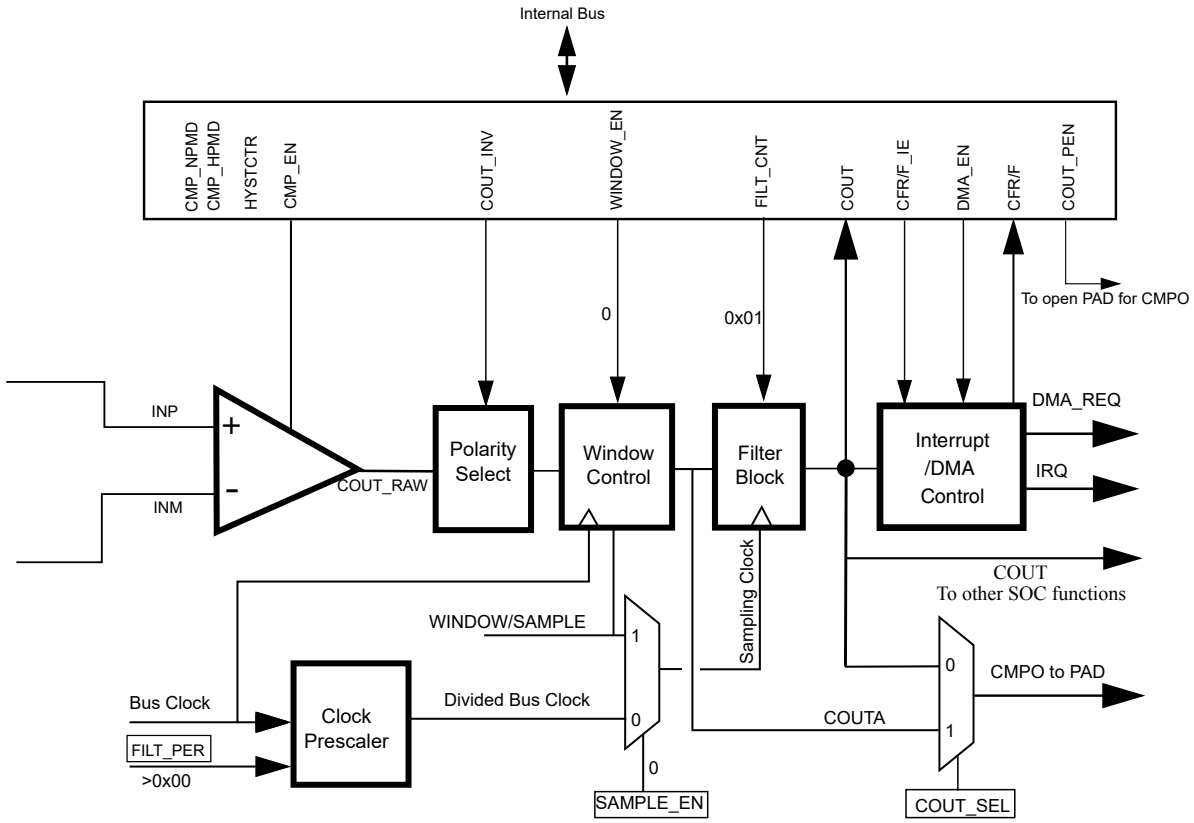


Figure 266. Sampled, Non-Filtered (#3B): sampling interval internally derived

In this mode, the path from analog inputs to COUTA is combinational (unclocked). Windowing control bypasses completely. You can sample COUTA whenever you detect a rising edge on the sampling clock.

The difference in two operation modes (#3A and #3B) of sampled, non-filtered mode is that how you drive the clock to the filter block. In #3A, the clock to filter block drives externally, and in #3B, the clock to filter block drives internally.

The filter block has no other function than sample or hold of the comparator output in this mode.

The following figure shows the comparator operation in this mode, assuming that the polarity select sets to a non-inverting state.

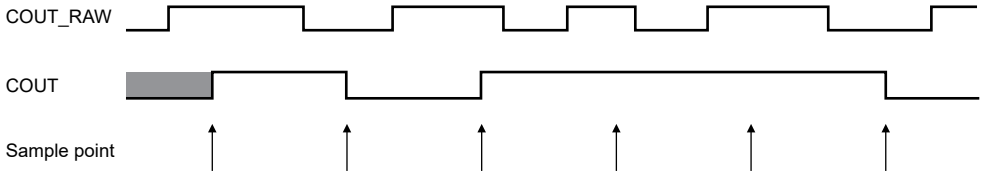


Figure 267. Sampled, Non-Filtered mode timing diagram

61.3.5.4 Sampled, filtered mode (#4A and #4B)

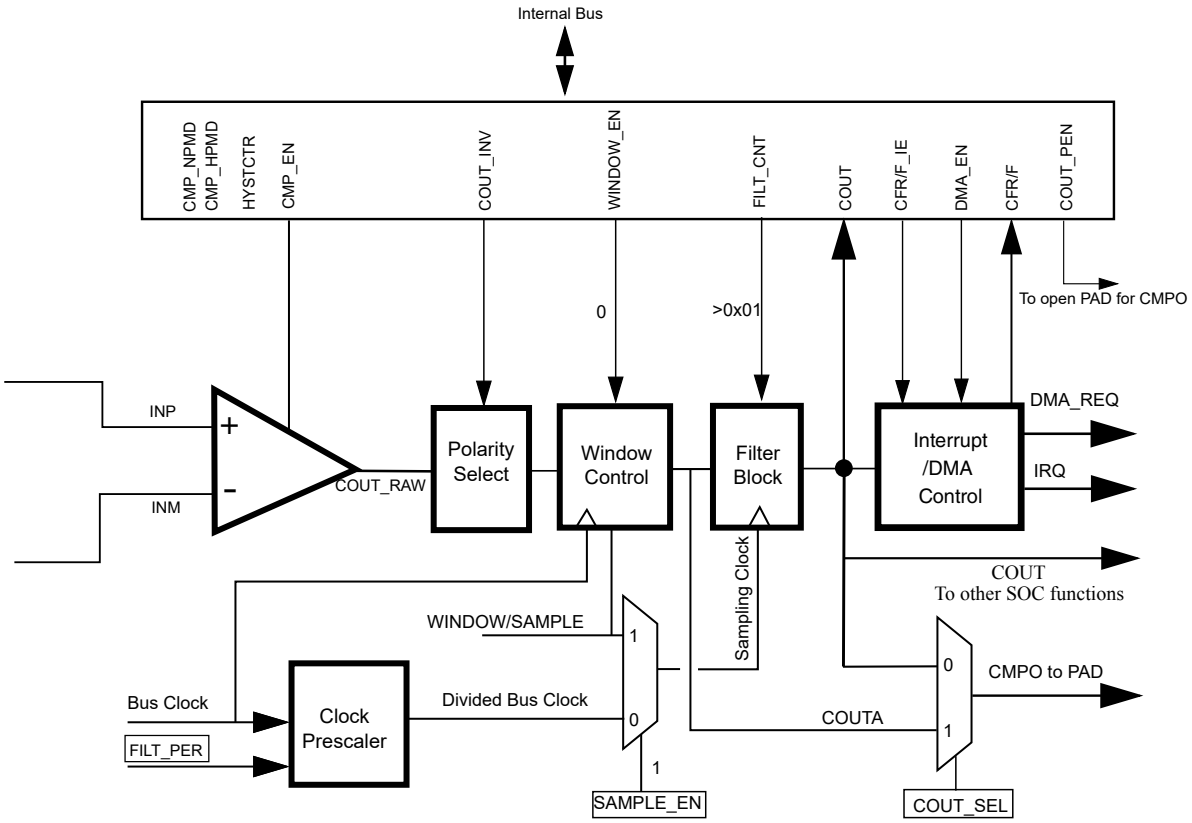


Figure 268. Sampled, filtered (#4A): sampling point externally derived

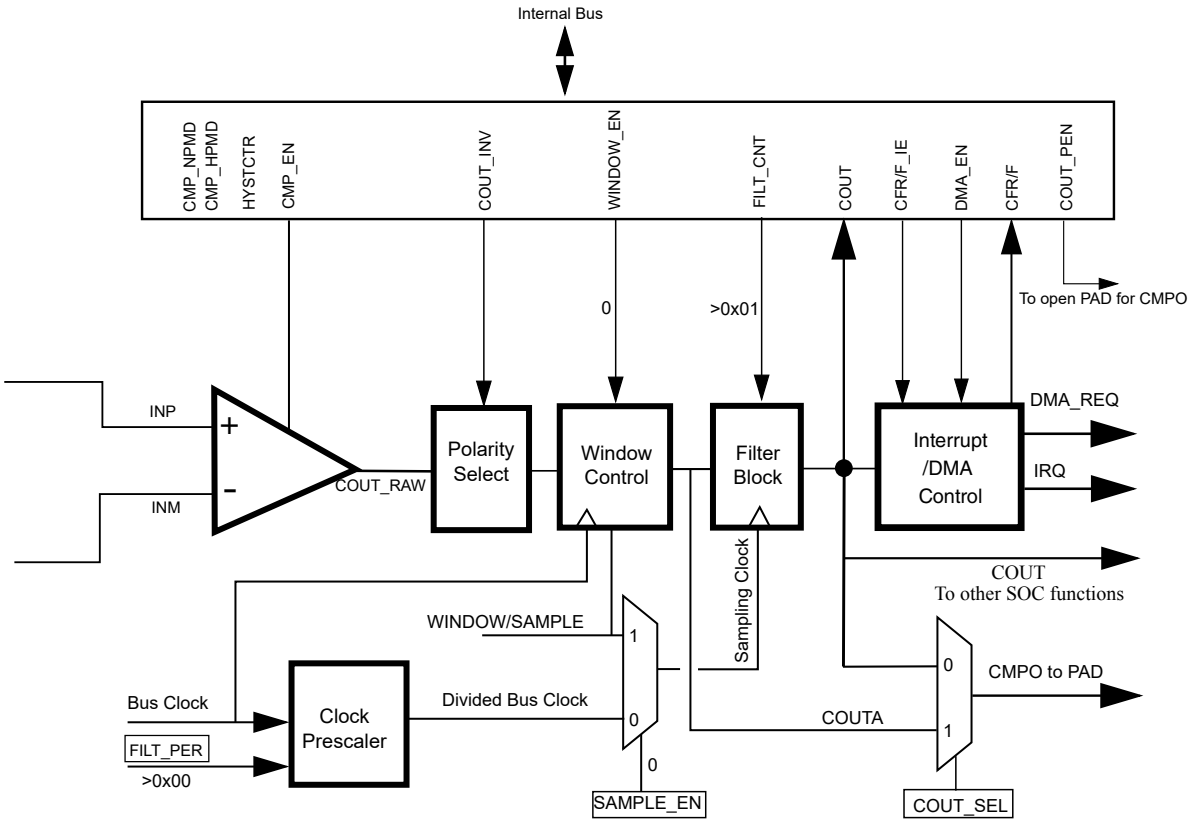


Figure 269. Sampled, filtered (#4B): sampling point internally derived

In this mode, the path from the analog inputs to COUTA is combinational(unclocked). Windowing control bypasses completely. You can sample COUTA whenever you detect a rising edge on the sampling clock.

The only difference in operation between sampled, non-filtered (#3A) mode and sampled, filtered (#4A) mode is that CCR1[FILT_CNT] is larger than 1, which activates filter operation.

The only difference in operation between sampled, non-filtered (#3B) mode and sampled, filtered (#4B) mode is that CCR1[FILT_CNT] is larger than 1, which activates filter operation.

61.3.5.5 Windowed mode (#5A and #5B)

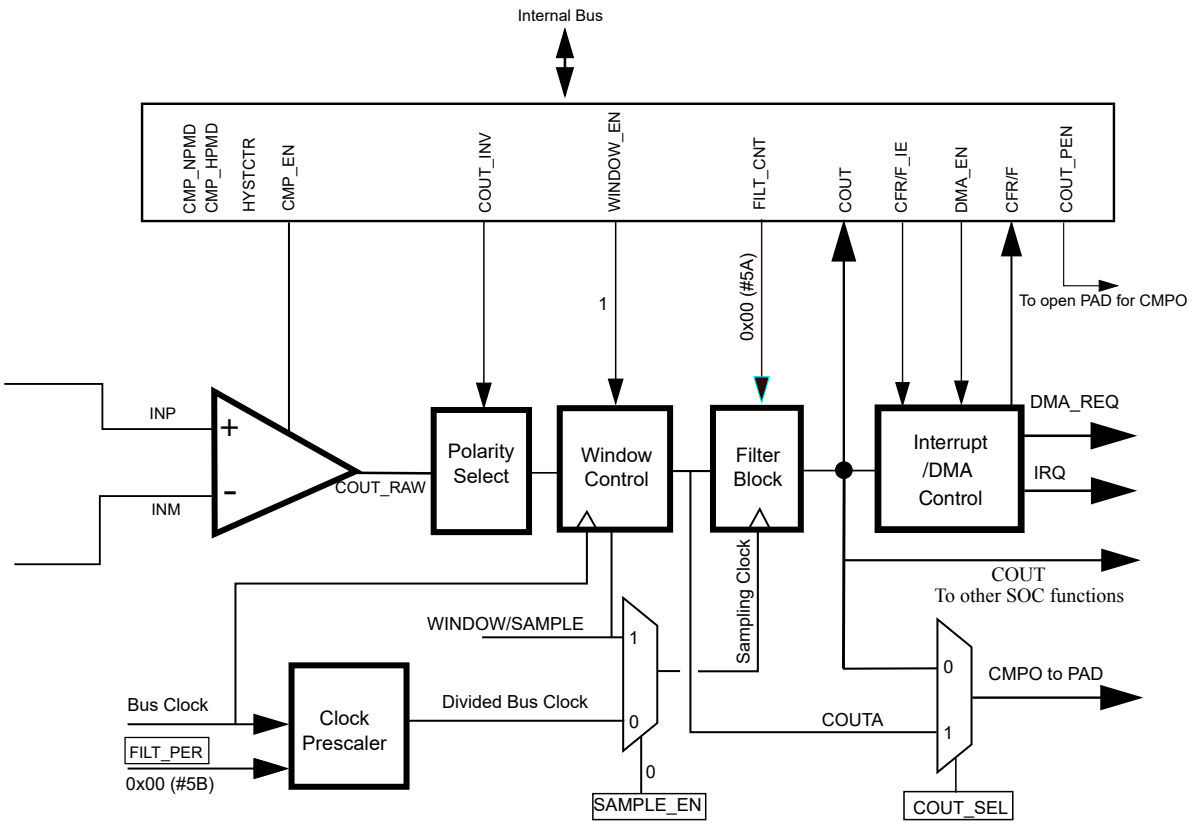


Figure 270. Windowed mode

The bus clock clocks COUTA whenever you enable the window in this mode. The last latched value holds after you disable the window and the filter block is bypassed.

The following figure shows the comparator operation in this mode, ignoring the latency of the analog comparator, polarity select, and window control block. The polarity select sets to a non-inverting state.

COUTA may lag the analog inputs by up to two functional clock cycles plus the combinational path delay through the comparator and polarity select logic in the actual operation.

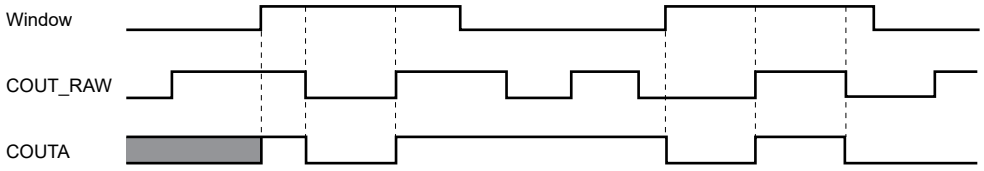


Figure 271. Windowed mode timing diagram

The following figure shows that if CCR1[COUTA_OWEN] becomes 1, you can define COUTA level as CCR1[COUTA_OW], after you closes the window.

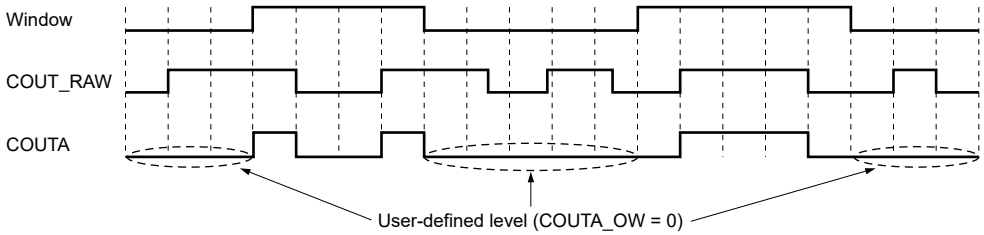


Figure 272. Windowed mode timing diagram with user defined value 0 outside window

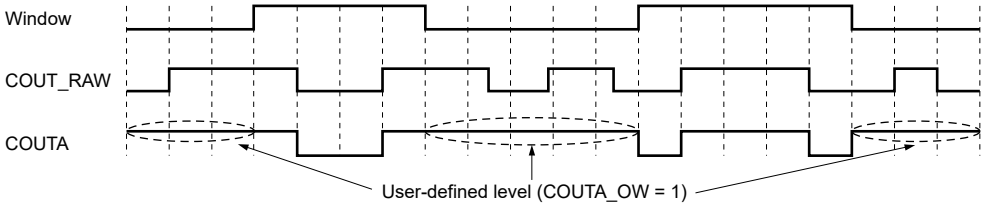


Figure 273. Windowed mode timing diagram with user defined value 1 outside window

NOTE

When the window is open, COUT_A will switch from COUTA_OW to COUT_RAW. When the window is closed, COUT_A will switch from COUT_RAW to COUTA_OW. This may generate unnecessary transition flags, for instance, CFR or CFF. User needs to choose COUTA_OW carefully according to the actual application, and select the appropriate flag CFR or CFF to generate interrupt.

If CCR1[WINDOW_CLS] becomes 1, you can define the CMPO event (rising edge, falling edge or both edges that CCR1[EVT_SEL] selects) to close the window. The external window signal has to go to zero and back to one to enable the internal window again. The following figure shows an example that CMPO rising edge closes the internal window.

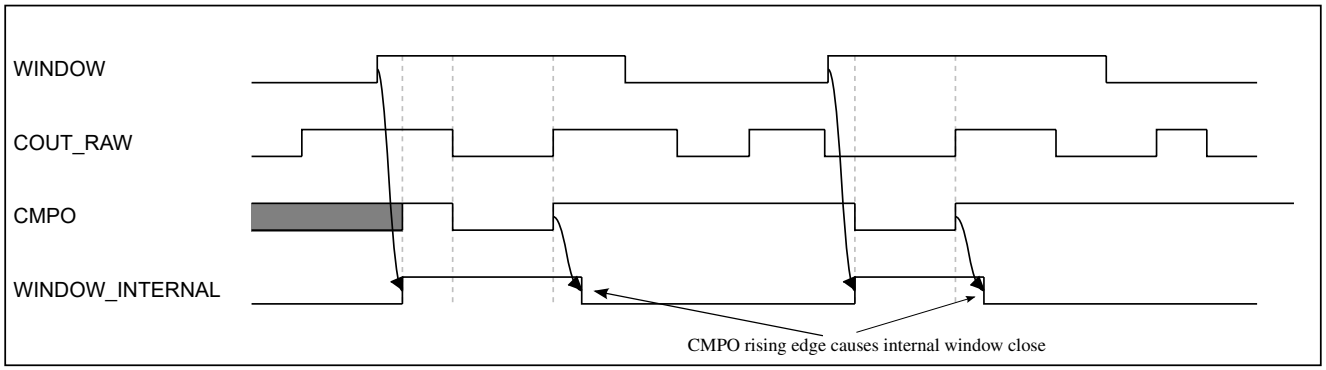


Figure 274. Windowed mode timing diagram with CMPO rising edge close window

The following figure shows that if CCR1[WINDOW_INV] becomes 1, you can invert the window signal before you use it.

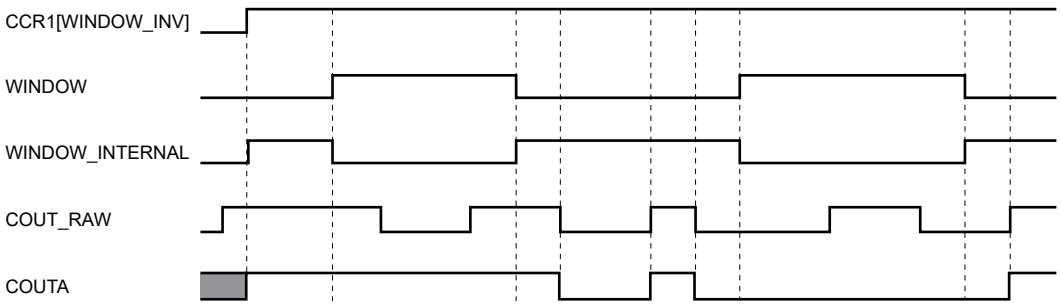


Figure 275. Windowed mode timing diagram with window signal inverted

61.3.5.6 Windowed/Resampled mode (#6)

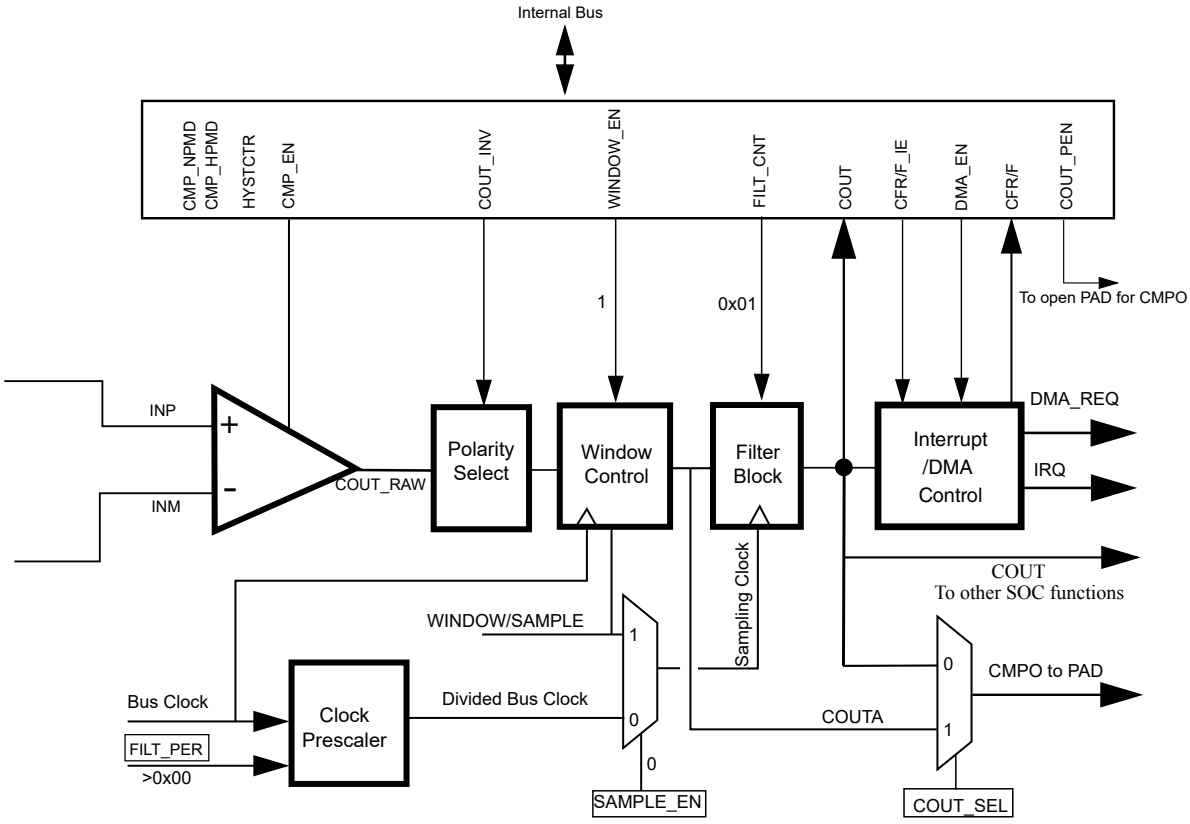


Figure 276. Windowed/Resampled mode

This mode of operation results in an unfiltered string of comparator samples where CCR1[FILT_PER] and the bus clock rate determines the interval between the samples. The following section shows that the configuration for this mode is virtually identical to that for the Windowed/Filtered mode. The only difference is that the value of CCR1[FILT_CNT] must be 1 in this mode.

The following figure uses the same input stimulus shown in Figure 271, and adds resampling of COUTA to generate COUT. The arrows in the figure indicate the time points at which the samples are taken. You can ignore prop delays and latency for clarity.

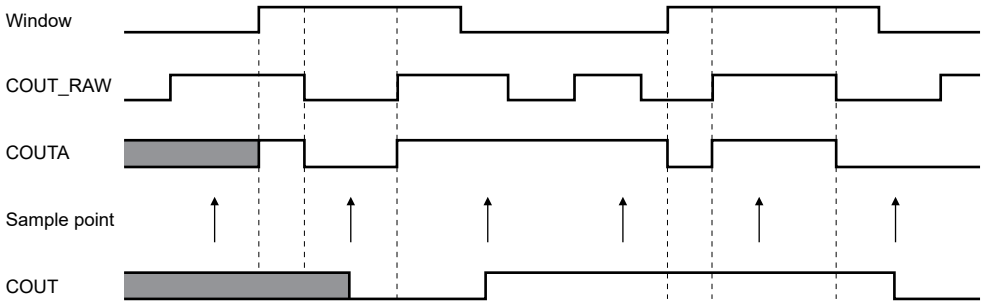


Figure 277. Windowed/Resampled mode operation

This example demonstrates the operation of the comparator in Windowed/Resampled mode, and does not reflect any specific application. Based on the sampling rate and window placement, COUT may not see zero-crossing events that the analog comparator detects. You must carefully consider the sampling period and/or window placement for a given application.

61.3.5.7 Windowed/Filtered mode (#7)

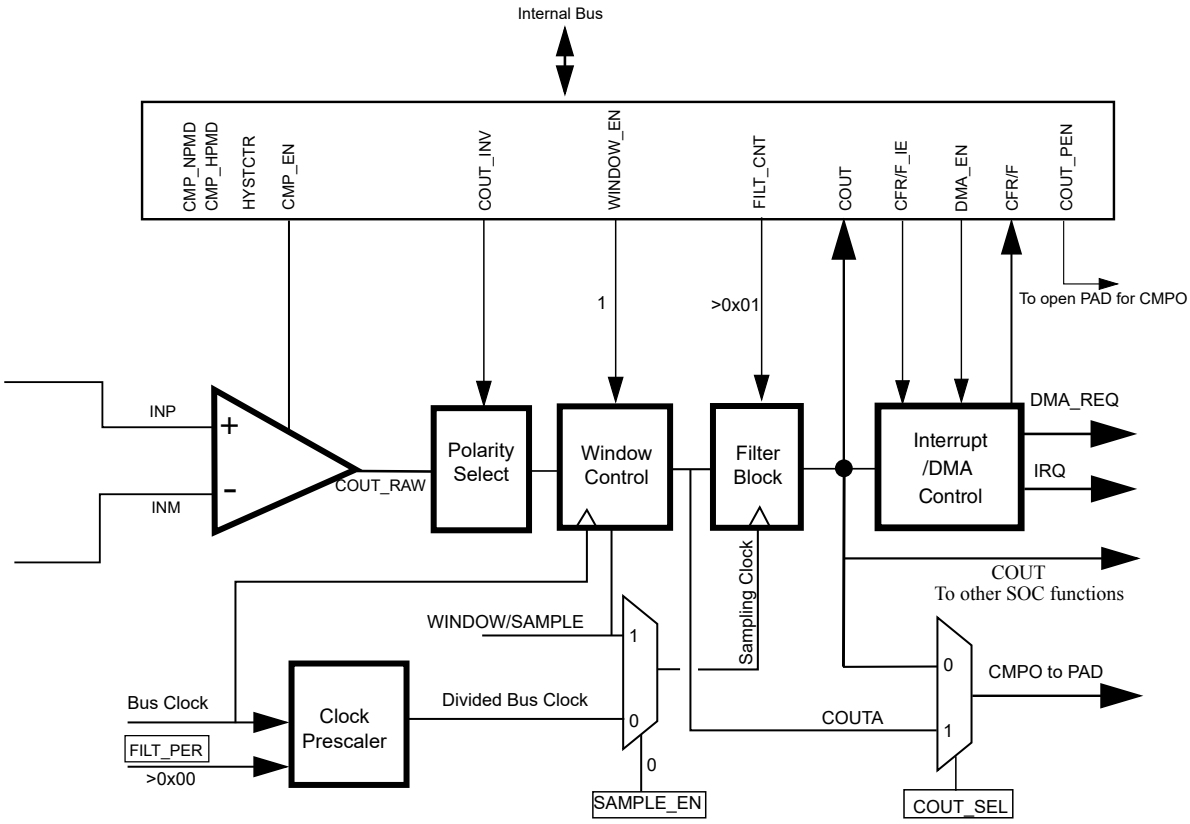


Figure 278. Windowed/Filtered mode

The only difference in operation between Windowed/Resampled mode (#6) and Windowed/Filtered mode (#7) is that CCR1[FILT_CNT] is >1, which activates filter operation.

This mode is the most complex mode of operation for the comparator block, as it utilizes both windowing and filtering features. It also has the highest latency of any of the modes. This is approximately: up to 2 peripheral clock synchronization in the window function + ((CCR1[FILT_CNT] x CCR1[FILT_PER]) + 1) x peripheral clock for the filter function.

61.3.6 DMA

After DMA is enabled by writing 1 to CCR1[DMA_EN] and interrupt is enabled by writing 1 to IER[CFR_IE], IER[CFF_IE], or both, the corresponding change on COUT forces a DMA transfer request rather than a CPU interrupt. After the DMA completes the transfer, it sends a transfer completing indicator signal that deasserts the DMA transfer request and clears the flags (both CSR[CFR] and CSR[CFF]) to allow a subsequent change on comparator output to occur and forces another DMA request.

61.3.7 Clocking

LPCMP requires the following clocks to operate:

Table 355. LPCMP clocks

Type of clock	Description
Bus	Controls the access to LPCMP registers and window/ filter function.
Round-robin clock (RCLK)	Controls Round-robin trigger mode.

61.3.8 Resets

The global chip reset signal resets LPCMP.

61.3.9 Interrupts

After the corresponding IER becomes 1, CSR[CFR], CSR[CFF], and CSR[RRF] can generate an interrupt, assuming that CCR1[DMA_EN] is not 1. You can clear either the flag or IER to deassert the interrupt.

61.4 External signal descriptions

Below table introduces external signals.

Table 356. External signal descriptions

Signal	Description	I/O
CMPO	Filtered or unfiltered comparator output	O
Input_Analog_Channels	Analog input channels (see the chip-specific information for more on the connections).	I
VREFH_EXT	External reference voltage for the CMP-DAC (see the chip-specific information for more on the connections).	I
RR_ACTIVE	Round-robin trigger mode enabled.	O

61.5 Initialization

You can enable LPCMP by writing 1 to CCR0[CMP_EN], and then configuring the control registers (CCR1, CCR2, DCR, and so on).

To disable LPCMP, write 0 to CCR0[CMP_EN]. Switching operation modes or changing control register fields on-the-fly (when CCR0[CMP_EN] is set to 1) may cause noise on the COUT or COUTA signals. To avoid unwanted signal noise, you must ensure to disable the module before switching modes or changing control fields.

The time required to stabilize COUT is the power-on delay of the comparators plus the largest propagation delay from a selected analog source through the analog comparator, windowing function, and filter (see the Comparator and 8-bit DAC electrical specifications section of LPCMP datasheet for more information on propagation delay and power-up delay). Table 354 specifies the delay that the windowing and filter function causes.

During operation, you must always consider the propagation delay of the selected data paths. It can take many bus clock cycles for COUT and CSR[CFR]/CSR[CFF] to reflect an input change or a configuration change to one of the components involved in the data path.

61.6 Application information

61.6.1 Round-robin trigger mode programing recommendation

Configure the Round-robin trigger mode as follows:

1. Configure the comparison cycles by RRCR0[RR_NSAM]. Note: It is a mandatory request that the round robin cycling period must set longer than the time that all the active channels complete the specified comparison cycles set by RRCR0[RR_NSAM].
2. Configure CMP initialization delay by RRCR0[RR_INITMOD]. Note: In programming RRCR0[RR_INITMOD], the RR_INITMOD x round robin clock period must be longer than the initialization delay, see the LPCMP datasheet for more information.
3. Configure RRCR1[FIXP] to select the fixed port of CMP and
    a. If you use one input channel to compare with other channels, configure RRCR1[FIXCH] to select the fixed channel.
    b. If you use DAC output to compare with input channels, configure CCR2[INPSEL] or CCR2[INMSEL] (according to RRCR1[FIXP]) to select the DAC output.
4. Configure channels for comparison by RRCR1[RR_CHnEN].
5. Write RRCSR[RR_CHnOUT] to define the pre-set state of channel n.
6. Clear channel flags RRSR[RR_CHnF].
7. Enable round robin interrupt by IER[RRF_IE] (disable IER[CFR_IE] and IER[CFF_IE]).
8. Enable round-robin trigger mode by setting RRCR0[RR_EN] to 1.
9. Enable comparator by setting CCR0[CMP_EN] to 1.

61.6.2 Round-robin clock (RCLK) frequency requirement

(1) RCLK high frequency limit

RCLK high frequency limit depends on two facts:

1. The analog CMP and DAC initialization time (see the chip data sheet for more information on the initialization time.)
    - RRCR0[RR_INITMOD] provides a maximum 63 RCLK cycles for the analog CMP and DAC initialization.
    - RCLK must be slow to assure: 63 * ($1/f_{RCLK}$) > $T_{initialization}$, where f_{RCLK} is in MHz, and $T_{initialization}$ is in microsecond.
    - so $f_{RCLK} < 63 / T_{initialization}$
    - Example: $T_{initialization}$ = 40 µs, then f_{RCLK} should be smaller than 1.575 MHz.
2. The analog CMP propagation delay (see the Comparator and 8-bit DAC electrical specifications section of LPCMP datasheet for more information on the CMP propagation delay.)
    - RRCR0[RR_NSAM] provides a maximum 4 RCLK cycles for the analog CMP propagation delay.
    - RCLK must be slow to assure: 4 * ($1/f_{RCLK}$) > $T_{propagation}$, where f_{RCLK} is in MHz, and $T_{propagation}$ is in microsecond.
    - $f_{RCLK} < 4 / T_{propagation}$
    - Example: $T_{propagation}$ = 0.1 µs, then f_{RCLK} must be smaller than 40 MHz.

(2) RCLK low frequency limit

In theory, RCLK frequency has no low limit. But the lower the RCLK frequency, the longer the scan time. Therefore, the lower limit of the RCLK frequency depends on the system application.

61.7 LPCMP register descriptions

The memory map comprises of 32-bit aligned registers, which you can access via 8-, 16- or 32-bit reads and 32-bit write. Attempted accesses using unsupported write data sizes, writes to read-only resources, or to reserved spaces terminate with an error. Read access to reserved address generates a transfer error and the read data bus shows all 0s.

61.7.1 LPCMP memory map

LPCMP_0 base address: 4037_0000h

LPCMP_1 base address: 4037_4000h

LPCMP_2 base address: 404E_8000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID (VERID)	32	R	0100_0001h
4h	Parameter (PARAM)	32	R	0000_0002h
8h	Comparator Control Register 0 (CCR0)	32	RW	0000_0002h
Ch	Comparator Control Register 1 (CCR1)	32	RW	0000_0000h
10h	Comparator Control Register 2 (CCR2)	32	RW	0000_0000h
18h	DAC Control (DCR)	32	RW	0000_0000h
1Ch	Interrupt Enable (IER)	32	RW	0000_0000h
20h	Comparator Status (CSR)	32	RW	0000_0000h
24h	Round Robin Control Register 0 (RRCR0)	32	RW	0000_0000h
28h	Round Robin Control Register 1 (RRCR1)	32	RW	0000_0000h
2Ch	Round Robin Control and Status (RRCSR)	32	RW	0000_0000h
30h	Round Robin Status (RRSR)	32	RW	0000_0000h

61.7.2 Version ID (VERID)

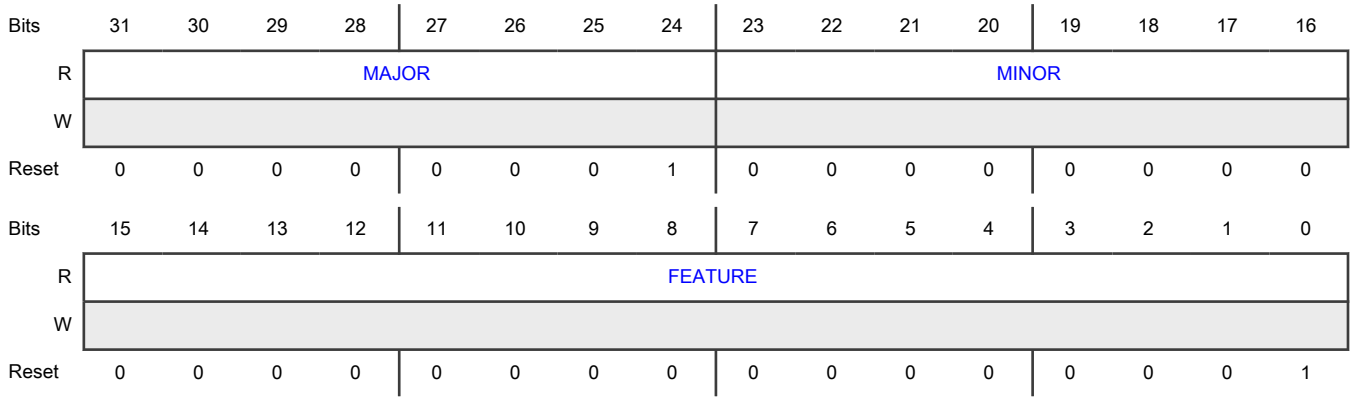
Offset

Register	Offset
VERID	0h

Function

Contains version numbers for the module design and feature set.

Diagram



Fields

Field	Function
31-24 MAJOR	Major Version Number Returns the major version number for the module design.
23-16 MINOR	Minor Version Number Returns the minor version number for the module design.
15-0 FEATURE	Feature Specification Number Returns the feature set number. 0000_0000_0000_0001b - Round robin feature

61.7.3 Parameter (PARAM)

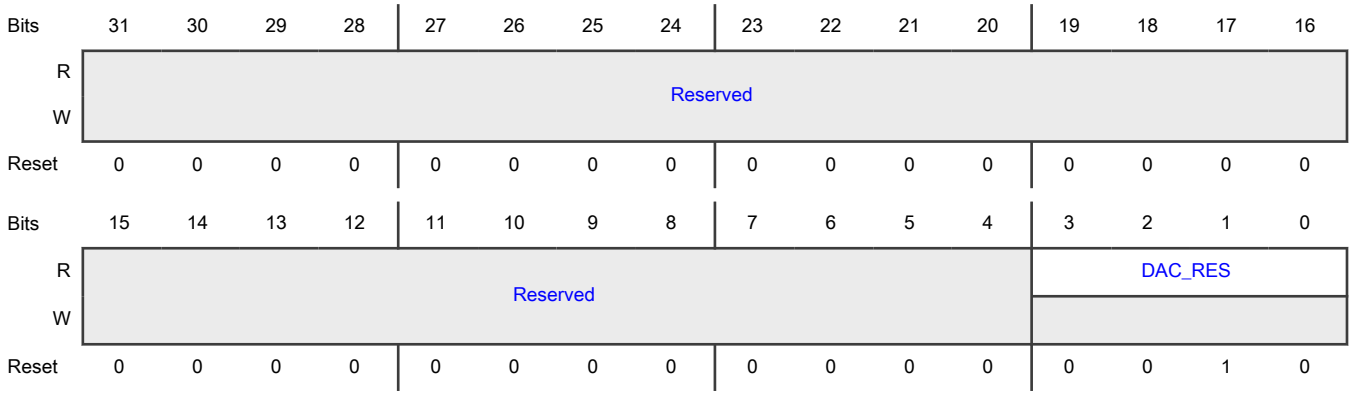
Offset

Register	Offset
PARAM	4h

Function

Contains parameter values that are implemented in the module.

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 DAC_RES	DAC Resolution Indicates supported DAC resolutions. **NOTE** All other bit field values are reserved. 0000b - 4-bit DAC 0001b - 6-bit DAC 0010b - 8-bit DAC 0011b - 10-bit DAC 0100b - 12-bit DAC 0101b - 14-bit DAC 0110b - 16-bit DAC

61.7.4 Comparator Control Register 0 (CCR0)

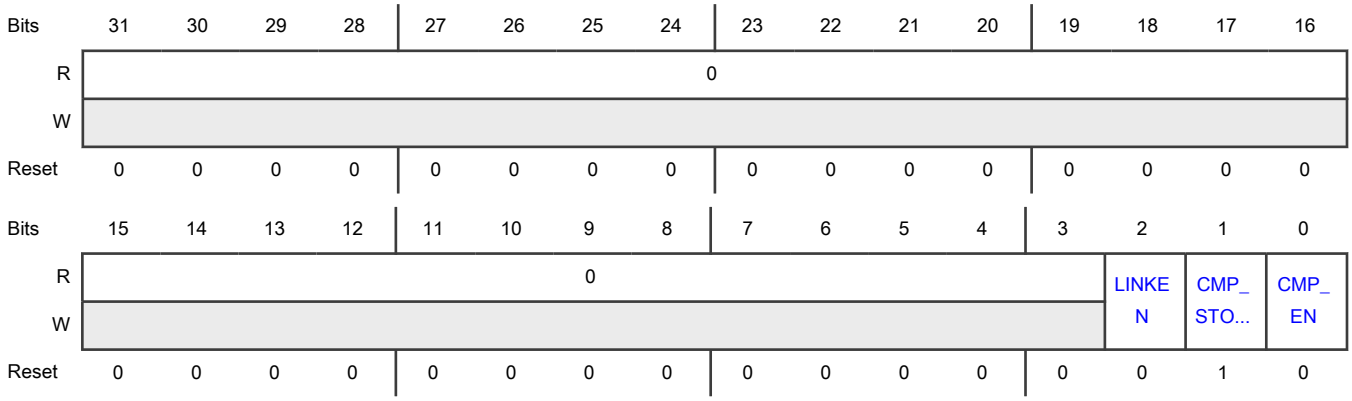
Offset

Register	Offset
CCR0	8h

Function

Contains configuration options for enabling the analog comparator and the DAC.

Diagram



Fields

Field	Function
31-3 —	Reserved
2 LINKEN	CMP-to-DAC Link Enable Enables the CMP-to-DAC link. 0b - Disable: enabling or disabling the DAC is independent from enabling or disabling the CMP. 1b - Enable: enabling/disabling DAC is controlled by the CMP_EN bit instead of DCR[DAC_EN]. Also, when the CMP is auto-disabled because software selects the same signal for both the plus and minus comparator inputs, the DAC is disabled too.
1 CMP_STOP_EN	Comparator STOP Mode Enable Enables the analog comparator or the DAC when the module is in STOP mode. **NOTE** This field has no effect in Round-robin Trigger mode. 0b - Disables the analog comparator regardless of CMP_EN. 1b - Allows CMP_EN to enable the analog comparator.
0 CMP_EN	Comparator Enable Enables the analog comparator. **NOTE** When CCR0[LINKEN]=1, CMP_EN also controls the enabling/disabling of the DAC instead of DCR[DAC_EN]. 0b - Disable (The analog logic remains off and consumes no power.) 1b - Enable

61.7.5 Comparator Control Register 1 (CCR1)

Offset

Register	Offset
CCR1	Ch

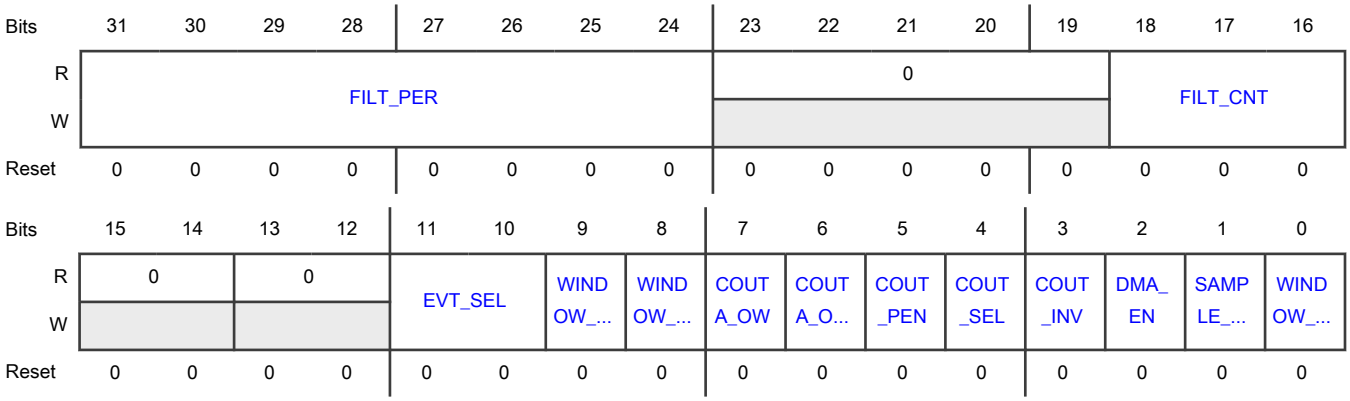
Function

Contains configuration options for the comparator operation, such as enabling Sampling or Windowing mode.

NOTE

You cannot enable Sampling and Windowing modes both at the same time. Sampling mode takes precedence over Windowing mode. If you write 1 to both SAMPLE_EN and WINDOW_EN, only SAMPLE_EN becomes 1.

Diagram



Fields

Field	Function
31-24 FILT_PER	Filter Sample Period Specifies the sampling period (in bus clock cycles) of the comparator output filter. Programming this field to 00h bypasses the filter. See Functional description for more information on filter programming and latency. **NOTE** FILT_PER has no effect in Sampling mode (CCR1[SAMPLE_EN] = 1).
23-19 —	Reserved
18-16 FILT_CNT	Filter Sample Count Specifies the number of consecutive samples that must agree before the comparator output filter accepts the sample as a new valid output state. See Functional description for more information on filter programming and latency.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	000b - Filter is bypassed: COUT = COUTA 001b - 1 consecutive sample (Comparator output is simply sampled.) 010b - 2 consecutive samples 011b - 3 consecutive samples 100b - 4 consecutive samples 101b - 5 consecutive samples 110b - 6 consecutive samples 111b - 7 consecutive samples
15-14 —	Reserved
13-12 —	Reserved
11-10 EVT_SEL	CMPO Event Select Selects which CMPO signal edge (rising, falling, or both) defines a CMPO event. **NOTE** Valid only in Windowing mode. 00b - Rising edge 01b - Falling edge 1xb - Both edges
9 WINDOW_CLS	CMPO Event Window Close Enables a CMPO event (defined as a CMPO rising edge, falling edge, or both) to close an active window. See EVT_SEL to configure the CMPO event. **NOTE** The WINDOW signal has to go to zero and back to one again to re-activate the window. Valid only in Windowing mode. 0b - CMPO event cannot close the window 1b - CMPO event can close the window
8 WINDOW_INV	WINDOW/SAMPLE Signal Invert Inverts the window/sample signal. 0b - Do not invert 1b - Invert
7	COUTA Output Level for Closed Window

Table continues on the next page...

Table continued from the previous page...

Field	Function
COUTA_OW	Defines the COUTA signal value when the window is closed. **NOTE** Valid only in Windowing mode and when COUTA_OWEN=1. 0b - COUTA is 0 1b - COUTA is 1
6 COUTA_OWEN	COUTA_OW Enable Enables the COUTA signal value to be defined by COUTA_OW when the window is closed. **NOTE** Valid only in Windowing mode. 0b - COUTA holds the last sampled value. 1b - Enables the COUTA signal value to be defined by COUTA_OW.
5 COUT_PEN	Comparator Output Pin Enable Enables the comparator output to become an available signal option for a selected package pin. 0b - Not available 1b - Available
4 COUT_SEL	Comparator Output Select Selects which comparator output option, COUT or COUTA, to use for CMPO. 0b - Use COUT (filtered) 1b - Use COUTA (unfiltered)
3 COUT_INV	Comparator Invert Selects the polarity of the analog comparator function, affecting the value driven to the COUT output (on both the chip pin and as CSR[COUT]) when CCR0[CMP_EN] is 0. **NOTE** COUT_INV has no effect in Trigger mode. 0b - Do not invert 1b - Invert
2 DMA_EN	DMA Enable Enables DMA transfers triggered from the LPCMP module. After this field and the corresponding interrupt enable field becomes 1, a DMA request is asserted when CFR or CFF becomes 1. 0b - Disable 1b - Enable
1	Sampling Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
SAMPLE_EN	Enables Sampling mode. 0b - Disable 1b - Enable
0 WINDOW_EN	Windowing Enable Enables Windowing mode. **NOTE** Valid only when SAMPLE_EN = 0. 0b - Disable 1b - Enable

61.7.6 Comparator Control Register 2 (CCR2)

Offset

Register	Offset
CCR2	10h

Function

Contains the configuration options for the comparator operation, such as selecting the plus and minus comparator inputs and the hysteresis levels.

NOTE

When an inappropriate operation selects the same signal for both the plus and minus comparator inputs, the analog comparator automatically shuts down (regardless of CMP_EN) to prevent itself from becoming a noise generator.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		INMSEL		0		INPSEL		0		MSEL		0		PSEL	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0										HYSTCTR		0	OFFSET	0	CMP_HPMD
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29-28 INMSEL	Input Minus Select Selects the minus input of the comparator. **NOTE** These selections connect directly to the minus input of the comparator. 00b - IN0: from the 8-bit DAC output 01b - IN1: from the analog 8-1 mux 10b - Reserved 11b - Reserved
27-26 —	Reserved
25-24 INPSEL	Input Plus Select Selects the plus input of the comparator. **NOTE** These selections connect directly to the plus input of the comparator. 00b - IN0: from the 8-bit DAC output 01b - IN1: from the analog 8-1 mux 10b - Reserved 11b - Reserved
23 —	Reserved
22-20 MSEL	Minus Input MUX Select Selects the input used for the negative mux. See the chip-specific LPCMP information for more on connections. **NOTE** MSEL has no effect in Trigger mode. 000b - Input channel 0 001b - Input channel 1 010b - Input channel 2 011b - Input channel 3

Table continues on the next page...

Table continued from the previous page...

Field	Function
	100b - Input channel 4 101b - Input channel 5 110b - Input channel 6 111b - Input channel 7
19 —	Reserved
18-16 PSEL	Plus Input MUX Select Selects the input used for the positive mux. See the chip-specific LPCMP information for more on connections. **NOTE** PSEL has no effect in Trigger mode. 000b - Input channel 0 001b - Input channel 1 010b - Input channel 2 011b - Input channel 3 100b - Input channel 4 101b - Input channel 5 110b - Input channel 6 111b - Input channel 7
15-6 —	Reserved
5-4 HYSTCTR	Comparator Hysteresis Control Selects the level of internally generated hysteresis for the comparator output. **NOTE** This applies to the comparator hard block. 00b - Level 0: Analog comparator hysteresis 0 mV. 01b - Level 1: Analog comparator hysteresis 10 mV. 10b - Level 2: Analog comparator hysteresis 20 mV. 11b - Level 3: Analog comparator hysteresis 30 mV.
3 —	Reserved
2	Comparator Offset Control

Table continues on the next page...

Table continued from the previous page...

Field	Function
OFFSET	Selects the level of internally generated voltage offset for the comparator output. See the chip data sheet to get the specific values for each offset level. 0b - Level 0: The hysteresis selected by HYSTCTR is valid for both directions (rising and falling). 1b - Level 1: Hysteresis does not apply when INP (input-plus) crosses INM (input-minus) in the rising direction or when INM crosses INP in the falling direction. Hysteresis still applies for INP crossing INM in the falling direction.
1 —	Reserved
0 CMP_HPMD	CMP High Power Mode Select Selects Low or High Power(Speed) mode for the comparator. 0b - Low power (speed) comparison mode 1b - High power (speed) comparison mode

61.7.7 DAC Control (DCR)

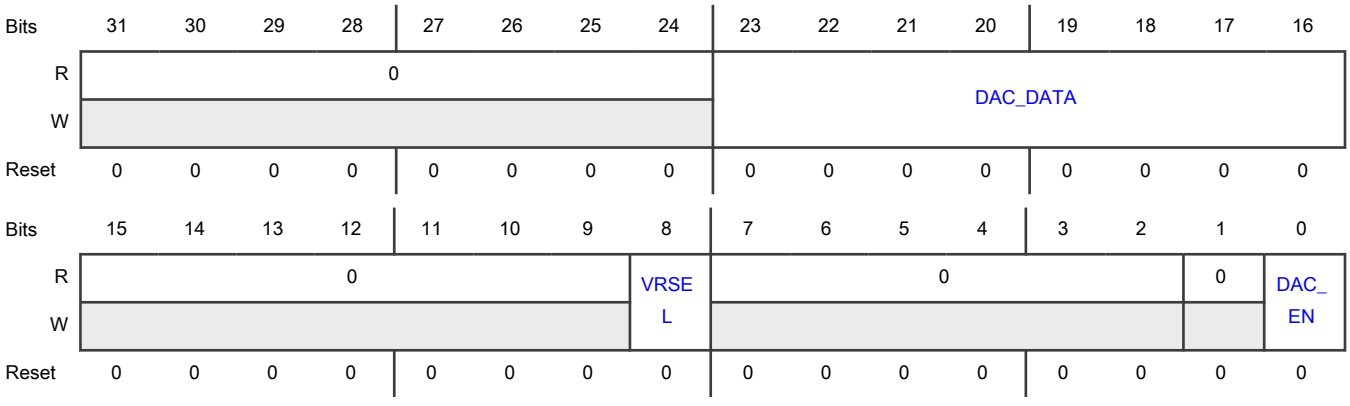
Offset

Register	Offset
DCR	18h

Function

Contains the configuration options to enable the DAC.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-16 DAC_DATA	DAC Output Voltage Select Selects the DAC output (DACO) voltage from one of 256 distinct levels by configuring the value of DAC_DATA. The DACO ranges from Vin/256 to Vin. **NOTE** DACO = (Vin/256) * (DAC_DATA + 1)
15-9 —	Reserved
8 VRSEL	DAC Reference High Voltage Source Select Selects the high voltage reference source for the Vin supply of the DAC's resistor ladder network. See the chip-specific LPCMP information for the source of vrefh0 and vrefh1. 0b - VREFH0 1b - VREFH1
7-2 —	Reserved
1 —	Reserved
0 DAC_EN	DAC Enable Enables the DAC. When disabled, power-down the DAC to conserve power. **NOTE** You can control the link from the CMP enable to the DAC enable by setting up CCR0[LINKEN]. 0b - Disable 1b - Enable

61.7.8 Interrupt Enable (IER)

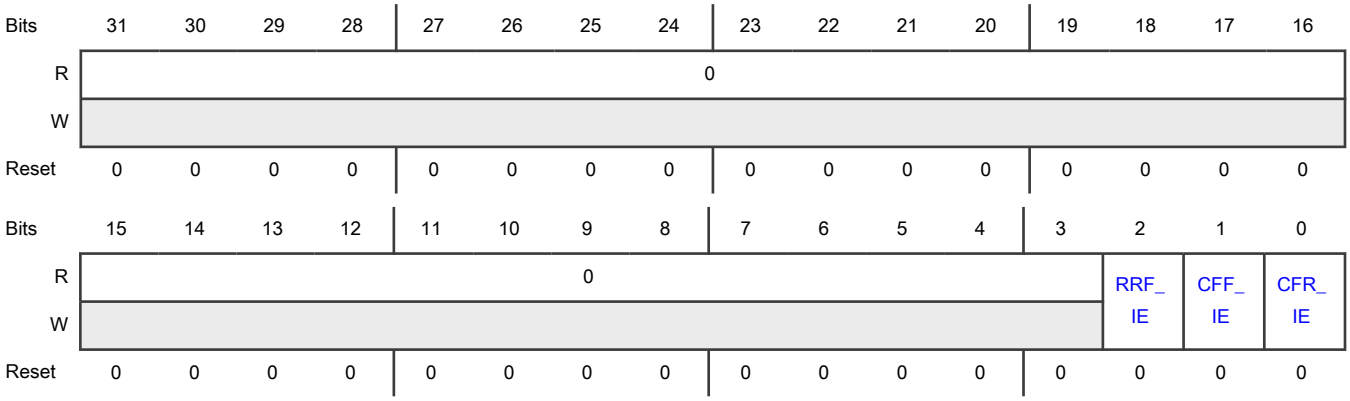
Offset

Register	Offset
IER	1Ch

Function

Provides enable fields for the comparator and round-robin flag interrupts.

Diagram



Fields

Field	Function
31-3 —	Reserved
2 RRF_IE	Round-Robin Flag Interrupt Enable Enables or disables the round-robin flag interrupt. 0b - Disables the round-robin flag interrupt. 1b - Enables the round-robin flag interrupt when the comparison result changes for a given channel.
1 CFF_IE	Comparator Flag Falling Interrupt Enable Enables or disables the comparator flag falling interrupt. 0b - Disables the comparator flag falling interrupt. 1b - Enables the comparator flag falling interrupt when CFF is set.
0 CFR_IE	Comparator Flag Rising Interrupt Enable Enables or disables the comparator flag rising interrupt. 0b - Disables the comparator flag rising interrupt. 1b - Enables the comparator flag rising interrupt when CFR is set.

61.7.9 Comparator Status (CSR)

Offset

Register	Offset
CSR	20h

Function

Indicates comparator status, including COUT, CFF, CFR, and RRF.

NOTE

LPCMP may output a glitch and affect the value of CSR[CFF] and CSR[CFR] at the moment of enabling CMP. In order to ensure correctness, it is recommended to write one to clear (W1C) CSR[CFF] and CSR[CFR] before further configuring CMP.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0							COUT	0					RRF	CFF	CFR
W														W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-9 —	Reserved
8 COUT	Analog Comparator Output Returns the current value of the analog comparator output when read. This field resets to 0 and reads as CCR1[COUT_INV] after the analog comparator module disables when CCR0[CMP_EN] = 0. Writing to this field is ignored.
7-3 —	Reserved
2 RRF	Round-Robin Flag Detects when any channel's last comparison result is different from the pre-set value in Trigger mode. Write 1 to clear this field. This field clears when CCR0[CMP_EN] or RRCR0[RR_EN] is not 1. 0b - Not detected 1b - Detected
1 CFF	Analog Comparator Flag Falling Detects when a falling edge on COUT occurs. Write 1 to clear this field when CCR1[DMA_EN] is disabled. If CCR1[DMA_EN] is enabled, the flag automatically clears after DMA is done. This field clears when CCR0[CMP_EN] is not 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Not detected 1b - Detected
0 CFR	Analog Comparator Flag Rising Detects when a rising edge on COUT occurs. Write 1 to clear this field when CCR1[DMA_EN] is disabled. If CCR1[DMA_EN] is enabled, the flag automatically clears after DMA is done. This field clears when CCR0[CMP_EN] is not 1. 0b - Not detected 1b - Detected

61.7.10 Round Robin Control Register 0 (RRCR0)

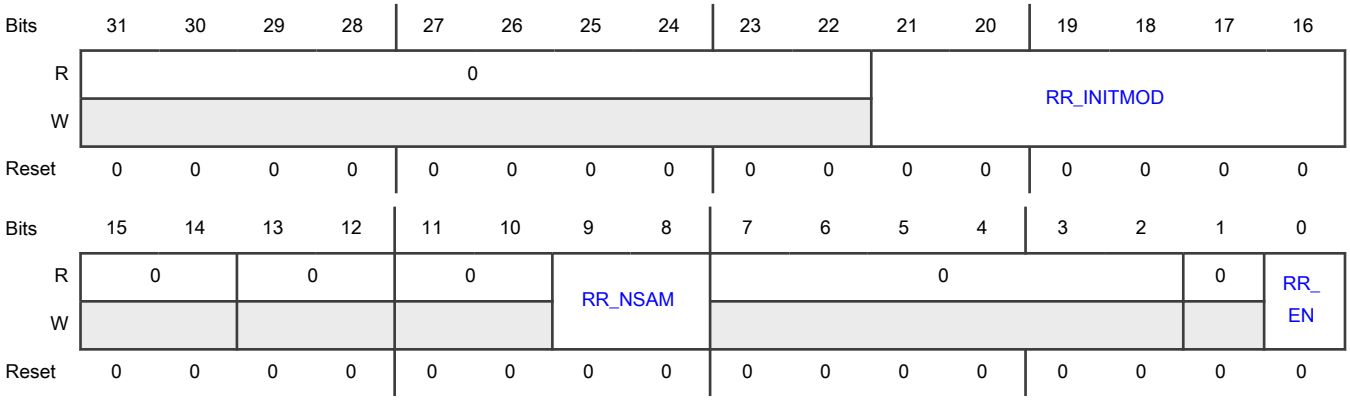
Offset

Register	Offset
RRCR0	24h

Function

Contains configuration options for the round-robin operation, such as enabling it and specifying the initialization delay.

Diagram



Fields

Field	Function
31-22 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
21-16 RR_INITMOD	Initialization Delay Modulus Specifies the number of round-robin clock cycles that determines the comparator and DAC initialization delay specified in the chip datasheet. Calculate the initialization delay as RR_INITMOD * (round-robin clock period). For example, if the initialization delay is 80us and the round-robin clock is 100kHz, program RR_INITMOD to be 80us/10us = 8. 00_0000b - 63 cycles (same as 111111b) 00_0001b-11_1111b - 1 to 63 cycles
15-14 —	Reserved
13-12 —	Reserved
11-10 —	Reserved
9-8 RR_NSAM	Number of Sample Clocks Specifies the number of the round-robin clock cycles to wait after scanning the active channel before sampling the channel's comparison result. After the next cycle of the round-robin clock, the sampling takes place RR_NSAM clocks later. 00b - 0 clock 01b - 1 clock 10b - 2 clocks 11b - 3 clocks
7-2 —	Reserved
1 —	Reserved
0 RR_EN	Round-Robin Enable Enables the round-robin operation. 0b - Disable 1b - Enable

61.7.11 Round Robin Control Register 1 (RRCR1)

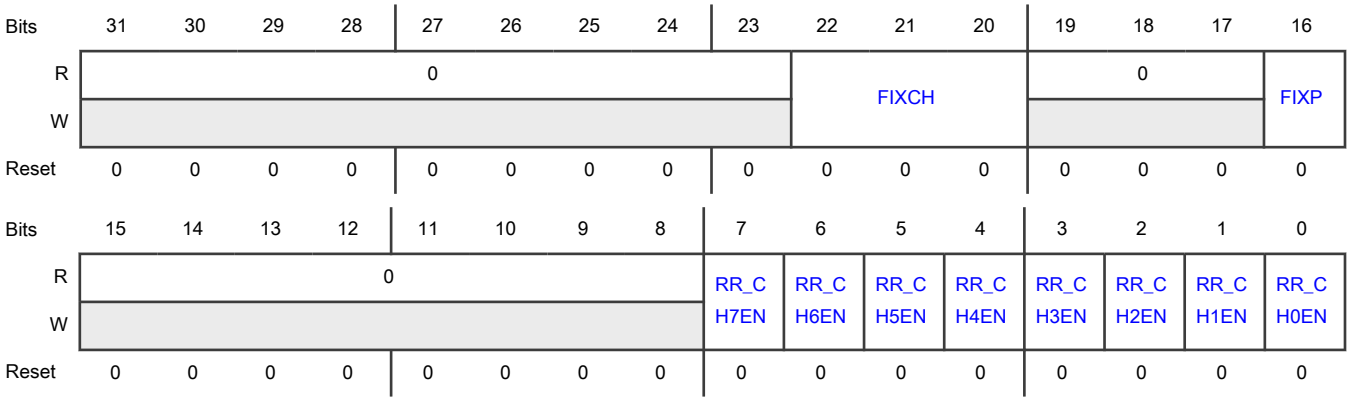
Offset

Register	Offset
RRCR1	28h

Function

Contains configuration options for the round-robin operation, such as enabling individual channels to participate.

Diagram



Fields

Field	Function
31-23 —	Reserved
22-20 FIXCH	Fixed Channel Select Selects which channel in the mux port to fix for a given round-robin trigger mode application. 000b - Channel 0 001b - Channel 1 010b - Channel 2 011b - Channel 3 100b - Channel 4 101b - Channel 5 110b - Channel 6 111b - Channel 7
19-17 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
16 FIXP	Fixed Port Fixes an analog mux port (plus or minus) for round-robin trigger mode. The inputs to the non-fixed port sweep during each round. 0b - Fix the plus port. Sweep only the inputs to the minus port. 1b - Fix the minus port. Sweep only the inputs to the plus port.
15-8 —	Reserved
7-0 RR_CHnEN	Channel n Input Enable in Trigger Mode Enables channel n of the non-fixed mux port to check its voltage value when in Trigger mode. **NOTE** RR_CHnEN has no effect when the same channel is selected as the reference voltage. 0b - Disable 1b - Enable

61.7.12 Round Robin Control and Status (RRCSR)

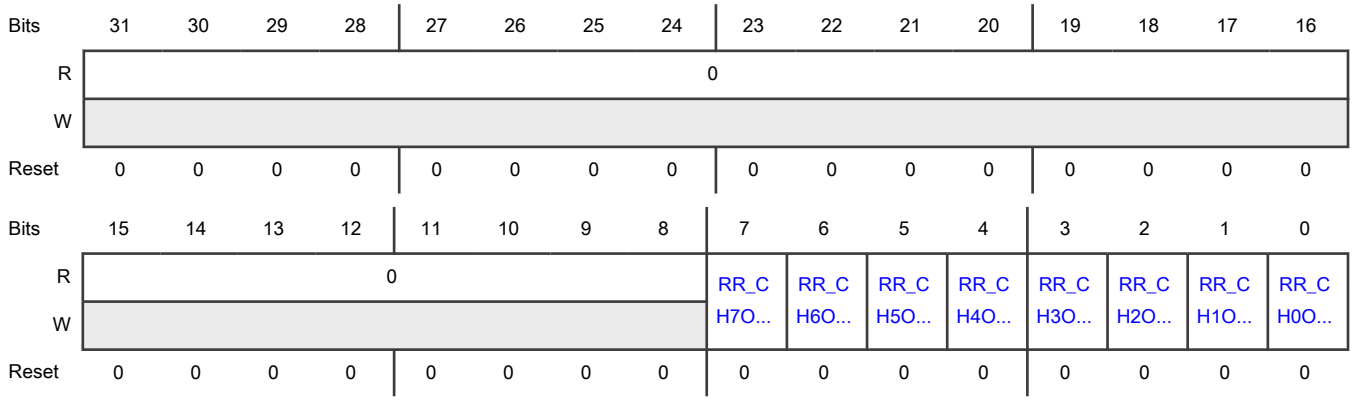
Offset

Register	Offset
RRCSR	2Ch

Function

Contains the latest comparison results of the individual channels with the fixed mux port. It also allows you to define the pre-set state for each channel.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 RR_CHnOUT	Comparison Result for Channel n Returns the latest comparison result for channel n when read and defines the pre-set state for channel n when written to.

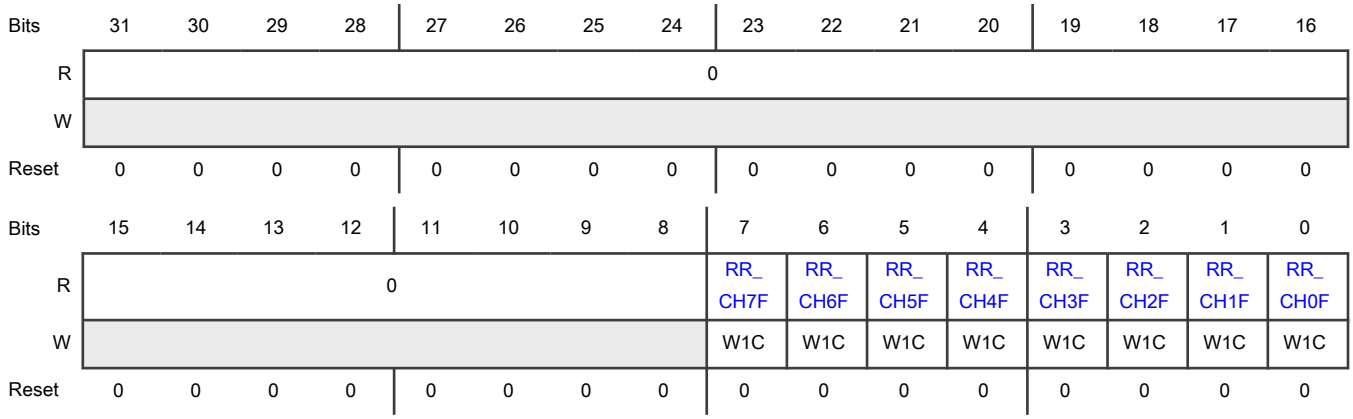
61.7.13 Round Robin Status (RRSR)

Offset

Register	Offset
RRSR	30h

Function
Contains individual channel flags that indicates when a channel's last comparison result is different from its pre-set value.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 RR_CHnF	Channel n Input Changed Flag Indicates when the corresponding channel's last comparison result is different from its pre-set value.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE To clear a flag, write a 1 to it. 0b - No different 1b - Different

61.8 Glossary

CMP	Comparator
DAC	Digital-to-analog convertor
ANMUX	Analog multiplexer

Chapter 62
Logic Control Unit (LCU)

62.1 Chip-specific LCU information

62.1.1 LCU instances

This chip has two identical LCU instances, LCU_0 and LCU_1.

62.2 Overview

LCU selects multiple inputs from timers, Pulse Width Modulation (PWM) signals, and Input/Output (I/O) pads, and combines them using a programmable logic function to create output waveforms.

62.2.1 Block diagram

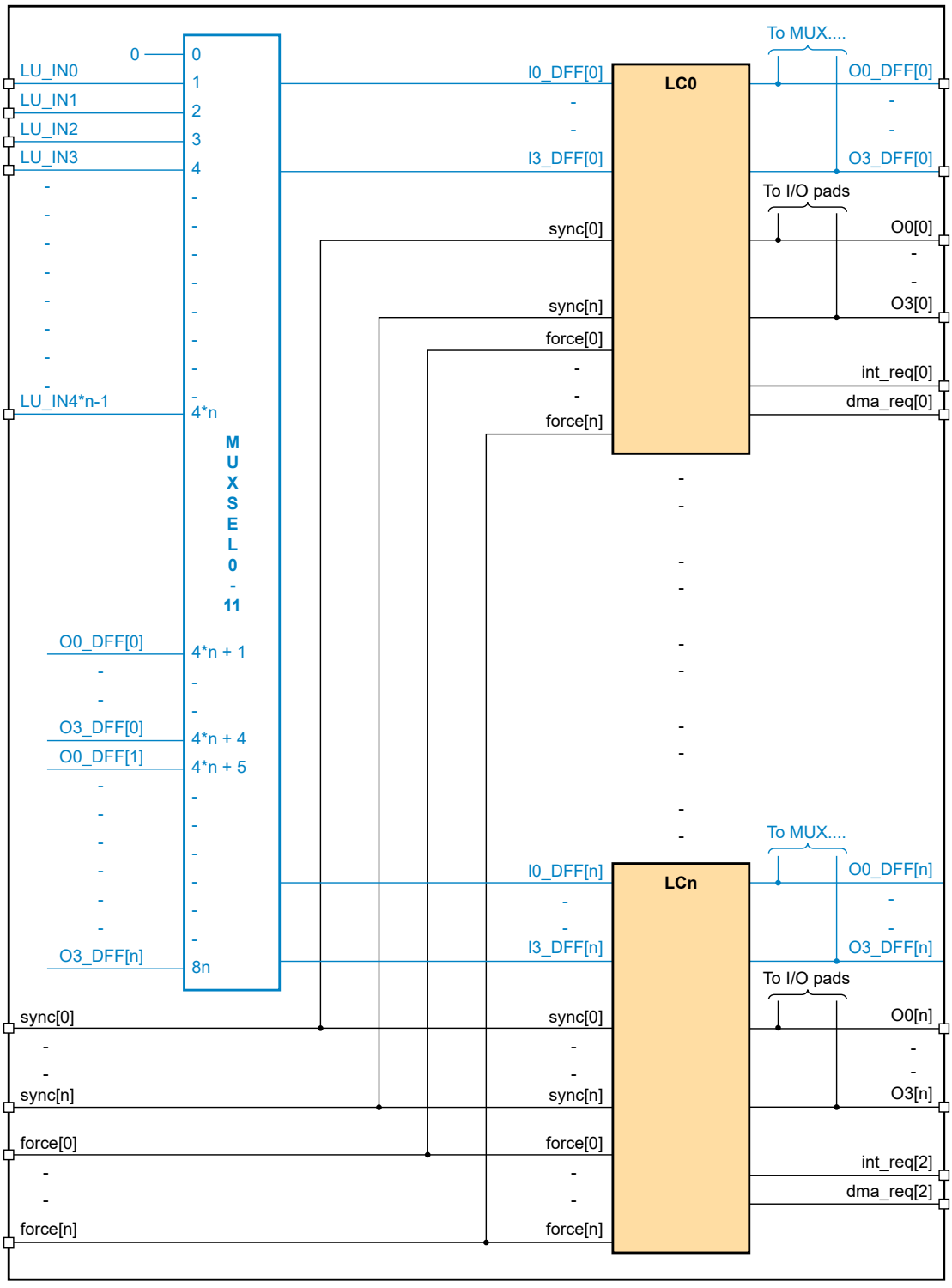


Figure 279. Block diagram

62.2.2 Features

- 3 LCs with programmable logic function for generating output results

- 12 inputs to and 12 outputs from LCs
- Muxing to map any input to any LC
- Independent filters for the rising and falling output states of each LC
- Software override logic for inputs using multiple modes
- 3 force inputs with programmable edge filtering to force output states using multiple modes
- 2 sync inputs to control transition timing
- Lock bit to block writes to CFG[WP]
- Interrupt request generation based on output change or force event
- DMA transfer request generation based on output change or force event

62.3 Functional description

62.3.1 LC operation

62.3.1.1 Overview

An LC performs logic operations based on the LUT principle, as described in Logic operations.

Each LC supports the following features:

- Independent filter thresholds for the rising and falling output states support different transition delays.
- Bypass of the output state filter by setting the filter threshold to zero. Bypassing the filter causes the asynchronous signal to propagate to the output of the LC.
- Force logic that instantly switches an LC output to the inactive state upon external fault signal assertion. The return from the inactive state to the output state is either:
  - Instantaneous: when LCU deasserts the force signal.
  - Instantaneous synchronized: when LCU deasserts the force signal and then the sync input asserts.
  - Manual: when you write 1 to the force status bit and then the LCU deasserts the force input.
  - Manual synchronized: when you write 1 to the force status bit and LCU deasserts the force input, and then the sync input asserts.
- Input software override logic to override external inputs by writing to the following registers:
  - LC*n* Sync Control (LC*n*_SCTRL[SW_MODE])
  - Software Override Value (SWVALUE[SWVALUE])
  - Software Override Enable (SWEN[SWEN])
- Generation of interrupt requests or DMA transfer requests.

62.3.1.2 LC diagram

This figure illustrates an example LC implementation with:

- One LC
- Four force inputs
- Two sync inputs

For the number of each resource in this chip, see Features.

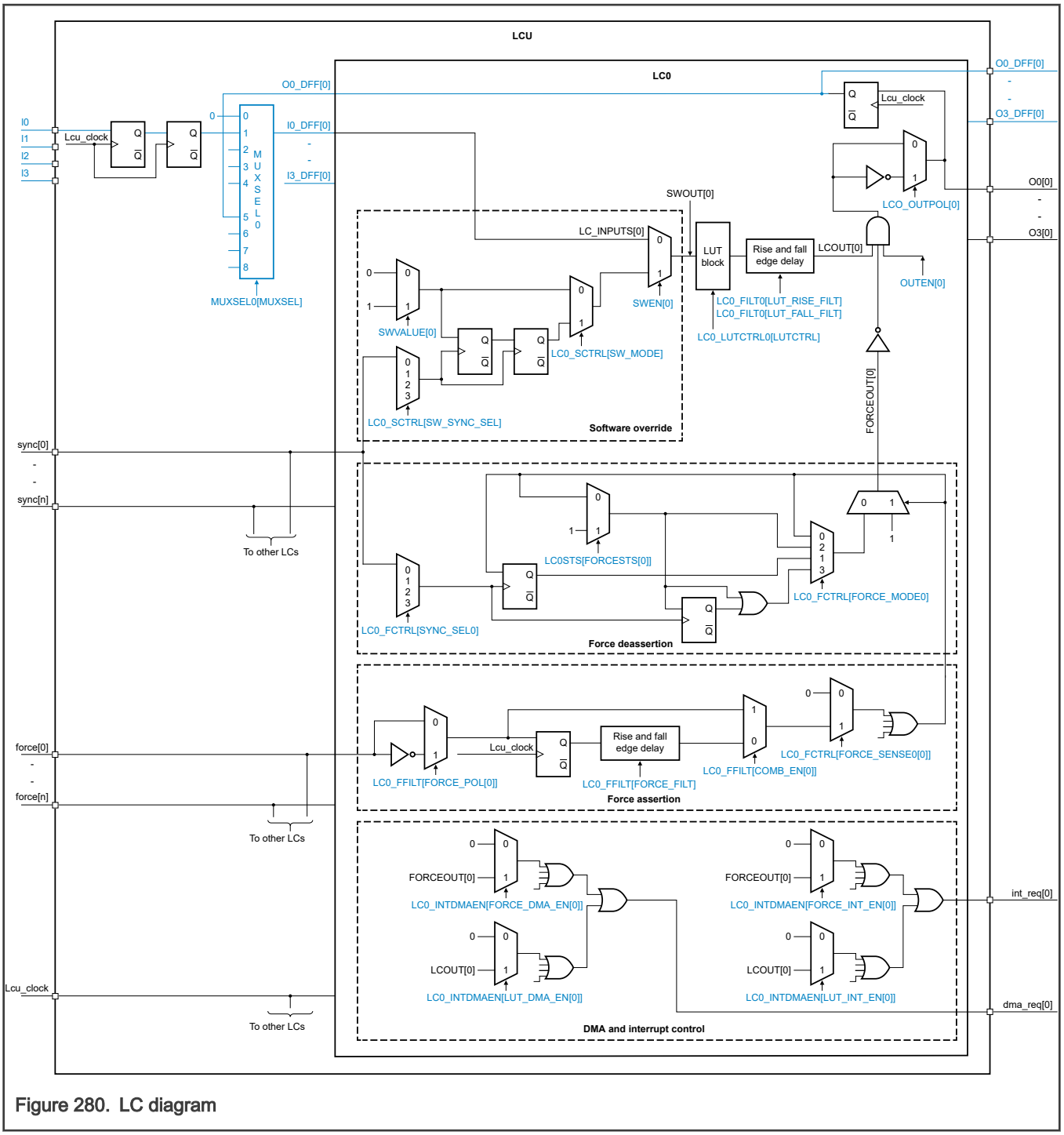


Figure 280. LC diagram

62.3.1.3 Logic operations

An LC reads its set of inputs (inputs 3, 2, 1, and 0) as a four-bit binary value, with Input 3 as the most-significant bit and Input 0 as the least-significant bit. For example, if:

Input 0 = 0

Input 1 = 1

Input 2 = 0

Input 3 = 1

then the combined input value is 1010b (10 decimal).

Each possible four-bit input value corresponds to a bit position in the LUT Control (LCn_LUTCTRLm[LUTCTRL]), as illustrated in LC LUT. For a given LC output, if the combined LC input value corresponds to a bit position in that output's LUTCTRL value that equals 1, then the LC asserts that output.

For example, if you write 35h to LUTCTRL (shown in the Example column of the LC LUT), then the combined input values 0 (0000b), 2 (0010b), 4 (0100b), and 5 (0101b) cause assertion of the output. All other input values cause deassertion of the output.

LCU processes each look-up table asynchronously.

62.3.1.4 LC LUT

The LUT maps the combined LC input value to a bit position in LC*n*_LUTCTRL*m*.

Table 357. LC LUT

Input 3	Input 2	Input 1	Input 0	LUTCTRL bit position	Example LUTCTRL value: 35h
0	0	0	0	0	1
0	0	0	1	1	0
0	0	1	0	2	1
0	0	1	1	3	0
0	1	0	0	4	1
0	1	0	1	5	1
0	1	1	0	6	0
0	1	1	1	7	0
1	0	0	0	8	0
1	0	0	1	9	0
1	0	1	0	10	0
1	0	1	1	11	0
1	1	0	0	12	0
1	1	0	1	13	0
1	1	1	0	14	0
1	1	1	1	15	0

62.3.1.5 LC output filters

Each LC supports two optional digital filters, a Rise Filter (LC*n*_FLIT*m*[LUT_RISE_FILT]) and a Fall Filter (LC*n*_FLIT*m*[LUT_FALL_FILT]), to post-process the LUT output and delay the output signal change.

Each filter starts accumulating clock cycles on the occurrence of the associated LUT event. The Rise Filter counts from an output assertion, and the Fall Filter counts from an output deassertion. When the number of clock cycles reaches the filter threshold (matches the specified filter value), the output flips to the new state. For example, if you specify a Rise Filter value of 100h, and the LUT event triggers an output assertion, the output signal does not assert until the 256th clock cycle after the event.

A filter value of 0 bypasses the filter.

NOTE

LCU usecase (intercell/feedback loop path along with TRGMUX Path) does not work with default RISE/FALL filter time. You must configure RISE/FALL Filter delay at least for 1 cycle to avoid such glitches/failures.

62.3.2 Behavior in different chip modes of operation

Table 358. Modes of operation

Operation mode	Status of outputs
Normal	• Controlled by LC*n*_LUTCTRL*m* and MUXSEL*n* • Toggled based on the states of the selected inputs
Debug	Inactive or normal operation controlled by Debug Mode Enable (DBGEN)

62.3.3 Clocking

This module has no clocking considerations.

62.3.4 Interrupts

Each LC has a single interrupt request output int_req[n]. Interrupt request enables bits for all the 4 outputs in each LC defined in LC*n*_INTDMAEN with LC*n*_INTDMAEN[FORCE_INT_EN] for interrupt request generation when a force event occurs and LC*n*_INTDMAEN[LUT_INT_EN] for interrupt request generation when a LUT event occurs.

When user enables interrupt for a force event on an output and the force event occurs on that output, LCU generates an interrupt request if LC*n*_STS[FORCESTS] and LC*n*_INTDMAEN[FORCE_INT_EN] for that output is 1. Similarly when user enables interrupt for a LUT event on an output and a LUT event occurs on that output, LCU generates an interrupt request if LC*n*_STS[LUT_STS] and LC*n*_INTDMAEN[LUT_INT_EN] for that output is 1.

62.3.5 DMA

Each LC has a single DMA request output dma_req[n]. DMA request enables bits for all the 4 outputs in each LC defined in LC*n*_INTDMAEN with LC*n*_INTDMAEN[FORCE_DMA_EN] for DMA request generation when a force event occurs and LC*n*_INTDMAEN[LUT_DMA_EN] for DMA request generation when a LUT event occurs.

When user enables DMA for a force event on an output and the force event occurs on that output, LCU generates a DMA request if LC*n*_STS[FORCESTS] and LC*n*_INTDMAEN[FORCE_DMA_EN] for that output is 1. Similarly when user enables DMA for a LUT event on an output and a LUT event occurs on that output, LCU generates a DMA request if LC*n*_STS[LUT_STS] and LC*n*_INTDMAEN[LUT_DMA_EN] for that output is 1.

62.4 External signals

This module has no external signals.

62.5 Initialization

This module does not require initialization.

62.6 Application information

62.6.1 Use-case examples

62.6.1.1 Two-channel multiplexer

62.6.1.1.1 Overview

A multiplexer (or mux) takes multiple inputs and, based on some control mechanism, passes only one of those inputs to the output. You can implement a two-channel multiplexer controlled by an external signal or a software override using three inputs and one output of a single LC, as illustrated in Logic diagram and Implementation.

This example implementation controls the multiplexer with the SEL input signal and a software override. The software override controls the multiplexer output regardless of the SEL signal state. To use software override, you must enable it for the corresponding output (SWEN[SWEN]). You assert software override by writing 1 to the corresponding bit of the Software Override Value (SWVALUE[SWVALUE]).

62.6.1.1.2 Logic diagram

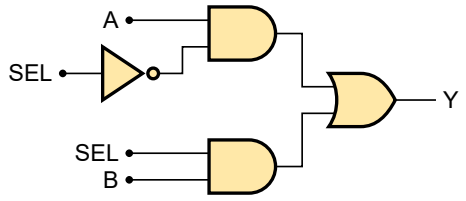


Figure 281. Logic diagram

62.6.1.1.3 Implementation

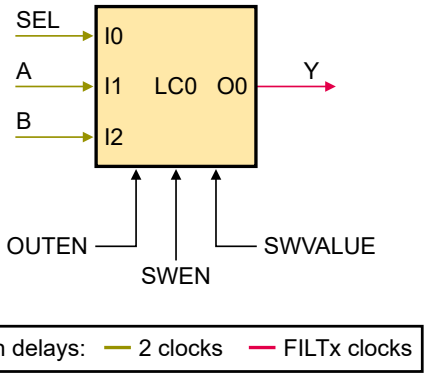


Figure 282. Implementation

62.6.1.1.4 Connections and controls

Table 359. Connections and controls

Name	Description
SEL	Input signal
A	
B	
OUTEN	Output enable
SWEN	Software override enable
SWVALUE	Software override
Y	Output signal

62.6.1.1.5 Truth table

Table 360. Truth table

Inputs			Output
A	B	SEL	Y
0	0	0	0
0	0	1	0
1	0	0	1
1	0	1	0
0	1	0	0
0	1	1	1
1	1	0	1
1	1	1	1

62.6.1.1.6 Register configuration

Table 361. Register configuration

Register	I3	I2	I1	I0	O0	O1	O2	O3
LUTCTRL0	—	—	—	—	E4E4h	—	—	—
LUTCTRL1	—	—	—	—	—	0h	—	—
LUTCTRL2	—	—	—	—	—	—	0h	—
LUTCTRL3	—	—	—	—	—	—	—	0h
FILT0	—	—	—	—	0h	—	—	—
FILT1	—	—	—	—	—	0h	—	—
FILT2	—	—	—	—	—	—	0h	—
FILT3	—	—	—	—	—	—	—	0h
INTDMAEN	—	—	—	—	0h			
OUTPOL	—	—	—	—	0h			
FFILT	—	—	—	—	0h			
FCTRL	—	—	—	—	0h			
SCTRL	—	—	—	—	0h			
MUXSEL0	—	—	—	1h	—	—	—	—
MUXSEL1	—	—	2h	—	—	—	—	—
MUXSEL2	—	3h	—	—	—	—	—	—
MUXSEL3	0h	—	—	—	—	—	—	—
SWEN	—	—	—	—	Disable software override for LC0 input 0: 0h			

Table continues on the next page...

Table 361. Register configuration (continued)

Register	I3	I2	I1	I0	O0	O1	O2	O3
					Enable software override for LC0 input 0: 1h			
SWVALUE	—	—	—	—	Deassert software override for LC0 input 0: 0h Assert software override for LC0 input 0: 1h			
OUTEN	—	—	—	—	1h			

62.6.1.1.7 Waveforms

In this figure, the SWEN and SWVALUE signals represent the states of the software override control registers. Writes to SWEN and SWVALUE have immediate impact on LC outputs, whereas LC inputs require input synchronization to prevent metastability.

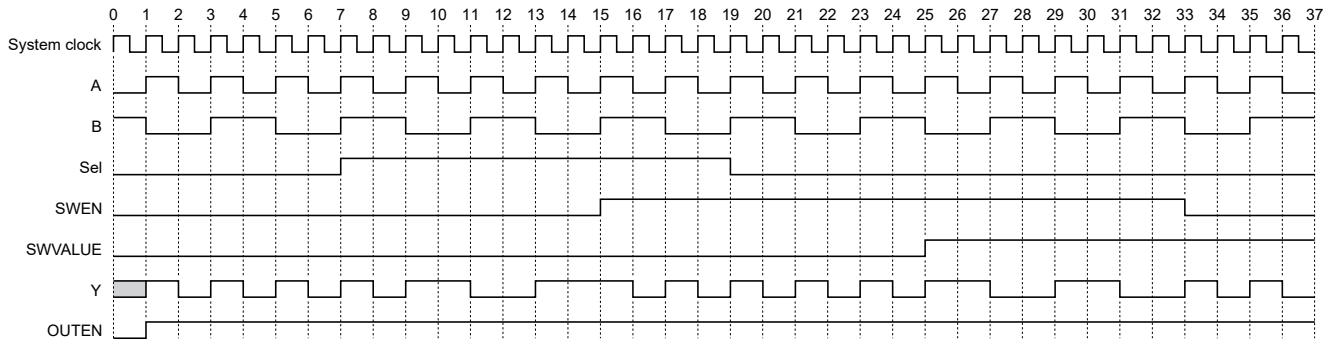


Figure 283. Waveforms

62.6.1.2 Binary to Gray code converter

62.6.1.2.1 Overview

Gray code, also known as reflected binary code (RBC), is a non-weighted, minimum change code. In this code, any two adjacent code numbers differ by only one bit. This code solves the problem of physical switch transitions by changing only one switch at a time, so there is never any ambiguity of position.

You can implement binary to Gray code conversion using a single LC, as illustrated in Logic diagram and Implementation.

In this example, the LC also generates a DMA request on each rising edge of output 3, which is bit 0 of the Gray code (G0).

62.6.1.2.2 Conversion table

Table 362. Conversion table

Decimal	Binary	Gray
0	0000	0000
1	0001	0001
2	0010	0011
3	0011	0010
4	0100	0110

Table continues on the next page...

Table 362. Conversion table (continued)

Decimal	Binary	Gray
5	0101	0111
6	0110	0101
7	0111	0100
8	1000	1100
9	1001	1101
10	1010	1111
11	1011	1110
12	1100	1010
13	1101	1011
14	1110	1001
15	1111	1000

62.6.1.2.3 Logic diagram

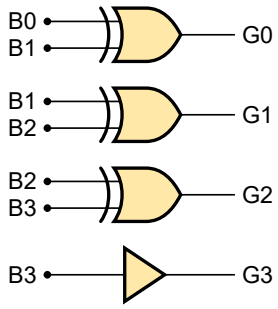


Figure 284. Logic diagram

62.6.1.2.4 Implementation

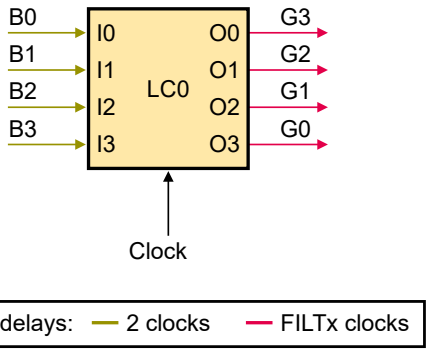


Figure 285. Implementation

62.6.1.2.5 Connections and controls

Table 363. Connections and controls

Name	Description
B0	Bit 0 of the input binary code
B1	Bit 1 of the input binary code
B2	Bit 2 of the input binary code
B3	Bit 3 of the input binary code
G0	Bit 0 of the output Gray code
G1	Bit 1 of the output Gray code
G2	Bit 2 of the output Gray code
G3	Bit 3 of the output Gray code
Clock	System clock input

62.6.1.2.6 Truth table

Table 364. Truth table

Inputs				Outputs			
B3	B2	B1	B0	G3	G2	G1	G0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	1
0	0	1	1	0	0	1	0
0	1	0	0	0	1	1	0
0	1	0	1	0	1	1	1
0	1	1	0	0	1	0	1
0	1	1	1	0	1	0	0
1	0	0	0	1	1	0	0
1	0	0	1	1	1	0	1
1	0	1	0	1	1	1	1
1	0	1	1	1	1	1	0
1	1	0	0	1	0	1	0
1	1	0	1	1	0	1	1
1	1	1	0	1	0	0	1
1	1	1	1	1	0	0	0

62.6.1.2.7 Register configuration

Table 365. Register configuration

Register	I3	I2	I1	I0	O0	O1	O2	O3
LUTCTRL0	—	—	—	—	FF00h	—	—	—
LUTCTRL1	—	—	—	—	—	FF0h	—	—
LUTCTRL2	—	—	—	—	—	—	3C3Ch	—
LUTCTRL3	—	—	—	—	—	—	—	6666h
FILT0	—	—	—	—	0h	—	—	—
FILT1	—	—	—	—	—	0h	—	—
FILT2	—	—	—	—	—	—	0h	—
FILT3	—	—	—	—	—	—	—	0h
INTDMAEN	—	—	—	—	800h			
OUTPOL	—	—	—	—	0h			
FFILT	—	—	—	—	0h			
FCTRL	—	—	—	—	0h			
SCTRL	—	—	—	—	0h			
MUXSEL0h	—	—	—	01h	—	—	—	—
MUXSEL1	—	—	2h	—	—	—	—	—
MUXSEL2	—	3h	—	—	—	—	—	—
MUXSEL3	4h	—	—	—	—	—	—	—
SWEN	—	—	—	—	0h			
SWVALUE	—	—	—	—	0h			
OUTEN	—	—	—	—	Fh			

62.6.1.2.8 Waveforms

The LC outputs are delayed by two clock cycles because of the two-stage input synchronizers.

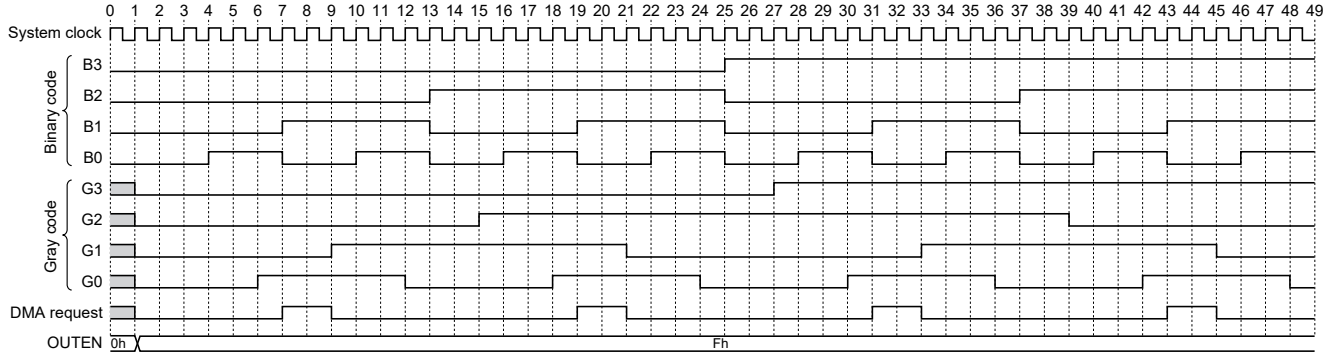


Figure 286. Waveforms

62.6.1.3 Manchester encoder and decoder

62.6.1.3.1 Overview

Manchester encoding (also known as phase encoding) is a data-modulation technique for binary data transfer based one of these signals:

- Analog
- RF
- Optical
- High-speed-digital
- Long-distance-digital

The fundamental idea behind Manchester encoding is to use voltage transitions, instead of voltage levels, to represent 1s and 0s. It performs an exclusive OR of data and clock signals to encode and decode data. A high-to-low transition on the falling clock edge indicates 0. A low-to-high transition on the rising clock edge indicates 1.

62.6.1.3.2 Logic diagram

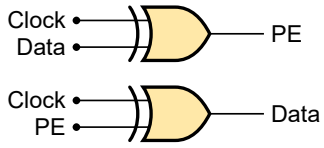


Figure 287. Logic diagram

62.6.1.3.3 Implementation

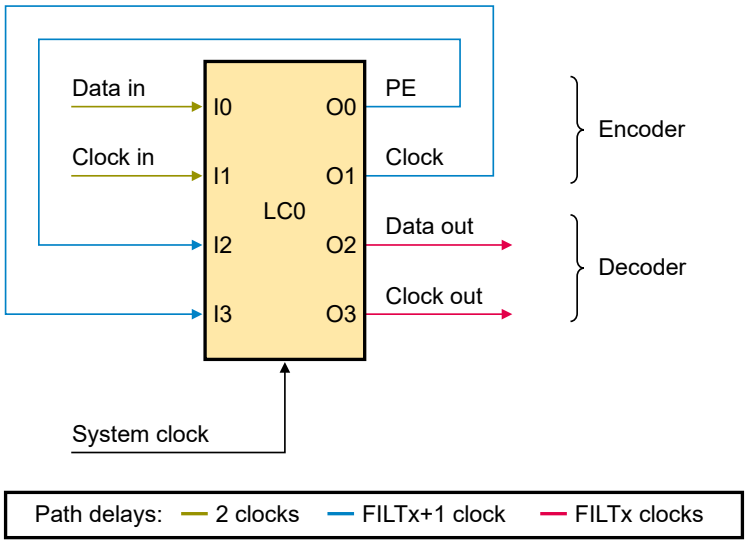


Figure 288. Implementation

62.6.1.3.4 Connections and controls

Table 366. Connections and controls

Name	Description
Data in	Input signal
Clock in	
PE	Manchester-encoded signal from LC
Clock	Auxiliary signal fed to the decoder to allow Manchester decoding
Data out	Output signal after Manchester decoding
System clock	Clock frequency supplied to LC
Clock out	Output signal

62.6.1.3.5 Truth table for encoder

Table 367. Truth table for encoder

Inputs		Output
Data in	Clock in	PE
0	0	0
0	1	1
1	0	1
1	1	0

62.6.1.3.6 Truth table for decoder

Table 368. Truth table for decoder

Inputs		Output
Clock	PE	Data out
0	0	0
0	1	1
1	0	1
1	1	0

62.6.1.3.7 Register configuration

Table 369. Register configuration

Register	I3	I2	I1	I0	O0	O1	O2	O3
LUTCTRL0	—	—	—	—	6666h	—	—	—
LUTCTRL1	—	—	—	—	—	AAAAh	—	—
LUTCTRL2	—	—	—	—	—	—	FF0h	—

Table continues on the next page...

Table 369. Register configuration (continued)

Register	I3	I2	I1	I0	O0	O1	O2	O3
LUTCTRL3	—	—	—	—	—	—	—	FF00h
FILT0	—	—	—	—	0h	—	—	—
FILT1	—	—	—	—	—	0h	—	—
FILT2	—	—	—	—	—	—	0h	—
FILT3	—	—	—	—	—	—	—	0h
INTDMAEN	—	—	—	—	0h			
OUTPOL	—	—	—	—	0h			
FFILT	—	—	—	—	0h			
FCTRL	—	—	—	—	0h			
SCTRL	—	—	—	—	0h			
MUXSEL0	—	—	—	1h	—	—	—	—
MUXSEL1	—	—	2h	—	—	—	—	—
MUXSEL2	—	5h	—	—	—	—	—	—
MUXSEL3	6h	—	—	—	—	—	—	—
SWEN	—	—	—	—	0h			
SWVALUE	—	—	—	—	0h			
OUTEN	—	—	—	—	Fh			

62.6.1.3.8 Waveforms for encoder

This figure includes the Manchester-encoded waveform (labeled PE) of the input waveform (labeled "Data in"). The PE waveform is delayed by two clock cycles because of two-stage input synchronization.

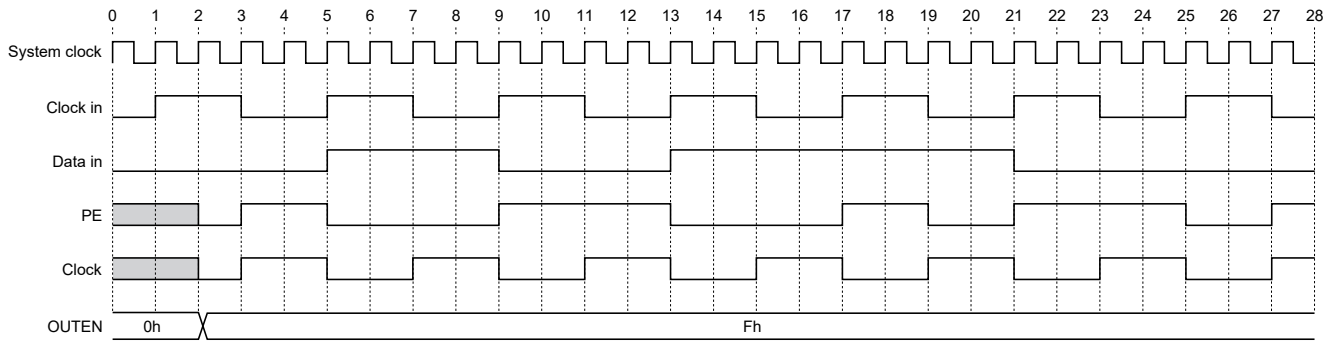


Figure 289. Waveforms for encoder

62.6.1.3.9 Waveforms for decoder

The output waveform (labeled "Data out") is delayed by four clock cycles from the input waveform (labeled "Data in") because of two-stage input synchronization. In this example, the Manchester encoder also propagates the Clock waveform that the decoder uses to reconstruct the "Data in" waveform from the Manchester-decoded waveform (labeled PE).

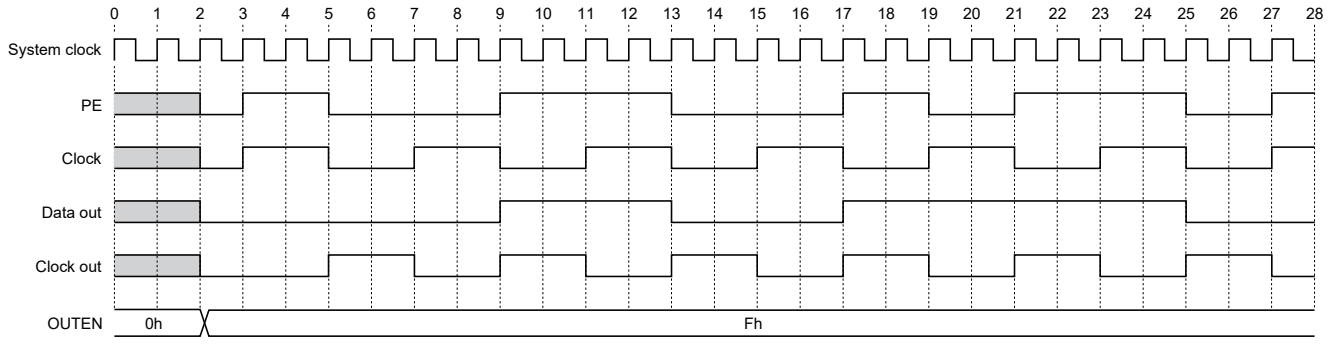


Figure 290. Waveforms for decoder

62.6.1.4 Adjustable PWM/FM generator

62.6.1.4.1 Overview

In many cases, it is useful to route an LC output back to one of its inputs, as illustrated in Implementation. In addition, you can delay outputs with a digital filter with different time constants for delaying rising and falling edges. Combining these capabilities, you can use an LC to generate output waveforms with adjustable pulse width and frequency.

In addition to waveform generation, the software-override feature forces the generator output low or high by software. You control the override via Software Override Enable (SWEN) and Software Override Value (SWVALUE).

62.6.1.4.2 Implementation

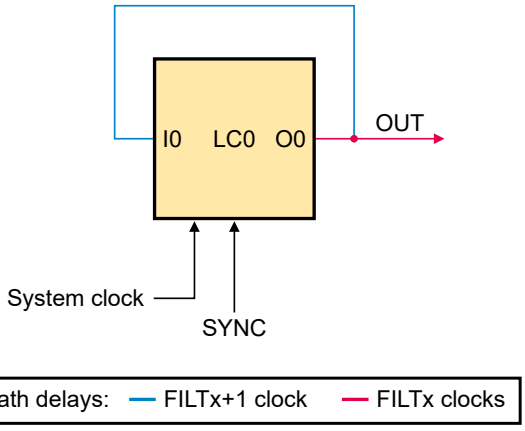


Figure 291. Implementation

62.6.1.4.3 Connections and controls

Table 370. Connections and controls

Name	Description
I0	Delayed output routed back to LC input
O0 and OUT	Delayed output
System clock	Clock frequency supplied to LC
SYNC	Software override synchronization signal

62.6.1.4.4 Truth table

This example generates an adjustable PWM/FM waveform according to the following table. In the table:

- OUT is the LC output signal that is also routed to the LC input using an internal multiplexer.
- TdH is the digital filter value to delay the rising edge of the output signal.
- TdL is the digital filter value to delay the falling edge of the output signal.

Table 371. Truth table

I0	OUT
0	1 delayed by TdH (see Output edge delays)
1	0 delayed by TdL (see Output edge delays)

62.6.1.4.5 Output edge delays

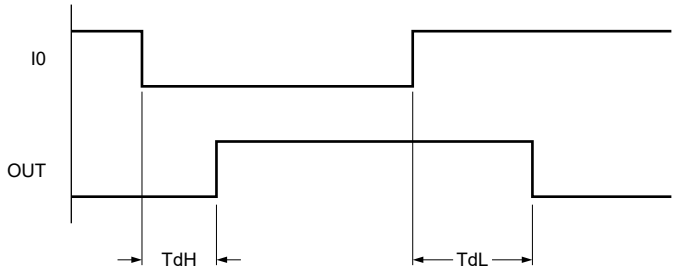


Figure 292. Output edge delays

62.6.1.4.6 Register configuration

Table 372. Register configuration

Register	I3	I2	I1	I0	O0	O1	O2	O3
LUTCTRL0	—	—	—	—	1h	—	—	—
LUTCTRL1	—	—	—	—	—	0h	—	—
LUTCTRL2	—	—	—	—	—	—	0h	—
LUTCTRL3	—	—	—	—	—	—	—	0h
FILT0	—	—	—	—	1_0001h	—	—	—
FILT1	—	—	—	—	—	0h	—	—
FILT2	—	—	—	—	—	—	0h	—
FILT3	—	—	—	—	—	—	—	0h
INTDMAEN	—	—	—	—	0h			
OUTPOL	—	—	—	—	0h			
FFILT	—	—	—	—	0h			
FCTRL	—	—	—	—	0h			

Table continues on the next page...

Table 372. Register configuration (continued)

Register	I3	I2	I1	I0	O0	O1	O2	O3
SCTRL	—	—	—	—	SWVALUE for input 0 changes immediately: 0h SWVALUE for input 0 changes on rising edge of sync: 1h			
MUXSEL0	—	—	—	5h	—	—	—	—
MUXSEL1	—	—	0h	—	—	—	—	—
MUXSEL2	—	0h	—	—	—	—	—	—
MUXSEL3	0h	—	—	—	—	—	—	—
SWEN	—	—	—	—	Disable software override: 0h Enable software override: 1h			
SWVALUE	—	—	—	—	Deassert software override: 0h Assert software override: 1h			
OUTEN	—	—	—	—	1h			

62.6.1.4.7 Waveforms for immediate software override

In this figure, the SWEN and SWVALUE signals represent the states of the software override control registers. Writes to Software Override Enable (SWEN) and Software Override Value (SWVALUE) have immediate impact on LC outputs, whereas LC inputs require input synchronization to prevent metastability.

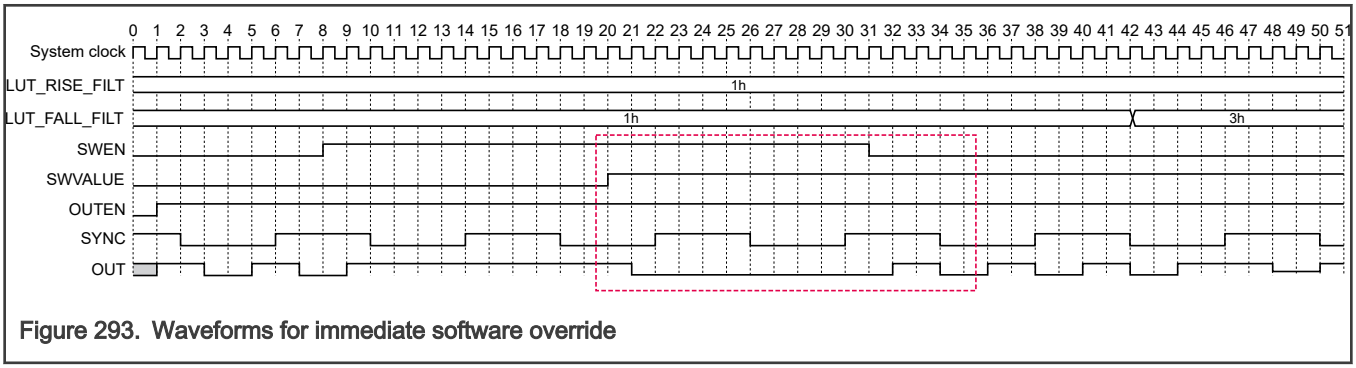


Figure 293. Waveforms for immediate software override

62.6.1.4.8 Waveforms for synced software override

In this example, software override control is synced with the rising edges of the SYNC waveform.

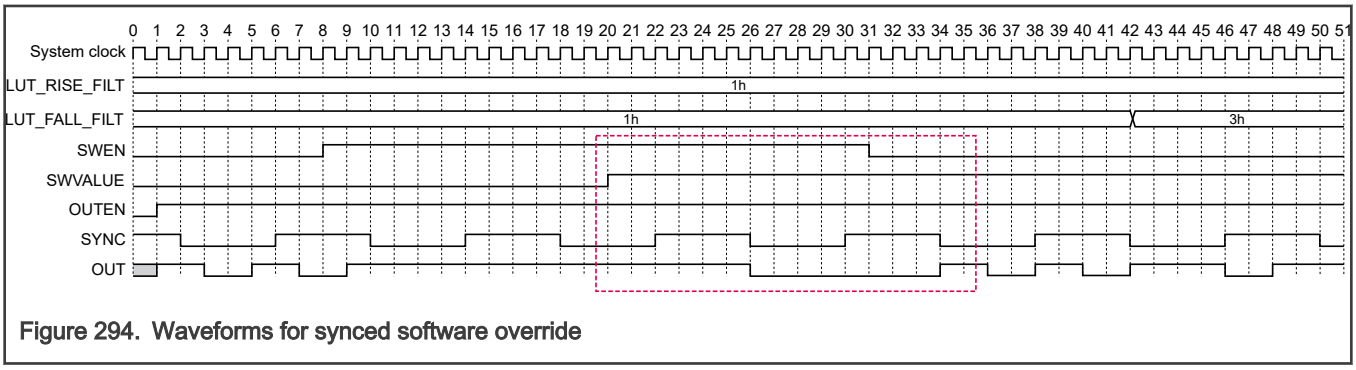


Figure 294. Waveforms for synced software override

62.6.1.5 SR latch

62.6.1.5.1 Overview

An SR latch is the simplest bistable device and a fundamental component of data storage.

This example implementation uses one LC that includes an internal feedback path configured by the internal multiplexer, as illustrated in Implementation. Output Q responds to inputs S (set) and R (reset). It changes with a delay of two clock cycles—the time required for the S and R signal inputs to pass through the internal two-stage synchronizers. This example implementation bypasses the Q digital filter; therefore, the loop of Q back to the input experiences a delay of one clock cycle." See Waveforms.

62.6.1.5.2 Implementation

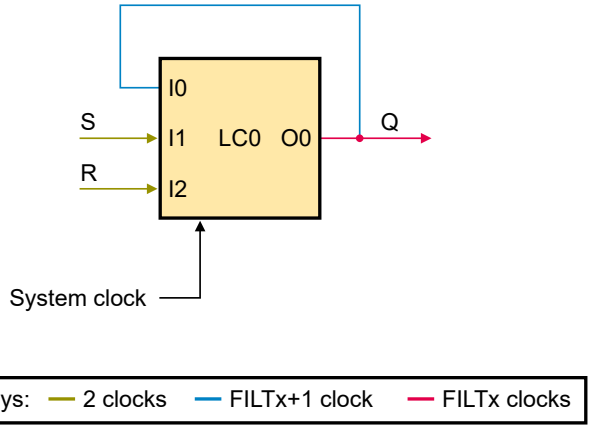


Figure 295. Implementation

62.6.1.5.3 Connections and controls

Table 373. Connections and controls

Name	Description
S	Set input
R	Reset input
Q	Both an output signal and an input that feeds back to LC0
System clock	Clock frequency supplied to LC

62.6.1.5.4 Truth table

Table 374. Truth table

Inputs			Output	State
R	S	Q	Q	
0	0	0	0	Latch
0	0	1	1	Latch
0	1	0	1	Set

Table continues on the next page...

Table 374. Truth table (continued)

Inputs			Output	State
R	S	Q	Q	
0	1	1	1	Set
1	0	0	0	Reset
1	0	1	0	Reset
1	1	0	0	Not allowed
1	1	1	0	

62.6.1.5.5 Register configuration

Table 375. Register configuration

Register	I3	I2	I1	I0	O0	O1	O2	O3
LUTCTRL0	—	—	—	—	Eh	—	—	—
LUTCTRL1	—	—	—	—	—	0h	—	—
LUTCTRL2	—	—	—	—	—	—	0h	—
LUTCTRL3	—	—	—	—	—	—	—	0h
FILT0	—	—	—	—	0h	—	—	—
FILT1	—	—	—	—	—	0h	—	—
FILT2	—	—	—	—	—	—	0h	—
FILT3	—	—	—	—	—	—	—	0h
INTDMAEN	—	—	—	—	0h			
OUTPOL	—	—	—	—	0h			
FFILT	—	—	—	—	0h			
FCTRL	—	—	—	—	0h			
SCTRL	—	—	—	—	0h			
MUXSEL0	—	—	—	5h	—	—	—	—
MUXSEL1	—	—	2h	—	—	—	—	—
MUXSEL2	—	3h	—	—	—	—	—	—
MUXSEL3	0h	—	—	—	—	—	—	—
SWEN	—	—	—	—	0h			
SWVALUE	—	—	—	—	0h			
OUTEN	—	—	—	—	1h			

62.6.1.5.6 Waveforms

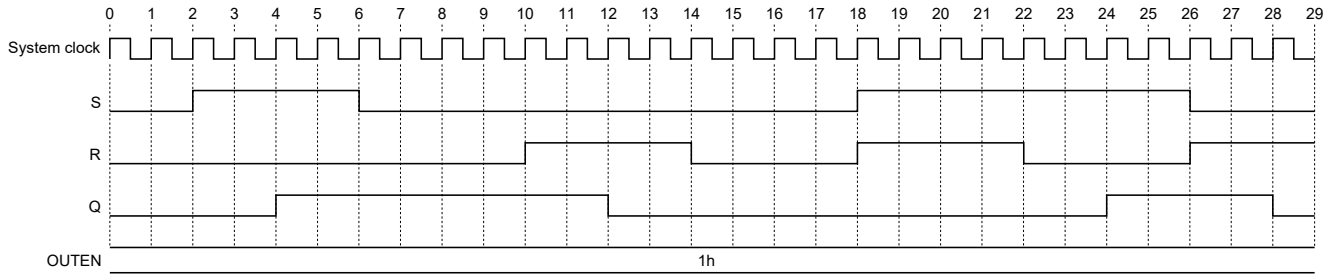


Figure 296. Waveforms

62.6.1.6 D flip-flop

62.6.1.6.1 Overview

A D flip-flop stores and outputs whatever logic level is applied to its data terminal at the time its clock input goes high.

You can implement a D flip-flop element using an LC as illustrated in Implementation. On a clock rising edge, the LC either asserts or deasserts the Q output based on the D input.

The Q output and CLKO digital filters are bypassed. Therefore, the states of these variables are fed to inputs with a delay of one clock cycle. See Waveforms.

62.6.1.6.2 Implementation

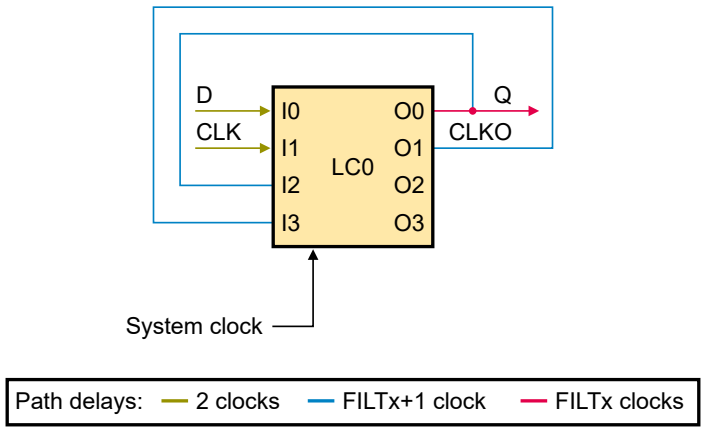


Figure 297. Implementation

62.6.1.6.3 Connections and controls

Table 376. Connections and controls

Name	Description
D	Input signal
CLK	Input clock
Q	Output signal; holds the logic state of the last Q output
CLKO	Holds the logic state of the last CLK input

62.6.1.6.4 Truth table

Table 377. Truth table

Inputs				Outputs		State
CLKO	Q	CLK	D	Q	CLKO	
0	0	0	0	0	0	Hold
0	0	0	1	0	0	Hold
0	0	1	0	0	1	Reset
0	0	1	1	1	1	Set
0	1	0	0	1	0	Hold
0	1	0	1	1	0	Hold
0	1	1	0	0	1	Reset
0	1	1	1	1	1	Set
1	0	0	0	0	0	Hold
1	0	0	1	0	0	Hold
1	0	1	0	0	1	Hold
1	0	1	1	0	1	Hold
1	1	0	0	1	0	Hold
1	1	0	1	1	0	Hold
1	1	1	0	1	1	Hold
1	1	1	1	1	1	Hold

62.6.1.6.5 Register configuration

Table 378. Register configuration

Register	I3	I2	I1	I0	O0	O1	O2	O3
LUTCTRL0	—	—	—	—	F0B8h	—	—	—
LUTCTRL1	—	—	—	—	—	CCCCh	—	—
LUTCTRL2	—	—	—	—	—	—	0h	—
LUTCTRL3	—	—	—	—	—	—	—	0h
FILT0	—	—	—	—	0h	—	—	—
FILT1	—	—	—	—	—	0h	—	—
FILT2	—	—	—	—	—	—	0h	—
FILT3	—	—	—	—	—	—	—	0h
INTDMAEN	—	—	—	—	0h			
OUTPOL	—	—	—	—	0h			
FFILT	—	—	—	—	0h			

Table continues on the next page...

Table 378. Register configuration (continued)

Register	I3	I2	I1	I0	O0	O1	O2	O3
FCTRL	—	—	—	—	0h			
SCTRL	—	—	—	—	0h			
MUXSEL0	—	—	—	1h	—	—	—	—
MUXSEL1	—	—	2h	—	—	—	—	—
MUXSEL2	—	5h	—	—	—	—	—	—
MUXSEL3	6h	—	—	—	—	—	—	—
SWEN	—	—	—	—	0h			
SWVALUE	—	—	—	—	0h			
OUTEN	—	—	—	—	3h			

62.6.1.6.6 Waveforms

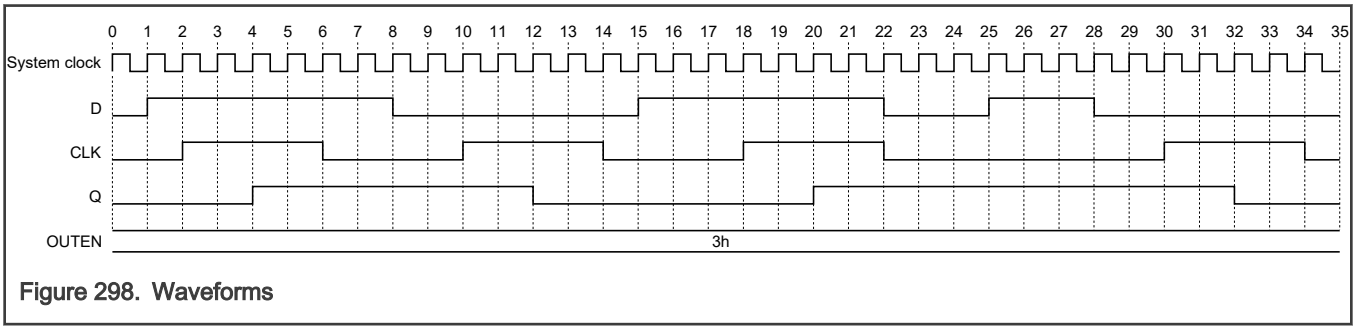


Figure 298. Waveforms

62.6.1.7 JK flip-flop

62.6.1.7.1 Overview

A JK flip-flop is a gated SR latch (see SR latch) with an added clock input circuit that prevents the illegal or invalid output condition that can occur when both SR-latch inputs (S and R) are asserted.

Truth table for JK flip-flop presents the four possible input combinations for a JK flip-flop. All output changes occur at the clock rising edge.

You can create a JK flip-flop using two LCs, as illustrated in Implementation. The first LC detects rising edges of the CLK clock input (see Truth table for LC0). The second LC processes J and K inputs at every rising clock edge (see Truth table for LC1).

Output Q and CLKO digital filters are bypassed. Therefore, the states of these variables are fed to inputs with a delay of one clock cycle (see Waveforms).

62.6.1.7.2 Implementation

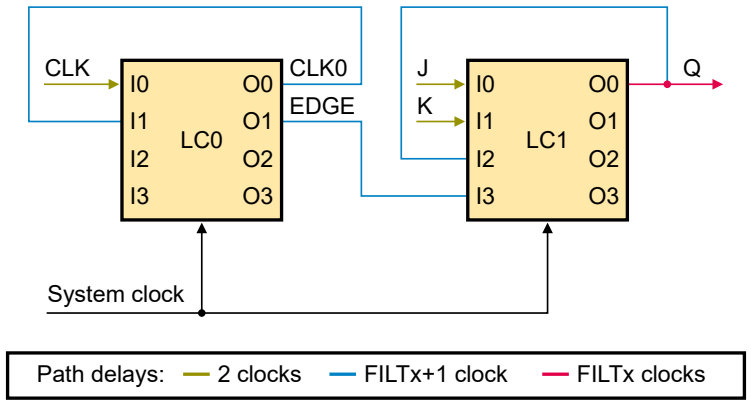


Figure 299. Implementation

62.6.1.7.3 Truth table for JK flip-flop

Table 379. Truth table for JK flip-flop

J	K	State
0	0	Hold
1	0	Set
0	1	Clear
1	1	Toggle

62.6.1.7.4 Truth table for LC0

Table 380. Truth table for LC0

Inputs		Outputs		State
CLKO	CLK	CLKO	EDGE	
0	0	0	0	No edge
0	1	1	1	Edge detected
1	0	0	0	No edge
1	1	1	0	No edge

62.6.1.7.5 Truth table for LC1

Table 381. Truth table for LC1

Inputs				Output	State
EDGE	Q	J	K	Q	
0	0	X	X	0	Hold
0	1	X	X	1	Hold

Table continues on the next page...

Table 381. Truth table for LC1 (continued)

Inputs				Output	State
EDGE	Q	J	K	Q	
1	0	0	0	0	Hold
1	1	0	0	1	Hold
1	X	1	0	1	Set
1	X	0	1	0	Clear
1	0	1	1	1	Toggle
1	1	1	1	0	Toggle

62.6.1.7.6 Register configuration for LC0

Table 382. Register configuration for LC0

Register	I3	I2	I1	I0	O0	O1	O2	O3
LUTCTRL0	—	—	—	—	Ah	—	—	—
LUTCTRL1	—	—	—	—	—	2h	—	—
LUTCTRL2	—	—	—	—	—	—	0h	—
LUTCTRL3	—	—	—	—	—	—	—	0h
FILT0	—	—	—	—	0h	—	—	—
FILT1	—	—	—	—	—	0h	—	—
FILT2	—	—	—	—	—	—	0h	—
FILT3	—	—	—	—	—	—	—	0h
INTDMAEN	—	—	—	—	0h			
OUTPOL	—	—	—	—	0h			
FFILT	—	—	—	—	0h			
FCTRL	—	—	—	—	0h			
SCTRL	—	—	—	—	0h			
MUXSEL0	—	—	—	1h	—	—	—	—
MUXSEL1	—	—	9h	—	—	—	—	—
MUXSEL2	—	0h	—	—	—	—	—	—
MUXSEL3	0h	—	—	—	—	—	—	—
SWEN	—	—	—	—	0h			
SWVALUE	—	—	—	—	0h			
OUTEN	—	—	—	—	3h			

62.6.1.7.7 Register configuration for LC1

Table 383. Register configuration for LC1

Register	I3	I2	I1	I0	O0	O1	O2	O3
LUTCTRL0	—	—	—	—	3AF0h	—	—	—
LUTCTRL1	—	—	—	—	—	0h	—	—
LUTCTRL2	—	—	—	—	—	—	0h	—
LUTCTRL3	—	—	—	—	—	—	—	0h
FILT0	—	—	—	—	0h	—	—	—
FILT1	—	—	—	—	—	0h	—	—
FILT2	—	—	—	—	—	—	0h	—
FILT3	—	—	—	—	—	—	—	0h
INTDMAEN	—	—	—	—	0h			
OUTPOL	—	—	—	—	0h			
FFILT	—	—	—	—	0h			
FCTRL	—	—	—	—	0h			
SCTRL	—	—	—	—	0h			
MUXSEL4	—	—	—	5h	—	—	—	—
MUXSEL5	—	—	6h	—	—	—	—	—
MUXSEL6	—	Dh	—	—	—	—	—	—
MUXSEL7	Ah	—	—	—	—	—	—	—
SWEN	—	—	—	—	0h			
SWVALUE	—	—	—	—	0h			
OUTEN	—	—	—	—	10h			

62.6.1.7.8 Waveforms

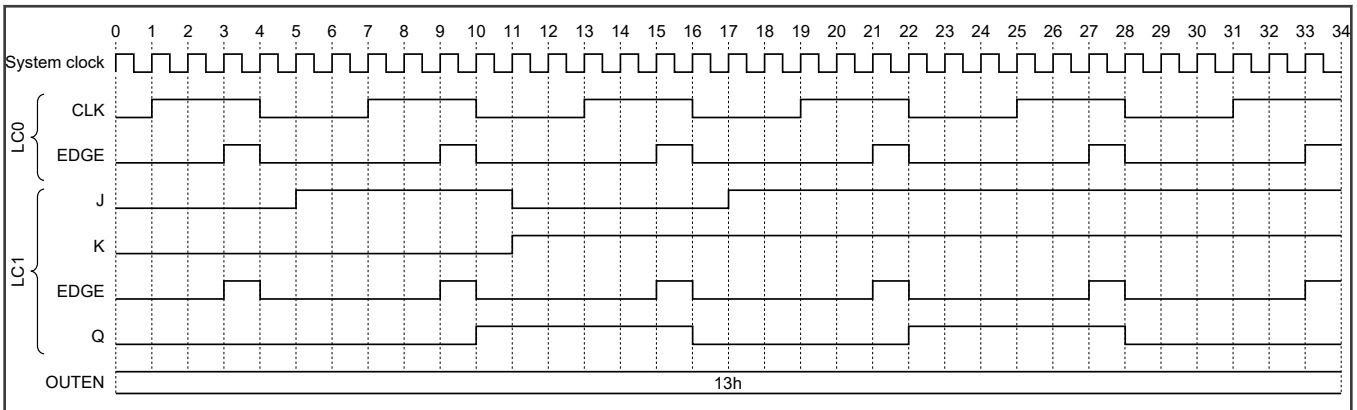


Figure 300. Waveforms

62.6.1.8 Incremental encoder

62.6.1.8.1 Overview

An incremental encoder provides A and B pulse outputs in the form of 90-degree shifted waveforms (see Outputs). The decoding circuit processes these waveforms to detect the rotor position and the direction of rotation.

This example implementation requires two LCs (see Implementation):

- LC0 provides pulse streams on CW or CCW output at every A and B signal edge. If the encoder rotation is CW (A signal leading), then pulses appear at the CW output. If the encoder rotation is CCW (B signal leading), then pulses appear at the CCW output. Two pulse counters accumulate these pulse streams to provide the actual absolute encoder position in the respective direction of rotation.
- LC1 then uses the LC0 outputs to detect the direction of the rotation.

62.6.1.8.2 Outputs

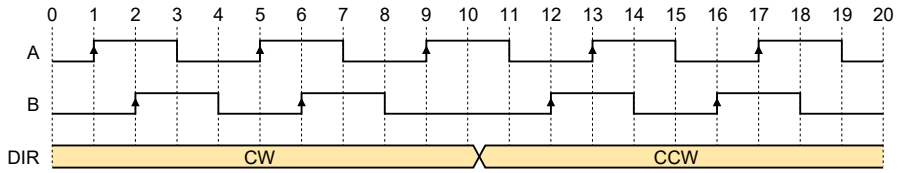


Figure 301. Outputs

62.6.1.8.3 Implementation

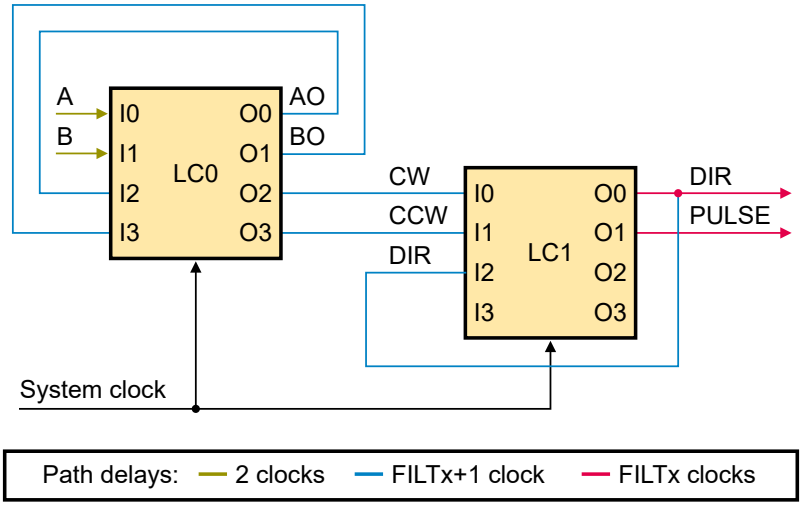


Figure 302. Implementation

62.6.1.8.4 Truth table for pulse generation

Table 384. Truth table for pulse generation

Inputs				Outputs			
BO	AO	B	A	AO	BO	CW	CCW
0	0	0	0	0	0	—	—
0	0	0	1	1	0	1	—
0	0	1	0	0	1	—	1
0	0	1	1	1	1		
0	1	0	0	0	0	—	1
0	1	0	1	1	0		
0	1	1	0	0	1	—	—
0	1	1	1	1	1	1	—
1	0	0	0	0	0	1	
1	0	0	1	1	0	—	—
1	0	1	0	0	1	—	—
1	0	1	1	1	1	—	1
1	1	0	0	0	0		
1	1	0	1	1	0	—	1
1	1	1	0	0	1	1	—
1	1	1	1	1	1	—	—

62.6.1.8.5 Truth table for decoding direction of rotation

In this table:

- DIR = 0 represents a CCW direction.
- DIR = 1 represents a CW direction.

Table 385. Truth table for decoding direction of rotation

Inputs			Output
DIR	CCW	CW	DIR
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	1

62.6.1.8.6 Register configuration for LC0

Table 386. Register configuration for LC0

Register	I3	I2	I1	I0	O0	O1	O2	O3
LUTCTRL0	—	—	—	—	AAAAh	—	—	—
LUTCTRL1	—	—	—	—	—	CCCCh	—	—
LUTCTRL2	—	—	—	—	—	—	4182h	—
LUTCTRL3	—	—	—	—	—	—	—	2814h
FILT0	—	—	—	—	1_0001h	—	—	—
FILT1	—	—	—	—	—	1_0001h	—	—
FILT2	—	—	—	—	—	—	0h	—
FILT3	—	—	—	—	—	—	—	0h
INTDMAEN	—	—	—	—	0h			
OUTPOL	—	—	—	—	0h			
FFILT	—	—	—	—	0h			
FCTRL	—	—	—	—	0h			
SCTRL	—	—	—	—	0h			
MUXSEL0	—	—	—	1h	—	—	—	—
MUXSEL1	—	—	2h	—	—	—	—	—
MUXSEL2	—	9h	—	—	—	—	—	—
MUXSEL3	Ah	—	—	—	—	—	—	—
SWEN	—	—	—	—	0h			
SWVALUE	—	—	—	—	0h			
OUTEN	—	—	—	—	Fh			

62.6.1.8.7 Register configuration for LC1

Table 387. Register configuration for LC1

Register	I3	I2	I1	I0	O0	O1	O2	O3
LUTCTRL0	—	—	—	—	BAh	—	—	—
LUTCTRL1	—	—	—	—	—	0h	—	—
LUTCTRL2	—	—	—	—	—	—	0h	—
LUTCTRL3	—	—	—	—	—	—	—	0h
FILT0	—	—	—	—	0h	—	—	—
FILT1	—	—	—	—	—	0h	—	—
FILT2	—	—	—	—	—	—	0h	—
FILT3	—	—	—	—	—	—	—	0h

Table continues on the next page...

Table 387. Register configuration for LC1 (continued)

Register	I3	I2	I1	I0	O0	O1	O2	O3
INTDMAEN	—	—	—	—	0h			
OUTPOL	—	—	—	—	0h			
FFILT	—	—	—	—	0h			
FCTRL	—	—	—	—	0h			
SCTRL	—	—	—	—	0h			
MUXSEL4	—	—	—	Bh	—	—	—	—
MUXSEL5	—	—	Ch	—	—	—	—	—
MUXSEL6	—	Dh	—	—	—	—	—	—
MUXSEL7	0h	—	—	—	—	—	—	—
SWEN	—	—	—	—	0h			
SWVALUE	—	—	—	—	0h			
OUTEN	—	—	—	—	10h			

62.6.1.8.8 Waveforms

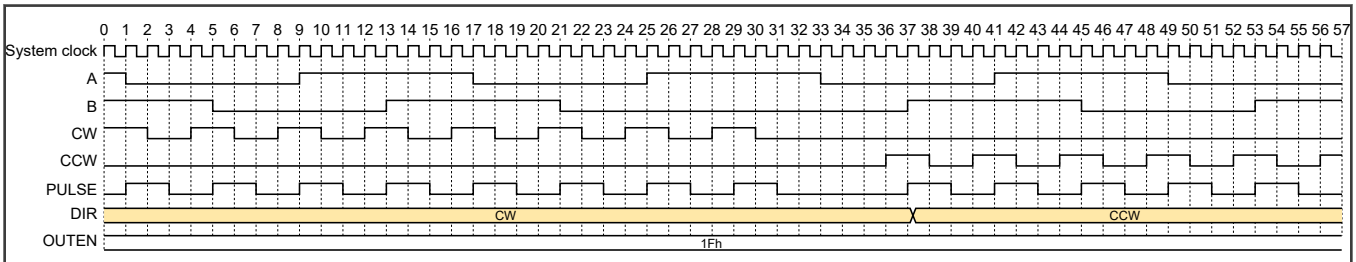


Figure 303. Waveforms

62.6.1.9 AC motor PWM controller

62.6.1.9.1 Overview

Driving a three-phase AC motor requires dedicated PWMs to perform:

- Complementary signal generation
- Dead time insertion
- Fault-state assertion
- Manual or automatic fault recovery

LCU enhances basic timer functionality with the necessary complementary features available only on dedicated motor control PWMs.

Implementation for AC motor controller shows all input and output signals of LCU in a typical three-phase motor control application.

TdH is the time delay preceding each rising edge of the complementary PWM output signals. See Truth table for details.

Fault signals are routed from analog comparators or external protection circuits to protect power switches against damage caused by overcurrent. Any active fault must immediately change all PWM outputs to the inactive state. Force logic causes PWM outputs to transition to the inactive state upon an external fault event. See Truth table for details.

The dead time is one clock cycle. On a fault event, both PWM outputs transition to the inactive state (logic 0). When the fault event deasserts, both PWM signals start operating according to FORCE_MODE = 00 (cleared on force deassertion). See Waveforms for force cleared on force deassertion for details.

62.6.1.9.2 Implementation for AC motor controller

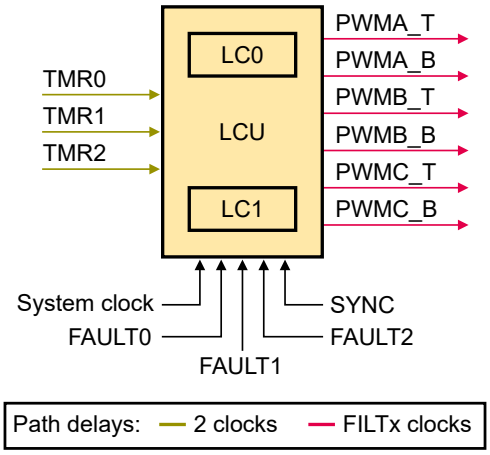


Figure 304. Implementation for AC motor controller

62.6.1.9.3 Implementation for controlling one phase of AC motor controller

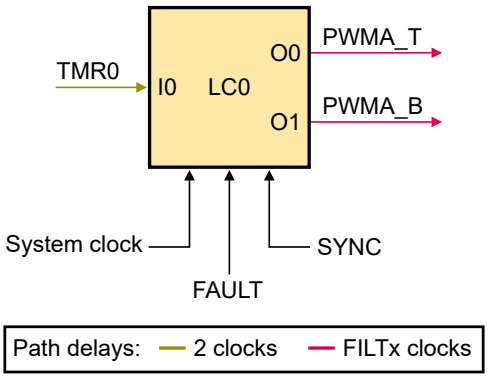


Figure 305. Implementation for controlling one phase of AC motor controller

62.6.1.9.4 Connections and controls

Table 388. Connections and controls

Name	Description
TMR*n*	Synchronized timer input signals
System clock	Clock supplied to the LC
FAULT*n*	Fault signal (see Overview)

Table continues on the next page...

Table 388. Connections and controls (continued)

Name	Description
SYNC	Sync signal for optional automatic fault recovery
PWM*x*_T	PWM output signal for controlling top power switch
PWM*x*_B	PWM output signal for controlling bottom power switch

62.6.1.9.5 Truth table

Table 389. Truth table

Input	Outputs	
TMR0	PWMA_T	PWMB_B
0	0	1 delayed by TdH (see Output edge delays)
1	1 delayed by TdH (see Output edge delays)	0

62.6.1.9.6 Register configuration

Table 390. Register configuration

Register	I3	I2	I1	I0	O0	O1	O2	O3
LUTCTRL0	—	—	—	—	2h	—	—	—
LUTCTRL1	—	—	—	—	—	1h	—	—
LUTCTRL2	—	—	—	—	—	—	0h	—
LUTCTRL3	—	—	—	—	—	—	—	0h
FILT0	—	—	—	—	1_0000h	—	—	—
FILT1	—	—	—	—	—	1_0000h	—	—
FILT2	—	—	—	—	—	—	0h	—
FILT3	—	—	—	—	—	—	—	0h
INTDMAEN	—	—	—	—	0h			
OUTPOL	—	—	—	—	0h			
FFILT	—	—	—	—	100_0000h			
FCTRL	—	—	—	—	0h			
SCTRL	—	—	—	—	Software sync on rising edge for inputs 2 and 0, immediate for 3 and 1: 101h Software sync on rising edge for all four inputs: 1111h			
MUXSEL0	—	—	—	01h	—	—	—	—
MUXSEL1	—	—	0h	—	—	—	—	—
MUXSEL2	—	0h	—	—	—	—	—	—
MUXSEL3	0h	—	—	—	—	—	—	—

Table continues on the next page...

Table 390. Register configuration (continued)

Register	I3	I2	I1	I0	O0	O1	O2	O3
SWEN	—	—	—	—	0h			
SWVALUE	—	—	—	—	0h			
OUTEN	—	—	—	—	3h			

62.6.1.9.7 Waveforms for force cleared on force deassertion

This example shows two behaviors, depending on the Force Clearing mode (LC0_FCTRL[FORCE_MODE0]).

00b - Both PWM signals start normal operation when the fault event deasserts. See Waveforms for force cleared on force deassertion for details.

01b - Both PWM signals start normal operation on rising sync after the fault event deasserts. See Waveforms for force cleared on rising sync after force deassertion for details.

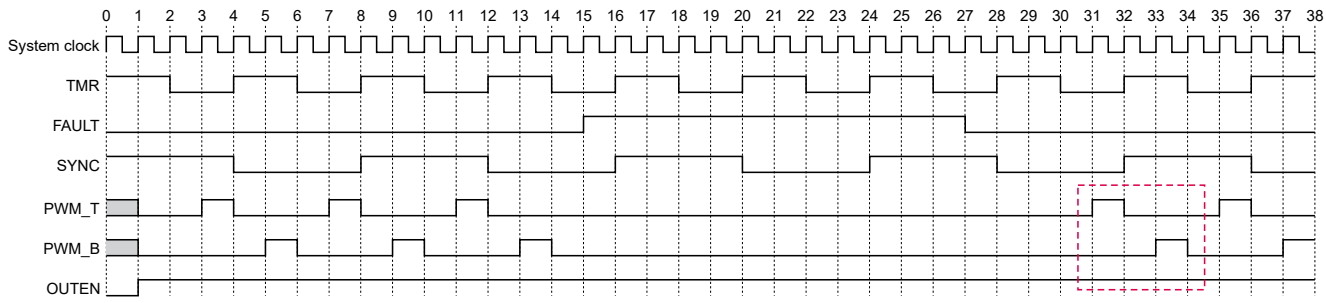


Figure 306. Waveforms for force cleared on force deassertion

62.6.1.9.8 Waveforms for force cleared on rising sync after force deassertion

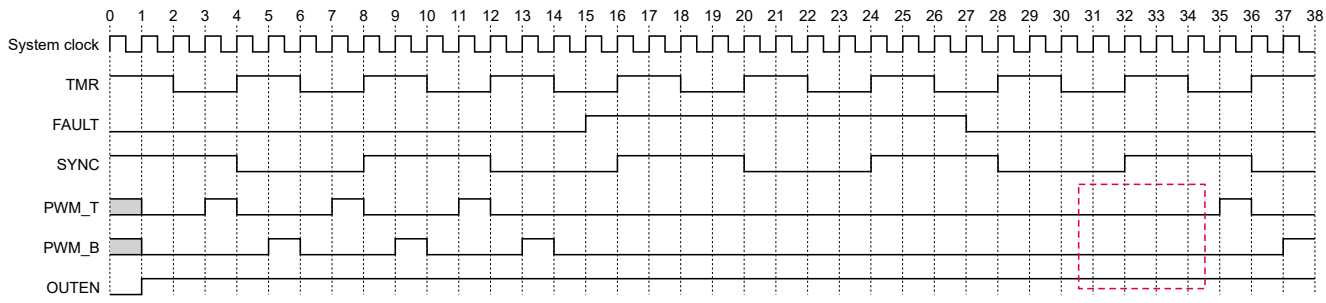


Figure 307. Waveforms for force cleared on rising sync after force deassertion

62.6.1.10 AC motor double-switching controller

62.6.1.10.1 Overview

The simplest method of obtaining motor winding currents is to measure the current in each phase by placing a shunt resistor in two or three inverter legs (phases).

This example implementation of an AC motor drive for low-cost and high-volume applications uses a single shunt on the DC bus return path, as illustrated in AC motor power stage schematic. LCU supports double-switching by XORing two timer channel outputs. The implementation uses two LCs arranged to perform double switching, dead time generation, and the processing of three external fault signals.

62.6.1.10.2 Double-switching

The double-switching technique requires:

- Six synchronized timer pulses
- Three fault signals
- Six PWM outputs for controlling top and bottom power switches
- A sync signal for automatic fault recovery

Waveforms for double-switching technique example applies the PWM switching cycle to an AC motor to reconstruct currents in three phases using a single DC bus shunt resistor (RSH). It applies the non-zero switching vectors 100, 110, and 101 to the power switches, which sample currents IA[0], -IC[0], -IC[1], and IA[1]. Because the durations (T1) of active vectors 110 and 101 are the same, it also applies the zero switching vector (000) with durations T2 and T3 in the middle of the switching cycle to allow phase current reconstruction. This technique can only be used by PWM modules targeted at motor control and power conversion applications.

62.6.1.10.3 Fault inputs and deadtime insertion

Fault signals are routed from analog comparators or external protection circuits to protect power switches against damage caused by overcurrent. You must program the Combinational Force Path (LCn_FFILT[COMB_EN]) to enable an active fault input to immediately force all PWM outputs to the inactive state.

62.6.1.10.4 Deadtime insertion

In Waveforms for generation of PWM complementary signals using an LC, the PWMA_T output is generated by XORing timer outputs TMR0 and TMR1. The PWMA_B output is complementary to PWMA_T. Digital filters insert a dead time of one system clock cycle to delay the rising edges of both PWM signals.

62.6.1.10.5 Fault events

Waveforms for force cleared on force deassertion shows the reaction to the fault event. On a fault event, both the PWM outputs transition to the inactive state (logic 0). When fault event deasserts, both the PWM signals start operating according to FORCE_MODE = 00 (cleared on force deassertion).

Alternatively, after the fault event deasserts, fault clearing can be performed according to FORCE_MODE = 01 (cleared on rising sync force deassertion). See Waveforms for force cleared on rising sync after force deassertion for details.

62.6.1.10.6 AC motor power stage schematic

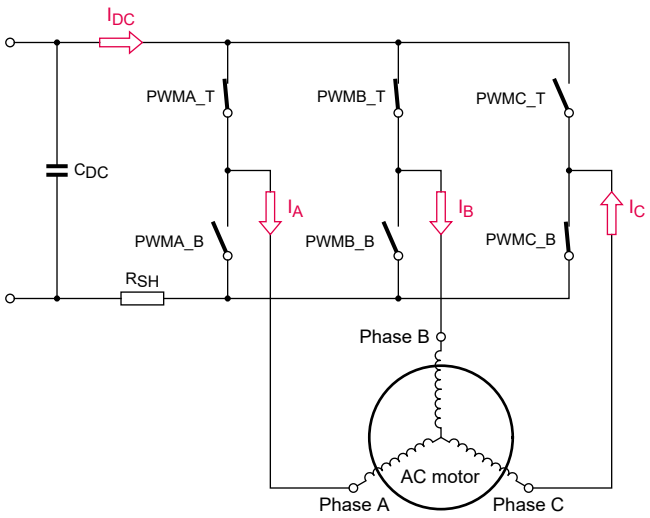


Figure 308. AC motor power stage schematic

62.6.1.10.7 Waveforms for double-switching technique example

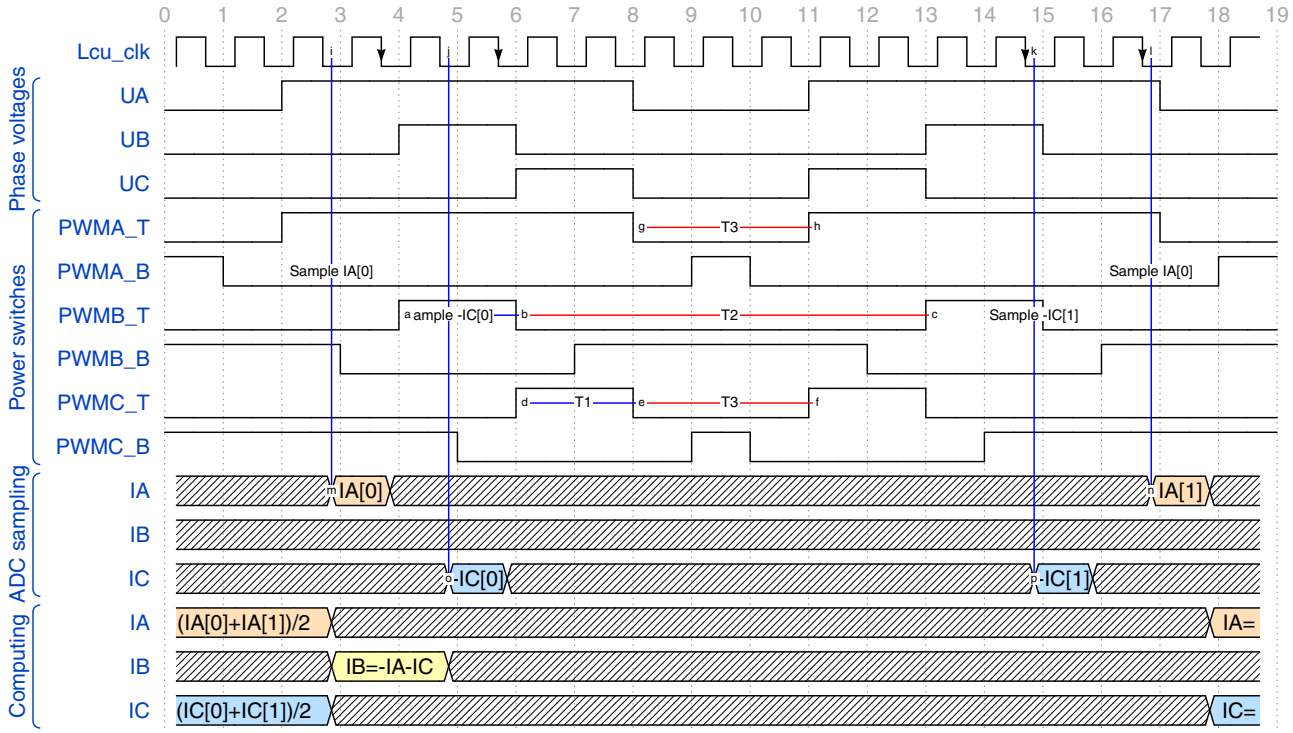


Figure 309. Waveforms for double-switching technique example

62.6.1.10.8 Implementation for AC motor double-switching controller

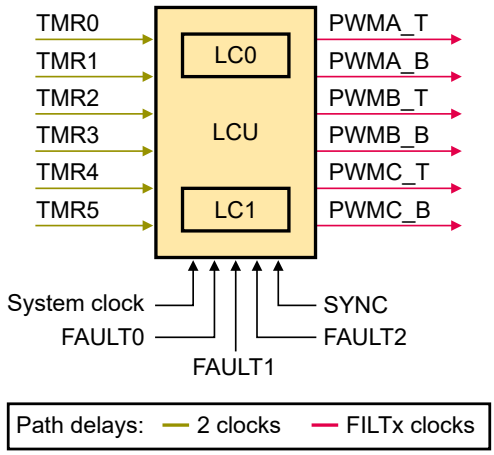


Figure 310. Implementation for AC motor double-switching controller

62.6.1.10.9 Implementation for controlling one phase

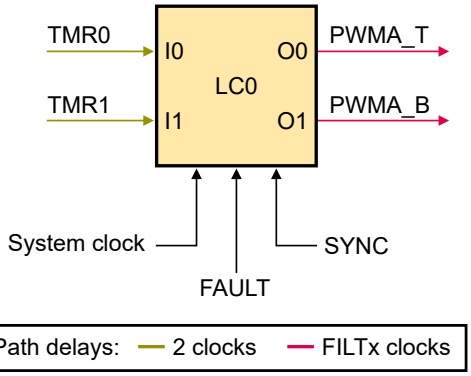


Figure 311. Implementation for controlling one phase

62.6.1.10.10 Connections and controls

Table 391. Connections and controls

Name	Description
TMR*n*	Synchronized timer input signals
System clock	Clock supplied to the LC
FAULT*n*	Fault signal (see Implementation for AC motor double-switching controller for more details)
SYNC	Sync signal for optional automatic fault recovery
PWM*x*_T	PWM output signal for controlling top power switch
PWM*x*_B	PWM output signal for controlling bottom power switch

62.6.1.10.11 Truth table

Table 392. Truth table

Inputs		Outputs	
TMR1	TMR0	PWMA_T	PWMB_B
0	0	0	1 delayed by TdH (see Output edge delays)
0	1	1 delayed by TdH (see Output edge delays)	0
1	0	1 delayed by TdH (see Output edge delays)	0
1	1	0	1 delayed by TdH (see Output edge delays)
0	0	0	0
0	1	0	0
1	0	0	0
1	1	0	0

62.6.1.10.12 Register configuration

Table 393. Register configuration

Register	I3	I2	I1	I0	O0	O1	O2	O3
LUTCTRL0	—	—	—	—	6h	—	—	—
LUTCTRL1	—	—	—	—	—	9h	—	—
LUTCTRL2	—	—	—	—	—	—	0h	—
LUTCTRL3	—	—	—	—	—	—	—	0h
FILT0	—	—	—	—	1_0000h	—	—	—
FILT1	—	—	—	—	—	1_0000h	—	—
FILT2	—	—	—	—	—	—	0h	—
FILT3	—	—	—	—	—	—	—	0h
INTDMAEN	—	—	—	—	0h			
OUTPOL	—	—	—	—	0h			
FFILT	—	—	—	—	100_0000h			
FCTRL	—	—	—	—	Force inputs 2 and 0 affect output 0, but force inputs 3 and 1 do not affect output 0: 101h All four inputs affect output 0: 1111h			
SCTRL	—	—	—	—	0h			
MUXSEL0	—	—	—	1h	—	—	—	—
MUXSEL1	—	—	2h	—	—	—	—	—
MUXSEL2	—	0h	—	—	—	—	—	—
MUXSEL3	0h	—	—	—	—	—	—	—

Table continues on the next page...

Table 393. Register configuration (continued)

Register	I3	I2	I1	I0	O0	O1	O2	O3
SWEN	—	—	—	—	0h			
SWVALUE	—	—	—	—	0h			
OUTEN	—	—	—	—	3h			

62.6.1.10.13 Waveforms for generation of PWM complementary signals using an LC

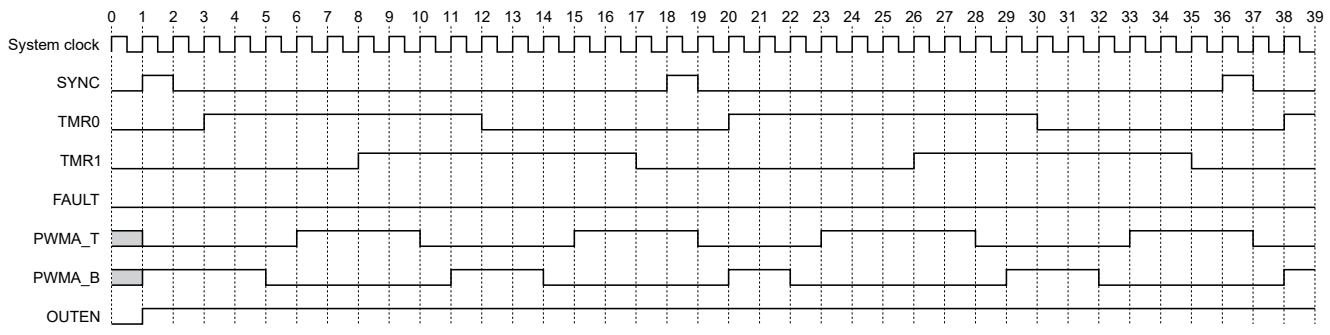


Figure 312. Waveforms for generation of PWM complementary signals using an LC

62.6.1.10.14 Waveforms for force cleared on force deassertion

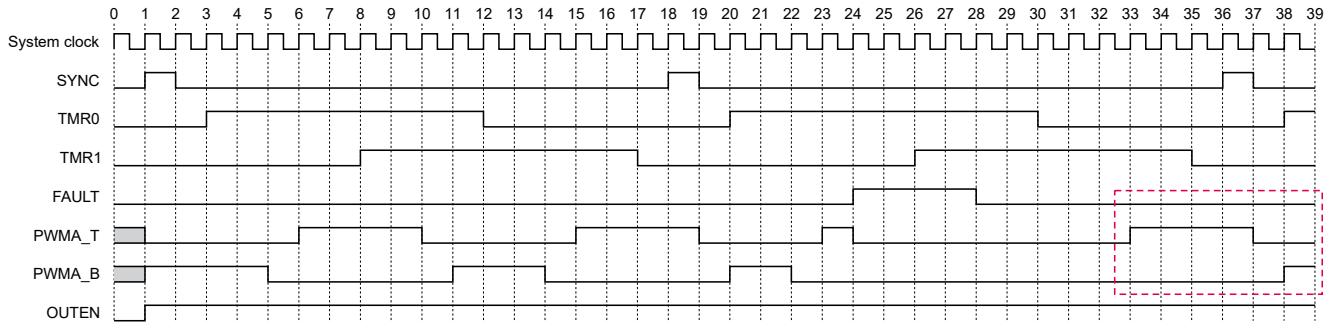


Figure 313. Waveforms for force cleared on force deassertion

62.6.1.10.15 Waveforms for force cleared on rising sync after force deassertion

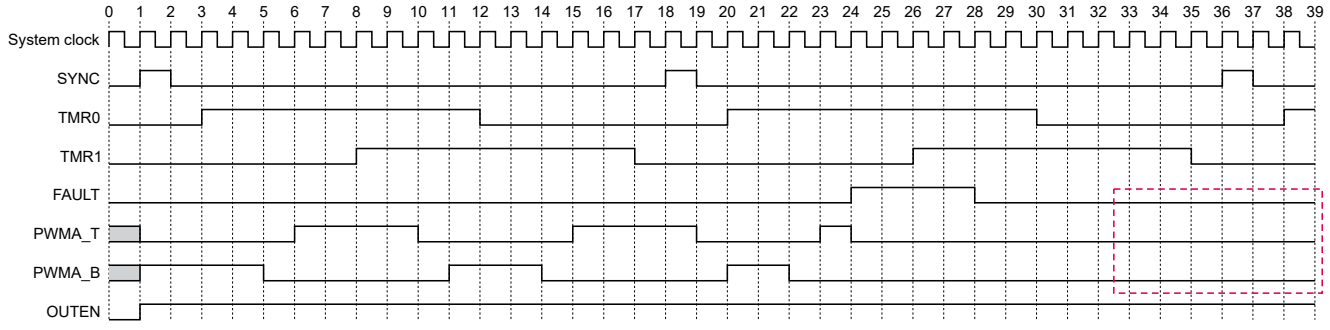


Figure 314. Waveforms for force cleared on rising sync after force deassertion

62.6.1.11 Brushless DC (BLDC) motor PWM controller

62.6.1.11.1 Overview

You can use LCU to replace the mechanical commutator in the BLDC motor control application with Hall-sensor position measurement.

In this example implementation, LC0 receives the direction (DIR) and Hall A, Hall B, and Hall C sensor output states, and from those inputs generates the switching states S1, S2, and S3. LC1 and LC2 convert the switching states to PWM or on-off signals for controlling power switches, as illustrated in Waveforms for BLDC motor control technique.

BLDC motor power stage schematic and BLDC motor electrical arrangement illustrate a three-phase BLDC motor power stage with switching pulses and phase currents controlling motor rotation in the CCW direction. The state of the Hall sensors is 010. As the rotor shaft rotates, Hall sensors track the motor position. You accelerate or decelerate the motor by controlling the width of the PWM signal. You can include one or more faults (force inputs) to force power switches instantaneously into the inactive state if there is an overcurrent.

62.6.1.11.2 Implementation

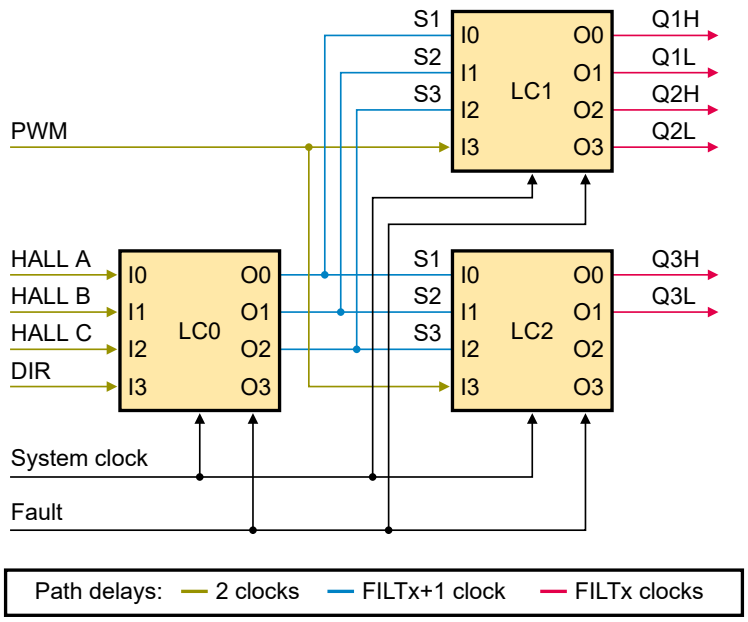


Figure 315. Implementation

62.6.1.11.3 BLDC motor power stage schematic

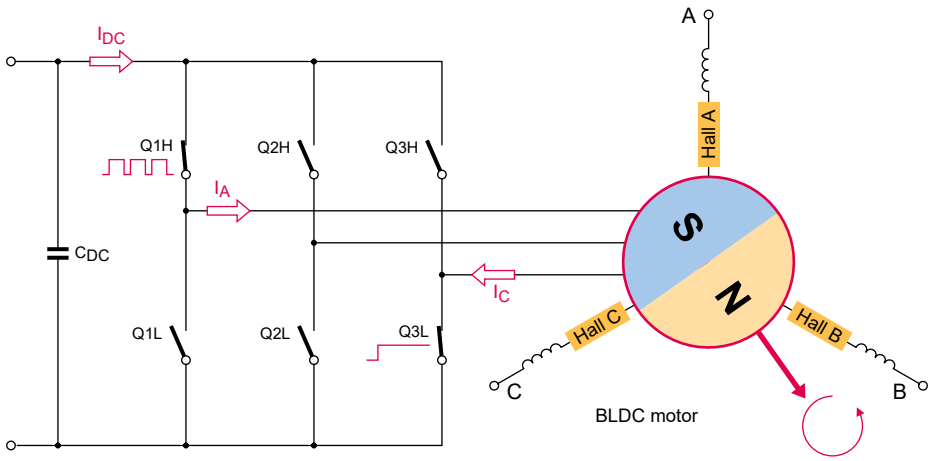


Figure 316. BLDC motor power stage schematic

62.6.1.11.4 BLDC motor electrical arrangement

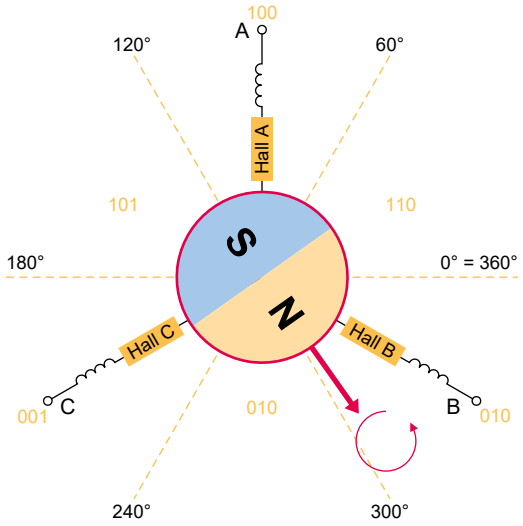


Figure 317. BLDC motor electrical arrangement

62.6.1.11.5 Waveforms for BLDC motor control technique

This figure illustrates the basic waveforms for controlling a BLDC motor with stator windings and Hall sensors arranged according to BLDC motor electrical arrangement. Configure the external hardware (motor or sensor) so that it can produce these waveforms; then you can configure LCU.

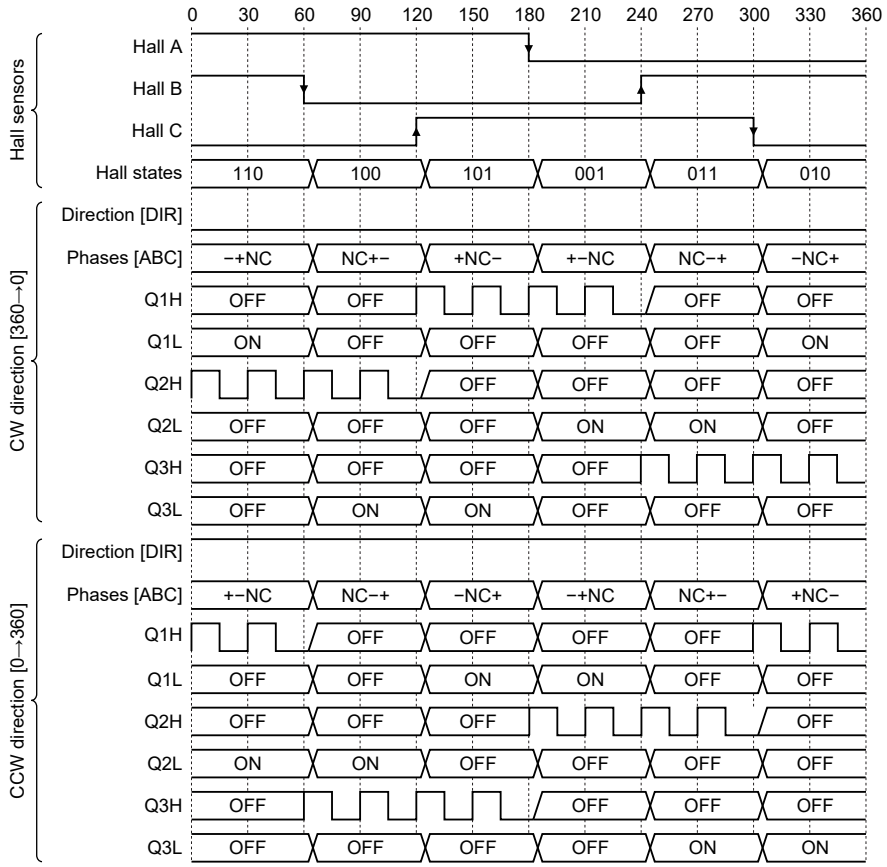


Figure 318. Waveforms for BLDC motor control technique

62.6.1.11.6 Connections and controls

Table 394. Connections and controls

Input-output group	Name	Description
LC0 inputs	HALL A	Hall sensor A output state
	HALL B	Hall sensor B output state
	HALL C	Hall sensor C output state
	DIR	Direction of shaft rotation, as illustrated in Waveforms for reversing BLDC motor.
LC1 and LC2 inputs	S1	Switching state 1 (output 0 from LC0)
	S2	Switching state 2 (output 1 from LC0)
	S2	Switching state 3 (output 2 from LC0)
	PWM	Output from PWM
LC0, LC1, and LC2 inputs	Clock	LCU clock input
	Fault	Optional fault input

Table continues on the next page...

Table 394. Connections and controls (continued)

Input-output group	Name	Description
LC1 outputs	Q1H	Motor control switch outputs, as illustrated in BLDC motor electrical arrangement
	Q1L	
	Q2H	
	Q2L	
LC2 outputs	Q3H	
	Q3L	

62.6.1.11.7 Truth table for generating switching states

This table includes all Hall sensor states, even those that never occur because of sensor displacement by 120°.

Table 395. Truth table for generating switching states

Inputs				Outputs			Comments
DIR	Hall C	Hall B	Hall A	S1	S2	S3	
0	0	0	0	0	0	0	Not used
0	0	0	1	1	0	0	Hall states
0	0	1	0	0	1	0	Hall states
0	0	1	1	1	1	0	Hall states
0	1	0	0	0	0	1	Hall states
0	1	0	1	1	0	1	Hall states
0	1	1	0	0	1	1	Hall states
0	1	1	1	0	0	0	Not used
1	0	0	0	0	0	0	Not used
1	0	0	1	0	1	1	Complement of Hall states
1	0	1	0	1	0	1	Complement of Hall states
1	0	1	1	0	0	1	Complement of Hall states
1	1	0	0	1	1	0	Complement of Hall states
1	1	0	1	0	1	0	Complement of Hall states
1	1	1	0	1	0	0	Complement of Hall states
1	1	1	1	0	0	0	Not used

62.6.1.11.8 Truth table for generating switching pulses for phases A and B

Table 396. Truth table for generating switching pulses for phases A and B

Inputs				Outputs				Comments
PWM	S3	S2	S1	Q1H	Q1L	Q2H	Q2L	
0	0	0	0	0	0	0	0	Not used
0	0	0	1	0	0	0	0	—
0	0	1	0	0	1	0	0	—
0	0	1	1	0	1	0	0	—
0	1	0	0	0	0	0	1	—
0	1	0	1	0	0	0	0	—
0	1	1	0	0	0	0	1	—
0	1	1	1	0	0	0	0	Not used
1	0	0	0	0	0	0	0	Not used
1	0	0	1	0	0	1	0	—
1	0	1	0	0	1	0	0	—
1	0	1	1	0	1	1	0	—
1	1	0	0	1	0	0	1	—
1	1	0	1	1	0	0	0	—
1	1	1	0	0	0	0	1	—
1	1	1	1	0	0	0	0	Not used

62.6.1.11.9 Truth table for generating switching pulses for phase C

Table 397. Truth table for generating switching pulses for phase C

Inputs				Outputs		Comments
PWM	S3	S2	S1	Q3H	Q3L	
0	0	0	0	0	0	Not used
0	0	0	1	0	1	—
0	0	1	0	0	0	—
0	0	1	1	0	0	—
0	1	0	0	0	0	—
0	1	0	1	0	1	—
0	1	1	0	0	0	—
0	1	1	1	0	0	Not used
1	0	0	0	0	0	Not used
1	0	0	1	0	1	—

Table continues on the next page...

Table 397. Truth table for generating switching pulses for phase C (continued)

Inputs				Outputs		Comments
PWM	S3	S2	S1	Q3H	Q3L	
1	0	1	0	1	0	—
1	0	1	1	0	0	—
1	1	0	0	0	0	—
1	1	0	1	0	1	—
1	1	1	0	1	0	—
1	1	1	1	0	0	Not used

62.6.1.11.10 Configuration for generating switching states

Table 398. Configuration for generating switching states

Register	I3	I2	I1	I0	O0	O1	O2	O3
LUTCTRL0	—	—	—	—	542Ah	—	—	—
LUTCTRL1	—	—	—	—	—	324Ch	—	—
LUTCTRL2	—	—	—	—	—	—	E70h	—
LUTCTRL3	—	—	—	—	—	—	—	0h
FILT0	—	—	—	—	0h	—	—	—
FILT1	—	—	—	—	—	0h	—	—
FILT2	—	—	—	—	—	—	0h	—
FILT3	—	—	—	—	—	—	—	0h
INTDMAEN	—	—	—	—	0h			
OUTPOL	—	—	—	—	0h			
FFILT	—	—	—	—	0h			
FCTRL	—	—	—	—	0h			
SCTRL	—	—	—	—	0h			
MUXSEL0	—	—	—	1h	—	—	—	—
MUXSEL1	—	—	2h	—	—	—	—	—
MUXSEL2	—	3h	—	—	—	—	—	—
MUXSEL3	4h	—	—	—	—	—	—	—
SWEN	—	—	—	—	0h			
SWVALUE	—	—	—	—	0h			
OUTEN	—	—	—	—	7h			

62.6.1.11.11 Configuration for switching pulses for phases A and B

Table 399. Configuration for switching pulses for phases A and B

Register	I3	I2	I1	I0	O0	O1	O2	O3
LUTCTRL0	—	—	—	—	3000h	—	—	—
LUTCTRL1	—	—	—	—	—	C0Ch	—	—
LUTCTRL2	—	—	—	—	—	—	A00h	—
LUTCTRL3	—	—	—	—	—	—	—	5050h
FILT0	—	—	—	—	0h	—	—	—
FILT1	—	—	—	—	—	0h	—	—
FILT2	—	—	—	—	—	—	0h	—
FILT3	—	—	—	—	—	—	—	0h
INTDMAEN	—	—	—	—	0h			
OUTPOL	—	—	—	—	0h			
FFILT	—	—	—	—	F00_0000h			
FCTRL	—	—	—	—	101_0101h			
SCTRL	—	—	—	—	0h			
MUXSEL4	—	—	—	Dh	—	—	—	—
MUXSEL5	—	—	Eh	—	—	—	—	—
MUXSEL6	—	Fh	—	—	—	—	—	—
MUXSEL7	8h	—	—	—	—	—	—	—
SWEN	—	—	—	—	0h			
SWVALUE	—	—	—	—	0h			
OUTEN	—	—	—	—	F0h			

62.6.1.11.12 Configuration for switching pulses for phase C

Table 400. Configuration for switching pulses for phase C

Register	I3	I2	I1	I0	O0	O1	O2	O3
LUTCTRL0	—	—	—	—	4400h	—	—	—
LUTCTRL1	—	—	—	—	—	2222h	—	—
LUTCTRL2	—	—	—	—	—	—	0h	—
LUTCTRL3	—	—	—	—	—	—	—	0h
FILT0	—	—	—	—	0h	—	—	—
FILT1	—	—	—	—	—	0h	—	—
FILT2	—	—	—	—	—	—	0h	—
FILT3	—	—	—	—	—	—	—	0h

Table continues on the next page...

Table 400. Configuration for switching pulses for phase C (continued)

Register	I3	I2	I1	I0	O0	O1	O2	O3
INTDMAEN	—	—	—	—	0h			
OUTPOL	—	—	—	—	0h			
FFILT	—	—	—	—	0300_0000h			
FCTRL	—	—	—	—	101h			
SCTRL	—	—	—	—	0h			
MUXSEL8	—	—	—	Dh	—	—	—	—
MUXSEL9	—	—	Eh	—	—	—	—	—
MUXSEL10	—	Fh	—	—	—	—	—	—
MUXSEL11	8h	—	—	—	—	—	—	—
SWEN	—	—	—	—	0h			
SWVALUE	—	—	—	—	0h			
OUTEN	—	—	—	—	300h			

62.6.1.11.13 Waveforms for reversing BLDC motor

This figure illustrates the waveforms for controlling a reversing BLDC motor. The DIR input indicates the direction of rotation:

- Logic 0 for CW rotation
- Logic 1 for CCW rotation

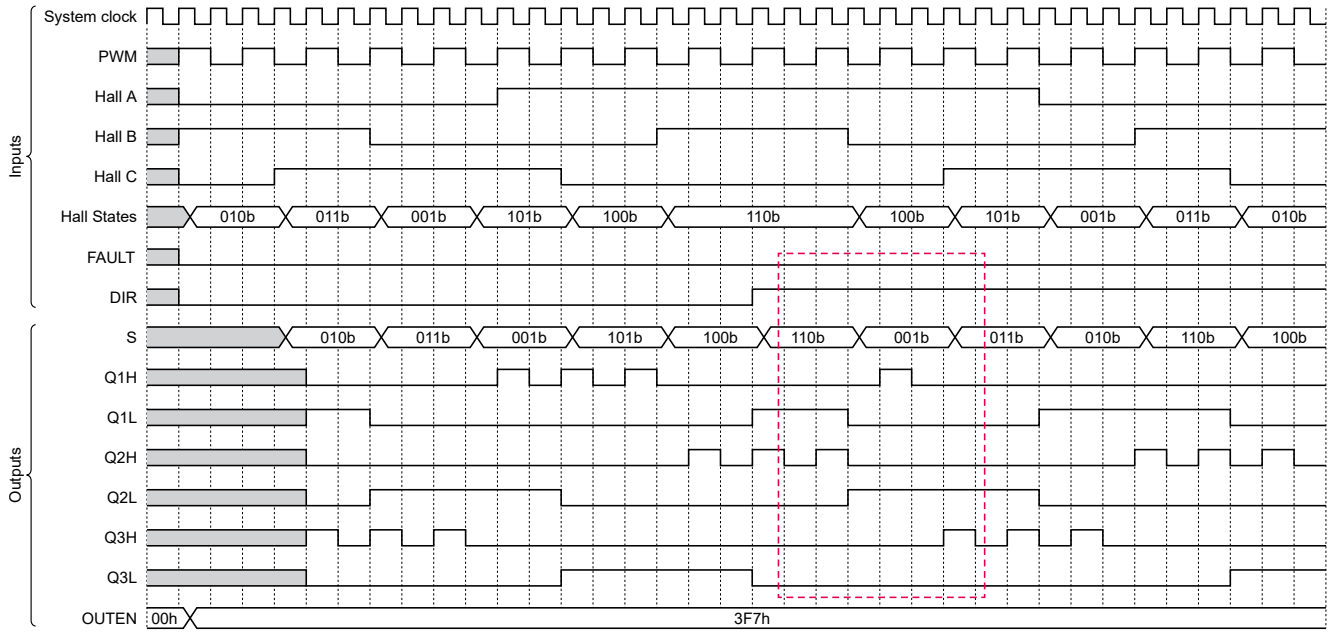


Figure 319. Waveforms for reversing BLDC motor

62.6.1.11.14 Waveforms for reversing BLDC motor with fault assertion

This figure illustrates the waveforms for the reaction to an external fault in addition to reversing the BLDC motor. On the occurrence of an external fault event, the power switches transition to the inactive state (logic 0). Two clock cycles after external fault deassertion, the power switches start operating normally.

Switching state outputs S1, S2, and S3, generated by LC0, receive an additional delay of one clock cycle on their way to the inputs of other LCs (digital filters bypassed).

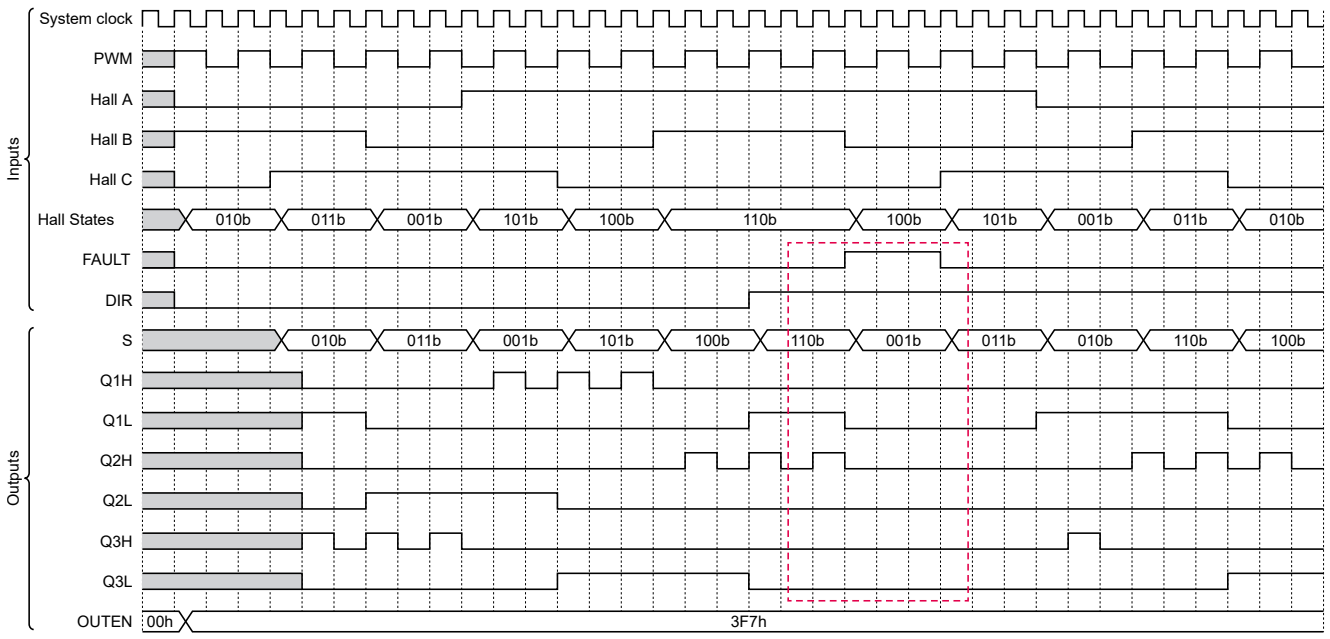


Figure 320. Waveforms for reversing BLDC motor with fault assertion

62.7 Mapping of 12-bit input, output, and state fields

This table maps inputs, outputs, or states to the bit positions in the following 12-bit fields:

- SWEN[SWEN]
- SWVALUE[SWVALUE]
- OUTEN[OUTEN]
- LCIN[LC_INPUTS]
- SWOUT[SWOUT]
- LCOUT[LCOUT]
- FORCEOUT[FORCEOUT]
- FORCESTS[FORCESTS]
- DBGEN[DBGEN]

LC	Input, output, or state	Bit
0	0	0
	1	1

Table continues on the next page...

Table continued from the previous page...

LC	Input, output, or state	Bit
	2	2
	3	3
1	0	4
	1	5
	2	6
	3	7
2	0	8
	1	9
	2	10
	3	11

62.8 LCU register descriptions

NOTE

Access to address offset 2A4h does not generate a transfer error.

62.8.1 LCU memory map

LCU_0 base address: 4009_8000h

LCU_1 base address: 4009_C000h

Offset	Register	Width (In bits)	Access	Reset value
0h	LC 0 Output 0 LUT Control (LC0_LUTCTRL0)	32	RW	0000_0000h
4h	LC 0 Output 1 LUT Control (LC0_LUTCTRL1)	32	RW	0000_0000h
8h	LC 0 Output 2 LUT Control (LC0_LUTCTRL2)	32	RW	0000_0000h
Ch	LC 0 Output 3 LUT Control (LC0_LUTCTRL3)	32	RW	0000_0000h
10h	LC 0 Output 0 Filter (LC0_FILT0)	32	RW	0000_0000h
14h	LC 0 Output 1 Filter (LC0_FILT1)	32	RW	0000_0000h
18h	LC 0 Output 2 Filter (LC0_FILT2)	32	RW	0000_0000h
1Ch	LC 0 Output 3 Filter (LC0_FILT3)	32	RW	0000_0000h
20h	LC 0 Interrupt and DMA Enable (LC0_INTDMAEN)	32	RW	0000_0000h
24h	LC 0 Status (LC0_STS)	32	RW	0000_0000h
28h	LC 0 Output Polarity Control (LC0_OUTPOL)	32	RW	0000_0000h
2Ch	LC 0 Force Filter (LC0_FFILT)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
30h	LC 0 Force Control (LC0_FCTRL)	32	RW	0000_0000h
34h	LC 0 Sync Control (LC0_SCTRL)	32	RW	0000_0000h
40h	LC 1 Output 0 LUT Control (LC1_LUTCTRL0)	32	RW	0000_0000h
44h	LC 1 Output 1 LUT Control (LC1_LUTCTRL1)	32	RW	0000_0000h
48h	LC 1 Output 2 LUT Control (LC1_LUTCTRL2)	32	RW	0000_0000h
4Ch	LC 1 Output 3 LUT Control (LC1_LUTCTRL3)	32	RW	0000_0000h
50h	LC 1 Output 0 Filter (LC1_FILT0)	32	RW	0000_0000h
54h	LC 1 Output 1 Filter (LC1_FILT1)	32	RW	0000_0000h
58h	LC 1 Output 2 Filter (LC1_FILT2)	32	RW	0000_0000h
5Ch	LC 1 Output 3 Filter (LC1_FILT3)	32	RW	0000_0000h
60h	LC 1 Interrupt and DMA Enable (LC1_INTDMAEN)	32	RW	0000_0000h
64h	LC 1 Status (LC1_STS)	32	RW	0000_0000h
68h	LC 1 Output Polarity Control (LC1_OUTPOL)	32	RW	0000_0000h
6Ch	LC 1 Force Filter (LC1_FFILT)	32	RW	0000_0000h
70h	LC 1 Force Control (LC1_FCTRL)	32	RW	0000_0000h
74h	LC 1 Sync Control (LC1_SCTRL)	32	RW	0000_0000h
80h	LC 2 Output 0 LUT Control (LC2_LUTCTRL0)	32	RW	0000_0000h
84h	LC 2 Output 1 LUT Control (LC2_LUTCTRL1)	32	RW	0000_0000h
88h	LC 2 Output 2 LUT Control (LC2_LUTCTRL2)	32	RW	0000_0000h
8Ch	LC 2 Output 3 LUT Control (LC2_LUTCTRL3)	32	RW	0000_0000h
90h	LC 2 Output 0 Filter (LC2_FILT0)	32	RW	0000_0000h
94h	LC 2 Output 1 Filter (LC2_FILT1)	32	RW	0000_0000h
98h	LC 2 Output 2 Filter (LC2_FILT2)	32	RW	0000_0000h
9Ch	LC 2 Output 3 Filter (LC2_FILT3)	32	RW	0000_0000h
A0h	LC 2 Interrupt and DMA Enable (LC2_INTDMAEN)	32	RW	0000_0000h
A4h	LC 2 Status (LC2_STS)	32	RW	0000_0000h
A8h	LC 2 Output Polarity Control (LC2_OUTPOL)	32	RW	0000_0000h
ACh	LC 2 Force Filter (LC2_FFILT)	32	RW	0000_0000h
B0h	LC 2 Force Control (LC2_FCTRL)	32	RW	0000_0000h
B4h	LC 2 Sync Control (LC2_SCTRL)	32	RW	0000_0000h
200h - 22Ch	Mux Select (MUXSEL0 - MUXSEL11)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
280h	Configuration (CFG)	32	RW	0303_0280h
284h	Software Override Enable (SWEN)	32	RW	0000_0000h
288h	Software Override Value (SWVALUE)	32	RW	0000_0000h
28Ch	Output Enable (OUTEN)	32	RW	0000_0000h
290h	Logic Inputs (LCIN)	32	R	0000_0000h
294h	Overridden Inputs (SWOUT)	32	R	0000_0000h
298h	Logic Outputs (LCOUT)	32	R	0000_0000h
29Ch	Forced Outputs (FORCEOUT)	32	R	0000_0000h
2A0h	Force Status (FORCESTS)	32	RW	0000_0000h
2A8h	Debug Mode Enable (DBGEN)	32	RW	0000_0000h

62.8.2 LC n Output m LUT Control (LC0_LUTCTRL0 - LC2_LUTCTRL3)

Offset

For n = 0 to 2; m = 0 to 3:

Register	Offset
LCn_LUTCTRLm	0h + (n × 40h) + (m × 4h)

Function

See LUTCTRL.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LUTCTRL															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-0 LUTCTRL	LUT Control Specifies the LUT positions, based on the combined LC input value, that result in assertion of this output. For more information, see Logic operations.

62.8.3 LC n Output m Filter (LC0_FILT0 - LC2_FILT3)

Offset

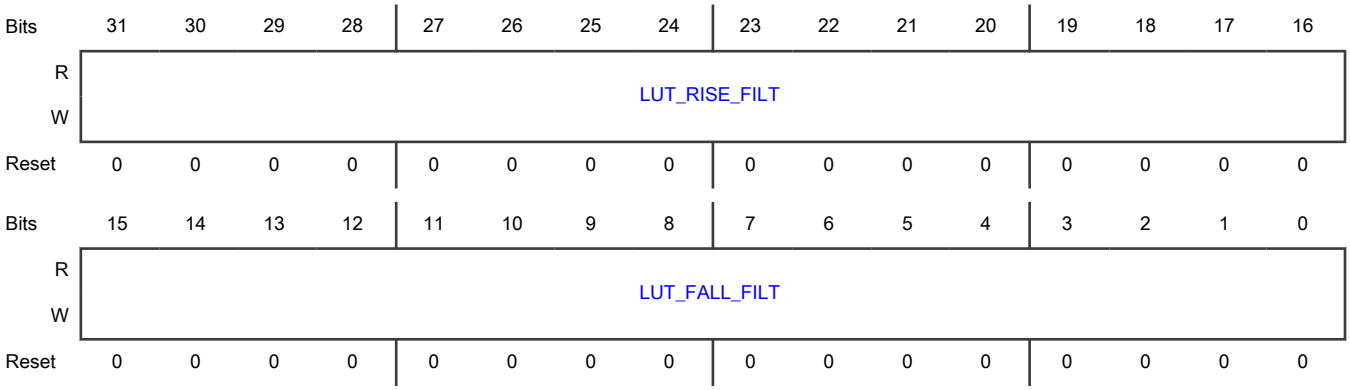
For n = 0 to 2; m = 0 to 3:

Register	Offset
LCn_FILTm	10h + (n × 40h) + (m × 4h)

Function

Specifies the rising- and falling-edge thresholds for the LC output filters.

Diagram



Fields

Field	Function
31-16 LUT_RISE_FILT	Rise Filter Specifies the number of consecutive clock cycles the filter output must be logic 1 before the output signal goes high. 0000_0000_0000_0000b - Bypass filter All other values - Filter threshold

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-0 LUT_FALL_FILT	Fall Filter Specifies the number of consecutive clock cycles the filter output must be logic 0 before the output signal goes low. 0000_0000_0000_0000b - Bypass filter All other values - Filter threshold

62.8.4 LC n Interrupt and DMA Enable (LC0_INTDMAEN - LC2_INTDMAEN)

Offset

Register	Offset
LC0_INTDMAEN	20h
LC1_INTDMAEN	60h
LC2_INTDMAEN	A0h

Function
Enables interrupt and DMA requests for LUT and force events.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				FORCE_DMA_EN				0				FORCE_INT_EN			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				LUT_DMA_EN				0				LUT_INT_EN			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved
27-24	Force DMA Enable Enables the generation of a DMA request when a force event occurs (LC*n*_STS[FORCESTS]).

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>FORCE_DMA_EN</td><td>For each bit:
0b - Disable
1b - Enable
Mapping of bits:<table><tr><th>Output</th><th>Register bit</th></tr><tr><td>0</td><td>24</td></tr><tr><td>1</td><td>25</td></tr><tr><td>2</td><td>26</td></tr><tr><td>3</td><td>27</td></tr></table></td></tr>
<tr><td>23-20
—</td><td>Reserved</td></tr>
<tr><td>19-16
FORCE_INT_EN</td><td>Force Interrupt Enable
Enables the generation of an interrupt request when a force event occurs (LCn_STS[FORCESTS]).
For each bit:
0b - Disable
1b - Enable
Mapping of bits:<table><tr><th>Output</th><th>Register bit</th></tr><tr><td>0</td><td>16</td></tr><tr><td>1</td><td>17</td></tr><tr><td>2</td><td>18</td></tr><tr><td>3</td><td>19</td></tr></table></td></tr>
<tr><td>15-12
—</td><td>Reserved</td></tr>
<tr><td>11-8
LUT_DMA_EN</td><td>LUT DMA Enable
Enables the generation of a DMA request when an LUT event occurs (LCn_STS[LUT_STS]).
For each bit:
0b - Disable</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>1b - Enable
Mapping of bits:<table><tr><th>Output</th><th>Register bit</th></tr><tr><td>0</td><td>8</td></tr><tr><td>1</td><td>9</td></tr><tr><td>2</td><td>10</td></tr><tr><td>3</td><td>11</td></tr></table></td></tr>
<tr><td>7-4
—</td><td>Reserved</td></tr>
<tr><td>3-0
LUT_INT_EN</td><td>LUT Interrupt Enable
Enables the generation of an interrupt request when an LUT event occurs (LCn_STS[LUT_STS]).
For each bit:
0b - Disable
1b - Enable
Mapping of bits:<table><tr><th>Output</th><th>Register bit</th></tr><tr><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td></tr><tr><td>2</td><td>2</td></tr><tr><td>3</td><td>3</td></tr></table></td></tr>
</table>

62.8.5 LC n Status (LC0_STS - LC2_STS)

Offset

Register	Offset
LC0_STS	24h
LC1_STS	64h
LC2_STS	A4h

Function

Indicates the occurrence of LUT and force events.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				FORCESTS				0				LUT_STS			
W					W1C								W1C			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-12 —	Reserved
11-8 FORCESTS	Force Event Indicates that a force event has occurred on the associated output. For each bit: 0b - No event 1b - Event occurred Mapping of bits:

Output	Register bit
0	8
1	9
2	10
3	11

When you enable DMA for a force output (LC*n*_INTDMAEN[FORCE_DMA_EN]) and a force event occurs on that output, LCU generates a DMA request. The resulting DMA done signal causes LCU to change the bit to 0, or you can write 1 to the bit.

You can also change these bits to 0 by writing to the corresponding Force Status bits (FORCESTS[FORCESTS]).

The timing of status clearing depends on the selected Force Mode (LC*n*_FCTRL[FORCE_MODEm]).

7-4 —	Reserved
3-0	LUT Event

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>LUT_STS</td><td>Indicates that an LUT event has occurred for the associated LC output.
For each bit:
0b - No event
1b - Event occurred
Mapping of bits:
<table>
<tr><th>Output</th><th>Register bit</th></tr>
<tr><td>0</td><td>0</td></tr>
<tr><td>1</td><td>1</td></tr>
<tr><td>2</td><td>2</td></tr>
<tr><td>3</td><td>3</td></tr>
</table>
When you enable DMA for an LUT event on an output (LCn_INTDMAEN[LUT_DMA_EN]) and an LUT event occurs on that output, LCU generates a DMA request. The resulting DMA done signal causes LCU to change the bit to 0.
When you enable interrupt request for an output (LCn_INTDMAEN[LUT_INT_EN]) and an LUT event occurs on that output, LCU generates an interrupt request. Write 1 to the bit to change it to 0.</td></tr>
</table>

62.8.6 LC n Output Polarity Control (LC0_OUTPOL - LC2_OUTPOL)

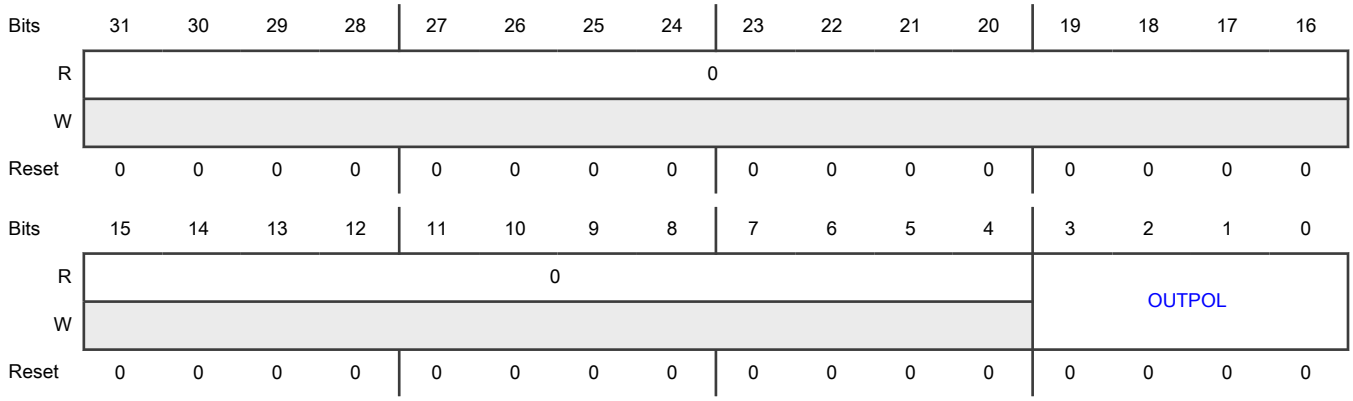
Offset

Register	Offset
LC0_OUTPOL	28h
LC1_OUTPOL	68h
LC2_OUTPOL	A8h

Function

Specifies the polarity of the LC outputs.

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 OUTPOL	Output Polarity Specifies the polarity of the outputs. For each bit: 0b - Not inverted 1b - Inverted Mapping of bits:

Output	Bit
0	0
1	1
2	2
3	3

62.8.7 LC n Force Filter (LC0_FFILT - LC2_FFILT)

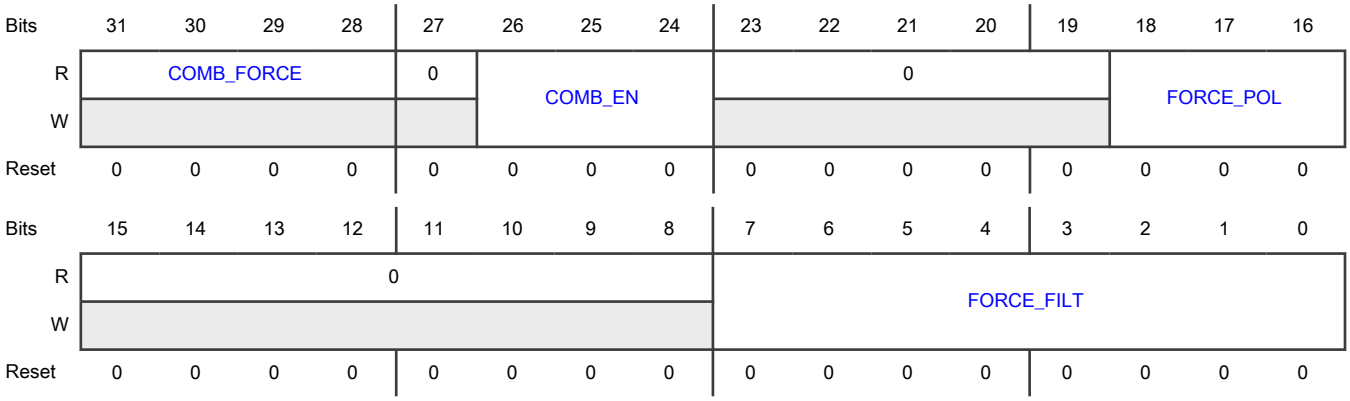
Offset

Register	Offset
LC0_FFILT	2Ch
LC1_FFILT	6Ch
LC2_FFILT	ACh

Function

Controls the force filter for this LC.

Diagram



Fields

Field	Function
31-28 COMB_FORCE	Combined Sensed Force Input Indicates the combined value of force inputs to each output. For each bit: 0b - Logic low 1b - Logic high Mapping of bits:

Output	Bit
0	28
1	29
2	30
3	31

27 —	Reserved
26-24 COMB_EN	Combinational Force Path (CFP) Enable Enables an active force input to combinationally affect the LC outputs. When CFP is not enabled, force inputs must be synchronized and then optionally filtered. When you enable CFP (write 1 to one of these bits), the corresponding force input bypasses synchronization and filtering and immediately affects the LC output. The Force Filter (FORCE_FILT) is still in effect. Force inputs that do not meet the pulse width requirements for synchronization and filtering are not registered in Force Status (FORCESTS) and you do not need to write 1 to change the bit to 0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	CFP provides a safety factor when the LC outputs drive a motor and immediate response is required. In some topologies, an oscillation can result if the assertion of the force input causes the LC output to turn off. This situation in turn leads to deassertion of the force input and the output turning back on. In general, you must disable CFP unless the LCU outputs drive a motor or similar safety-critical application, which requires an immediate response without synchronization delays.
23-19 —	Reserved
18-16 FORCE_POL	Force Input Polarity Specifies the polarity of the force inputs to this LC. For each bit: 0b - Not inverted 1b - Inverted Mapping of bits:

Input	Bit
0	16
1	17
2	18

15-8 —	Reserved
7-0 FORCE_FILT	Force Filter Specifies the count, in clock cycles, that a force input must remain at a given logic state before the filtered force input switches state. 0000_0000b - Bypass filter All other values - Filter threshold

62.8.8 LC n Force Control (LC0_FCTRL - LC2_FCTRL)

Offset

Register	Offset
LC0_FCTRL	30h
LC1_FCTRL	70h
LC2_FCTRL	B0h

Function

Provides control of the force inputs.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SYNC_SEL3		FORCE_MODE 3		FORCE_SENSE3				SYNC_SEL2		FORCE_MODE 2		FORCE_SENSE2			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SYNC_SEL1		FORCE_MODE 1		FORCE_SENSE1				SYNC_SEL0		FORCE_MODE 0		FORCE_SENSE0			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 SYNC_SEL3	Sync Select Selects which sync input to use for output 3 of this LC. 00b - Sync input 0 01b - Sync input 1 10b - Reserved 11b - Reserved
29-28 FORCE_MODE 3	Force Clearing Mode Specifies the timing for clearing force events for output 3 in this LC. 00b - Deassertion. Cleared on deassertion of force inputs 01b - Rising sync after deassertion. Cleared on rising sync after deassertion of force inputs 10b - Writing 1 after deassertion. Cleared by writing 1 to the correct LCn_STS[FORCESTS] or FORCESTS[FORCESTS] bits after deassertion of force inputs 11b - Rising sync after writing 1 and deassertion. Cleared on rising sync after writing 1 to the correct LCn_STS[FORCESTS] or FORCESTS[FORCESTS] bits and deassertion of force inputs
27-24 FORCE_SENSE3	Force Input Sensitivity Selects which force inputs affect output 3 of this LC. For each bit: 0b - Does not affect 1b - Affects Mapping of bits:

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td><table>
<tr><th>Input</th><th>Bit</th></tr>
<tr><td>0</td><td>24</td></tr>
<tr><td>1</td><td>25</td></tr>
<tr><td>2</td><td>26</td></tr>
<tr><td>Reserved</td><td>27</td></tr>
</table>
Example: 011b selects force inputs 0 and 1, but not 2.</td></tr>
<tr><td>23-22
SYNC_SEL2</td><td>Sync Select
Selects which sync input to use for output 2 of this LC.
00b - Sync input 0
01b - Sync input 1
10b - Reserved
11b - Reserved</td></tr>
<tr><td>21-20
FORCE_MODE2</td><td>Force Clearing Mode
Specifies the timing for clearing force events for output 2 in this LC.
00b - Deassertion. Cleared on deassertion of force inputs
01b - Rising sync after deassertion. Cleared on rising sync after deassertion of force inputs
10b - Writing 1 after deassertion. Cleared by writing 1 to the correct LCn_STS[FORCESTS] or FORCESTS[FORCESTS] bits after deassertion of force inputs
11b - Rising sync after writing 1 and deassertion. Cleared on rising sync after writing 1 to the correct LCn_STS[FORCESTS] or FORCESTS[FORCESTS] bits and deassertion of force inputs</td></tr>
<tr><td>19-16
FORCE_SENSE2</td><td>Force Input Sensitivity
Selects which force inputs affect output 2 of this LC.
For each bit:
0b - Does not affect
1b - Affects
Mapping of bits:<table>
<tr><th>Input</th><th>Bit</th></tr>
<tr><td>0</td><td>16</td></tr>
<tr><td>1</td><td>17</td></tr>
<tr><td>2</td><td>18</td></tr>
</table></td></tr>
</table>

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td><table>
<tr><th>Input</th><th>Bit</th></tr>
<tr><td>Reserved</td><td>19</td></tr>
</table>
Example: 011b selects force inputs 0 and 1, but not 2.</td></tr>
<tr><td>15-14
SYNC_SEL1</td><td>Sync Select
Selects which sync input to use for output 1 of this LC.
00b - Sync input 0
01b - Sync input 1
10b - Reserved
11b - Reserved</td></tr>
<tr><td>13-12
FORCE_MODE1</td><td>Force Clearing Mode
Specifies the timing for clearing force events for output 1 in this LC.
00b - Deassertion. Cleared on deassertion of force inputs
01b - Rising sync after deassertion. Cleared on rising sync after deassertion of force inputs
10b - Writing 1 after deassertion. Cleared by writing 1 to the correct LCn_STS[FORCESTS] or FORCESTS[FORCESTS] bits after deassertion of force inputs
11b - Rising sync after writing 1 and deassertion. Cleared on rising sync after writing 1 to the correct LCn_STS[FORCESTS] or FORCESTS[FORCESTS] bits and deassertion of force inputs</td></tr>
<tr><td>11-8
FORCE_SENSE1</td><td>Force Input Sensitivity
Selects which force inputs affect output 1 of this LC.
For each bit:
0b - Does not affect
1b - Affects
Mapping of bits:<table>
<tr><th>Input</th><th>Bit</th></tr>
<tr><td>0</td><td>8</td></tr>
<tr><td>1</td><td>9</td></tr>
<tr><td>2</td><td>10</td></tr>
<tr><td>Reserved</td><td>11</td></tr>
</table>
Example: 011b selects force inputs 0 and 1, but not 2.</td></tr>
<tr><td>7-6</td><td>Sync Select</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
SYNC_SEL0	Selects which sync input to use for output 0 of this LC. 00b - Sync input 0 01b - Sync input 1 10b - Reserved 11b - Reserved
5-4 FORCE_MODE0	Force Clearing Mode Specifies the timing for clearing force events for output 0 in this LC. 00b - Deassertion. Cleared on deassertion of force inputs 01b - Rising sync after deassertion. Cleared on rising sync after deassertion of force inputs 10b - Writing 1 after deassertion. Cleared by writing 1 to the correct LCn_STS[FORCESTS] or FORCESTS[FORCESTS] bits after deassertion of force inputs 11b - Rising sync after writing 1 and deassertion. Cleared on rising sync after writing 1 to the correct LCn_STS[FORCESTS] or FORCESTS[FORCESTS] bits and deassertion of force inputs
3-0 FORCE_SENSE0	Force Input Sensitivity Selects which force inputs affect output 0 of this LC. For each bit: 0b - Does not affect 1b - Affects Mapping of bits: Example: 011b selects force inputs 0 and 1, but not 2.

Input	Bit
0	0
1	1
2	2
Reserved	3

62.8.9 LC n Sync Control (LC0_SCTRL - LC2_SCTRL)

Offset

Register	Offset
LC0_SCTRL	34h

Table continues on the next page...

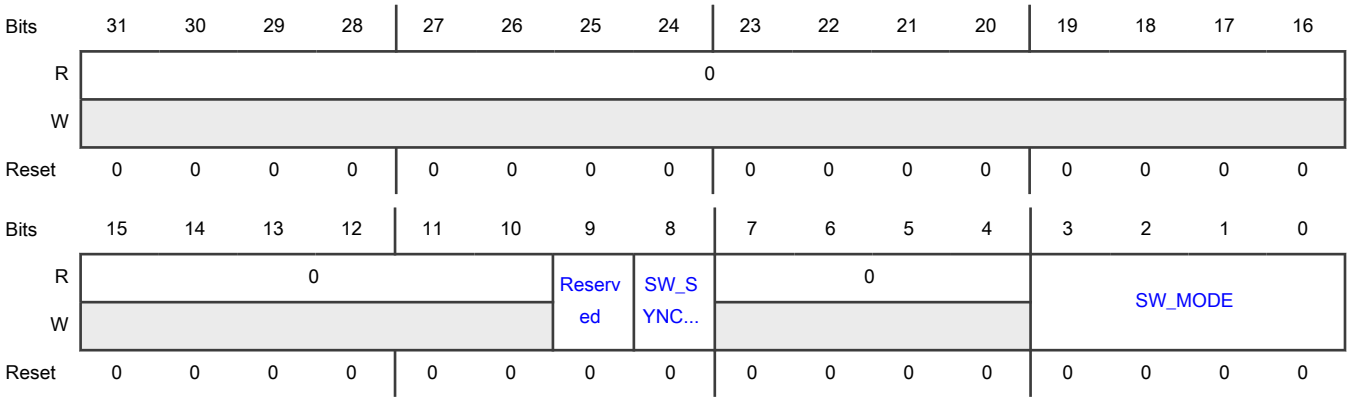
Table continued from the previous page...

Register	Offset
LC1_SCTRL	74h
LC2_SCTRL	B4h

Function

Controls the software sync behavior.

Diagram



Fields

Field	Function
31-10 —	Reserved
9 —	Reserved
8 SW_SYNC_SEL	Software Sync Select Selects which sync input to use for software synced mode. 0b - Sync input 0 1b - Sync input 1
7-4 —	Reserved
3-0 SW_MODE	Software Sync Mode Specifies the software sync mode for the inputs to this LC. When Software Override is enabled (SWEN), these bits control whether Software Override Value (SWVALUE) changes occur immediately or on the rising edge of the selected sync pulse. For each bit:

Table continued from the previous page...

Field	Function
	0b - Immediate 1b - On rising edge of sync Mapping of bits:

Input	Bit
0	0
1	1
2	2
3	3

62.8.10 Mux Select (MUXSEL0 - MUXSEL11)

Offset

For a = 0 to 11:

Register	Offset
MUXSELa	200h + (a × 4h)

Function

Selects the sources for inputs to the LCs.

NOTE

Access to Address space 200h + (48 to (124)) does not generate a transfer error.

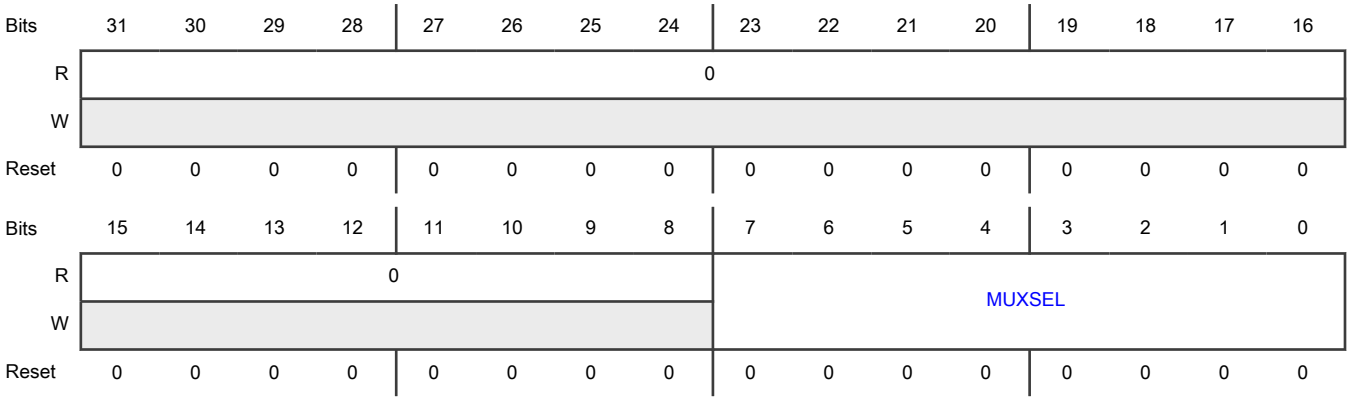
Register	LC	Input
MUXSEL0	0	0
MUXSEL1		1
MUXSEL2		2
MUXSEL3		3
MUXSEL4	1	0
MUXSEL5		1
MUXSEL6		2
MUXSEL7		3
MUXSEL8	2	0

Table continues on the next page...

Table continued from the previous page...

Register	LC	Input
MUXSEL9		1
MUXSEL10		2
MUXSEL11		3

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 MUXSEL	Mux Select Selects the source of the LC input. All LU_IN inputs go through a two-stage synchronizer, but the LU_OUT signals fed back to this mux do not have this two-stage delay because they are already synchronous to LCU clock. 0000_0000b - Logic 0 0000_0001b-0000_1100b - LU_IN0 to LU_IN11 0000_1101b-0001_1000b - LU_OUT0 to LU_OUT11 All other values are reserved.

62.8.11 Configuration (CFG)

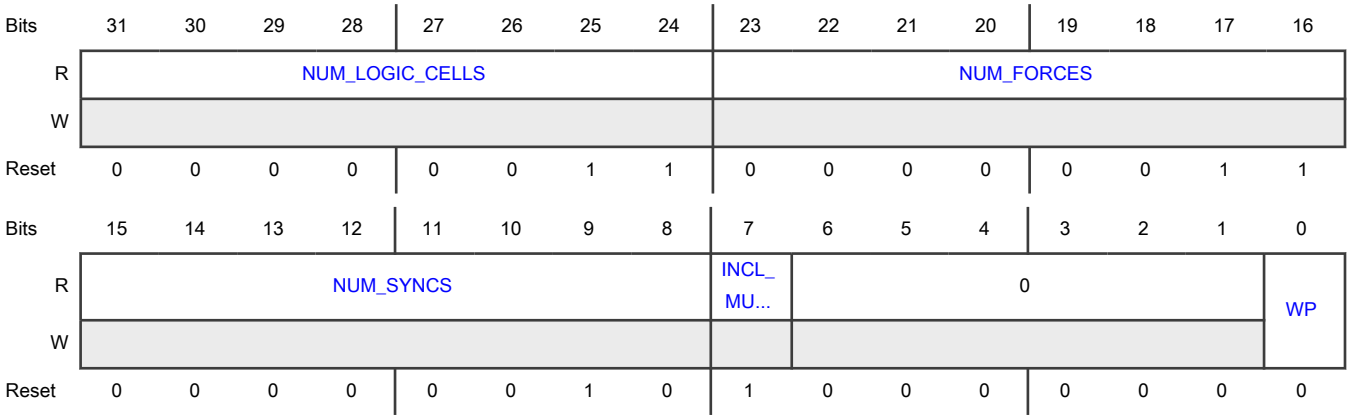
Offset

Register	Offset
CFG	280h

Function

Indicates the LCU configuration, and provides write protection.

Diagram



Fields

Field	Function
31-24 NUM_LOGIC_CELLS	LCs Indicates the number of LCs.
23-16 NUM_FORCES	Force Inputs Indicates the number of force inputs for each LC.
15-8 NUM_SYNCS	Sync Inputs Indicates the number of sync inputs for each LC.
7 INCL_MUXES	Input Muxing Indicates whether LCU supports input muxing. 0b - Not supported 1b - Supported
6-1 —	Reserved
0 WP	Write Protect Turns on write protection for all LCU registers except SWVALUE, LC*n*_STS, and FORCESTS. 0b - No effect 1b - Turn on write protection

62.8.12 Software Override Enable (SWEN)

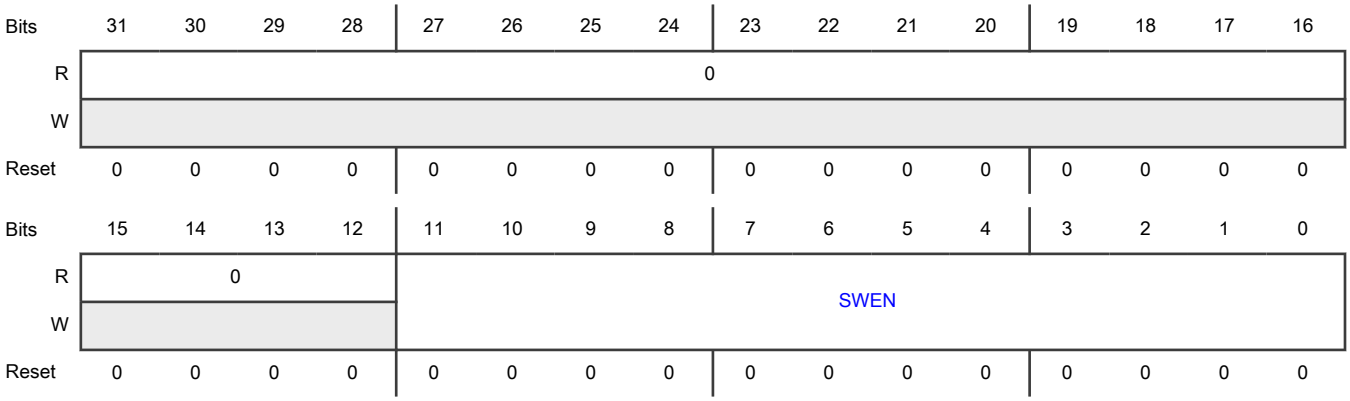
Offset

Register	Offset
SWEN	284h

Function

Enables overrides for LC inputs.

Diagram



Fields

Field	Function
31-12 —	Reserved
11-0 SWEN	Software Override Enable Enables software override of LC inputs. For each bit: 0b - Disable 1b - Enable Mapping of bits: See Mapping of 12-bit input, output, and state fields.

62.8.13 Software Override Value (SWVALUE)

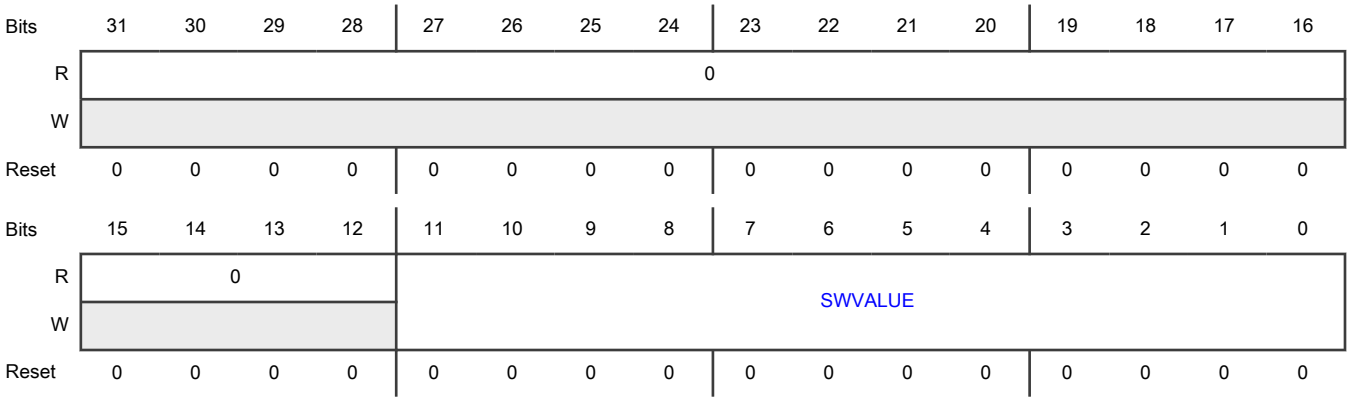
Offset

Register	Offset
SWVALUE	288h

Function

Specifies the software override value for each LC input.

Diagram



Fields

Field	Function
31-12 —	Reserved
11-0 SWVALUE	Software Override Value Specifies the software override value for each LC input. For each bit: 0b - 0 1b - 1 Mapping of bits: See Mapping of 12-bit input, output, and state fields.

62.8.14 Output Enable (OUTEN)

Offset

Register	Offset
OUTEN	28Ch

Function

Enables LC outputs.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				OUTEN											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-12 —	Reserved
11-0 OUTEN	Output Enables Enables LC outputs. For each bit: 0b - Disable 1b - Enable Mapping of bits: See Mapping of 12-bit input, output, and state fields.

62.8.15 Logic Inputs (LCIN)

Offset

Register	Offset
LCIN	290h

Function

Indicates states of LC inputs.

If you write to this register, LCU generates a bus transfer error.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				LC_INPUTS											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-12 —	Reserved
11-0 LC_INPUTS	Logic Inputs Indicates states of LC inputs. For each bit: 0b - 0 1b - 1 Mapping of bits: See Mapping of 12-bit input, output, and state fields.

62.8.16 Overridden Inputs (SWOUT)

Offset

Register	Offset
SWOUT	294h

Function

Indicates states of LC inputs or states of software-overridden inputs, depending upon the state of the corresponding SWEN bit.

If you write to this register, LCU generates a bus transfer error.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				SWOUT											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-12 —	Reserved
11-0 SWOUT	Overridden Inputs Indicates states of LC inputs or software-overridden inputs. For each bit, when the corresponding SWEN bit is 0: 0b - LC input is 0 1b - LC input is 1 For each bit, when the corresponding SWEN bit is 1: 0b - Software-overridden input is 0 1b - Software-overridden input is 1 Mapping of bits: See Mapping of 12-bit input, output, and state fields.

62.8.17 Logic Outputs (LCOUT)

Offset

Register	Offset
LCOUT	298h

Function

Indicates states of LC outputs.

If you write to this register, LCU generates a bus transfer error.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				LCOUT											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-12 —	Reserved
11-0 LCOUT	Logic Outputs Indicates states of LC outputs. For each bit: 0b - 0 1b - 1 Mapping of bits: See Mapping of 12-bit input, output, and state fields.

62.8.18 Forced Outputs (FORCEOUT)

Offset

Register	Offset
FORCEOUT	29Ch

Function

Indicates the current state of the force outputs for all LCU logic outputs. Writing to this register generates a bus transfer error.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				FORCEOUT											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-12 —	Reserved
11-0 FORCEOUT	Forced Outputs Indicates the current state of force outputs for the logic outputs. For each bit: 0b - Not asserted 1b - Asserted Mapping of bits: See Mapping of 12-bit input, output, and state fields.

62.8.19 Force Status (FORCESTS)

Offset

Register	Offset
FORCESTS	2A0h

Function

See FORCESTS.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				FORCESTS											
W					W1C											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-12 —	Reserved
11-0 FORCESTS	Force Status Indicates the current force states of all LCs, mirrored from LC*n*_STS[FORCESTS]. Change these bits to 0 by writing 1 to them or to the corresponding bit in the specific LC*n*_STS register. This field allows you to simultaneously change FORCESTS bits to 0 across multiple LCs. For each bit: 0b - Not in force state 1b - Read: In force state — Write: Clear force state bit Mapping of bits: See Mapping of 12-bit input, output, and state fields.

62.8.20 Debug Mode Enable (DBGEN)

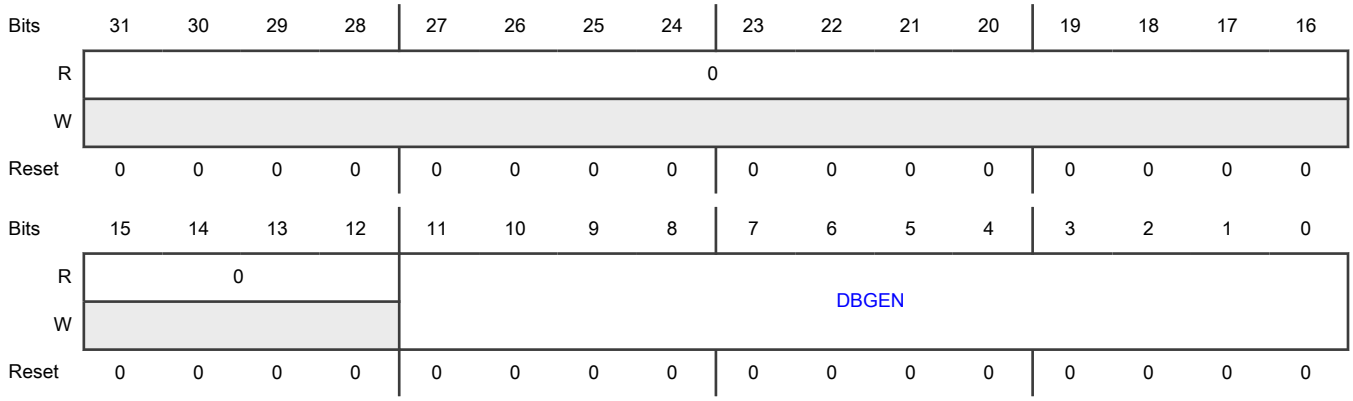
Offset

Register	Offset
DBGEN	2A8h

Function

Enables outputs to continue operation when the chip is in Debug mode.

Diagram



Fields

Field	Function
31-12 —	Reserved
11-0 DBGEN	Debug Mode Enable Enables outputs to continue operation in Debug mode. For each bit: 0b - Debug mode inactive 1b - Debug mode active Mapping of bits: See Mapping of 12-bit input, output, and state fields.

62.9 Glossary

CCW	Counterclockwise
CW	Clockwise
LC	Logic cell
LUT	Lookup table

Chapter 63
Enhanced Modular IO Subsystem (eMIOS)

63.1 Chip-specific eMIOS information

63.1.1 eMIOS configuration

This chip has up to three instances of eMIOS that are each configured as shown in the following tables.

Table 401. eMIOS instances

Instance	S32K314/S32K324/S32K344/S32K358/ S32K356/S32K348/S32K338/S32K328/ S32K388/S32K389	S32K310/S32K311/S32K312/S32K322/ S32K342/S32K341
eMIOS_0	Yes	Yes
eMIOS_1	Yes	Yes
eMIOS_2	Yes	No

Table 402. eMIOS configuration

Parameter	eMIOS_0	eMIOS_1	eMIOS_2
Timer width	16 bits[1]	16 bits[1]	16 bits[1]
Number of channels	24	24	24
Channels configuration	See EMIOS channel configuration figure below		
Local Channel prescaler width	4	4	4
Number of global counter buses	2	2	2
Number of global prescaler	1	1	1
Global channel prescaler width	8	8	8

1. For S32K388/S32K389: 24 bits

NOTE

eMIOS is clocked by CORE_CLK.

63.1.2 Channel types

The eMIOS contain 4 different types of channel, which are listed below.

Table 403. eMIOS channel types

Mode Description	Name	Ch TypeX	Ch TypeY	Ch TypeG	Ch TypeH
General Purpose Input / Output	GPIO	X	X	X	X
Single Action Input Capture	SAIC	X	X	X	X

Table continues on the next page...

Table 403. eMIOS channel types (continued)

Mode Description	Name	Ch TypeX	Ch TypeY	Ch TypeG	Ch TypeH
Single Action Output Compare	SAOC	X	X	X	X
Modulus Counter	MC	X			
Modulus Counter Buffered (Up / Down)	MCB	X		X	
Input Pulse Width Measurement	IPWM			X	X
Input Period Measurement	IPM			X	X
Double Action Output Compare	DAOC			X	X
Output Pulse Width and Frequency Modulation Buffered	OPWFMB	X		X	
Center aligned Output PWM Buffered with dead time	OPWMCB			X	
Output Pulse Width Modulation Buffered	OPWMB	X	X	X	X
Output Pulse Width Modulation Trigger	OPWMT	X	X	X	X
Pulse Edge Counting	PEC			X	

NOTE

ChType X and G has internal counters.

Table 404. eMIOS channel configuration

Channel number	eMIOS0	eMIOS1	eMIOS2
CH0	X	X	X
CH8	X	X	X
CH16	X	X	X
CH22	X	X	X
CH23	X	X	X
CH1	G	H	H
CH2	G	H	H

Table continues on the next page...

Table 404. eMIOS channel configuration (continued)

CH3	G	H	H
CH4	G	H	H
CH5	G	H	H
CH6	G	H	H
CH7	G	H	H
CH9	H	H	H
CH10	H	H	H
CH11	H	H	H
CH12	H	H	H
CH13	H	H	H
CH14	H	H	H
CH15	H	H	H
CH17	Y	Y	Y
CH18	Y	Y	Y
CH19	Y	Y	Y
CH20	Y	Y	Y
CH21	Y	Y	Y

63.1.3 Channel configuration

The eMIOS channel configuration with respect to Type and Counter_bus is shown below.

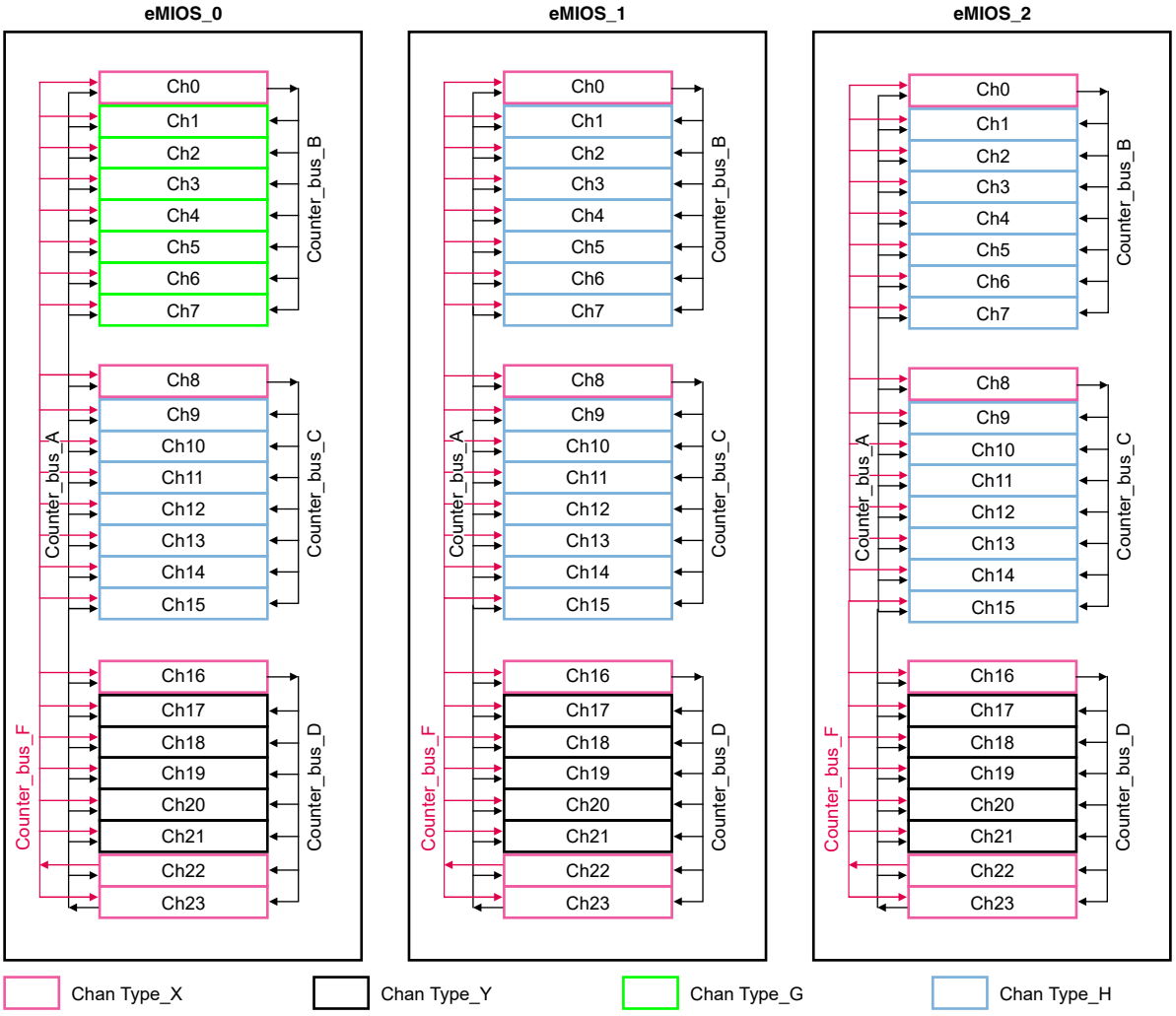


Figure 321. EMIOS channel types

63.1.4 eMIOS disabling

eMIOS can disable it's output using it's disable inputs. Disable inputs[3:0] are driven from the flag out bits[11:8].

Disable inputs	
eMIOS0 Output disable input[3:0]	trgmux_output[31:28]
eMIOS1 Output disable input[3:0]	trgmux_output[31:28]
eMIOS2 Output disable input[3:0]	trgmux_output[31:28]

For details, see the TRGMUX connectivity file attached to this document.

63.1.5 BCTU Interface

Each eMIOS channel, with the exception of the main counterbus channel (ch[23]) is capable of triggering the BCTU. See BCTU chapter for mapping. BCTU allows triggering from as many eMIOS modes as possible.

63.2 Introduction

eMIOS provides independent channels, UCs, that you configure to generate or measure time events for different functions in different chip applications.

eMIOS distributes these channels across a number of global and local counter buses. Each local bus is dedicated to a group of eight contiguous channels. Each channel can generate its own timebase, and each counter bus has its timebase provided by a dedicated channel. For a list of counter buses with their assigned channels and timebase sources, see Counter buses, channels, and timebase sources.

63.3 Block diagram

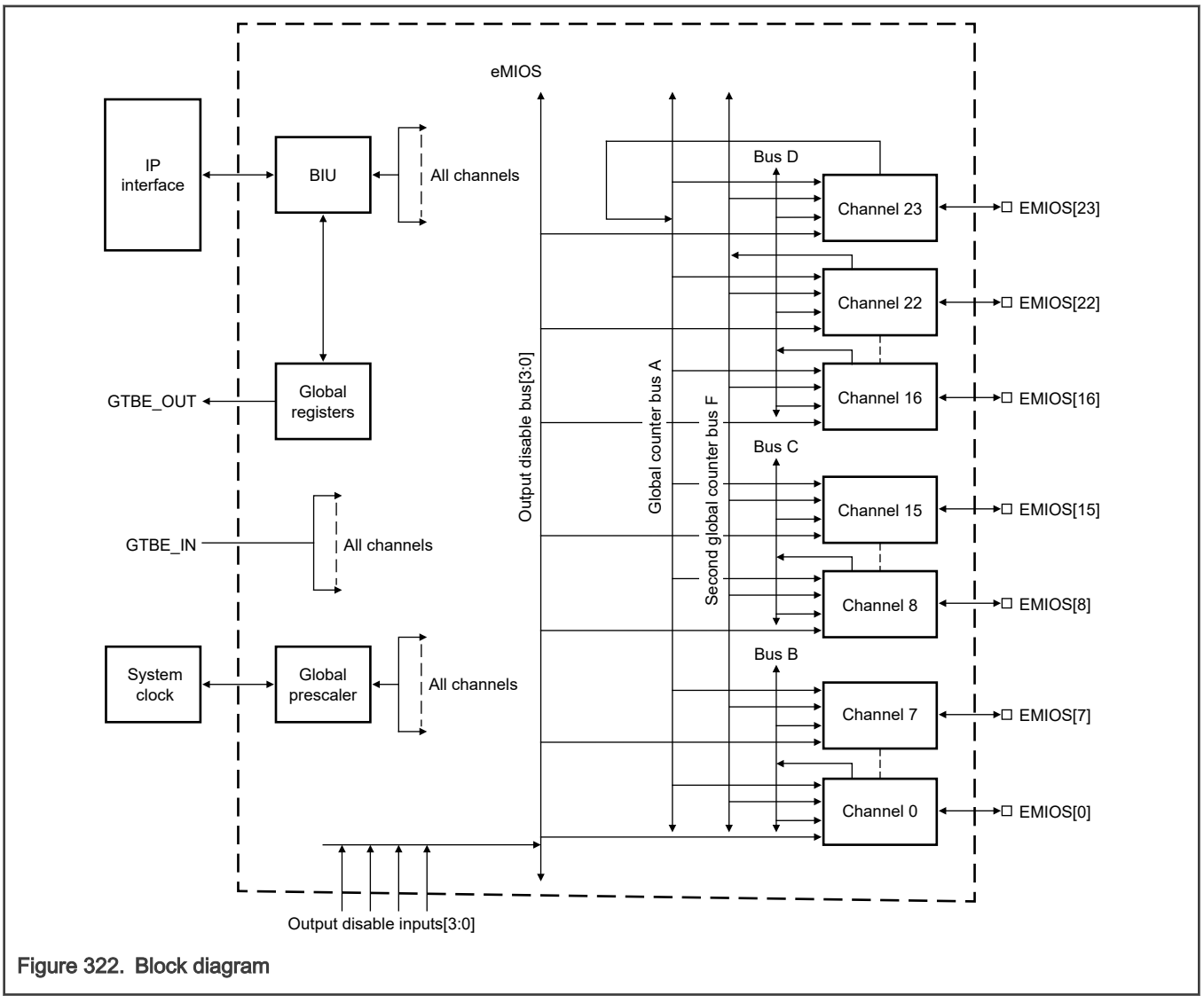


Figure 322. Block diagram

63.4 Features

- 24 UCs distributed across local counter buses as shown in Counter buses, channels, and timebase sources
- Global counter bus A driven by UC23
- Global counter bus F driven by UC22
- Synchronized timebases shared through the counter buses

- Dedicated timebase for each channel, distinct from the counter buses
- Global clock prescaler (GCP)
- One CP per channel
- Dedicated set of control and status registers for each UC
- 24-bit–wide data registers
- Shadow flag register (GFLAG) to access all channel flags with one read access
- Ability to freeze the UC state for debug purposes
- Motor control capability

63.5 Functional description

63.5.1 Reset

eMIOS resets asynchronously. Reset affects all registers.

On reset, UCs enter GPIO input mode.

63.5.2 Counter buses, channels, and timebase sources

Table 405. Counter buses, channels, and timebase sources

Counter bus	Channels	Timebase channel
Global bus A	All UCs	23
Local bus B	• UC0 • UC1 • UC2 • UC3 • UC4 • UC5 • UC6 • UC7	0
Local bus C	• UC8 • UC9 • UC10 • UC11 • UC12 • UC13 • UC14 • UC15	8
Local bus D	• UC16	16

Table continues on the next page...

Table 405. Counter buses, channels, and timebase sources (continued)

Counter bus	Channels	Timebase channel
	• UC17 • UC18 • UC19 • UC10 • UC21 • UC22 • UC23	
Global bus F	All UCs	22

63.5.3 Unified channels (UC)

63.5.3.1 Overview

Each UC contains the following:

- Two double-buffered data registers, A*n* and B*n*, that allow up to two events—input capture, output compare, or both—to occur before software intervention is needed
- Two comparators, A and B, that indicate when the selected counter bus is equal to the value in A*n* and B*n*
- An internal counter (CNT*n*[C]) that runs in all modes except GPIO.

  You can use this counter as a local timebase or to count input events. You can use it as a timebase if CNT*n*[C] is not used in the current mode.
- An output flip-flop that holds the logic level to be applied to the output pin
- A status register, UC Status n (S0 - S23), that flags input capture and match events, indicates flag overruns and overflows, and shows the input and output pin states
- A control register, UC Control n (C0 - C23), that controls UC operation

You control the following operational characteristics for each UC:

- The logic (C*n*[MODE]) that specifies the behavior of the UC (see UC modes)
- The counter bus (C*n*[BSL]) that provides the timebase for all timing functions
- The CP (C*n*[UCPRE] and C2_*n*[UCEXTPRE])
- The minimum input pulse width (C*n*[IF]) for the input filter that determines valid pin transitions
- Which input edges to detect (C*n*[EDSEL])
- Which of four output disable input signals (C*n*[ODISSL]) to use for disabling outputs
- For all output modes except GPIO, whether to disable the output flip-flop (C*n*[ODIS])

UC block diagram illustrates the basic UC architecture.

For a list of counter buses and their assigned channels and timebase sources, see Counter buses, channels, and timebase sources.

63.5.3.2 UC block diagram

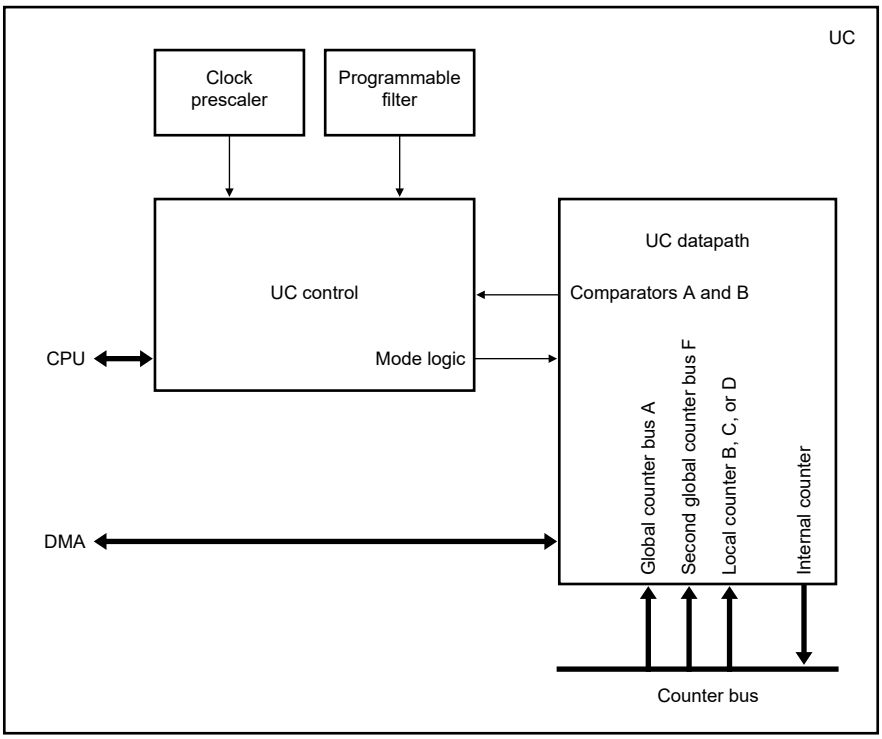


Figure 323. UC block diagram

63.5.3.3 Buffered modes

To provide smooth waveform generation even when A*n*[A] and B*n*[B] are changing, the following modes double buffer the A*n* and B*n* registers:

- Modulus Counter Buffered (MCB) mode
- Output Pulse Width and Frequency Modulation Buffered (OPWFMB) mode
- Output PWM Buffered (OPWMB) mode

These modes appear in separate sections because there are several basic differences in UC operation between them and their corresponding nonbuffered modes.

63.5.3.4 UC control and datapath

As illustrated in UC control and datapath diagram, the UC contains control and datapath sub-blocks.

The control sub-block generates signals to control the multiplexers in the datapath sub-block. eMIOS implements each UC mode in dedicated logic independent from the other modes. The UC modes share a set of registers allowing the storing of sequential events.

The datapath block contains the channel A and B registers, the internal timebase, and the comparators. Multiplexers receive inputs from the control sub-block to configure the datapath for the specific channel modes. This configuration consists of selecting the input of comparators and data for the register inputs. The outputs of the A and B comparators connect to the control block.

63.5.3.5 UC control and datapath diagram

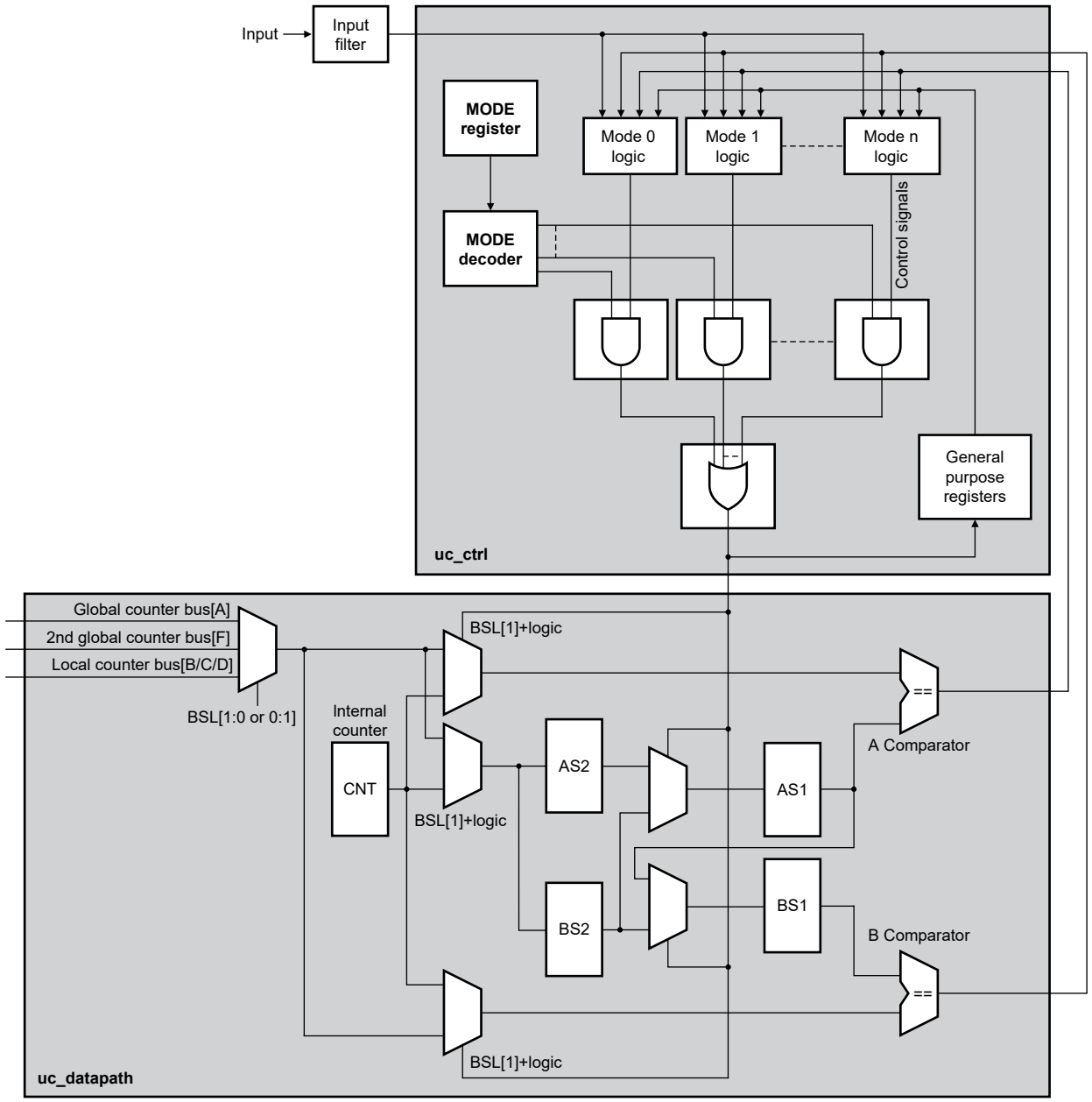


Figure 324. UC control and datapath diagram

63.5.3.6 UC modes

You can configure UCs to operate in the following modes:

- General-Purpose Input and Output (GPIO) mode
- Single Action Input Capture (SAIC) mode
- Single Action Output Capture (SAOC) mode
- Input Pulse Width Measurement (IPWM) mode
- Input Period Measurement (IPM) Mode

- Double Action Output Compare (DAOC) mode
- Pulse Edge Counting (PEC) mode
- Modulus Counter (MC) mode
- Modulus Counter Buffered (MCB) mode
- Output Pulse Width and Frequency Modulation Buffered (OPWFMB) mode
- Center Aligned Output PWM with Dead Time Insertion Buffered (OPWMCB) mode
- Output PWM Buffered (OPWMB) mode
- Output PWM with Trigger (OPWMT) mode

Each channel supports a specific subset of these modes. See the chip-configuration information to see which modes each channel supports.

63.5.3.7 General-Purpose Input and Output (GPIO) mode

63.5.3.7.1 Overview

This mode disables all UC input-capture and output-compare functions, and resets and disables the internal counter (C*n*[C]). All control fields remain accessible.

To change the UC from one operation mode to another, you must first change to GPIO mode and then change to the desired final mode. To prepare the UC for a new operation mode, you must configure AS1, BS1, AS2, and BS2 before exiting GPIO mode and entering the desired final mode (see AS1, AS2, BS1, and BS2 shadow registers). Specifically:

- Writing to A*n*[A] stores the same value in AS1 and AS2.
- Writing to B*n*[B] stores the same value in BS1 and BS2.
- Writing to ALTA*n*[ALTA] stores a value only in AS2.

As its name implies, GPIO mode supports both input and output submodes. See Differences for input and output submodes.

63.5.3.7.2 Differences for input and output submodes

Table 406. Differences for input and output submodes

Submode	To enter submode, write this value to C*n*[MODE]	Characteristics
Input	0	• You control the generation of input-capture flags with C*n*[EDPOL] and C*n*[EDSEL]. • You can determine the input pin status by reading S*n*[UCIN].
Output	1	• The UC performs as a single output port pin. • The value of C*n*[EDPOL] permanently drives the output flip-flop.

63.5.3.8 Single Action Input Capture (SAIC) mode

63.5.3.8.1 Overview

In this mode, when a triggering event occurs on the input pin, eMIOS:

- Captures the selected timebase into AS2
- Changes S*n*[FLAG] to 1 to indicate that an input capture has occurred

A*n*[A] returns the value of AS2. As soon as the UC enters SAIC mode after exiting GPIO mode, the channel is ready to capture events.

eMIOS captures the events as soon as they occur. Therefore, reading A*n*[A] always returns the value of the latest captured event. Subsequent capture events occur regardless of whether you read the previous event from A*n*[A]. Each new capture event sets the input capture flag (writes 1 to S*n*[FLAG]).

The input capture is triggered by an edge transition on the input pin as configured by C*n*[EDPOL] and C*n*[EDSEL].

SAIC with rising edge triggering and SAIC with both edges triggering show how to use the UC for input capture.

63.5.3.8.2 SAIC submodes and MODE field values

Table 407. SAIC submodes and MODE field values

Capture behavior	C*n*[MODE] (binary)
The UC only captures the timestamp.	000_0010
• The UC captures a timestamp. • AS2 bit 31 indicates the input signal level.	100_0010

63.5.3.8.3 Effect of C*n*[EDPOL] on edge capturing

Table 408. Effect of C*n*[EDPOL] on edge capturing

C*n*[EDPOL]	Input signal transition	Resulting value of AS2 bit 31
0	0→1	0
	1→0	1
1	0→1	1
	1→0	0

63.5.3.8.4 SAIC with rising edge triggering

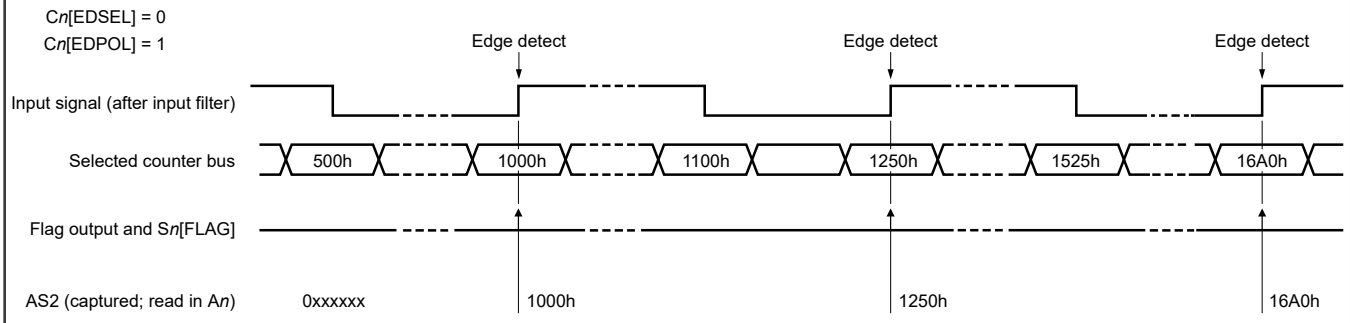


Figure 325. SAIC with rising edge triggering

63.5.3.8.5 SAIC with both edges triggering

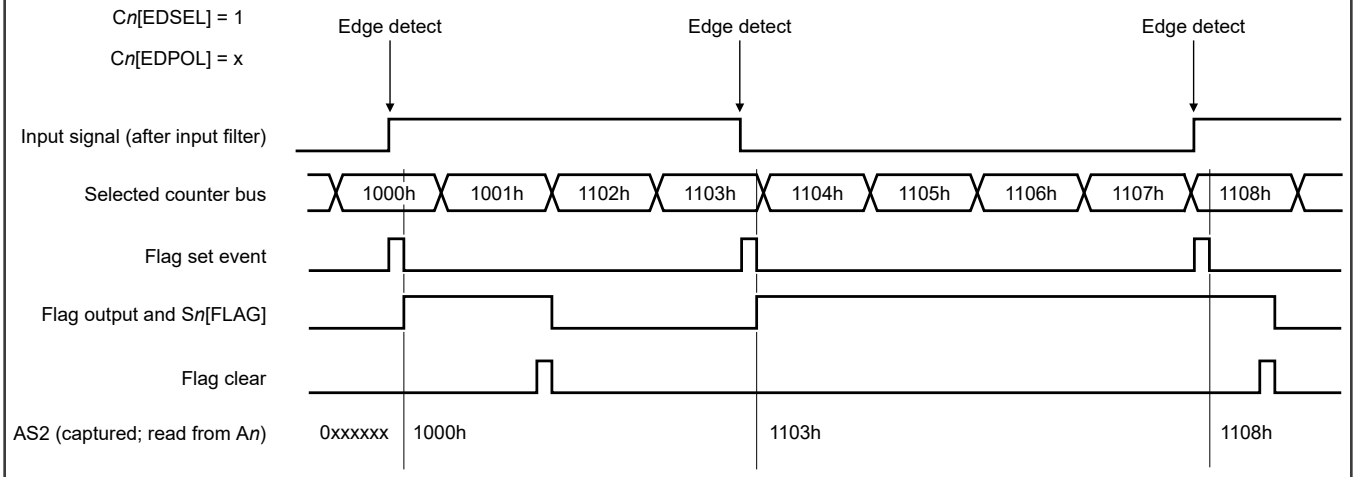


Figure 326. SAIC with both edges triggering

63.5.3.9 Single Action Output Capture (SAOC) mode

63.5.3.9.1 Overview

In this mode, eMIOS loads a match value into AS2 and then immediately transfers it to AS1 for comparison with the selected timebase. When an output-compare match occurs, eMIOS:

- Either toggles the output flip-flop or transfers C*n*[EDPOL] to the output flip-flop, as determined by C*n*[EDSEL].
- Sets the input capture flag (changes S*n*[FLAG] to 1).

Along with the match, S*n*[FLAG] becomes 1 to indicate that the output-compare match has occurred. Writing to A*n*[A] stores the value in AS2, and reading A*n*[A] returns the AS1 value.

The channel internal counter in SAOC mode is free-running. It starts counting as soon as the UC enters SAOC mode.

You can force an output-compare match by writing 1 to C*n*[FORCMA]. A forced output-compare match does not set the input capture flag (S*n*[FLAG]).

When the UC enters SAOC mode after exiting GPIO mode, the output flip-flop value becomes the complement of C*n*[EDPOL].

You select the internal or external counter bus with C*n*[BSL].

SAOC with C*n*[EDPOL] transferred to the output flip-flop shows how to perform a single output compare and transfer C*n*[EDPOL] to the output flip-flop.

SAOC toggling the output flip-flop shows how to perform a single output compare and toggle the output flip-flop at each match.

63.5.3.9.2 Preload the desired match value

eMIOS enables matches immediately after the UC enters SAOC mode. Therefore, you must write the desired match value to AS1 before the UC enters SAOC mode. You can update AS1 at any time. This modifies the match value reflected in the channel-generated output signal. Subsequent capture events occur regardless of whether you write to A*n*[A] again. Each new capture event sets the input capture flag (S*n*[FLAG]). See SAOC with flag behavior.

63.5.3.9.3 SAOC with C*n*[EDPOL] transferred to the output flip-flop

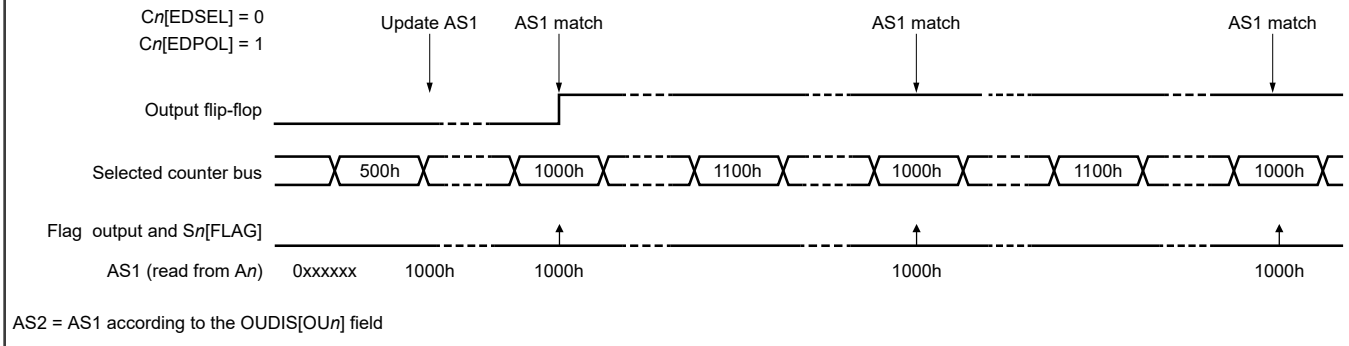


Figure 327. SAOC with C*n*[EDPOL] transferred to the output flip-flop

63.5.3.9.4 SAOC toggling the output flip-flop

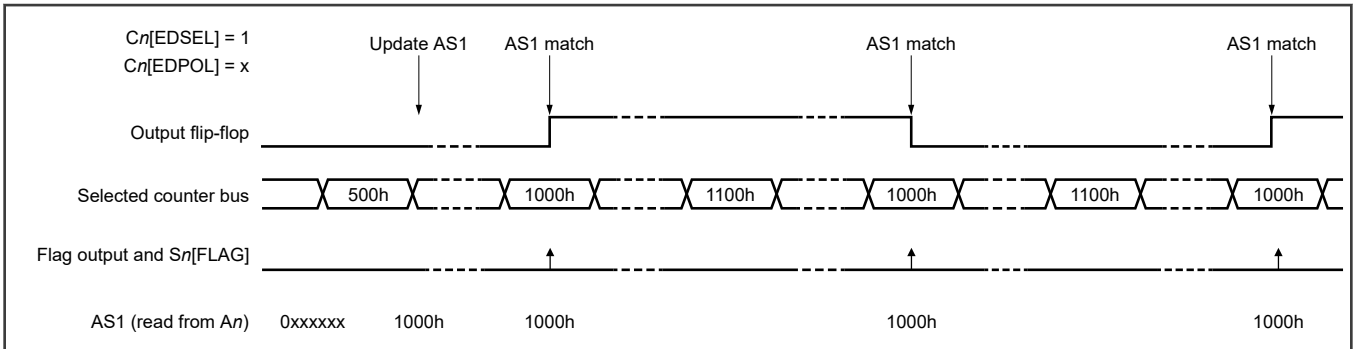


Figure 328. SAOC toggling the output flip-flop

63.5.3.9.5 SAOC with flag behavior

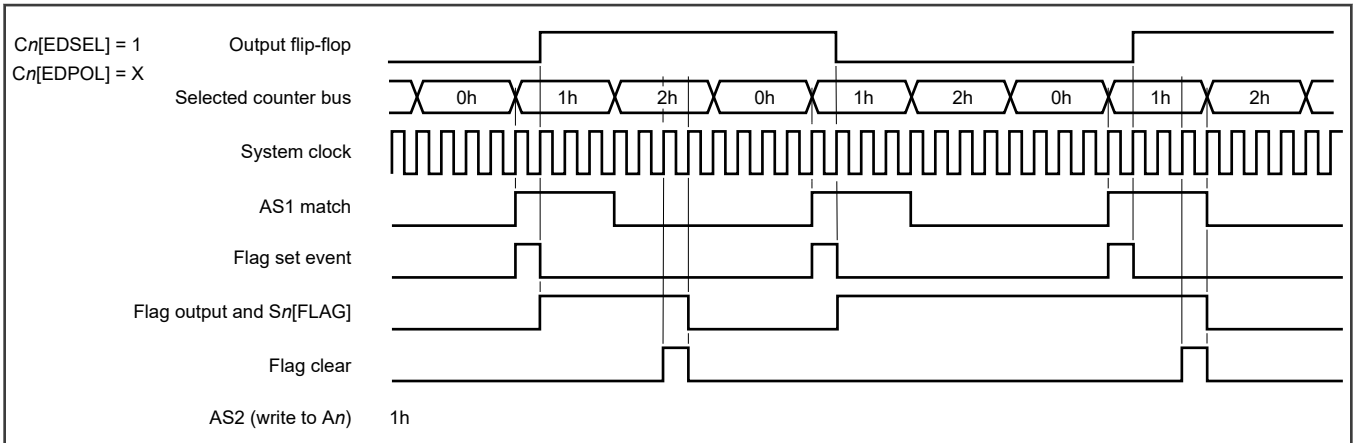


Figure 329. SAOC with flag behavior

63.5.3.10 Input Pulse Width Measurement (IPWM) mode

63.5.3.10.1 Overview

This mode allows you to measure the width of a positive or negative pulse. Select whether to measure from the rising or falling edge with Edge Polarity (Cn[EDPOL]).

The first leading edge:

- Triggers the first input capture, which latches the count value of the selected timebase into BS2 (see AS1, AS2, BS1, and BS2 shadow registers)
- Does not set the input-capture flag (Sn[FLAG])
- Enables AS2 capture

The next trailing edge:

- Latches the selected timebase into AS2
- Sets the input-capture flag (Sn[FLAG])
- Transfers BS2 to BS1 and AS1

eMIOS captures successive values on consecutive edges of opposite polarity. If you initiate subsequent input capture events while the input-capture flag (Sn[FLAG]) is set, eMIOS updates AS2, BS1, and AS1 with the latest captured values and keeps the flag set.

63.5.3.10.2 Coherency

Reading A*n*[A] updates BS1 with AS1. It also disables transfers from BS2 and BS1 until the next read of B*n*[B].

Reading Bn updates BS1 with AS1. It also re-enables transfers from BS2 to BS1, which takes effect at the next trailing edge capture.

Because transfers from BS2 to AS1 are never blocked, to guarantee data coherency you must read An, then read Bn.

If you do not require coherent data, you should read Bn before you read An, although doing so requires a second read of Bn to unblock BS1 updates.

63.5.3.10.3 Timing example: measure pulse width

This figure shows how to measure the input pulse width by subtracting BS1 from AS2.

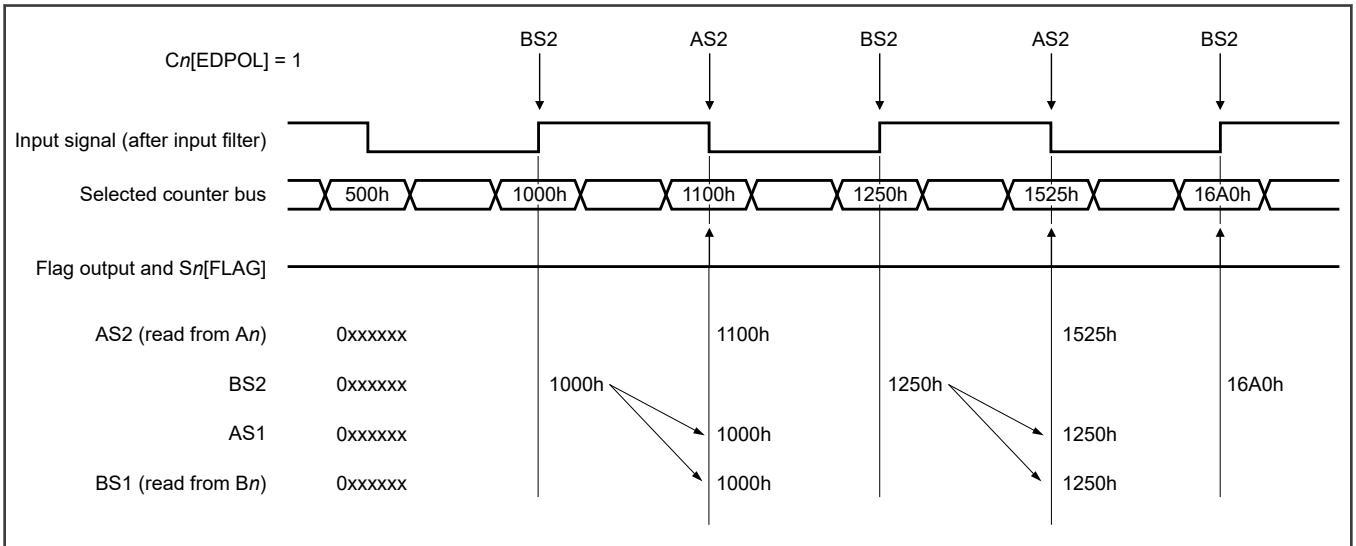


Figure 330. Timing example: measure pulse width

63.5.3.10.4 Timing example: AS1 and BS1 updates on channel reads

This example illustrates AS1 and BS1 updates when you read A*n*[A] and B*n*[B]. AS1 always has coherent data related to AS2 (see Coherency).

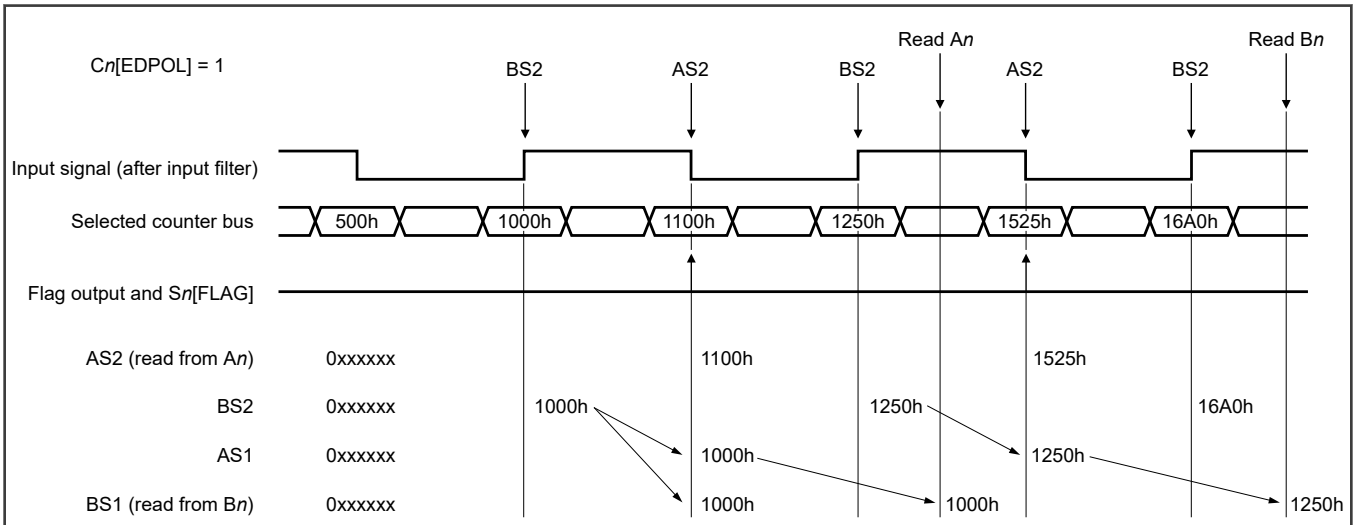


Figure 331. Timing example: BS1 and AS1 updates on channel reads

63.5.3.11 Input Period Measurement (IPM) Mode

63.5.3.11.1 Overview

This mode allows you to measure the period of an input signal. Select whether to measure from the rising or falling edges with Edge Polarity (C*n*[EDPOL]).

The first edge of selected polarity:

- Latches the selected timebase into AS2 and BS2 (see AS1, AS2, BS1, and BS2 shadow registers)
- Transfers BS2 to BS1
- Does not set the input-capture flag (S*n*[FLAG])

The second edge of the same polarity:

- Latches the counter bus value into AS2 and BS2
- Transfers BS2 to BS1 and AS1
- Sets the input-capture flag (S*n*[FLAG])

eMIOS repeats this process for each subsequent capture.

eMIOS captures successive values on two consecutive edges of the same polarity. The measured input signal must have a period of at least four system clock cycles to be properly captured by the synchronization logic at the channel input, even if the input filter is in bypass mode.

63.5.3.11.2 Coherency

Reading A*n*[A] updates BS1 with AS1, which provides coherent data in AS2 and BS1 (see Timing example: AS1 and BS1 updates on channel reads). It also disables transfers from BS2 and BS1 until the next read of B*n*[B].

Reading B*n*[B] updates BS1 with AS1. It also re-enables transfers from BS2 to BS1, which takes effect at the next edge capture.

63.5.3.11.3 Timing example: measure input period

This example illustrates how to measure the input period by subtracting BS1 from AS2.

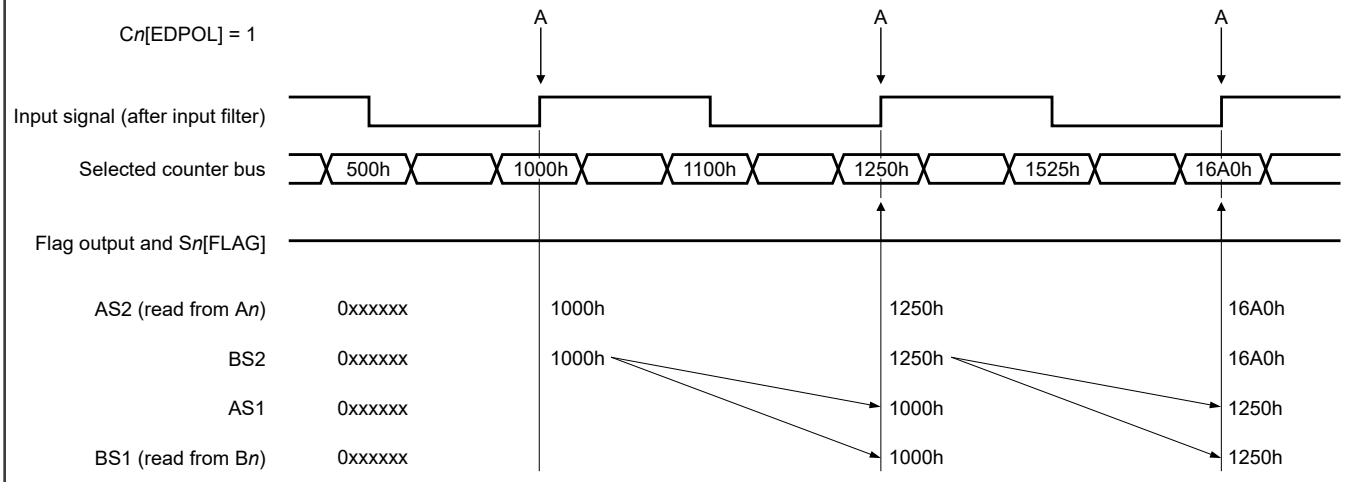


Figure 332. Timing example: measure input period

63.5.3.11.4 Timing example: AS1 and BS1 updates on channel reads

This diagram illustrates AS1 and BS1 updates when you read A*n*[A] and B*n*[B].

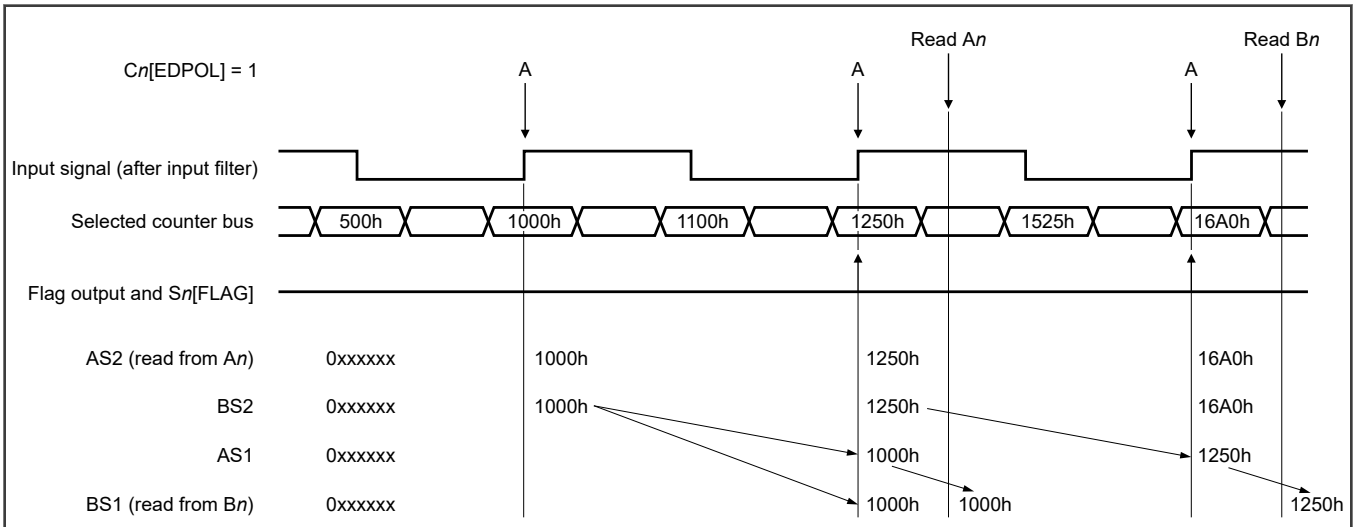


Figure 333. Timing example: AS1 and BS1 updates on channel reads

63.5.3.12 Double Action Output Compare (DAOC) mode

63.5.3.12.1 Overview

In this mode, matches on comparators A and B cause the UC to generate the leading and trailing edges of the variable pulse width output. See DAOC submodes and MODE field values. There is no restriction on the order in which A and B matches occur.

When the UC enters DAOC mode from GPIO mode, eMIOS disables both comparators and drives the complement of Edge Polarity (C*n*[EDPOL]) onto the output flip-flop.

63.5.3.12.2 DAOC submodes and MODE field values

Table 409. DAOC submodes and MODE field values

When set flag event occurs	C*n*[MODE] (binary)
Only on B matches	000_0110
On A or B matches	000_0111

63.5.3.12.3 Data transfer and compare

If you enable output (write 0 to OUDIS[OU*n*]), on the next clock cycle the UC transfers AS2 and BS2 to AS1 and BS1 respectively (see Timing example: leading dead time insertion). If you disable output (write 1 to OUDIS[OU*n*]), no transfer occurs.

eMIOS enables and disables the A and B comparators independently. It enables comparator A only after the transfer to AS1 completes, and disables comparator A on the next A match. Similarly, eMIOS enables comparator B only after the transfer to BS1 completes, and disables comparator B on the next B match.

When a match occurs on comparator A, the output flip-flop acquires the value of Edge Polarity (C*n*[EDPOL]). When a match occurs on comparator B, the output flip-flop acquires the complement of Edge Polarity (C*n*[EDPOL]).

63.5.3.12.4 Differences between flag on A match and flag on A and B matches

In the flag on B match submode (see DAOC submodes and MODE field values), the UC sets the input-capture flag (S*n*[FLAG]) only for matches on the B comparator. In the flag on both A and B matches submode, it sets the flag for matches on either comparator.

If subsequent enabled output compares occur on AS1 or BS1, the UC continues generating pulses regardless of the flag state.

If you load both AS1 and BS1 with the same value, the B match takes precedence and the UC drives the complement of Edge Polarity (C*n*[EDPOL]) onto the output flip-flop.

63.5.3.12.5 Forced operation by Cn[FORCMA] and Cn[FORCMB]

Force Match A (C*n*[FORCMA]) and Force Match B (C*n*[FORCMB]) allow you to force the output flip-flop to the level corresponding to the comparator, Edge Polarity (C*n*[EDPOL]) for an A match, or the complement of Edge Polarity for a B match. The force-match operations do not set the input-capture flag (S*n*[FLAG]).

63.5.3.12.6 Timing example: flag on both matches

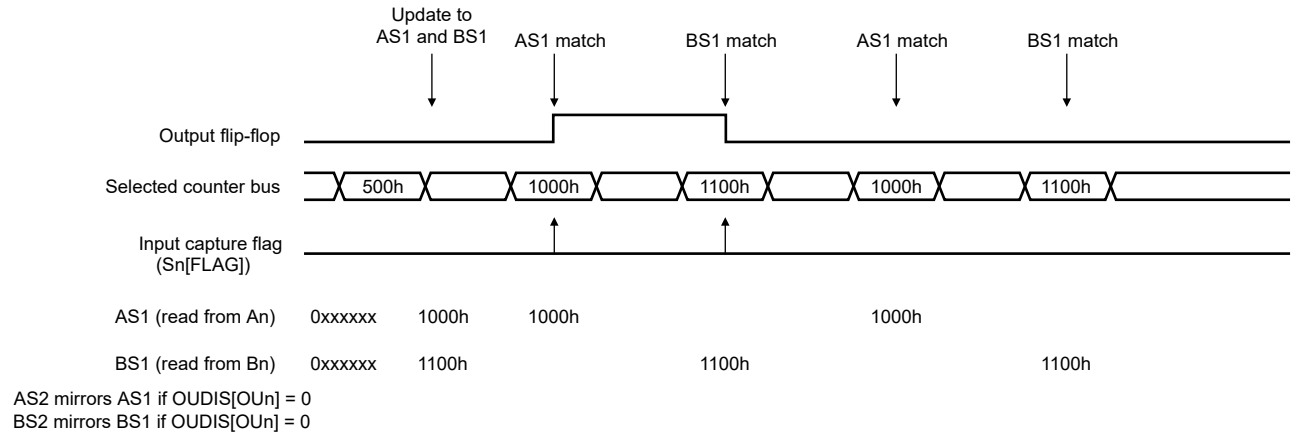


Figure 334. Timing example: flag on both matches

63.5.3.12.7 Timing example: flag on second match

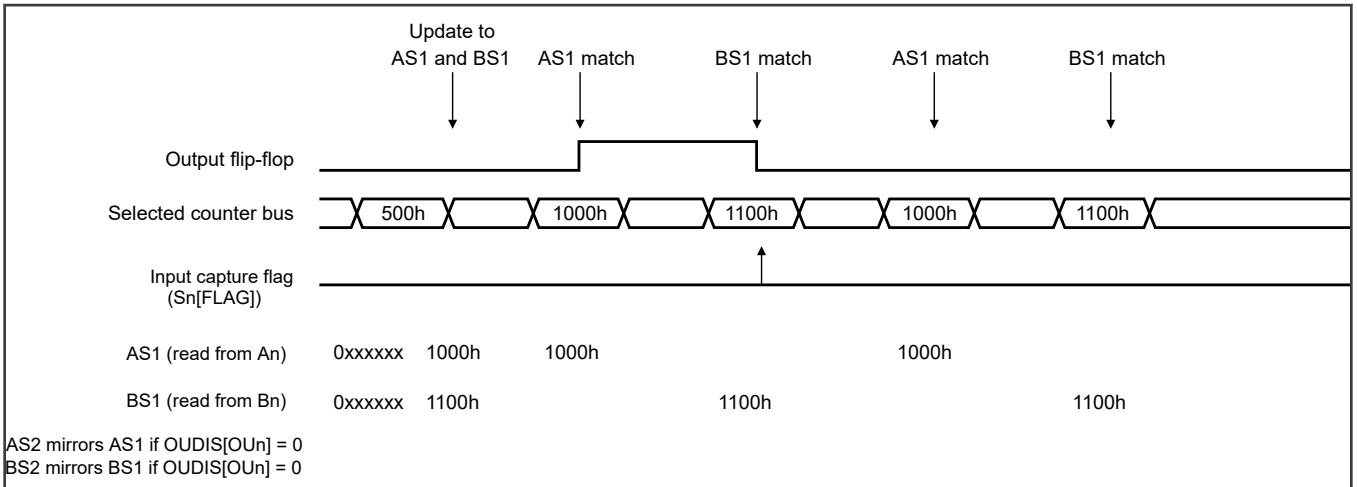


Figure 335. Timing example: flag on second match

63.5.3.12.8 Timing example: transfer disabling

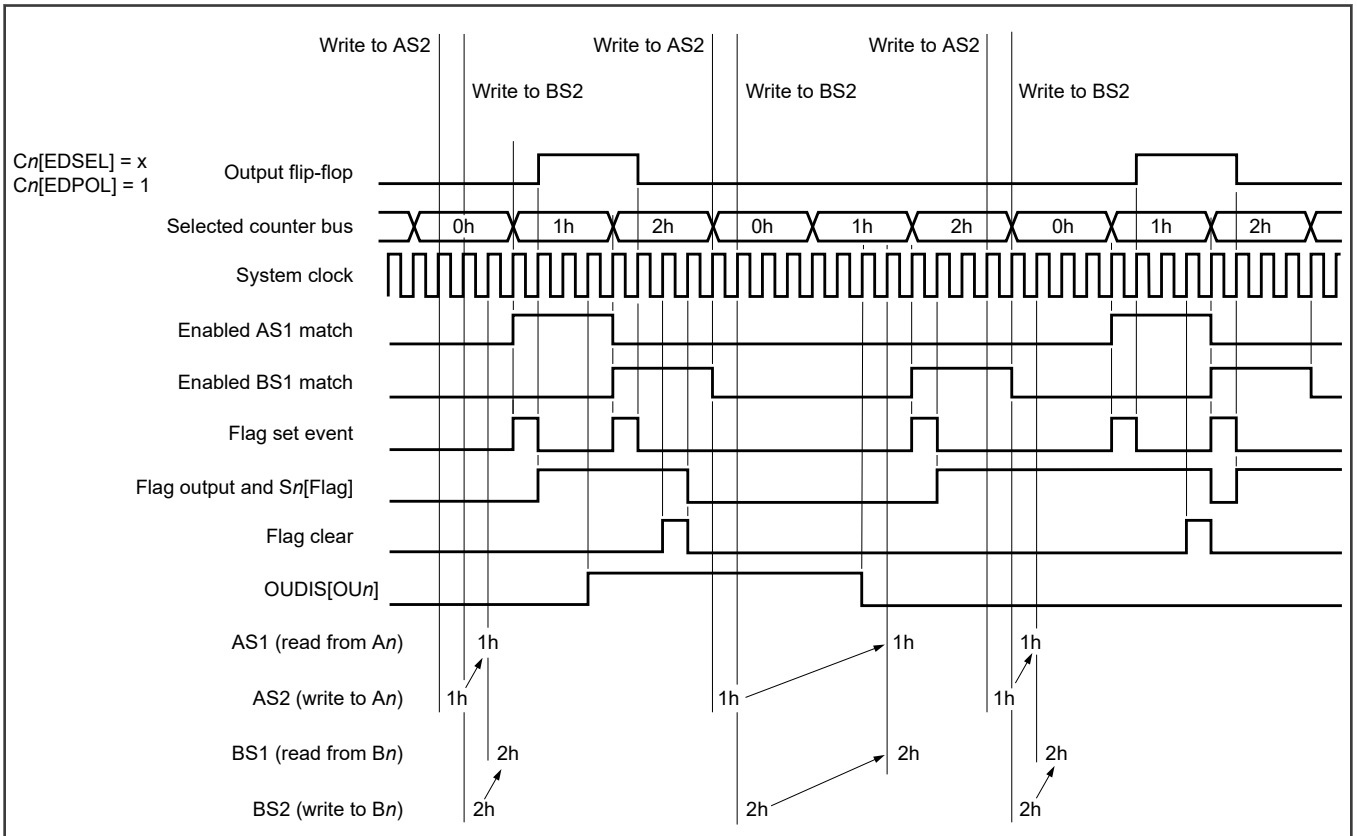


Figure 336. Timing example: transfer disabling

63.5.3.13 Pulse Edge Counting (PEC) mode

63.5.3.13.1 Overview

This mode counts the number of pulses or edges detected on the input for a desired time window. C*n*[EDSEL] and C*n*[EDPOL] determine which edges the UC counts.

Specify the window start time in AS1 and the end time in BS1. After you write to AS1, when the selected timebase matches comparator A, eMIOS resets the internal counter and starts counting input events. When the timebase matches comparator B, eMIOS:

- Disables the internal counter.
- Transfers the counter's content to AS2.
- Sets the input-capture flag (changes S*n*[FLAG] to 1).

You obtain the number of detected pulses by reading C*n*[C] or AS2.

Timing example continuous and Timing example single-shot show how to use the UC in continuous or single-shot operation.

63.5.3.13.2 PEC submodes and MODE field values

Table 410. PEC submodes and MODE field values

Operation	C*n*[MODE] (binary)
Continuous; the next match between comparator A and the selected timebase resets the internal counter and enables counting again. To guarantee coherent measurements when you read C*n*[C] after S*n*[FLAG] becomes 1, you must check if the timebase value is out of the time interval defined by AS1 and BS1. Alternately, AS2 (available in ALTA*n*[ALTA]) always holds the latest available measurement, providing coherent data at any time after the first set-flag event.	000_1010
Single-shot; The next match between comparator A and the selected timebase has no effect until you write to A*n*[A]. eMIOS also transfers the C*n*[C] value to AS2 when a match in the B comparator occurs.	000_1011

63.5.3.13.3 Timing example continuous

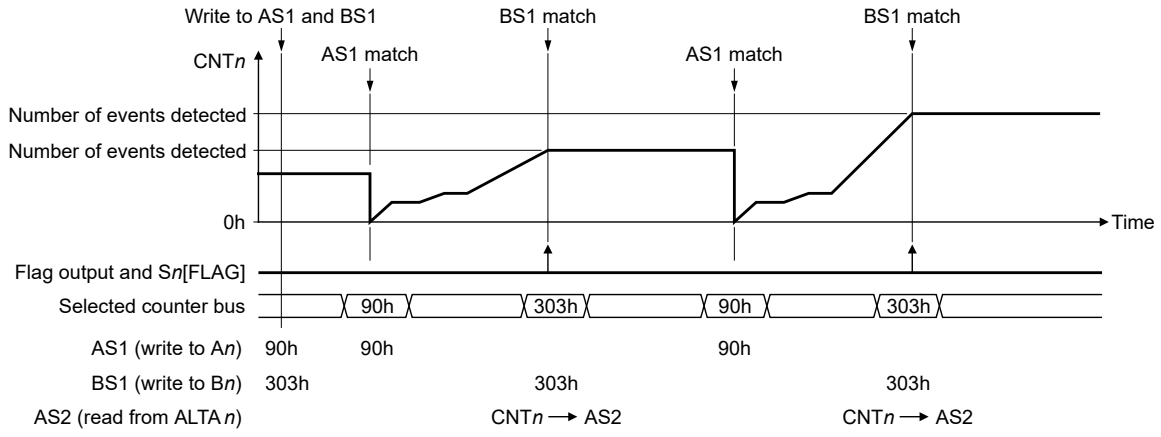


Figure 337. Timing example continuous

63.5.3.13.4 Timing example single-shot

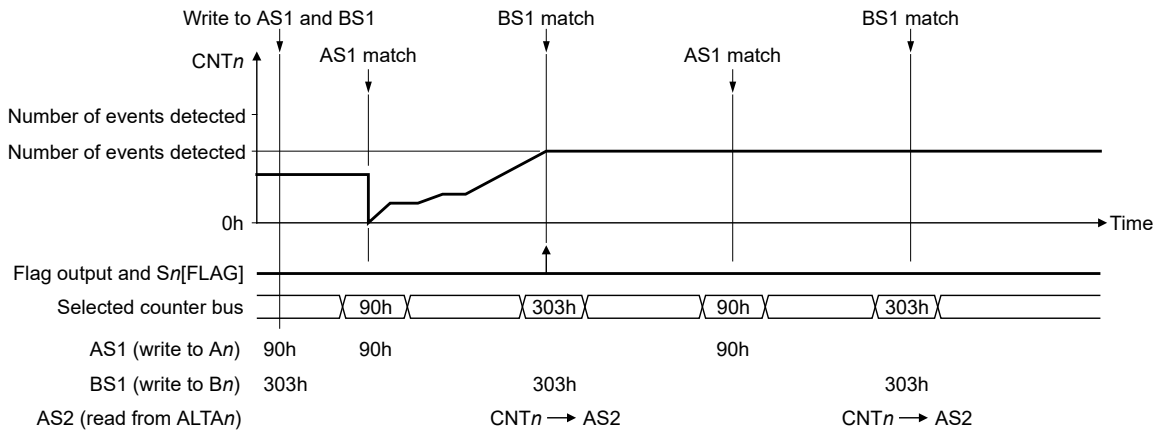


Figure 338. Timing example single-shot

63.5.3.14 Modulus Counter (MC) mode

63.5.3.14.1 Overview

In this mode, the UC produces a timebase for a counter bus or as a general purpose timer. The internal counter (C*n*[C]) counts up from the current value until it matches AS1.

Selecting external clock source (see MC submodes and MODE field values) causes the UC to uses the input signal pin as the source. You select the triggering polarity edge using Edge Polarity (C*n*[EDPOL]) and Edge Select (C*n*[EDSEL]).

You must write only non-zero values to A*n*[A]. Writing to A*n*[A] or the internal counter may cause a match to be missed and the counter to roll over and resume operation in the next cycle. The channel writes 0 to BS1, which is not accessible by the MCU. You can read and write BS2, but it is not used in this mode.

63.5.3.14.2 MC submodes and MODE field values

Table 411. MC submodes and MODE field values

Submode or function	C*n*[MODE] (binary)
Internal clock source	001_0pp0[1]
External clock source	001_0pp1[1]
Internal counter reset on match start	001_0p0p[1]
Internal counter reset on match end	001_0p1p[1]
Up Count	001_00pp[1]
Up Count-Down Count	001_01pp[1]

1. *p* = Adjust parameters for submodes and options. See Unified channels (UC).

63.5.3.14.3 Up Count submode

Table 412. Up Count submode operation

Counter reset timing[1]	C*n*[MODE]		
	Bit 1	External clock (Bit 0 = 1)	Internal clock (Bit 0 = 0)
Reset on match start	0	When the AS1 match occurs, the UC: • Writes 0 to the internal counter. • Sets the input-capture flag (S*n*[FLAG]). • Increments the internal counter at the next input event, resulting in a short zero count.	When the AS1 match occurs, the UC: • Writes 0 to the internal counter. • Sets the input-capture flag (S*n*[FLAG]). • Maintains the internal counter at 0 on the next prescaler tick after the match. • Resumes counting on the next prescaler tick.
		See Timebase with the fastest prescaler ratio and Timebase generation with clear on match start and internal clock.	
Clear on match end	1	When the next input event after the AS1 match occurs, the UC: • Writes 0 to the internal counter. • Sets the input-capture flag (S*n*[FLAG]).	When the next internal counter tick after the AS1 match occurs, the UC: • Writes 0 to the internal counter. • Sets the input-capture flag (S*n*[FLAG]).
		See Timebase with the fastest prescaler ratio and Timebase generation with clear on match end.	

1. If you select internal clock source and a prescaler value of 1 for the internal counter, the behavior is the same for both submodes.

See Timing example: Up Count submode.

63.5.3.14.4 Timing example: Up Count submode

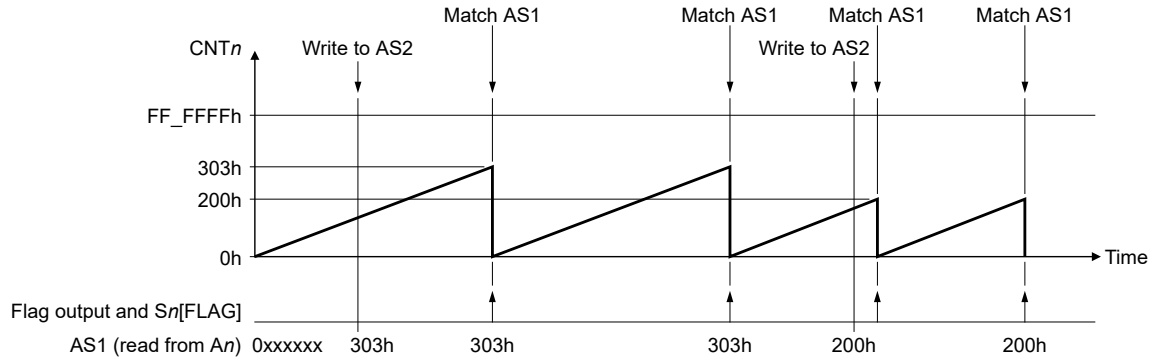


Figure 339. Timing example: Up Count submode

63.5.3.14.5 Up Count-Down Count submode operation

In Up Count-Down Count submode:

- When the internal counter matches AS1, the counter begins decrementing and the UC sets the input-capture flag (S*n*[FLAG]).
- When the internal counter matches BS1, the counter begins incrementing and, if in clear on match end operation (C*n*[1] = 1), the UC sets the input-capture flag (S*n*[FLAG]).

See Timing example: Up Count-Down Count submode.

63.5.3.14.6 Timing example: Up Count-Down Count submode

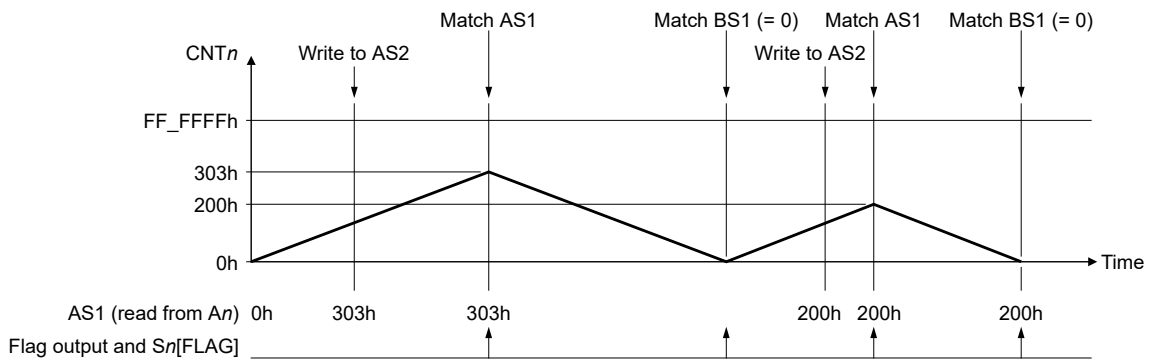


Figure 340. Timing example: Up Count-Down Count submode

63.5.3.14.7 Up Count with delayed reload operation

When the internal counter matches AS1 in the Up Count submode or BS1 in the Up Count-Down Count submode, the UC asserts the counter bus reload signal. You can read the counter bus reload signal in the following modes:

- Output PWM Buffered (OPWMB) mode
- Output Pulse Width and Frequency Modulation Buffered (OPWFMB) mode
- Center Aligned Output PWM with Dead Time Insertion Buffered (OPWMCB) mode

You can use the Reload Signal Output Delay Interval (C2*n*[UCRELDEL_INT]) to cause the UC to assert the reload signal only after a specified number of match events (from 2 to 32).

See Timing example: Up Count mode with delayed reload.

63.5.3.14.8 Timing example: Up Count mode with delayed reload

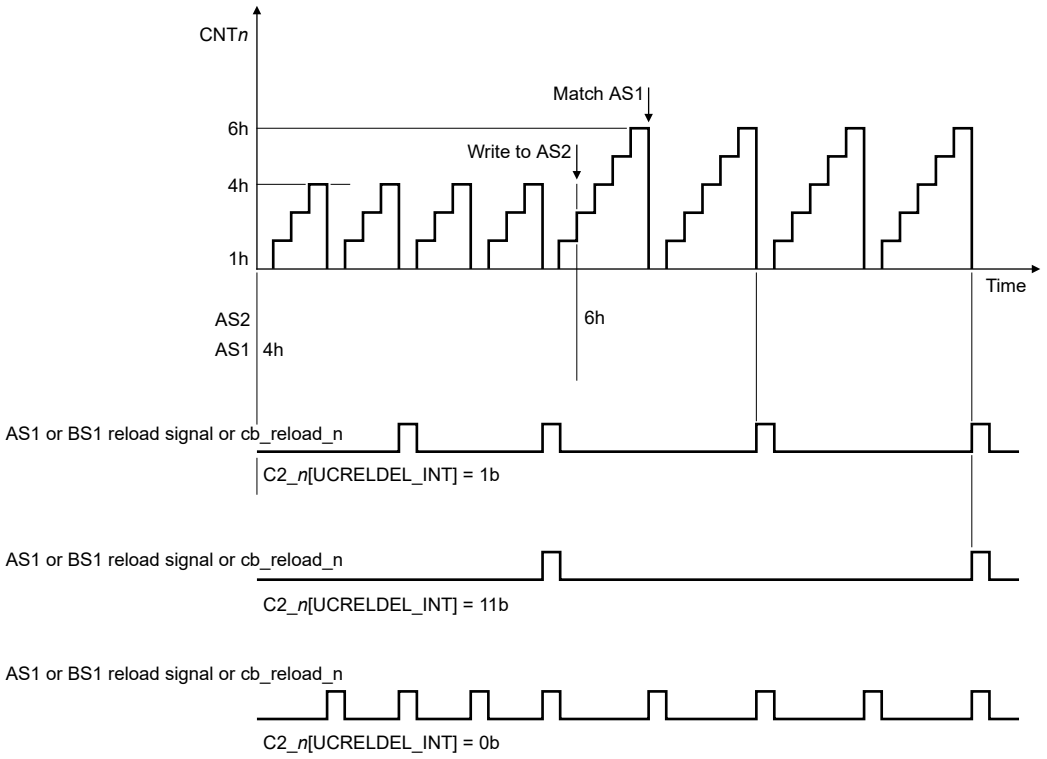


Figure 341. Timing example: Up Count mode with delayed reload

63.5.3.15 Modulus Counter Buffered (MCB) mode

63.5.3.15.1 Overview

In this mode, the UC generates a timebase that can be shared with other channels through the internal counter buses. AS1 is double-buffered, so you can change AS2 at any time with smooth transitions. The UC updates AS1 at the counter period boundary, which is when the internal counter (CNT*n*[C]) transitions to 1h.

Selecting external clock (see MCB submodes and MODE field values) causes the UC to use the input signal as the source. You select the triggering edge with Edge Polarity (C*n*[EDPOL]) and Edge Select (C*n*[EDSEL]).

You must ensure the internal counter is in the range from 1h to the AS1 value; otherwise, the A*n*[A] match does not occur and this causes the internal counter to wrap at the maximum counter value (FF_FFFFh for a 24-bit counter). After a counter wrap occurs, the UC resets the counter to 1h and resumes normal operation.

NOTE

Do not write 1h to the internal counter when the UC is in Freeze state. Doing so can cause the match to be missed and the counter to overflow.

63.5.3.15.2 MCB submodes and MODE field values

Table 413. MCB submodes and MODE field values

Submode or function	C*n*[MODE] (binary)
Internal clock source	101_0pp0b[1]

Table continues on the next page...

Table 413. MCB submodes and MODE field values (continued)

Submode or function	C*n*[MODE] (binary)
External clock source	101_0pp1b[1]
Up Count	101_000p[1]
Reserved	101_001p[1]
Up Count-Down Count	101_01pp[1]
Flag set only on match start	101_010p[1]
Flags also set at counter period boundary	101_011p[1]

1. *p* = Adjust parameters for submodes and options. See Unified channels (UC).

63.5.3.15.3 Up Count submode

In the Up Count submode:

- The internal counter (CNT*n*[C]) starts counting up (incrementing) from its current value.
- When the internal counter matches AS1 and a clock tick occurs (either prescaled clock or input pin event), the UC resets the internal counter to 1h.
- The UC sets the input-capture flag (S*n*[FLAG]) one system clock cycle after the match occurs.

You must write only values greater than 1h to AS1.

The boundary between counter period *n* and period *n*+1 occurs when the UC changes the internal counter from the AS1 value in period *n* to 1h in period *n*+1. AS1 defines the counter period.

The UC asserts the AS1 load signal (which has a duration of one system clock cycle) at the last system clock cycle of a counter period. This causes the UC to update AS1 with AS2 at the period boundary, at the same time that it resets the counter to 1h. Therefore, any value you write to AS2 during period *n* updates AS1 at the next period boundary and the UC uses it in period *n*+1. See Timing example: Up Count submode.

You can use Output Update Disable (OUDIS) to delay the update of AS1 for synchronization.

The UC sets the input-capture flag (S*n*[FLAG]) at the period boundary. See Timing example: Up Count submode flag assertion.

63.5.3.15.4 Timing example: Up Count submode

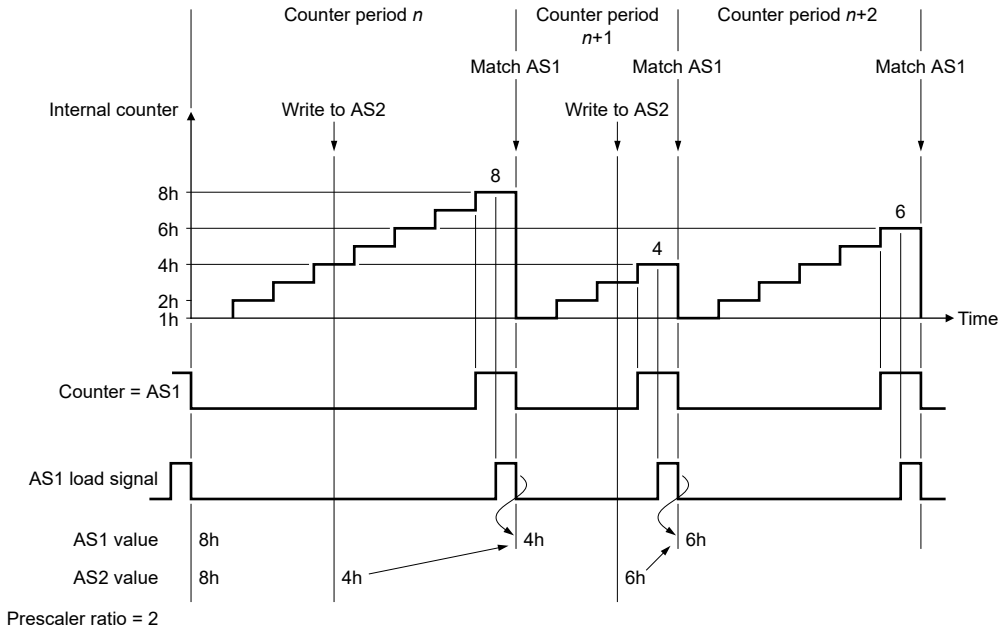


Figure 342. Timing example: Up Count submode

63.5.3.15.5 Timing example: Up Count submode flag assertion

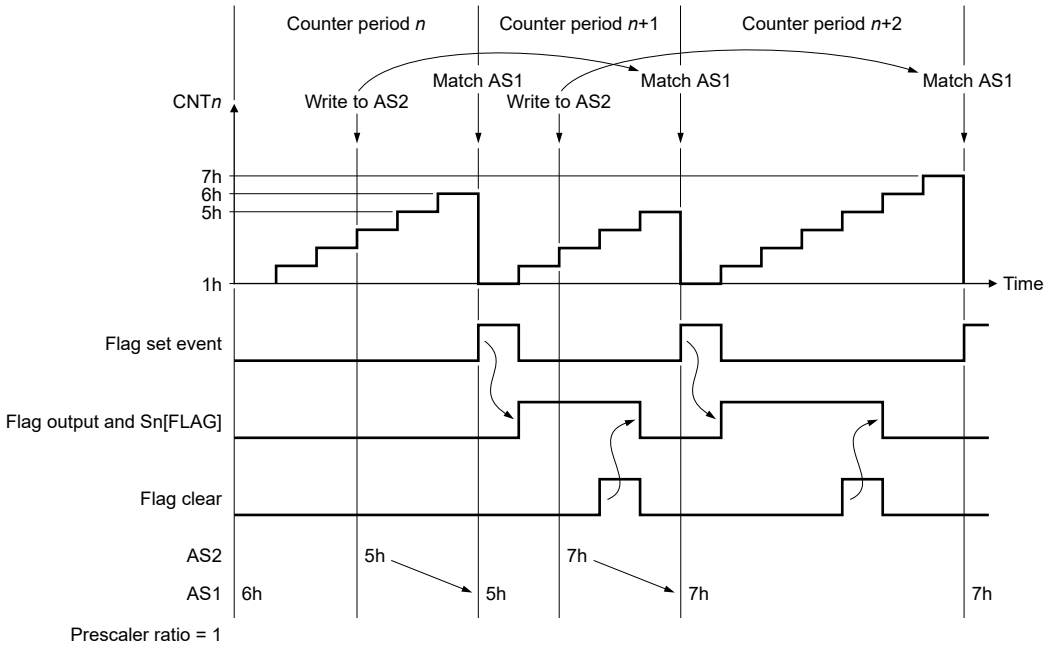


Figure 343. Timing example: Up Count submode flag assertion

63.5.3.15.6 Up Count-Down Count submode

In Up Count-Down Count submode:

- The counter period is: (2 x AS1) – 2. The counter changes direction at AS1 match and counts down (decrements) until it reaches 1h, then counts up (increments) again.
- BS1 generates a match to start the internal counter incrementing, and BS1 cannot be changed in this submode.

The UC updates AS1 at the counter period boundary. If you write AS2 (write A*n*[A]) during period *n*, the UC uses this new value in period *n*+1 for AS1 match. You can disable updates of AS1 (OUDIS[OU*n*]). See Timing example: Up Count-Down Count submode.

The UC sets the input-capture flag (S*n*[FLAG]) at AS1 match start, and may also do so at the period boundary. See Overview and Timing example: Up Count-Down Count submode flag assertion.

63.5.3.15.7 Timing example: Up Count-Down Count submode

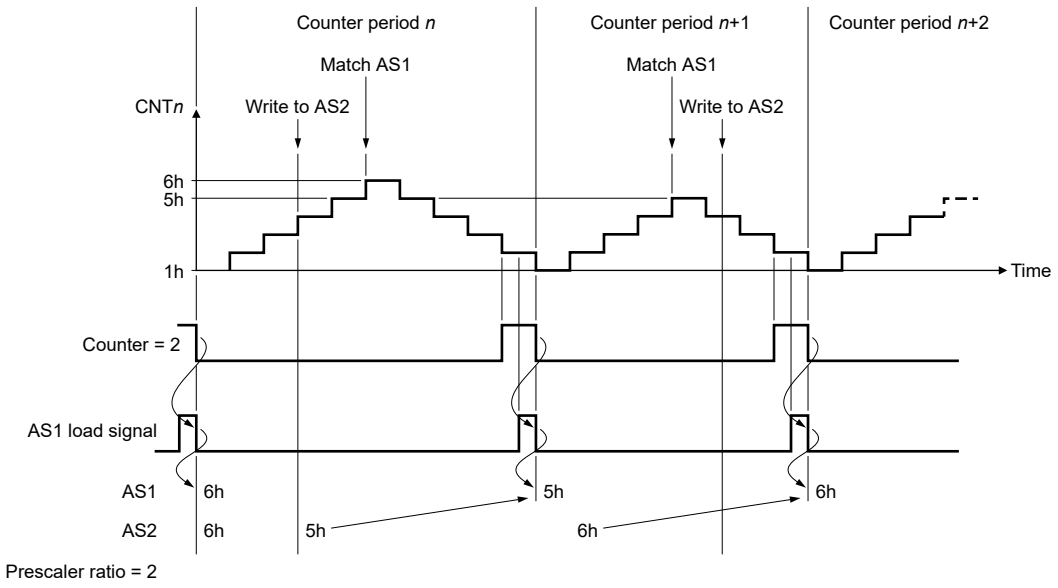


Figure 344. Timing example: Up Count-Down Count submode

63.5.3.15.8 Timing example: Up Count-Down Count submode flag assertion

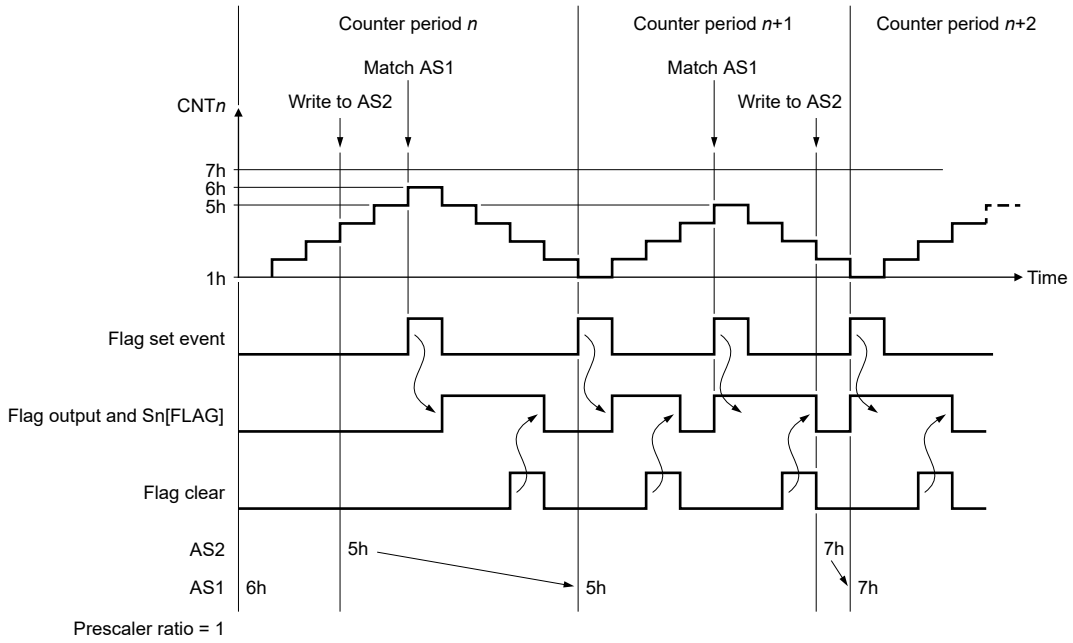


Figure 345. Timing example: Up Count-Down Count submode flag assertion

63.5.3.15.9 Up Count mode with delayed reload operation

When the internal counter matches AS1 in the Up Count submode or 1h in the Up Count-Down Count submode (see MCB submodes and MODE field values), the UC asserts the counter bus reload signal. You can read the counter bus reload signal in the following modes:

- Output PWM Buffered (OPWMB) mode
- Output Pulse Width and Frequency Modulation Buffered (OPWFMB) mode
- Center Aligned Output PWM with Dead Time Insertion Buffered (OPWMCB) mode

You can use the Reload Signal Output Delay Interval (C2[UCRELDEL_INT]) to cause the UC to assert the reload signal only after a specified number of match events (from 2 to 32).

See Timing example: Up Count mode with delayed reload.

63.5.3.15.10 Timing example: Up Count mode with delayed reload

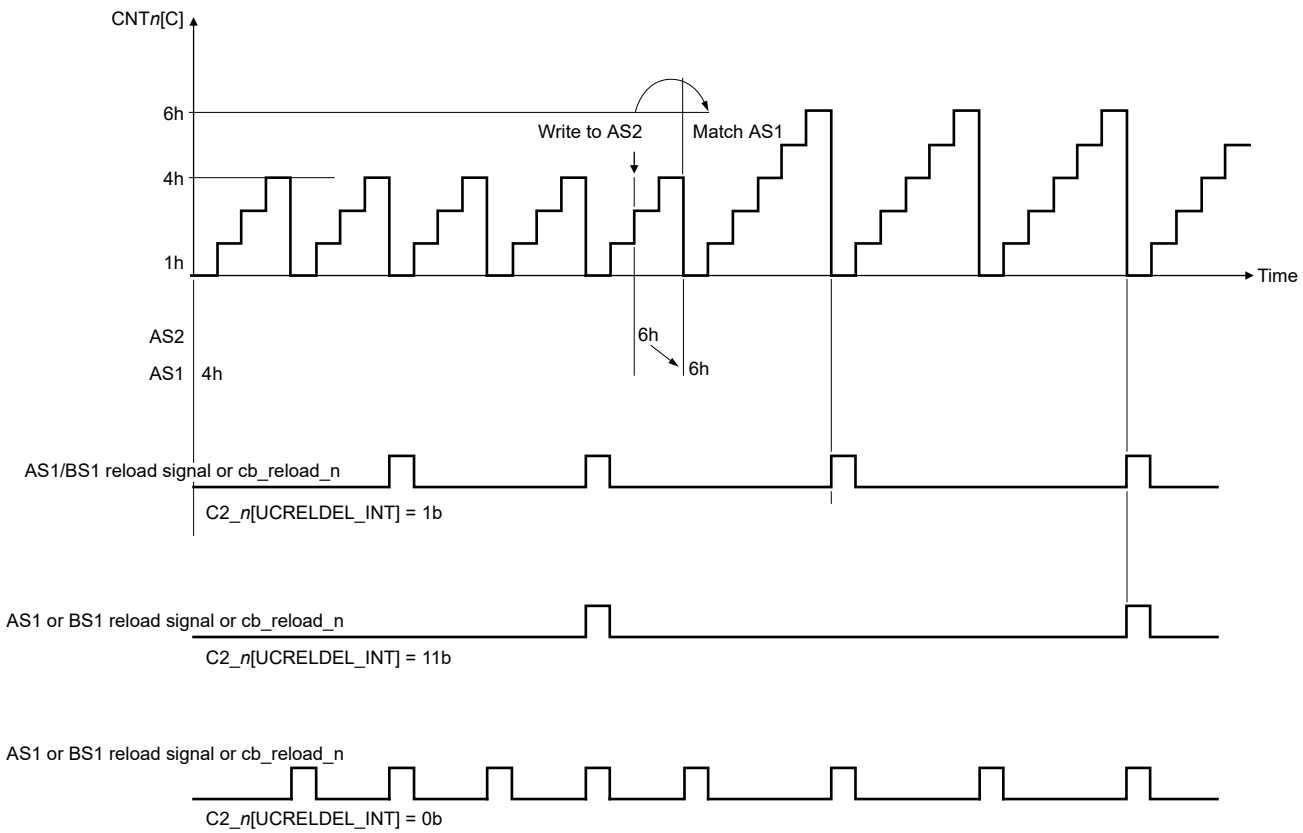


Figure 346. Timing example: Up Count mode with delayed reload

63.5.3.16 Output Pulse Width and Frequency Modulation Buffered (OPWFMB) mode

63.5.3.16.1 Overview

In this mode, the UC produces an output signal that has:

- A duty cycle determined by AS1.
- A period of BS1.

The double-buffered AS1 and BS1 registers produce smooth duty-cycle and period transitions when you change AS1 and BS1 during runtime.

See Table 420 for the values that you must write to C*n*[MODE] to enter this mode.

To provide smooth and consistent channel operation:

- AS1 and BS1 are both double-buffered to allow smooth signal generation when changing the register values.
- There is a delay from the AS1 match to the output pin transition (see AS1 and BS1 match timing).
- The internal counter ranges from 1h to BS1. When you write to BS1, you must write only values greater than 1h. Writing 0h or 1h leads to unpredictable results.

When a match on comparator A occurs, the UC drives C*n*[EDPOL] onto the output flip-flop. When a match on comparator B occurs, the UC drives the complement of C*n*[EDPOL] onto the output flip-flop. A BS1 match also causes the internal counter to transition to 1h, thus restarting the counter cycle.

The UC automatically selects the internal channel counter as the timebase when you select this mode. The mode supports duty cycles ranging from 0% to 100%.

63.5.3.16.2 OPWFMB submodes and MODE field values

Table 414. OPWFMB submodes and MODE field values

Match behavior	C*n*[MODE] (binary)
When BS1 matches the selected counter	101_1000
When AS1 or BS1 match the selected counter	101_1010

63.5.3.16.3 Avoid counter wrapping

When the UC enters OPWFMB mode, the output flip-flop acquires the value of C*n*[EDPOL]. If the UC transitions from GPIO to OPWFMB mode when the internal counter value is not within the range 1h–BS1, then the B match never occurs. Because the match never occurs, the internal counter wraps at the maximum counter value—FF_FFFFh for a 24-bit counter. When the counter wrap occurs, the counter returns to 1h and normal OPWFMB operation resumes. Therefore, to avoid this counter wrapping, ensure that the counter value is within the range 1h–BS1 before the UC enters OPWFMB mode.

63.5.3.16.4 AS1 and BS1 match timing

As illustrated in Timing example: AS1 and BS1 match, the output pin transition occurs when the AS1 or BS1 match signal deasserts, as indicated by the AS1 match negative-edge detection signal. If you write 4h to AS1, the output pin transitions 4 counter increments plus one system clock cycle after the counter period starts.

63.5.3.16.5 Timing example: AS1 and BS1 match

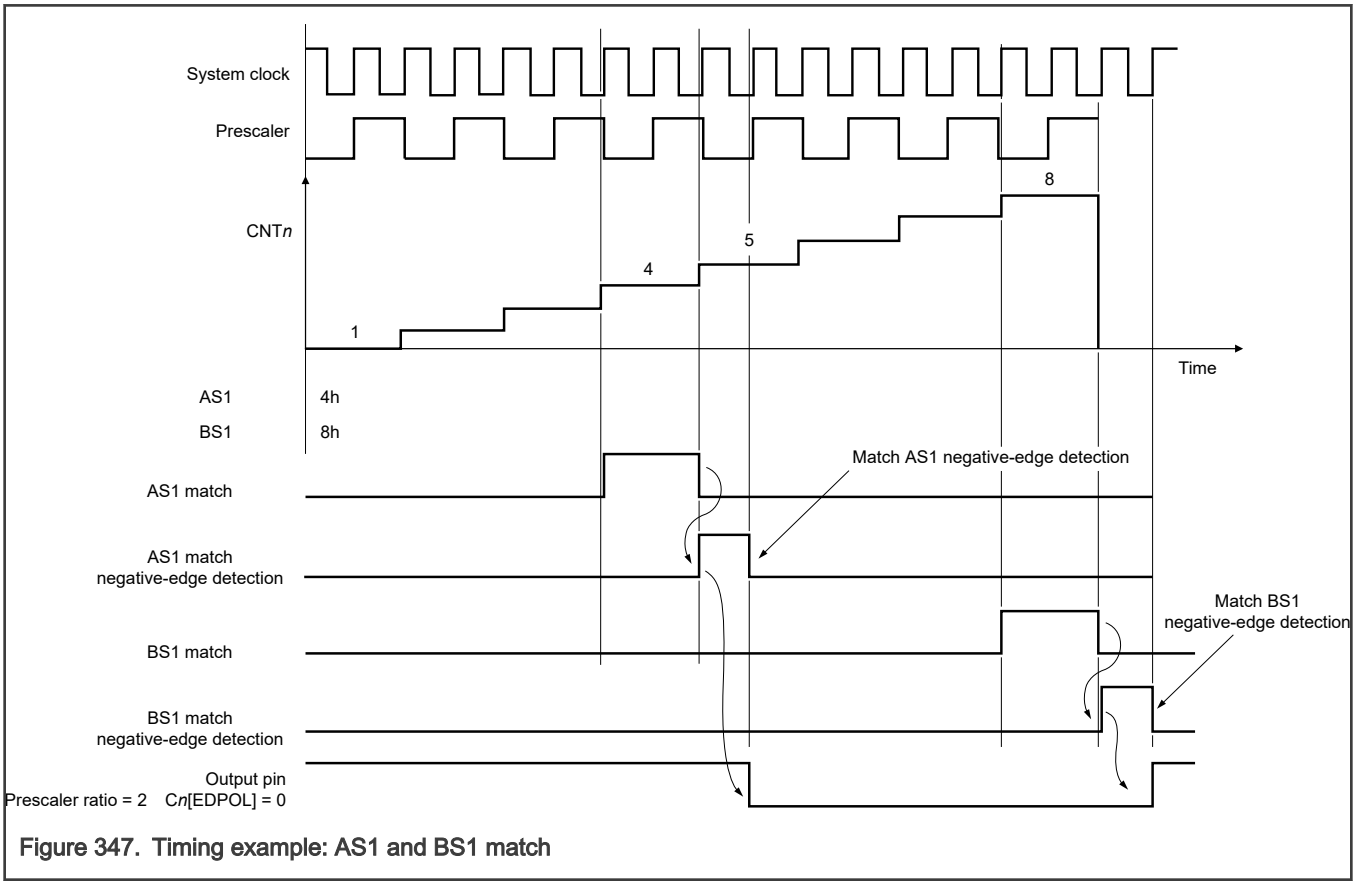


Figure 347. Timing example: AS1 and BS1 match

63.5.3.16.6 0% duty cycle

Timing example: AS1 = 0 illustrates the generated output signal when AS1 = 0, which results in a 0% duty cycle.

Because the internal counter never reaches zero in this situation, the UC infers a match as if AS1 = 1h. However, it is the positive edge of the match signal that triggers the output pin transition instead of the negative edge, which is what happens when AS1 = 1h. The AS1 positive-edge match signal from period *n*+1 occurs at the same time as the BS1 negative-edge match signal from period *n*. This timing allows the AS1 positive-edge match to mask the BS1 negative-edge match when they occur at the same time. As a result, no transition occurs on the output flip-flop and the UC generates a 0% duty cycle.

63.5.3.16.7 Timing example: AS1 = 0

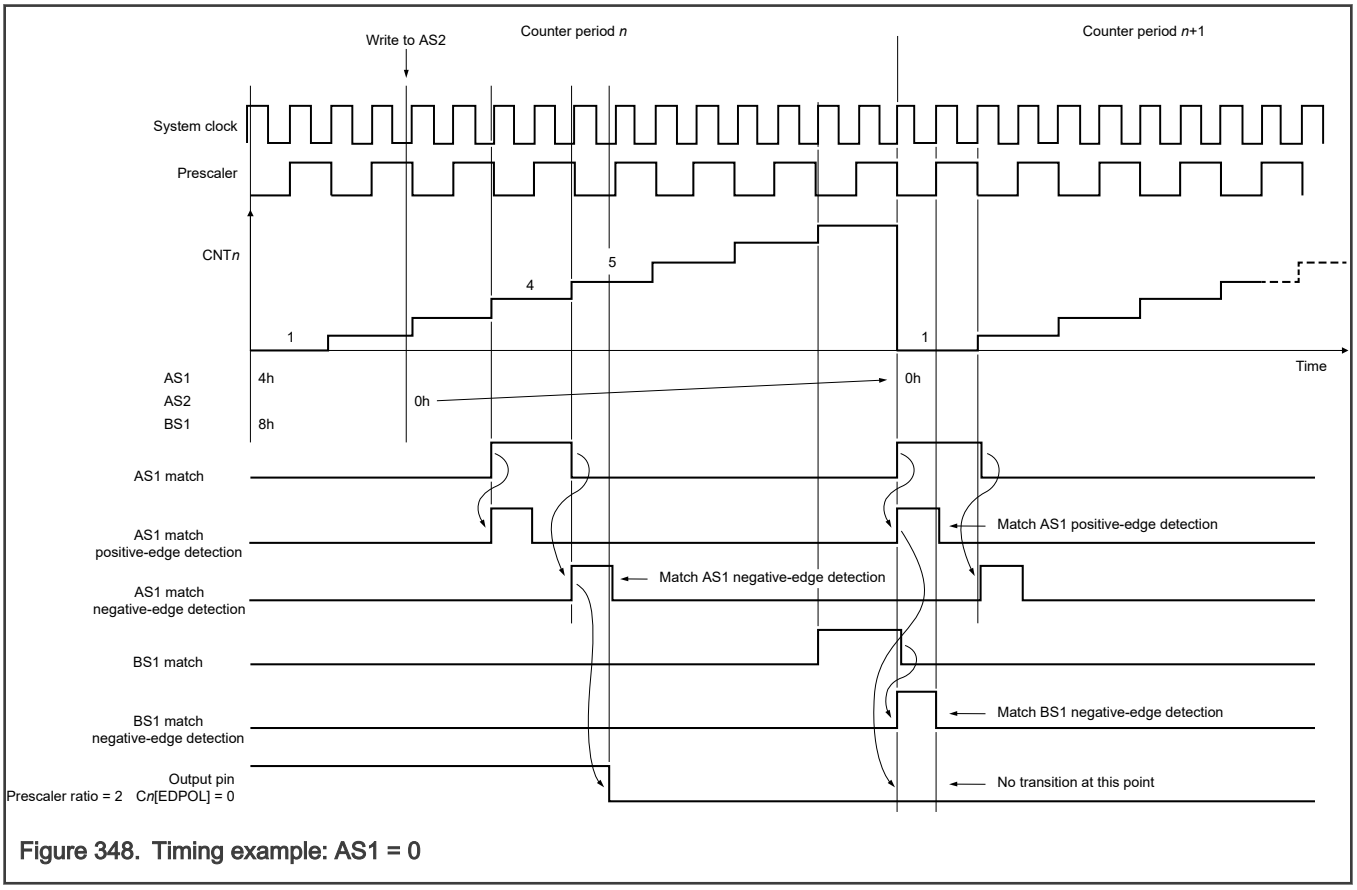


Figure 348. Timing example: AS1 = 0

63.5.3.16.8 Loading AS1 and BS1

As illustrated in Timing example: AS1 and BS1 update and FLAGs, AS1 and BS1 both use the same signal, generated at the last system clock cycle of a counter period. Therefore, eMIOS updates AS1 and BS1 with AS2 and BS2 values, respectively, at the same time that it changes the internal counter (CNT*n*[C]) to 1h. This event is the counter period boundary. The load signal pulse lasts for one system clock cycle. If you write AS2 and BS2 within counter period *n*, their values are available in AS1 and BS1, respectively, at the first clock cycle of counter period *n*+1. The UC then uses these new values for matches at counter period *n*+1. You can use Output Update Disable (OUDIS) to delay updating AS1 and BS1 for synchronization purposes.

63.5.3.16.9 Timing example: AS1 and BS1 update and FLAGs

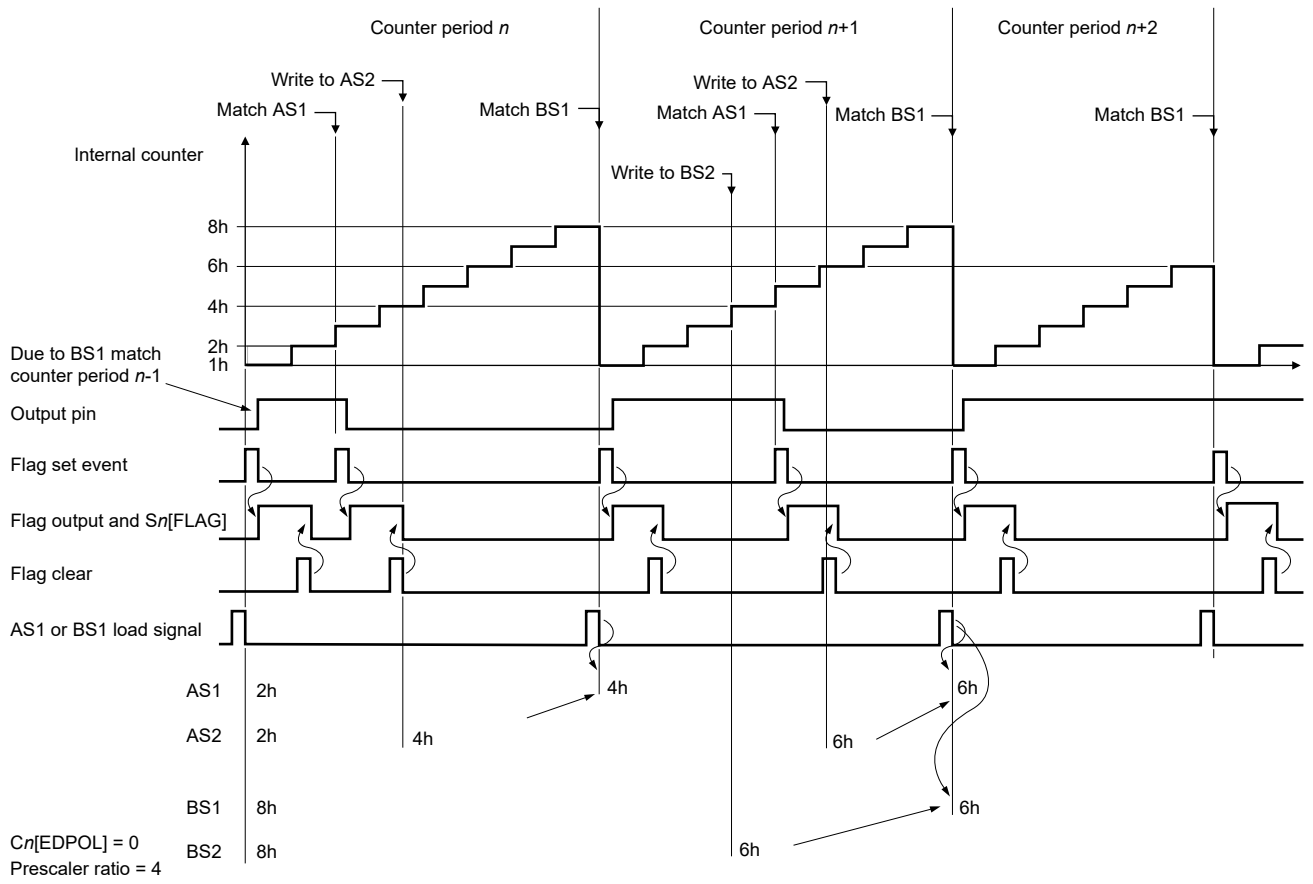


Figure 349. Timing example: AS1 and BS1 update and FLAGs

63.5.3.16.10 Flag generation

Timing example: AS1 and BS1 update and FLAGs assumes that both the channel and global prescalers are 1h (divide ratio of 2), causing the internal counter to transition every four system clock cycles. You can configure flags to be generated only on BS1 matches when bit 1 of C*n*[MODE] is 0, or on both AS1 and BS1 matches when that bit is 1. Because the BS1 flag occurs at the period boundary, you can use this flag to indicate that the AS2 or BS2 data written in period *n* has been loaded to AS1 or BS1, respectively, thereby generating matches in period *n*+1. Flag operation is synchronous, which means the flag sets one system clock cycle after the set-flag event.

63.5.3.16.11 Output disable

As illustrated in Timing example: active output disabled, the output-disable feature in OPWFMB mode forces the channel output flip-flop to the value of C*n*[EDPOL]. This functionality supports applications that use active-high signals and a high-to-low transition at an AS1 match. In this case, you must write 0 to C*n*[EDPOL]. You also must configure the internal counter to transition on every system clock cycle (write 0 to both GCP and CP).

The output-disable feature operates synchronously, meaning that the assertion of the Output Disable input signal causes the channel output flip-flop to transition to C*n*[EDPOL] at the next system clock cycle. If the Output Disable input pin deasserts, the output-signal transition occurs at the next AS1 or BS1 match.

Timing example: active output disabled assumes that the Output Disable input is enabled (using C*n*[ODIS]) and selected (using C*n*[ODISSL]). See UC Control n (C0 - C23) for a detailed description of the ODIS and ODISSL fields that enable and select the Output Disable inputs, respectively.

63.5.3.16.12 Timing example: active output disabled

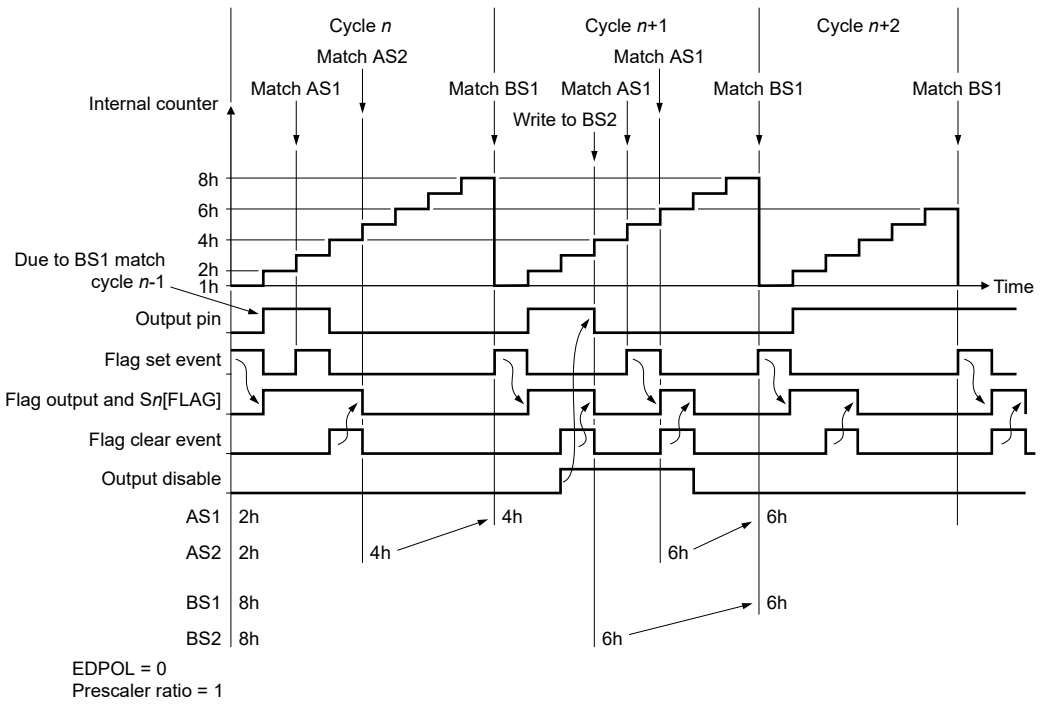


Figure 350. Timing example: active output disabled

63.5.3.16.13 Force match

Force Match A (C*n*[FORCMA]) and Force Match B (C*n*[FORCMB]) allow you to force the output flip-flop to the level corresponding to a match on comparators A or B, respectively. Similarly to a BS1 match, Force Match B changes the internal counter to 1h. The force-match operations do not set the input-capture flag (S*n*[FLAG]).

63.5.3.16.14 100% and 0% duty cycles

Timing example: from 100% to 0% duty cycle illustrates the generation of 100% and 0% duty-cycle signals. Initially, AS1 = 8h and BS1 = 8h. In this case, a BS1 match has precedence over an AS1 match—therefore, the output flip-flop acquires the complement of C*n*[EDPOL]. This configuration corresponds to a 100% duty-cycle signal. The same output signal can be generated for any AS1 value greater than or equal to BS1.

If AS1 is 0, the UC generates a 0% duty-cycle signal. In this case, the BS1 = 8h match from period 8 occurs at the same time as the AS1 = 0h match from period 9. See Timing example: AS1 = 0 for AS1 and BS1 match generation. In this case, an AS1 match has precedence over a BS1 match, and the output signal transitions to C*n*[EDPOL].

63.5.3.16.15 Timing example: from 100% to 0% duty cycle

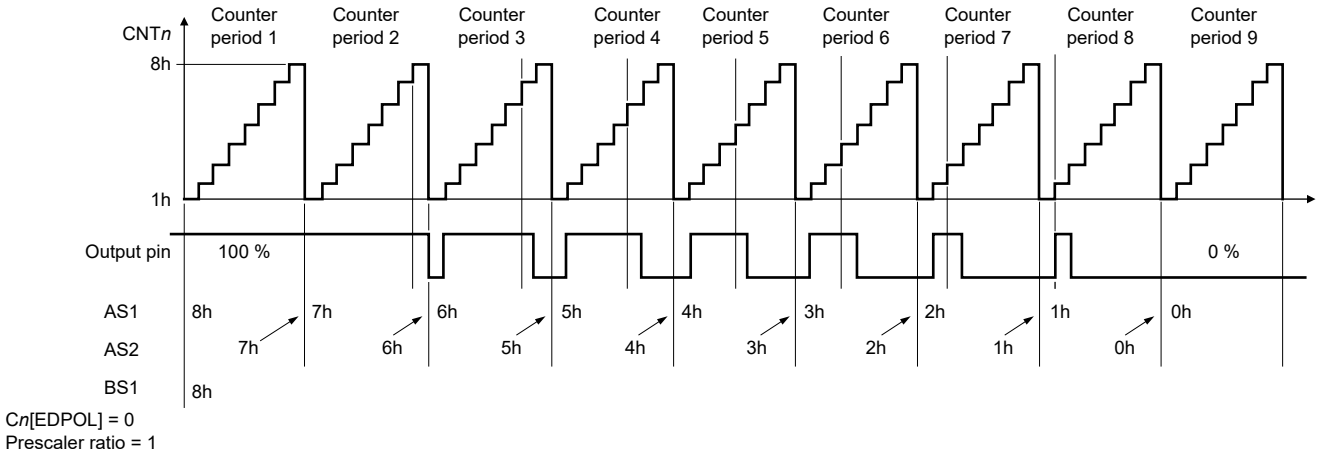


Figure 351. Timing example: from 100% to 0% duty cycle

63.5.3.17 Center Aligned Output PWM with Dead Time Insertion Buffered (OPWMCB) mode

63.5.3.17.1 Overview

In this mode, the UC generates a center-aligned PWM pulse with dead time insertion in the leading or trailing edge. See OPWMCB functions and MODE field values. AS1 and BS1 are double-buffered to allow smooth output signal generation when you change the AS2 or BS2 values.

When the UC enters OPWMCB mode from GPIO mode:

- The UC drives the complement of Edge Polarity (C*n*[EDPOL]) on the output flip-flop.
- Select the timebase using Bus Select (C*n*[BSL]). The timebase you select must be running in Up Count-Down Count mode. See Timing example: Up Count submode flag assertion. You must start the MCB timebase after the UC enters OPWMCB mode to avoid missing a match on the first duty cycle.
- Store the ideal duty cycle for the PWM signal in AS1 (write to A*n*[A]), which the UC compares with the selected timebase.
- Store the dead time value in BS1 (write to B*n*[B]), which the UC compares with the internal counter (CNT*n*[C]).

The internal counter may use the internal prescaler ratio, while the selected timebase may use a different prescaler ratio. The UC always resets the internal counter to 1h when an AS1 match occurs.

As in OPWMB and OPWFMB modes, the UC generates the match signal which compares the selected timebase with AS1 or BS1. When the UC deasserts the channel's combinational comparator output signal, it uses the match signal to assert or negate the channel's output flip-flop. See Timing example: AS1 and BS1 match for an illustration of the delay from match to output flip-flop transition in OPWFMB mode. The operation of OPWMCB mode is similar to OPWFMB mode with respect to match behavior and output pin transition.

See Entering OPWMCB mode.

63.5.3.17.2 OPWMCB functions and MODE field values

Table 415. OPWMCB functions and mode field values

Function	C*n*[MODE] (binary)
Insert trailing edge dead time with duty cycle determined by BS1 + AS1	101_11p0[1]

Table continues on the next page...

Table 415. OPWMCB functions and mode field values (continued)

Function	C*n*[MODE] (binary)
Insert leading edge dead time with duty cycle determined by BS1 – AS1	101_11p1[1]
Assert the input capture flag (Sn[FLAG]) on the trailing edge of the output PWM signal	101_110p[1]
Assert the input capture flag (Sn[FLAG]) on both edges of the output PWM signal	101_111p[1]

1. *p* = Adjust parameters for submodes and options. See Unified channels (UC).

63.5.3.17.3 Entering OPWMCB mode

To enter OPWMCB mode (this procedure assumes the channel is initially in GPIO mode):

1. Disable the GCP (write 0 to MCR[GPREN]).
2. For the timebase (MCB) channel:
   a. Disable the CP (write 0 to C*n*[UCPREN]).
   b. Initialize the internal counter (write 1h to CNT*n*[C]).
   c. Initialize the A data (write appropriate value to A*n*[A]).
   d. Configure the UC for MCB Up Count-Down Count mode (write 101_01*xy*b to C*n*[MODE], where *x* = 0 for flag set on match start or *x* = 1 for flag set at period boundary, and *y* = 0 for internal clock source or *y* = 1 for external clock source).
   e. Select the CP ratio (C*n*[UCPRE]).
   f. Enable the CP (write 1 to C*n*[UCPREN]).
3. For the OPWMCB channel:
   a. Disable the CP (write 0 to C*n*[UCPREN]).
   b. Initialize the A data (write appropriate value to A*n*[A]).
   c. Initialize the B data (write appropriate value to B*n*[B]).
   d. Select the timebase input (C*n*[BSL]).
   e. Configure the UC for MCB Up Count-Down Count mode (write 101_01*xy*b to C*n*[MODE], where *x* = 0 for flag set on match start or *x* = 1 for flag set at period boundary, and *y* = 0 for internal clock source or *y* = 1 for external clock source).
   f. Select the CP ratio (C*n*[UCPRE]).
   g. Enable the CP (write 1 to C*n*[UCPREN]).
4. Enable the GPC (write 1 to MCR[GPREN]).

63.5.3.17.4 Timing example: AS1 and BS1 load

When the selected counter bus transitions from 2h to 1h:

- The UC loads AS1 and BS1 as shown in the following figure.
- This event defines the counter period boundary.

The UC loads the values you write to AS2 or BS2 within period *n* into AS1 or BS1, respectively, and uses them to generate matches in period *n*+1.

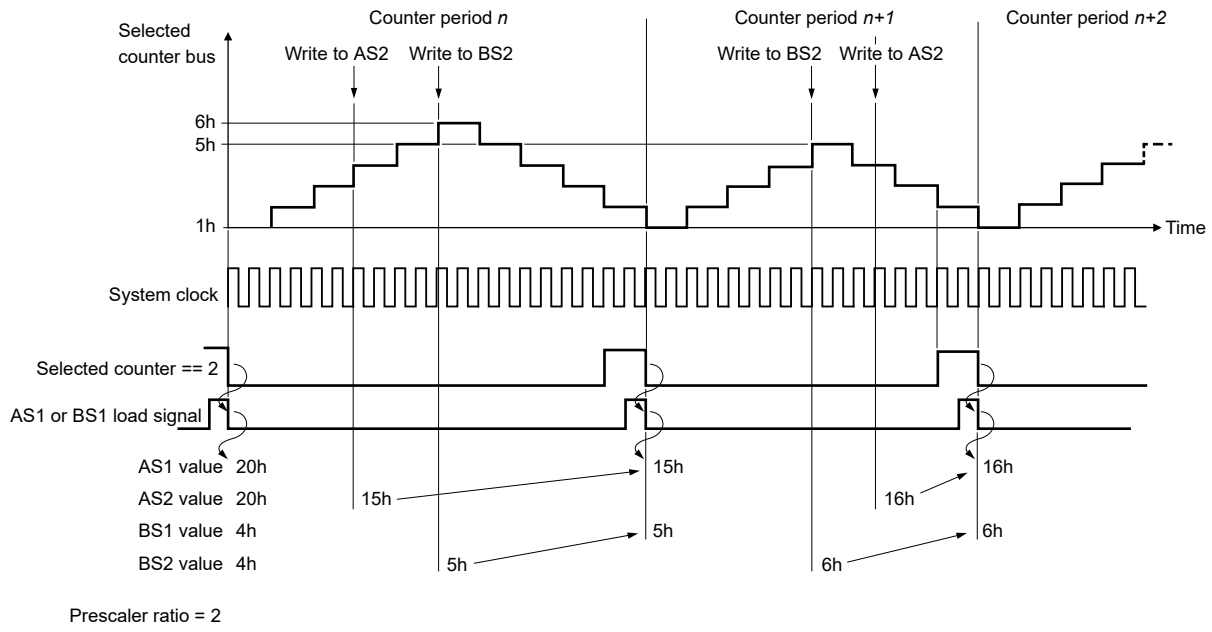


Figure 352. Timing example: AS1 and BS1 load

63.5.3.17.5 Operation sequence with leading edge dead time insertion

When operating with leading edge dead time insertion:

1. When the first AS1 match occurs, the UC writes 1h to the internal counter.
2. When a match between BS1 and the internal timebase occurs, the UC drives the value of Edge Polarity (C*n*[EDPOL]) on the output flip-flop.
3. When the next match between AS1 and the selected timebase occurs, the UC drives the complement of Edge Polarity (C*n*[EDPOL]) on the output flip-flop.

The UC repeats the above sequence continuously.

You must not allow the internal counter to reach 0h as the result of a rollover. To avoid this, you must not write a value greater than

2 x ((external Count Up limit) – A*n*[A])

into B*n*[B].

See Timing example: leading dead time insertion.

63.5.3.17.6 Timing example: leading dead time insertion

This example illustrates two cycles of a center-aligned PWM signal. AS1 and BS1 values change within the same period, which allows you to change the duty cycle and dead time values at the same time.

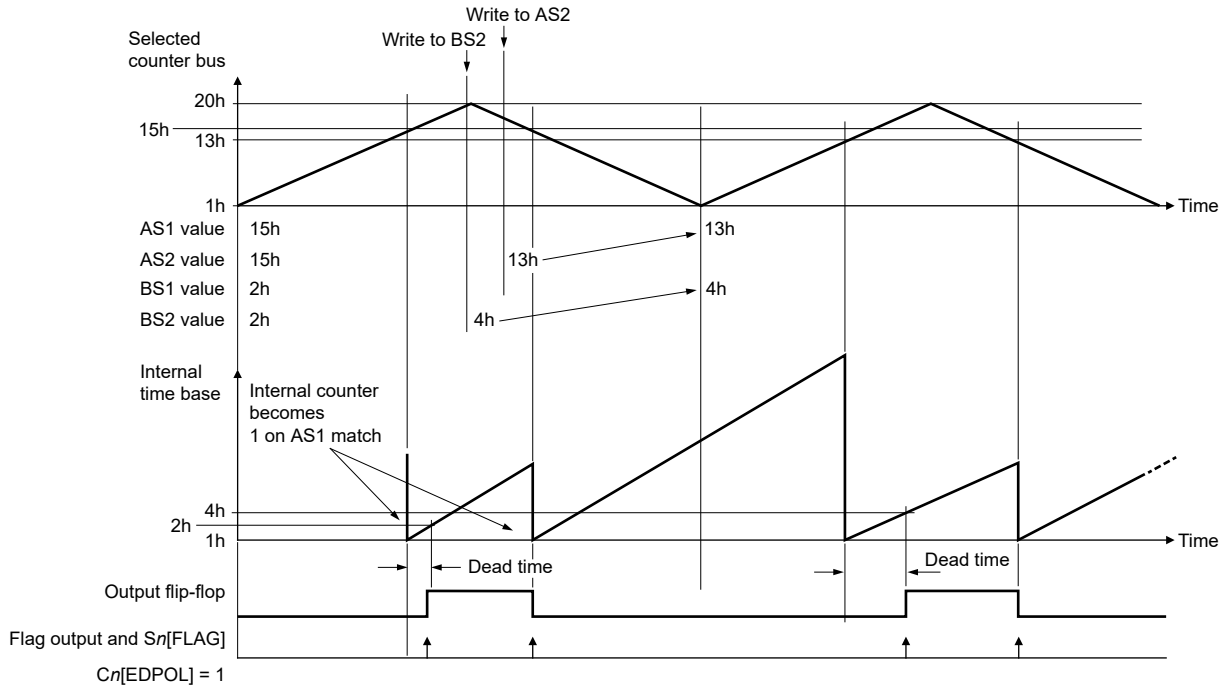


Figure 353. Timing example: leading dead time insertion

63.5.3.17.7 Operation sequence with trailing edge dead time insertion

When operating with trailing edge dead time insertion:

1. When the first match between AS1 and the selected timebase occurs, the UC:
   a. Drives the value of Edge Polarity (C*n*[EDPOL]) on the output flip-flop.
   b. Writes the internal counter (CNT*n*[C]) to 1h.
2. When the second match between AS1 and the selected timebase occurs, the UC:
   a. Writes the internal counter to 1h.
   b. Enables BS1 matches.
3. When a match between BS1 and the selected timebase occurs, the UC drives the complement of Edge Polarity (C*n*[EDPOL]) on the output flip-flop.

The UC repeats this sequence continuously.

63.5.3.17.8 Timing example: trailing dead time insertion

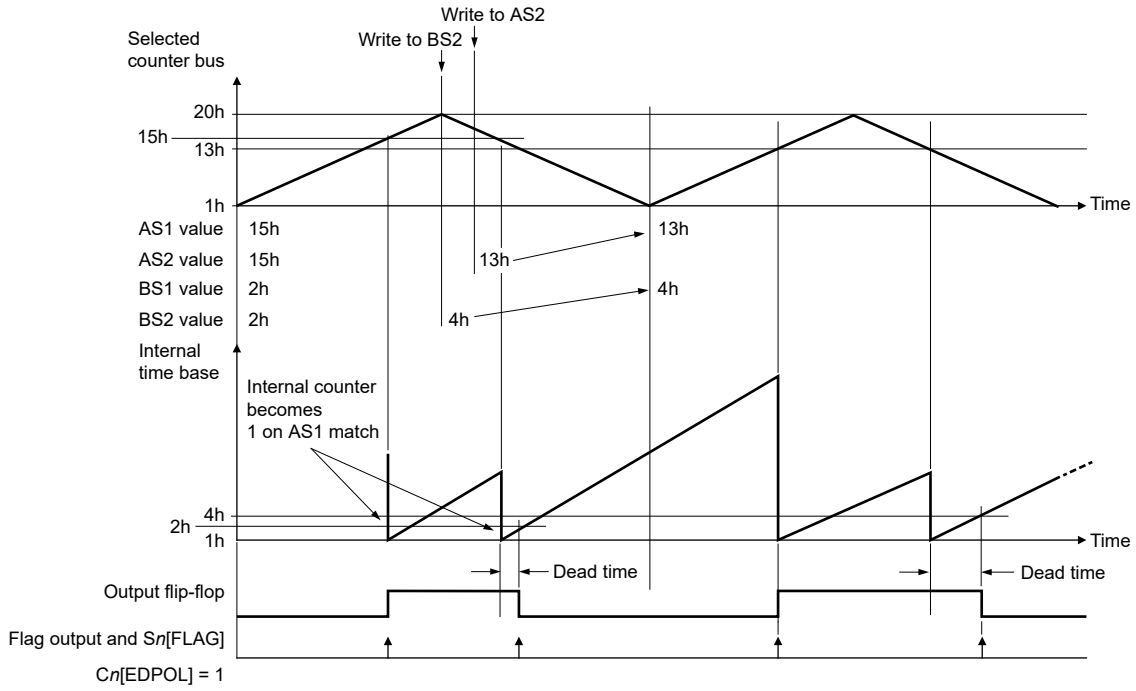


Figure 354. Timing example: trailing dead time insertion

63.5.3.17.9 Force match

Force Match A (C*n*[FORCMA]) and Force Match B (C*n*[FORCMB]) allow you to force the output flip-flop to the level corresponding to a match on comparators A or B, respectively. If subsequent matches occur on comparators A and B, the UC continues to generate PWM pulses regardless of the input capture flag (S*n*[FLAG]). The force-match operations do not set the input-capture flag (S*n*[FLAG]).

When you select leading edge dead time insertion:

- Force Match A drives the complement of Edge Polarity (C*n*[EDPOL]) on the output flip-flop.
- Force Match B drives the value of Edge Polarity (C*n*[EDPOL]) on the output flip-flop.

When you select trailing edge dead time insertion:

- Force Match A drives the value of Edge Polarity (C*n*[EDPOL]) on the output flip-flop.
- Force Match B drives the complement of Edge Polarity (C*n*[EDPOL]) on the output flip-flop.

Force Match operation does not write 1h to the internal timebase.

Force Match operation does not set the input-capture flag.

When you assert Force Match A and Force Match B at the same time, the UC drives the complement of Edge Polarity (C*n*[EDPOL]) on the output flip-flop. This is the equivalent of saying that Force Match A has precedence over Force Match B when you select leading edge dead time insertion, and Force Match B has precedence over Force Match A when you select trailing edge dead time insertion.

Force Match A and Force Match B have the same output transition behavior in both Freeze state and unfrozen state.

63.5.3.17.10 Duty cycle

You generate a duty cycle from 0% to 100% by writing appropriate values to AS1 and BS1 relative to the period of the external timebase.

To generate a 100% duty cycle waveform:

- Select trailing edge dead time insertion (see OPWMCB functions and MODE field values).
- Write 1h to Edge Polarity (C*n*[EDPOL]).
- The BS1 match must occur on or after the counter period boundary (external counter = 1).

You can also generate a 100% duty cycle waveform by writing 1h to AS1. However, if you do this before the UC enters OPWMCB mode, a 100% duty cycle may not occur in the first PWM period depending upon the pin state at mode entry.

To generate a 0% duty cycle waveform:

- AS1 must be greater than the maximum value of the selected counter period.
- The pin must start the current period with the complement value of Edge Polarity (C*n*[EDPOL]). If starting at 100% duty cycle, you can change Edge Polarity (C*n*[EDPOL]) to its complement by forcing the pin (see Force match).

You must write only non-zero values to AS1. If you write the maximum value of the external counter given by

(external counter period)/2

to AS1, the UC constantly drives the value of Edge Polarity (C*n*[EDPOL]) on the output flip-flop.

UC logic prevents matches from one period propagating to the next period. In trailing edge dead time insertion, the UC masks out BS1 matches that occur late in period *n*, so matches in period *n*+1 are unaffected and the output flip-flop does not transition.

63.5.3.17.11 Timing example: duty cycle

To generate a 100% duty cycle output waveform, write 4h to AS1 and 3h to BS1, as illustrated in this diagram. In this case, the trailing edge is at the boundary of counter period *n*+1, which actually belongs to period *n*+2 and therefore does not cause the output flip-flip to transition.

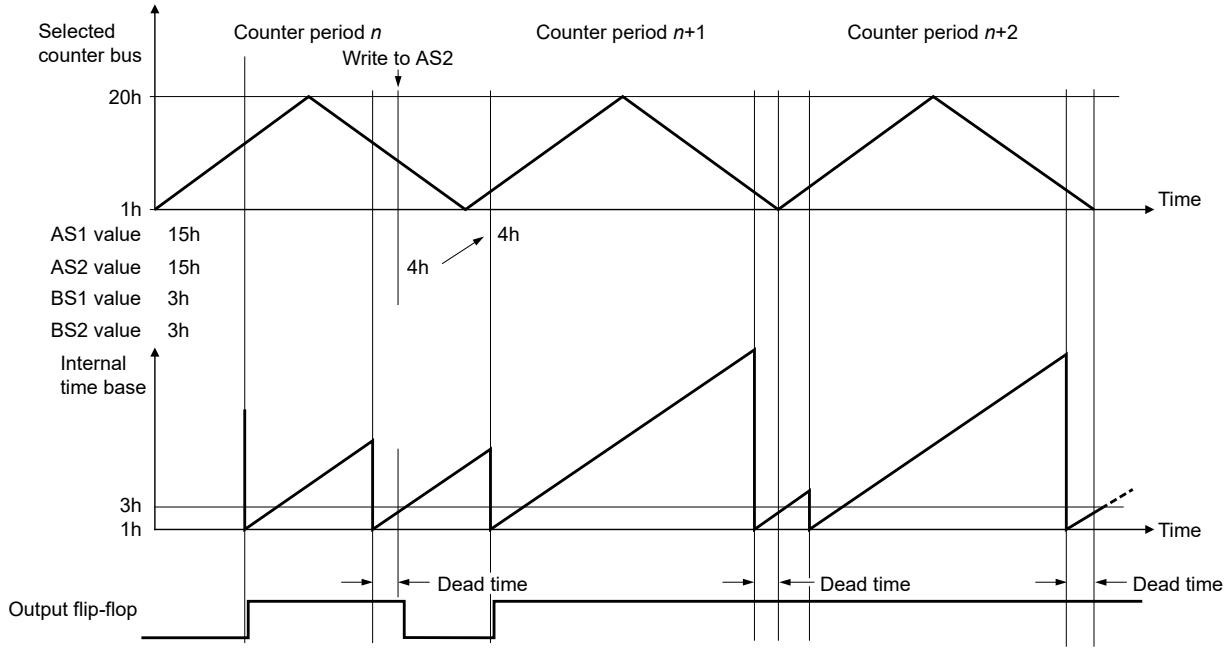


Figure 355. Timing example: duty cycle

63.5.3.17.12 Output disable

You can use Output Update Disable (write 0 to OUDIS[OU*n*]) to disable AS1 and BS1 updates. This allows you to update AS1 and BS1 in the same counter period and synchronize the loading of these registers with the loading of AS1 or BS1 in other channels.

Asserting Output Update Disable drives the complement of Edge Polarity (C*n*[EDPOL]) on output flip-flop. The internal channel matches continue to occur, and therefore the UC continues to set flags.

When you deassert Output Update Disable, the output flip-flop is again controlled by AS1 and BS1 matches. The UC transitions the output flip-flop only on system clock edges.

63.5.3.18 Output PWM Buffered (OPWMB) mode

63.5.3.18.1 Overview

In this mode, the UC produces PWM pulses with programmable leading and trailing edge placement. You must select an external counter driven in MCB Up mode from one of the counter buses. AS1 and BS1 define the first and second edges, respectively. C*n*[EDPOL] defines the output signal polarity. If C*n*[EDPOL] = 0, a negative edge occurs when AS1 matches the selected counter bus and a positive edge occurs when BS1 matches the selected counter bus.

See OPWMB submodes and MODE field values for the values that you must write to C*n*[MODE] to enter this mode.

AS1 and BS1 are double-buffered and updated from AS2 and BS2, respectively, at the cycle boundary. The load operation is similar to the one that occurs in OPWFMB mode. See Timing example: AS1 and BS1 update and FLAGs for more information about AS1 and BS1 updates.

When the UC enters OPWMB mode, the UC drives C*n*[EDPOL] on the output flip-flop.

These rules apply to OPWMB mode:

- If the AS1 and BS1 matches occur at the same time within the same counter period, the BS1 match has precedence over the AS1 match.
- An AS1 = 0 match in period *n* has precedence over a BS1 match in period *n*-1.
- The UC masks out AS1 matches if they occur after a BS1 match within the same period.
- The UC loads any value that you write to AS2 or BS2 in period *n* into AS1 and BS1 at the following period boundary (assuming that the corresponding OU*n* field of Output Update Disable (OUDIS) is 0). Therefore, the UC uses the new values for AS1 and BS1 matches in period *n*+1.

Timing example: 100% to 0% duty cycle illustrates a waveform changing from 100% to 0% duty cycle. In this example, BS1 is programmed to the same value as the period of the external selected timebase. If you write a value smaller than 8h to BS1, you cannot achieve a 0% duty cycle by only changing AS1. Because BS1 matches have precedence over AS1 matches, the flag output transitions to the complement of C*n*[EDPOL] at a BS1 match. For another example, if you write 9h to BS1, a BS1 match does not occur, and the UC generates a 0% duty-cycle signal.

63.5.3.18.2 OPWMB submodes and MODE field values

Table 416. OPWMB submodes and MODE field values

When FLAG generation occurs	C*n*[MODE] (binary)
At a BS1 match	110_0000
At an AS1 or BS1 match	110_0010

63.5.3.18.3 Force match

Force Match A (C*n*[FORCMA]) and Force Match B (C*n*[FORCMB]) allow you to force the output flip-flop to the level corresponding to a match on comparators A or B, respectively. If subsequent matches occur on comparators A and B, the UC continues to generate PWM pulses regardless of the input capture flag (S*n*[FLAG]). The force-match operations do not set the input-capture flag (S*n*[FLAG]).

63.5.3.18.4 Matches and flag operation

Timing example: matches and flags illustrates the operation of OPWMB mode with an emphasis on AS1 and BS1 matches and the transition of the channel output pin. The output pin transitions are based on the negative edges of the AS1 and BS1 match signals. AS1 becomes zero in period *n*+1. In this case, the UC uses the match positive edge instead of the negative edge to transition the output flip-flop.

63.5.3.18.5 Timing example: matches and flags

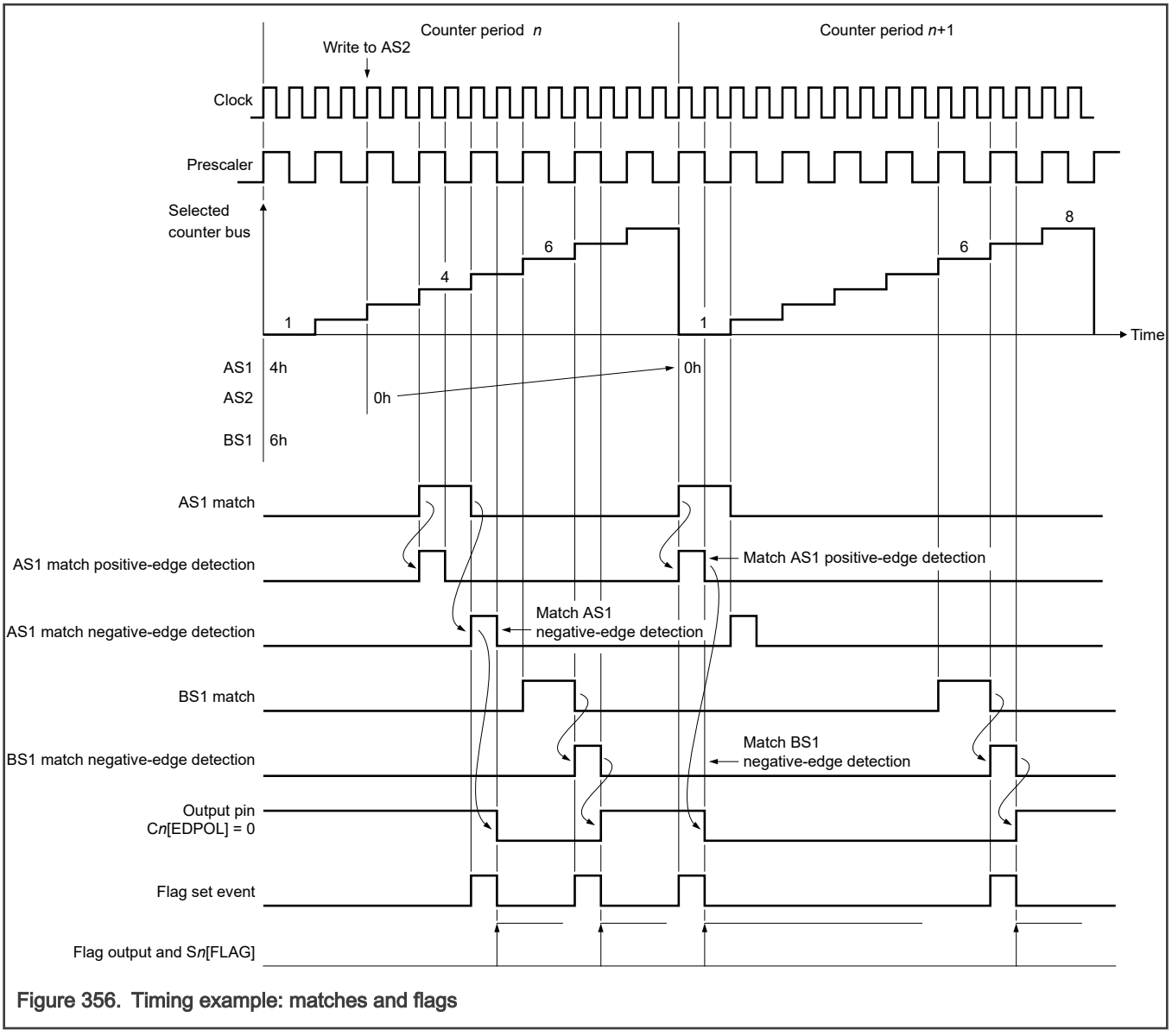


Figure 356. Timing example: matches and flags

63.5.3.18.6 0% duty cycle

Timing example: 0% duty cycle illustrates the channel operation for a 0% duty cycle. The AS1 match positive-edge signal occurs at the same time as the BS1 = 8h negative-edge signal. In this case, the AS1 match has precedence over the BS1 match, causing the output pin to remain at the value of C*n*[EDPOL] and generating a 0% duty-cycle signal.

63.5.3.18.7 Timing example: 0% duty cycle

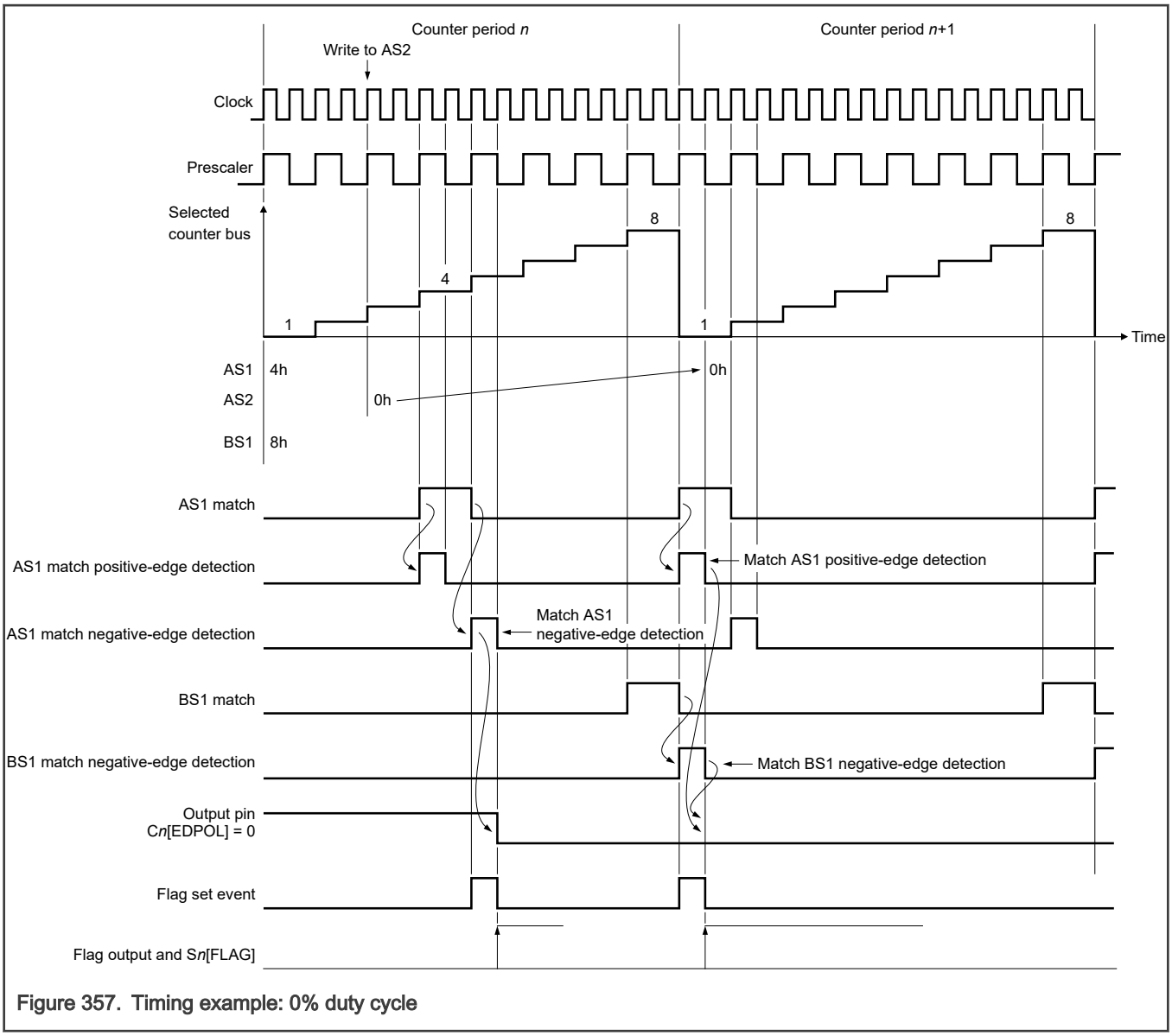


Figure 357. Timing example: 0% duty cycle

63.5.3.18.8 Active output disabled

Timing example: active output disabled illustrates the operation of the OPWMB mode with an emphasis on the assertion of the output-disable signal. The output disable forces a transition in the output pin to the value of C*n*[EDPOL]. The deassertion of the output-disable signal allows the output pin to transition at the following AS1 or BS1 match. The output disable does not modify the behavior of S*n*[FLAG]. A delay of one system clock cycle exists between the assertion of the output-disable signal and the transition of the output pin to C*n*[EDPOL].

63.5.3.18.9 Timing example: active output disabled

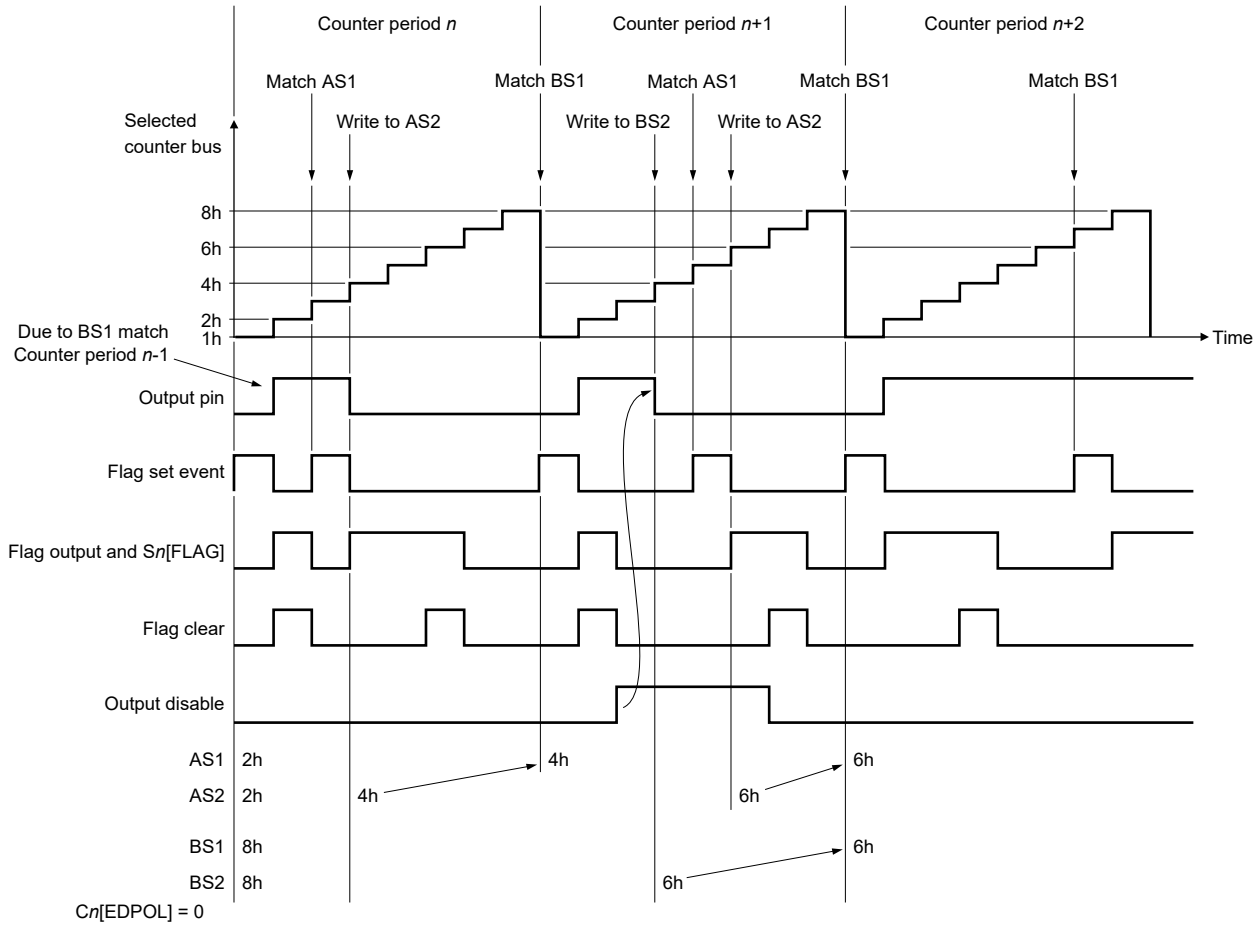


Figure 358. Timing example: active output disabled

63.5.3.18.10 Changing from 100% to 0% duty cycle

Timing example: 100% to 0% duty cycle illustrates a waveform changing from 100% to 0% duty cycle. In this example, BS1 has the same value as the period of the external selected timebase. If you write a value smaller than 8h to BS1, you cannot achieve a 0% duty cycle by only changing AS1. Because BS1 matches have precedence over AS1 matches, the output pin transitions to the complement of C*n*[EDPOL] at a BS1 match. For another example, if you write 9h to BS1, a BS1 match does not occur, and the UC generates a 0% duty-cycle signal.

63.5.3.18.11 Timing example: 100% to 0% duty cycle

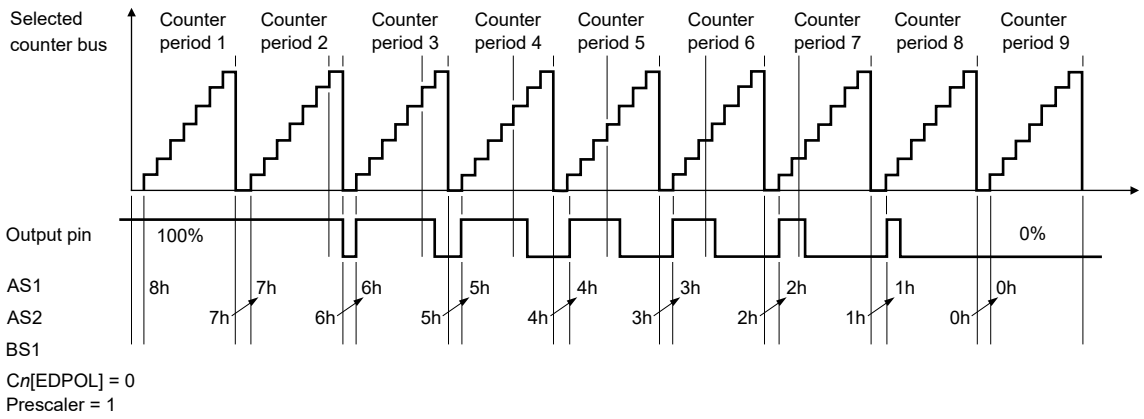


Figure 359. Timing example: 100% to 0% duty cycle

63.5.3.19 Output PWM with Trigger (OPWMT) mode

63.5.3.19.1 Overview

In this mode, the UC generates PWM pulses where the period does not change while the UC outputs the signal, but the duty cycle changes. When the duty cycle changes, it must not create glitches. The mode supports multiple UCs executing in the same mode and sharing a common timebase. Each UC can have a fixed PWM leading-edge position with respect to the other UCs. It can also generate a trigger signal at any point in the period. eMIOS can output this trigger signal to initiate activity in other parts of the chip—for example, to start ADC conversions.

You must select an external counter, driven in either MC Up or MCB Up mode (see Modulus Counter Buffered (MCB) mode), from one of the counter buses.

AS1 in OPWMT mode, AS2 in OPWMT mode, and BS1 and BS2 in OPWMT mode describe the configuration and behavior of the individual shadow registers in this mode.

To account for the shift in the AS1-defined leading edge of the waveform, the BS1-defined trailing edge may roll over into the next period. This means that a match against BS1 does not have to be qualified by a match in AS1. This also means that if you incorrectly write a value less than AS1 to BS1, the output persists over a period boundary until the UC encounters the BS1 value.

This mode provides a buffered update of the trailing edge by updating BS1 with BS2 contents only at a match of AS1.

The positive edges of the AS1, BS1, and AS2 match signals drive the output pin and flag transitions. See Timing example: matches and flags for details on positive-edge matches.

The UC sets the input capture flag (S*n*[FLAG]) only at an AS2 match. An AS1, BS1, or BS2 match has no effect on the flag.

When the UC enters OPWMT mode, it drives the complement of C*n*[EDPOL] on the output flip-flop.

Timing example: OPWMT illustrates the UC running in OPWMT mode with trigger-event generation and duty-cycle update after the next-period update.

63.5.3.19.2 Timing example: OPWMT

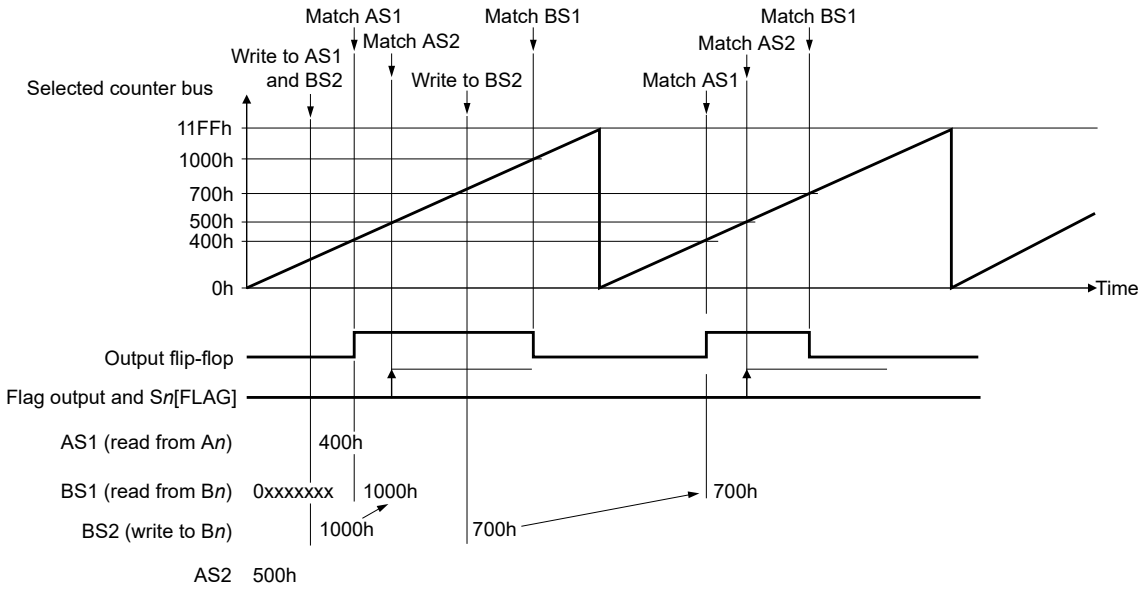


Figure 360. Timing example: OPWMT

63.5.3.19.3 AS1 in OPWMT mode

AS1 defines the leading edge of the PWM pulse output, and therefore the beginning of the PWM period. Using this fact, you can ensure that, when using a shared timebase, the leading edges of multiple UCs in OPWMT mode occur at specific times with respect to the other UCs. You can also introduce a fixed offset for each UC, which is particularly useful in the generation of lighting control signals. For this application, to reduce or eliminate noise generation, you can shift the PWM pulse of each UC with respect to the selected timebase by adjusting the AS1 value for each channel. The value you write to AS1 must be within the range of the selected timebase. Shadow registers (AS1, AS2, BS1, and BS2) loaded with 0h do not produce matches if the channel driving the timebase is in MCB mode.

AS1 is not buffered, because the shift of a PWM channel must not change while the UC generates the PWM signal. If you modify AS1, it updates immediately and one PWM pulse could be lost.

The UC compares AS1 to the selected timebase. When a match on AS1 occurs, the UC drives C*n*[EDPOL] on the output flip-flop. When a match occurs on comparator B, the UC drives the complement of C*n*[EDPOL] on the output flip-flop.

63.5.3.19.4 AS2 in OPWMT mode

AS2 defines the generation of a trigger event within the PWM period. The value you write to AS2 must be within the range of the selected timebase—otherwise the UC does not generate a trigger. A match on the comparator asserts the input-capture flag, but the match has no effect on the PWM output signal generation. The typical setup to obtain a trigger with a flag is to enable DMA and drive the DMA acknowledgement or DMA completion input high.

AS2 is not buffered, and therefore the UC updates it immediately. If the UC is running when you make a change, this could cause either of the following:

- The loss of one trigger event
- The generation of two trigger events within the same period

You access AS2 by reading or writing ALTA*n*[ALTA].

63.5.3.19.5 BS1 and BS2 in OPWMT mode

You access BS1 and BS2 using B*n*[B]:

- When you read B*n*[B], you read BS1.
- When you write to B*n*[B], you write to BS2.

BS1 defines the trailing edge of the PWM pulse output, and therefore the duty cycle of the PWM signal. To synchronize the BS1 update with the PWM signal and to ensure a correct output pulse generation, the BS2→BS1 transfer occurs at every AS1 match. This behavior is the same as in OPWM mode with next-period update.

Output Update Disable (OUDIS) affects transfers between BS2 and BS1 only.

63.5.3.19.6 Force match

At any time, Force Match A (C*n*[FORCMA]) and Force Match B (C*n*[FORCMB]) allow you to force the output flip-flop to the level corresponding to a match on AS1 or BS1, respectively. Similarly to a BS1 match, Force Match B changes the internal counter to 1h. The force-match operations do not set the input-capture flag (S*n*[FLAG]). Any match resulting from writing 1 to FORCMA or FORCMB has priority over any simultaneous match regarding output pin transitions. BS2→BS1 transfer at an A match is not inhibited by writing 1 simultaneously to both FORCMA or FORCMB. If you write 1 to both FORCMA and FORCMB simultaneously, the output pin acquires the complement of C*n*[EDPOL], as if AS1 and BS1 had the same value. FORCMA assertion causes the UC to process the BS2→BS1 transfer as a regular A match comparison and not due to FORCMA assertion, regardless of FORCMB assertion. If subsequent matches occur on comparators AS1 and B, the UC continues to generate PWM pulses, regardless of the state of S*n*[FLAG].

63.5.3.19.7 Duty cycle

To achieve 0% duty cycle, you must write the same value to AS1 and BS2. When a simultaneous match on comparators A and B occurs, the UC drives the complement of C*n*[EDPOL] on the output flip-flop at every period.

To achieve 100% duty cycle, you must write a value to BS1 that is greater than the maximum value of the selected timebase. The maximum counter value for the timebase is FF_FFFEh for a 24-bit counter and FFFEh for a 16-bit counter. When a match on comparator AS1 occurs, the UC drives C*n*[EDPOL] on the output flip-flop at every period. The match at comparator A still triggers the BS2→BS1 transfer.

Timing example: 0% duty cycle illustrates the UC running in OPWMT mode with trigger-event generation and a 0% duty cycle.

Timing example: 100% duty cycle illustrates the UC running in OPWMT mode with trigger-event generation and a 100% duty cycle.

63.5.3.19.8 Timing example: 0% duty cycle

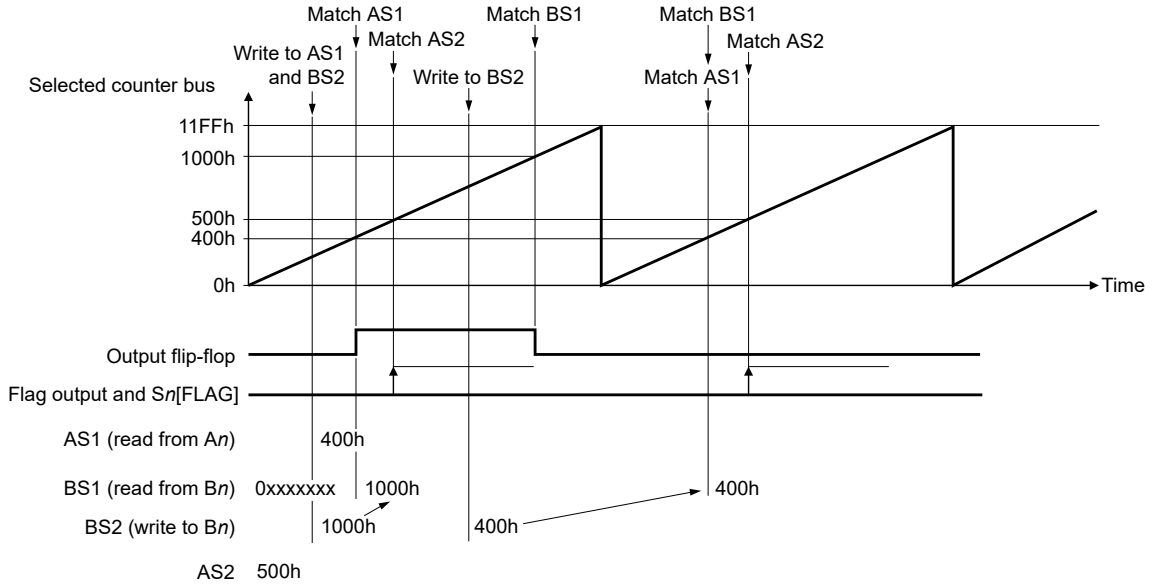


Figure 361. Timing example: 0% duty cycle

63.5.3.19.9 Timing example: 100% duty cycle

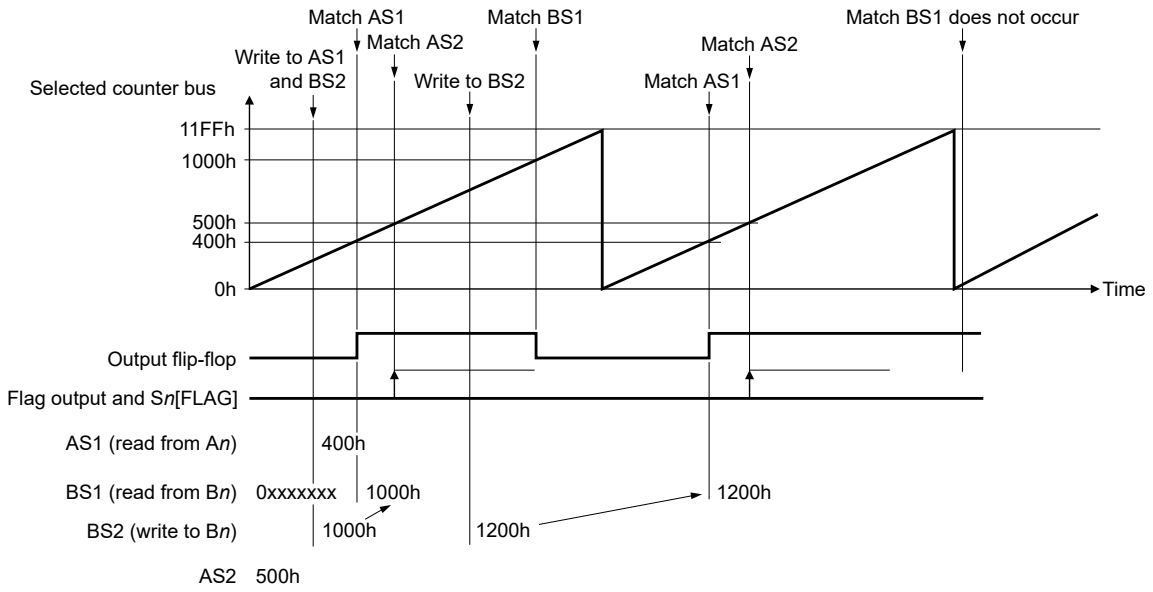


Figure 362. Timing example: 100% duty cycle

63.5.3.19.10 Output disable

As with other UC modes, OPWMT mode implements the output-disable function. When C*n*[ODIS] = 1, assertion of the output-disable input causes the UC to drive C*n*[EDPOL] on the output. Other than the fixed output, the UC continues to operate normally. When the output-disable signal deasserts, the UC returns to full normal operation.

63.5.3.20 Programmable input filter (PIF)

63.5.3.20.1 Overview

The PIF specifies the minimum input pulse width, in clock cycles, that can pass through the filter. It supports bypass operation (C*n*[IF]), with the input signal synchronized before arriving at the digital filter, or from 2 to 16 clock cycles in powers of 2.

63.5.3.20.2 Counter

The PIF is a 5-bit programmable up counter. The selected clock source increments the up counter after the number of clock cycles specified by the input filter (C*n*[IF]0.

The system clock synchronizes the input signal. When this signal changes state, the counter begins incrementing. As long as the new signal state is stable, the counter continues incrementing. If the counter overflows, eMIOS validates the new signal value. In this case, it transmits the value as a pulse edge to the edge detector. If the opposite edge appears on the signal before validation (overflow), eMIOS resets the counter. At the next transition, the counter starts incrementing again. A glitch does not occur on the edge detector for any pulse shorter than the full range of the masked counter.

Neither Freeze state nor deasserted GTBE disables the filter.

63.5.3.20.3 UC PIF logic

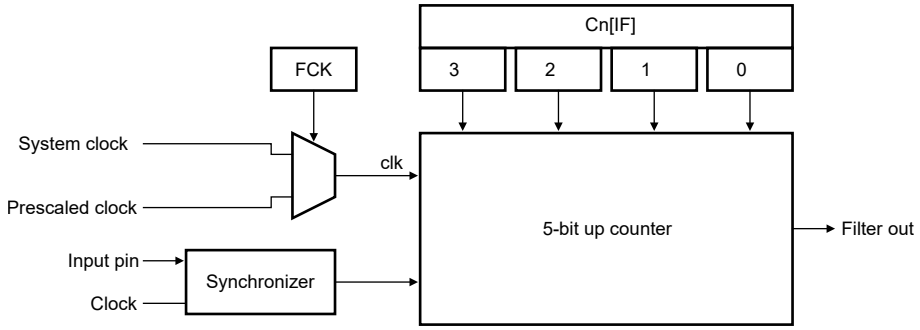


Figure 363. UC PIF logic

63.5.3.20.4 Timing example: UC PIF at 4 cycles

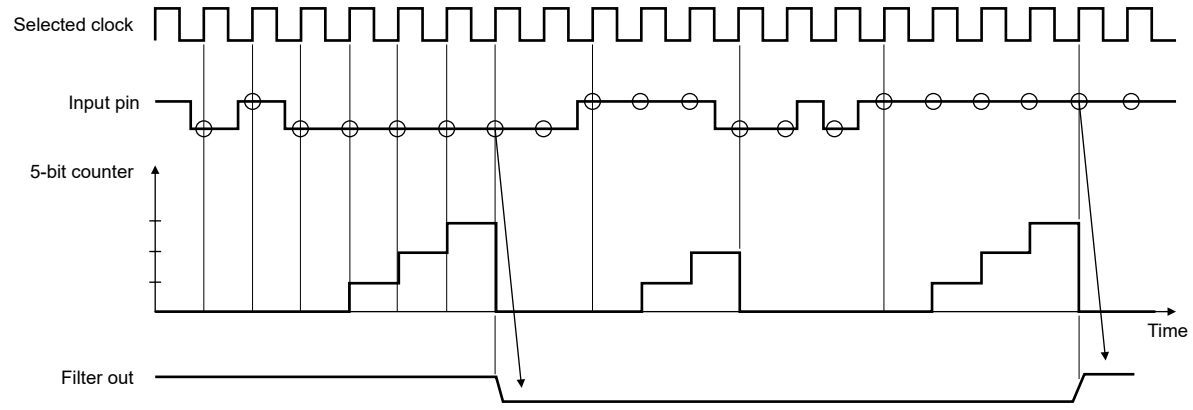


Figure 364. Timing example: UC PIF at 4 cycles

63.5.3.21 UC clock prescaler (CP)

eMIOS divides the GCP output signal by the UC CP (C*n*[UCPRE] + 1) to generate a clock enable for the UC internal counter. See Global clock prescaler (GCP). You enable the UC CP by writing 1 to C*n*[UCPREN]. If you disable the CP (write 0 to the same field), eMIOS stops the UC internal counter.

To ensure safe operation and avoid glitches, you must perform the following steps to change the UC CP:

1. Disable the GCP (write 0 to MCR[GPREN]).
2. Disable the UC CP (write 0 to C*n*[UCPREN]).
3. Specify the new prescaler value (write CP – 1 to C*n*[GPRE].
4. Enable the UC CP (write 1 to C*n*[UCPREN]).
5. Enable the GCP (write 1 to MCR[GPREN]).

63.5.4 Peripheral bus interface unit (BIU)

The BIU provides the interface between the IIB and the peripheral bus, which allows communication among all submodules and the IP interface. The BIU supports 8-, 16-, and 32-bit accesses performed over a 32-bit data bus in a single clock cycle.

63.5.5 Global clock prescaler (GCP)

eMIOS divides the system clock by the GCP (MCR[GPRE] + 1) and routes the resulting prescaled clock output to the channel CPs. You enable the GCP by writing 1 to MCR[GPREN]. If you disable the GCP (write 0 to the same field), eMIOS clears the prescaler counter and stops the prescaler clock.

To ensure safe operation and avoid glitches, you must perform the following steps to change the GCP:

1. Disable the GCP (write 0 to MCR[GPREN]).
2. Specify the new prescaler value (write GCP – 1 to MCR[GPRE].
3. Enable the GCP (write 1 to MCR[GPREN]).

63.5.6 Freeze and chip Debug mode

63.5.6.1 Enter and exit Freeze state

When the chip is in Debug mode (the chip Debug signal is asserted), you can freeze individual UCs. For the effects of Freeze state on eMIOS functions, see Freeze effects.

To put a channel in Freeze state, you must meet all of the following conditions:

- The chip must assert Debug.
- Prepare eMIOS for the Freeze state (write 1 to MCR[FRZ]).
- Enable Freeze for each channel you want to freeze:
  - Write 1 to C*n*[FREN].

A channel exits Freeze state when any of the following conditions occur:

- The chip deasserts Debug.
- You take eMIOS out of the Freeze state (write 0 to MCR[FRZ]).
- You disable Freeze for the channel:
  - write 0 to C*n*[FREN].

63.5.6.2 Freeze effects

Function	Effects of freeze
GCP	No effect
UC	• Halts counter, capture, and compare operation. • Freezes UC in its current state. • Makes all registers accessible. • In output modes, force match remains operational, allowing software to force the output to the desired level. • In input modes, the UC ignores input events. • When eMIOS exits Freeze state then UC operations resume, but they may be inconsistent until the UC reenters GPIO mode.
STAC	No effect
BIU	No effect

63.6 Initialization information

When initializing eMIOS, take the following considerations into account.

For eMIOS behavior on reset, see Reset.

Before changing the UC operating mode, you must:

1. Put the UC in GPIO mode (write 0b or 1b to C*n*[MODE]).
2. Write the correct values for the next operating mode to A*n* and B*n*.
3. Put the UC in the new operating mode (write the appropriate value to C*n*[MODE]).

If you change a UC from one mode to another without performing this procedure, the first operation cycle of the selected timebase can be random; that is, matches can occur in random time because you did not update A*n* and B*n* with correct values before the timebase matches the previous contents of either register.

When you enable interrupts, any interrupt service routine must clear the flag before exiting.

63.7 Application information

63.7.1 Overview

All output operation modes can generate correlated output signals. You can disable updates of the output signals for each UC (OUDIS).

To guarantee that incrementing of the internal counters of correlated channels occurs in the same clock cycle, you must configure the internal CPs before enabling the GCP. If you configure the internal CPs after enabling the GCP, the internal counters may increment in the same ratio, but at different clock cycles.

You should drive the Output Disable Input signals with the flag signals of some UCs running in SAIC mode. When an output disable condition happens, the interrupt service routine must process the output channels before processing the channels running SAIC. This sequence avoids glitches in the output pins.

63.7.2 Timebase generation

In the MC and OPWFM modes with internal clock sources, you can modify the internal counter rate by configuring the CP ratio. Timebase with the fastest prescaler ratio illustrates a timebase with a prescaler ratio of one.

NOTE

The buffered modes, MCB and OPWFMB, behave differently.

If the prescaler ratio is greater than one or you select an external clock, the internal counter behaves in one of four different ways depending on the channel mode:

- In MC mode with Clear on Match Start and External Clock: Timebase generation with clear on match start and external clock.
- In MC mode with Clear on Match Start and Internal Clock source: Timebase generation with clear on match start and internal clock.
- In MC mode with Clear on Match End: Timebase generation with clear on match end.
- In OPWFM mode: Timebase generation with clear on match start and internal clock. The internal counter clears at the start of the match signal, skips the next prescaled clock edge, and then increments in the subsequent prescaled clock edge.

63.7.3 Timebase with the fastest prescaler ratio

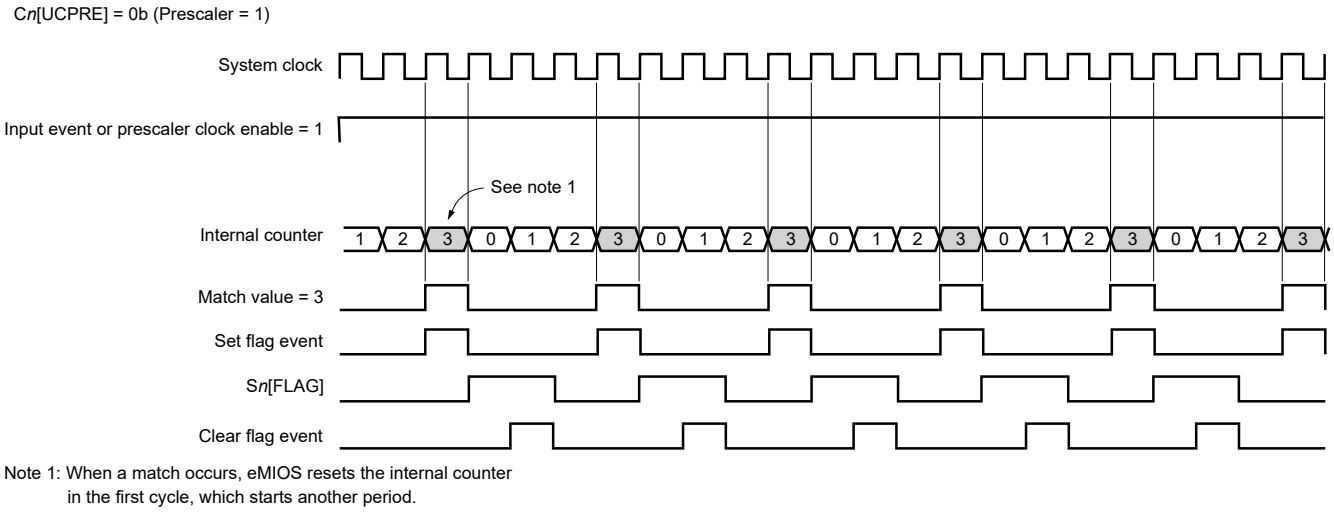
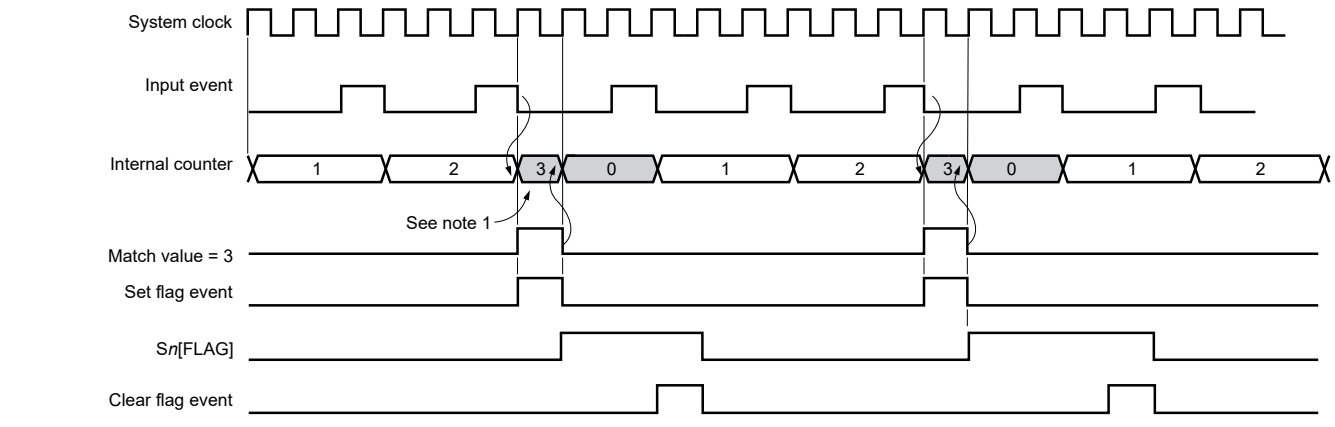


Figure 365. Timebase with fastest prescaler ratio

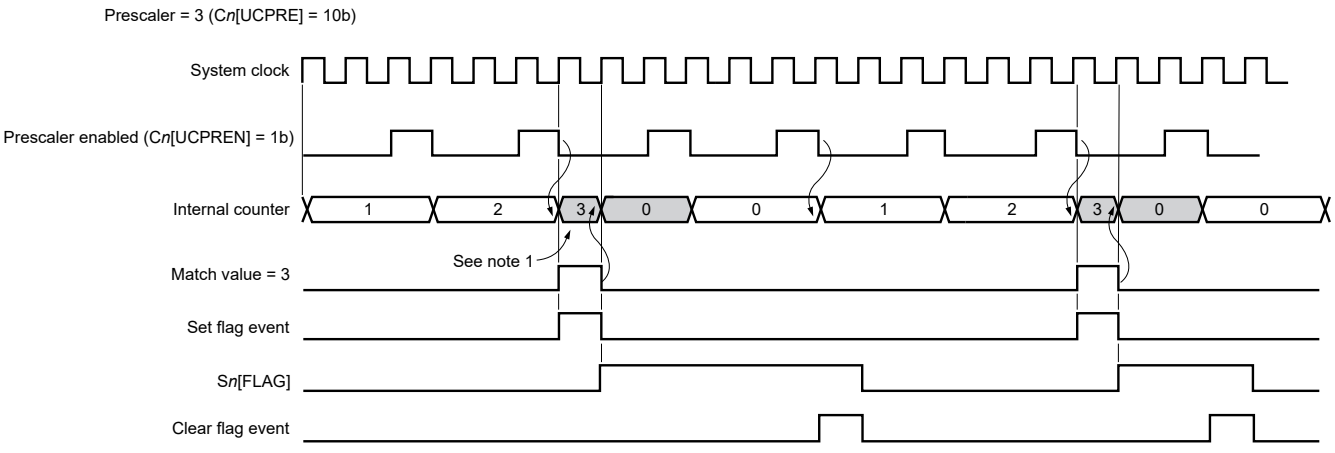
63.7.4 Timebase generation with clear on match start and external clock



Note 1: When a match occurs, eMIOS resets the internal counter in the first clock cycle.
The counter starts counting at the next edge of the prescaler clock enable.

Figure 366. Timebase generation with clear on match start and external clock

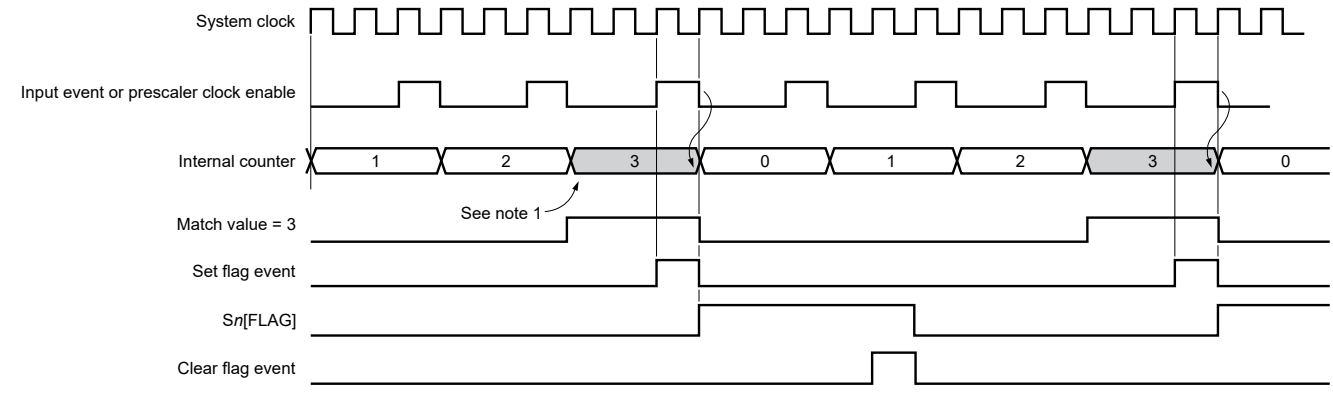
63.7.5 Timebase generation with clear on match start and internal clock



Note 1: When a match occurs, eMIOS clears the internal counter in the first clock cycle.
The counter starts counting again only after the second edge of the prescaled clock.

Figure 367. Timebase generation with clear on match start and internal clock

63.7.6 Timebase generation with clear on match end



Note 1: The match occurs only when the input event or prescaler clock enable is active.
When the match occurs, eMIOS resets the internal counter immediately.

Figure 368. Timebase generation with clear on match end

63.7.7 Coherent accesses

Reading A*n*[A] again in the same period as the last read of B*n*[B] results in incoherent results if the last read of B*n*[B] occurred after a disabled BS2→BS1 transfer.

63.7.8 Initializing channels and modes

Perform the following basic steps for output mode startup. This procedure assumes the channels are initially in GPIO mode:

1. Disable the GPC (write 0 to MCR[GPREN]).
2. For each timebase UC:
    a. Disable the CP (write 0 to C*n*[UCPREN]).
    b. Write an initial value to the internal counter (CNT*n*[C]).
    c. Write initial values to A*n*[A] and B*n*[B].
    d. Configure the UC for MC or MCB Up mode (write the appropriate value to C*n*[MODE]).
    e. Select the CP ratio (C*n*[UCPRE]).
    f. Enable the CP (write 1 to C*n*[UCPREN]).
3. For each output UC:
    a. Disable the CP (write 0 to C*n*[UCPREN]).
    b. Write initial values to A*n*[A] and B*n*[B].
    c. Select the timebase input (C*n*[BSL]).
    d. Specify the output mode (write the appropriate value to C*n*[MODE]).
    e. Select the same CP ratio as the timebase UC (C*n*[UCPRE]).
    f. Enable the CP (write 1 to C*n*[UCPREN]).
4. Enable the GPC (write 1 to MCR[GPREN]).
5. Enable GBTE (write 1 to MCR[GBTE]).

The timebase channel and the output channel may be the same for some applications such as in OPWFM or OPWFMB mode, or whenever the output channel drives the timebase itself.

You may configure flags at any time.

63.8 Memory maps and registers

63.8.1 Overall address map organization

All address locations not explicitly mentioned in this table are reserved. When you access an absent register, absent channel, or reserved address space, the transfer terminates with an error.

Table 417. Overall address map organization

Base address (hex)	Description
0	Module Configuration (MCR)
4	Global Flag (GFLAG)
8	Output Update Disable (OUDIS)
C	Disable Channel (UCDIS)
20	Channels 0–7
120	Channels 8–15
220	Channels 16–23

63.8.2 UC memory overview

Table 418. UC memory overview

Offset from UC *n* base address (hex)	Description
0	UC A n (A0 - A23)
4	UC B n (B0 - B23)
8	UC Counter n (CNT0 - CNT23)
C	UC Control n (C0 - C23)
10	UC Status n (S0 - S23)
14	Alternate Address n (ALTA0 - ALTA23)
18	UC Control 2 n (C2_0 - C2_23)
1C–1F	Reserved

63.8.3 AS1, AS2, BS1, and BS2 shadow registers

When you use eMIOS UC modes, you rely heavily on four shadow registers:

- AS1
- AS2
- BS1
- BS2

You use A*n*[A], B*n*[B], and ALTA*n*[ALTA] to access these shadow registers:

- Each instance of A*n*[A] has a separate associated instance of AS1 and AS2. In some UC modes, you also use ALTA*n*[ALTA] to access AS2.
- Each instance of B*n*[B] has a separate associated instance of BS1 and BS2.

See Relationship between A*n*, B*n*, ALTA*n*, and shadow registers.

Each UC mode uses the AS1-AS2 and BS1-BS2 shadow register pairs for various match and capture functions, as described in Unified channels (UC).

A reset clears all shadow registers.

Assignment of values for A*n*[A], B*n*[B], and ALTA*n*[ALTA] describes how you write to and read from the shadow registers for all operation modes. Values not listed are reserved. For more information, see Unified channels (UC).

63.8.4 Relationship between A*n*, B*n*, ALTA*n*, and shadow registers

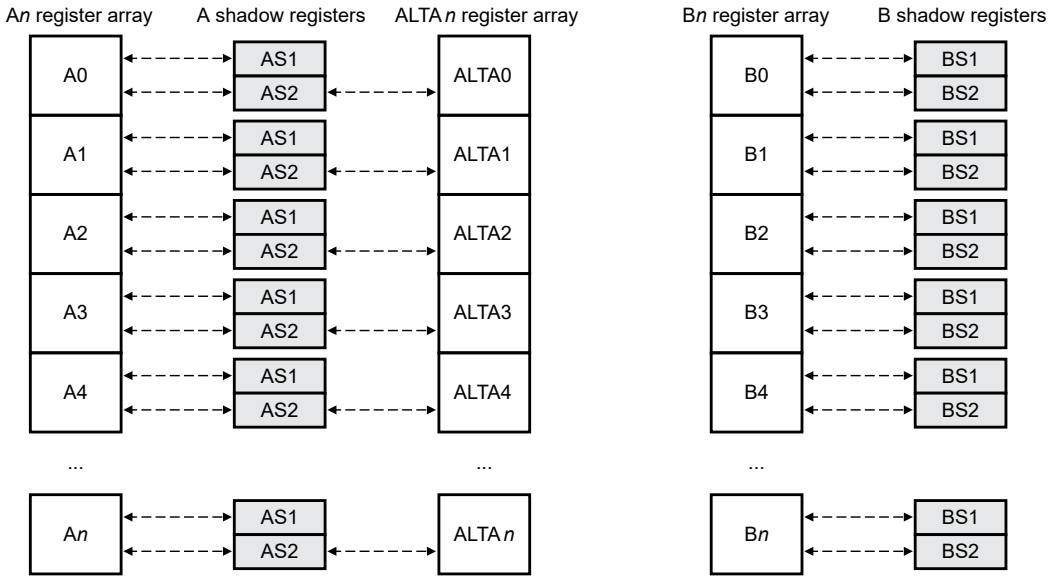


Figure 369. Relationship between A*n*, B*n*, ALTA*n*, and shadow registers

63.8.5 Assignment of values for A*n*[A], B*n*[B], and ALTA*n*[ALTA]

This table describes how you write to and read from the following registers for all operation modes:

- UC A n (A0 - A23)
- UC B n (B0 - B23)
- Alternate Address n (ALTA0 - ALTA23)

Values not listed are reserved. For more information, see Unified channels (UC).

Table 419. Assignment of values for A*n*[A], B*n*[B], and ALTA*n*[ALTA]

Operation mode	Register access					
	A*n*		B*n*		ALTA*n*	
	Write	Read	Write	Read	Write	Read
GPIO	AS1 and AS2[1]	AS1	BS1 and BS2[2]	BS1	AS2	AS2

Table continues on the next page...

Table 419. Assignment of values for A*n*[A], B*n*[B], and ALTA*n*[ALTA] (continued)

Operation mode	Register access					
	A*n*		B*n*		ALTA*n*	
	Write	Read	Write	Read	Write	Read
SAIC[3]	—	AS2	BS2	BS2	—	—
SAOC[3]	AS2	AS1	BS2	BS2	—	—
IPWM	—	AS2	—	BS1	—	—
IPM	—	AS2	—	BS1	—	—
DAOC	AS2	AS1	BS2	BS1	—	—
PEA	AS1	AS2	—	BS1	—	—
PEC[3]	AS1	AS1	BS1	BS1	—	AS2
WPTA	AS1	AS1	BS1	BS1	—	AS2
MC[3]	AS2	AS1	BS2	BS2	—	—
OPWFM	AS2	AS1	BS2	BS1	—	—
OPWMC	AS2	AS1	BS2	BS1	—	—
OPWM	AS2	AS1	BS2	BS1	—	—
OPWMT	AS1	AS1	BS2	BS1	AS2	AS2
MCB[3]	AS2	AS1	BS2	BS1	—	—
OPWFMB	AS2	AS1	BS2	BS1	—	—
OPWMCB	AS2	AS1	BS2	BS1	—	—
OPWMB	AS2	AS1	BS2	BS1	—	—

1. This operation writes the same value to AS1 and AS2.
2. This operation writes the same value to BS1 and BS2.
3. These modes do not require UC B n (B0 - B23), but you can still use it to access BS2.

63.8.6 eMIOS register descriptions

All control registers are 32 bits wide. Data fields are 24 bits wide. There are 24 UCs, as mapped in Counter buses, channels, and timebase sources.

63.8.6.1 eMIOS memory map

eMIOS_0 base address: 4008_8000h

eMIOS_1 base address: 4008_C000h

eMIOS_2 base address: 4009_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Module Configuration (MCR)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
4h	Global Flag (GFLAG)	32	R	0000_0000h
8h	Output Update Disable (OUDIS)	32	RW	0000_0000h
Ch	Disable Channel (UCDIS)	32	RW	0000_0000h
20h	UC A 0 (A0)	32	RW	0000_0000h
24h	UC B 0 (B0)	32	RW	0000_0000h
28h	UC Counter 0 (CNT0)	32	RW	0000_0000h
2Ch	UC Control 0 (C0)	32	RW	0000_0000h
30h	UC Status 0 (S0)	32	RW	See section
34h	Alternate Address 0 (ALTA0)	32	RW	0000_0000h
38h	UC Control 2 0 (C2_0)	32	RW	0000_0000h
40h	UC A 1 (A1)	32	RW	0000_0000h
44h	UC B 1 (B1)	32	RW	0000_0000h
48h	UC Counter 1 (CNT1)	32	RW	0000_0000h
4Ch	UC Control 1 (C1)	32	RW	0000_0000h
50h	UC Status 1 (S1)	32	RW	See section
54h	Alternate Address 1 (ALTA1)	32	RW	0000_0000h
58h	UC Control 2 1 (C2_1)	32	RW	0000_0000h
60h	UC A 2 (A2)	32	RW	0000_0000h
64h	UC B 2 (B2)	32	RW	0000_0000h
68h	UC Counter 2 (CNT2)	32	RW	0000_0000h
6Ch	UC Control 2 (C2)	32	RW	0000_0000h
70h	UC Status 2 (S2)	32	RW	See section
74h	Alternate Address 2 (ALTA2)	32	RW	0000_0000h
78h	UC Control 2 2 (C2_2)	32	RW	0000_0000h
80h	UC A 3 (A3)	32	RW	0000_0000h
84h	UC B 3 (B3)	32	RW	0000_0000h
88h	UC Counter 3 (CNT3)	32	RW	0000_0000h
8Ch	UC Control 3 (C3)	32	RW	0000_0000h
90h	UC Status 3 (S3)	32	RW	See section
94h	Alternate Address 3 (ALTA3)	32	RW	0000_0000h
98h	UC Control 2 3 (C2_3)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
A0h	UC A 4 (A4)	32	RW	0000_0000h
A4h	UC B 4 (B4)	32	RW	0000_0000h
A8h	UC Counter 4 (CNT4)	32	RW	0000_0000h
ACh	UC Control 4 (C4)	32	RW	0000_0000h
B0h	UC Status 4 (S4)	32	RW	See section
B4h	Alternate Address 4 (ALTA4)	32	RW	0000_0000h
B8h	UC Control 2 4 (C2_4)	32	RW	0000_0000h
C0h	UC A 5 (A5)	32	RW	0000_0000h
C4h	UC B 5 (B5)	32	RW	0000_0000h
C8h	UC Counter 5 (CNT5)	32	RW	0000_0000h
CCh	UC Control 5 (C5)	32	RW	0000_0000h
D0h	UC Status 5 (S5)	32	RW	See section
D4h	Alternate Address 5 (ALTA5)	32	RW	0000_0000h
D8h	UC Control 2 5 (C2_5)	32	RW	0000_0000h
E0h	UC A 6 (A6)	32	RW	0000_0000h
E4h	UC B 6 (B6)	32	RW	0000_0000h
E8h	UC Counter 6 (CNT6)	32	RW	0000_0000h
ECh	UC Control 6 (C6)	32	RW	0000_0000h
F0h	UC Status 6 (S6)	32	RW	See section
F4h	Alternate Address 6 (ALTA6)	32	RW	0000_0000h
F8h	UC Control 2 6 (C2_6)	32	RW	0000_0000h
100h	UC A 7 (A7)	32	RW	0000_0000h
104h	UC B 7 (B7)	32	RW	0000_0000h
108h	UC Counter 7 (CNT7)	32	RW	0000_0000h
10Ch	UC Control 7 (C7)	32	RW	0000_0000h
110h	UC Status 7 (S7)	32	RW	See section
114h	Alternate Address 7 (ALTA7)	32	RW	0000_0000h
118h	UC Control 2 7 (C2_7)	32	RW	0000_0000h
120h	UC A 8 (A8)	32	RW	0000_0000h
124h	UC B 8 (B8)	32	RW	0000_0000h
128h	UC Counter 8 (CNT8)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
12Ch	UC Control 8 (C8)	32	RW	0000_0000h
130h	UC Status 8 (S8)	32	RW	See section
134h	Alternate Address 8 (ALTA8)	32	RW	0000_0000h
138h	UC Control 2 8 (C2_8)	32	RW	0000_0000h
140h	UC A 9 (A9)	32	RW	0000_0000h
144h	UC B 9 (B9)	32	RW	0000_0000h
14Ch	UC Control 9 (C9)	32	RW	0000_0000h
150h	UC Status 9 (S9)	32	RW	See section
154h	Alternate Address 9 (ALTA9)	32	RW	0000_0000h
158h	UC Control 2 9 (C2_9)	32	RW	0000_0000h
160h	UC A 10 (A10)	32	RW	0000_0000h
164h	UC B 10 (B10)	32	RW	0000_0000h
16Ch	UC Control 10 (C10)	32	RW	0000_0000h
170h	UC Status 10 (S10)	32	RW	See section
174h	Alternate Address 10 (ALTA10)	32	RW	0000_0000h
178h	UC Control 2 10 (C2_10)	32	RW	0000_0000h
180h	UC A 11 (A11)	32	RW	0000_0000h
184h	UC B 11 (B11)	32	RW	0000_0000h
18Ch	UC Control 11 (C11)	32	RW	0000_0000h
190h	UC Status 11 (S11)	32	RW	See section
194h	Alternate Address 11 (ALTA11)	32	RW	0000_0000h
198h	UC Control 2 11 (C2_11)	32	RW	0000_0000h
1A0h	UC A 12 (A12)	32	RW	0000_0000h
1A4h	UC B 12 (B12)	32	RW	0000_0000h
1ACh	UC Control 12 (C12)	32	RW	0000_0000h
1B0h	UC Status 12 (S12)	32	RW	See section
1B4h	Alternate Address 12 (ALTA12)	32	RW	0000_0000h
1B8h	UC Control 2 12 (C2_12)	32	RW	0000_0000h
1C0h	UC A 13 (A13)	32	RW	0000_0000h
1C4h	UC B 13 (B13)	32	RW	0000_0000h
1CCh	UC Control 13 (C13)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1D0h	UC Status 13 (S13)	32	RW	See section
1D4h	Alternate Address 13 (ALTA13)	32	RW	0000_0000h
1D8h	UC Control 2 13 (C2_13)	32	RW	0000_0000h
1E0h	UC A 14 (A14)	32	RW	0000_0000h
1E4h	UC B 14 (B14)	32	RW	0000_0000h
1ECh	UC Control 14 (C14)	32	RW	0000_0000h
1F0h	UC Status 14 (S14)	32	RW	See section
1F4h	Alternate Address 14 (ALTA14)	32	RW	0000_0000h
1F8h	UC Control 2 14 (C2_14)	32	RW	0000_0000h
200h	UC A 15 (A15)	32	RW	0000_0000h
204h	UC B 15 (B15)	32	RW	0000_0000h
20Ch	UC Control 15 (C15)	32	RW	0000_0000h
210h	UC Status 15 (S15)	32	RW	See section
214h	Alternate Address 15 (ALTA15)	32	RW	0000_0000h
218h	UC Control 2 15 (C2_15)	32	RW	0000_0000h
220h	UC A 16 (A16)	32	RW	0000_0000h
224h	UC B 16 (B16)	32	RW	0000_0000h
228h	UC Counter 16 (CNT16)	32	RW	0000_0000h
22Ch	UC Control 16 (C16)	32	RW	0000_0000h
230h	UC Status 16 (S16)	32	RW	See section
234h	Alternate Address 16 (ALTA16)	32	RW	0000_0000h
238h	UC Control 2 16 (C2_16)	32	RW	0000_0000h
240h	UC A 17 (A17)	32	RW	0000_0000h
244h	UC B 17 (B17)	32	RW	0000_0000h
24Ch	UC Control 17 (C17)	32	RW	0000_0000h
250h	UC Status 17 (S17)	32	RW	See section
254h	Alternate Address 17 (ALTA17)	32	RW	0000_0000h
258h	UC Control 2 17 (C2_17)	32	RW	0000_0000h
260h	UC A 18 (A18)	32	RW	0000_0000h
264h	UC B 18 (B18)	32	RW	0000_0000h
26Ch	UC Control 18 (C18)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
270h	UC Status 18 (S18)	32	RW	See section
274h	Alternate Address 18 (ALTA18)	32	RW	0000_0000h
278h	UC Control 2 18 (C2_18)	32	RW	0000_0000h
280h	UC A 19 (A19)	32	RW	0000_0000h
284h	UC B 19 (B19)	32	RW	0000_0000h
28Ch	UC Control 19 (C19)	32	RW	0000_0000h
290h	UC Status 19 (S19)	32	RW	See section
294h	Alternate Address 19 (ALTA19)	32	RW	0000_0000h
298h	UC Control 2 19 (C2_19)	32	RW	0000_0000h
2A0h	UC A 20 (A20)	32	RW	0000_0000h
2A4h	UC B 20 (B20)	32	RW	0000_0000h
2ACh	UC Control 20 (C20)	32	RW	0000_0000h
2B0h	UC Status 20 (S20)	32	RW	See section
2B4h	Alternate Address 20 (ALTA20)	32	RW	0000_0000h
2B8h	UC Control 2 20 (C2_20)	32	RW	0000_0000h
2C0h	UC A 21 (A21)	32	RW	0000_0000h
2C4h	UC B 21 (B21)	32	RW	0000_0000h
2CCh	UC Control 21 (C21)	32	RW	0000_0000h
2D0h	UC Status 21 (S21)	32	RW	See section
2D4h	Alternate Address 21 (ALTA21)	32	RW	0000_0000h
2D8h	UC Control 2 21 (C2_21)	32	RW	0000_0000h
2E0h	UC A 22 (A22)	32	RW	0000_0000h
2E4h	UC B 22 (B22)	32	RW	0000_0000h
2E8h	UC Counter 22 (CNT22)	32	RW	0000_0000h
2ECh	UC Control 22 (C22)	32	RW	0000_0000h
2F0h	UC Status 22 (S22)	32	RW	See section
2F4h	Alternate Address 22 (ALTA22)	32	RW	0000_0000h
2F8h	UC Control 2 22 (C2_22)	32	RW	0000_0000h
300h	UC A 23 (A23)	32	RW	0000_0000h
304h	UC B 23 (B23)	32	RW	0000_0000h
308h	UC Counter 23 (CNT23)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
30Ch	UC Control 23 (C23)	32	RW	0000_0000h
310h	UC Status 23 (S23)	32	RW	See section
314h	Alternate Address 23 (ALTA23)	32	RW	0000_0000h
318h	UC Control 2 23 (C2_23)	32	RW	0000_0000h

63.8.6.2 Module Configuration (MCR)

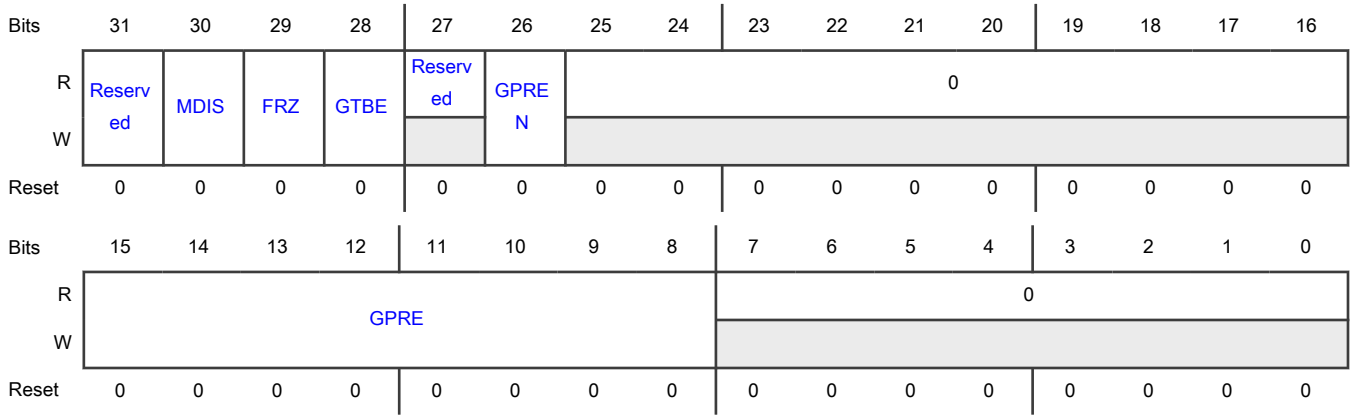
Offset

Register	Offset
MCR	0h

Function

Configures eMIOS operation.

Diagram



Fields

Field	Function
31 —	Reserved Write only the reset value to this field.
30 MDIS	Module Disable Disables eMIOS by stopping the clock.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When you disable eMIOS, its clock stops, it consumes less power, and you can access only the following registers: • Module Configuration (MCR) • Output Update Disable (OUDIS) • Disable Channel (UCDIS) 0b - Enable 1b - Disable
29 FRZ	Freeze Prepares eMIOS to enter Freeze state (see Freeze and chip Debug mode). eMIOS does not enter Freeze state until both of the following conditions occur: • The chip asserts Debug mode. • You enable Freeze for each UC (write 1 to C*n*[FREN]). A channel remains frozen until one or more of the following occur: • You force eMIOS to exit the Freeze state (write 0 to FRZ). • The chip deasserts Debug mode. • You disable Freeze for that UC (write 0 to C*n*[FREN]). 0b - Exit Freeze state 1b - Enter Freeze state
28 GTBE	Global Timebase Enable Asserts the GTBE_OUT signal. If GTBE_OUT is connected to the GTBE_IN input of one or more eMIOS instances, asserting the signal turns on the internal counters of those eMIOS instances simultaneously. **NOTE** The GTBE_IN input enables (asserted) or disables (deasserted) the internal counters. 0b - Deassert GTBE_OUT 1b - Assert GTBE_OUT
27 —	Reserved
26 GPREN	Global Prescaler Enable Enables the global prescaler counter. Disabling the global prescaler clears the prescaler counter and stops the prescaler clock. 0b - Disable 1b - Enable
25-16	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
15-8 GPRE	Global Prescaler Specifies the global clock prescaler. Write the desired clock divide ratio minus 1. For example, for a divide ratio of 8, write 111b. The prescaler can range from 1 (0b) to 256 (11111111b).
7-0 —	Reserved

63.8.6.3 Global Flag (GFLAG)

Offset

Register	Offset
GFLAG	4h

Function

Groups the flag fields from all channels into one register to improve interrupt handling.

For UCs, each field mirrors the corresponding channel flag (S*n*[FLAG]).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	F23	F22	F21	F20	F19	F18	F17	F16
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	F15	F14	F13	F12	F11	F10	F9	F8	F7	F6	F5	F4	F3	F2	F1	F0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
29 —	Reserved
28 —	Reserved
27 —	Reserved
26 —	Reserved
25 —	Reserved
24 —	Reserved
23 F23	Mirror of UC 23 FLAG
22 F22	Mirror of UC 22 FLAG
21 F21	Mirror of UC 21 FLAG
20 F20	Mirror of UC 20 FLAG
19 F19	Mirror of UC 19 FLAG
18 F18	Mirror of UC 18 FLAG
17 F17	Mirror of UC 17 FLAG
16 F16	Mirror of UC 16 FLAG

Table continues on the next page...

Table continued from the previous page...

Field	Function
15 F15	Mirror of UC 15 FLAG
14 F14	Mirror of UC 14 FLAG
13 F13	Mirror of UC 13 FLAG
12 F12	Mirror of UC 12 FLAG
11 F11	Mirror of UC 11 FLAG
10 F10	Mirror of UC 10 FLAG
9 F9	Mirror of UC 9 FLAG
8 F8	Mirror of UC 8 FLAG
7 F7	Mirror of UC 7 FLAG
6 F6	Mirror of UC 6 FLAG
5 F5	Mirror of UC 5 FLAG
4 F4	Mirror of UC 4 FLAG
3 F3	Mirror of UC 3 FLAG
2 F2	Mirror of UC 2 FLAG
1	Mirror of UC 1 FLAG

Table continues on the next page...

Table continued from the previous page...

Field	Function
F1	
0 F0	Mirror of UC 0 FLAG

63.8.6.4 Output Update Disable (OUDIS)

Offset

Register	Offset
OUDIS	8h

Function

Disables transfers from AS2 to AS1 and BS2 to BS1 on a per-channel basis. This applies to any channel operating in MC, MCB, or any output mode that writes to AS2 and BS2.

When a field is 0, transfers on that channel occur immediately or in the next period, depending on the operation mode. Unless that mode's description states otherwise, the transfer occurs immediately.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	OU23	OU22	OU21	OU20	OU19	OU18	OU17	OU16
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	OU15	OU14	OU13	OU12	OU11	OU10	OU9	OU8	OU7	OU6	OU5	OU4	OU3	OU2	OU1	OU0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
29 —	Reserved
28 —	Reserved
27 —	Reserved
26 —	Reserved
25 —	Reserved
24 —	Reserved
23 OU23	Channel 23 Output Update Disable 0b - Enable 1b - Disable
22 OU22	Channel 22 Output Update Disable 0b - Enable 1b - Disable
21 OU21	Channel 21 Output Update Disable 0b - Enable 1b - Disable
20 OU20	Channel 20 Output Update Disable 0b - Enable 1b - Disable
19 OU19	Channel 19 Output Update Disable 0b - Enable 1b - Disable
18 OU18	Channel 18 Output Update Disable 0b - Enable 1b - Disable

Table continues on the next page...

Table continued from the previous page...

Field	Function
17 OU17	Channel 17 Output Update Disable 0b - Enable 1b - Disable
16 OU16	Channel 16 Output Update Disable 0b - Enable 1b - Disable
15 OU15	Channel 15 Output Update Disable 0b - Enable 1b - Disable
14 OU14	Channel 14 Output Update Disable 0b - Enable 1b - Disable
13 OU13	Channel 13 Output Update Disable 0b - Enable 1b - Disable
12 OU12	Channel 12 Output Update Disable 0b - Enable 1b - Disable
11 OU11	Channel 11 Output Update Disable 0b - Enable 1b - Disable
10 OU10	Channel 10 Output Update Disable 0b - Enable 1b - Disable
9 OU9	Channel 9 Output Update Disable 0b - Enable 1b - Disable
8 OU8	Channel 8 Output Update Disable 0b - Enable 1b - Disable
7 OU7	Channel 7 Output Update Disable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Enable 1b - Disable
6 OU6	Channel 6 Output Update Disable 0b - Enable 1b - Disable
5 OU5	Channel 5 Output Update Disable 0b - Enable 1b - Disable
4 OU4	Channel 4 Output Update Disable 0b - Enable 1b - Disable
3 OU3	Channel 3 Output Update Disable 0b - Enable 1b - Disable
2 OU2	Channel 2 Output Update Disable 0b - Enable 1b - Disable
1 OU1	Channel 1 Output Update Disable 0b - Enable 1b - Disable
0 OU0	Channel 0 Output Update Disable 0b - Enable 1b - Disable

63.8.6.5 Disable Channel (UCDIS)

Offset

Register	Offset
UCDIS	Ch

Function

Allows you to disable a UC by stopping its clock. Each field controls the clock for the corresponding UC.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	UCDIS 23	UCDIS 22	UCDIS 21	UCDIS 20	UCDIS 19	UCDIS 18	UCDIS 17	UCDIS 16
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	UCDIS 15	UCDIS 14	UCDIS 13	UCDIS 12	UCDIS 11	UCDIS 10	UCDIS 9	UCDIS 8	UCDIS 7	UCDIS 6	UCDIS 5	UCDIS 4	UCDIS 3	UCDIS 2	UCDIS 1	UCDIS 0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30 —	Reserved
29 —	Reserved
28 —	Reserved
27 —	Reserved
26 —	Reserved
25 —	Reserved
24 —	Reserved
23 UCDIS23	Disable UC 23 0b - Enable 1b - Disable
22	Disable UC 22

Table continues on the next page...

Table continued from the previous page...

Field	Function
UCDIS22	0b - Enable 1b - Disable
21 UCDIS21	Disable UC 21 0b - Enable 1b - Disable
20 UCDIS20	Disable UC 20 0b - Enable 1b - Disable
19 UCDIS19	Disable UC 19 0b - Enable 1b - Disable
18 UCDIS18	Disable UC 18 0b - Enable 1b - Disable
17 UCDIS17	Disable UC 17 0b - Enable 1b - Disable
16 UCDIS16	Disable UC 16 0b - Enable 1b - Disable
15 UCDIS15	Disable UC 15 0b - Enable 1b - Disable
14 UCDIS14	Disable UC 14 0b - Enable 1b - Disable
13 UCDIS13	Disable UC 13 0b - Enable 1b - Disable
12 UCDIS12	Disable UC 12 0b - Enable 1b - Disable

Table continues on the next page...

Table continued from the previous page...

Field	Function
11 UCDIS11	Disable UC 11 0b - Enable 1b - Disable
10 UCDIS10	Disable UC 10 0b - Enable 1b - Disable
9 UCDIS9	Disable UC 9 0b - Enable 1b - Disable
8 UCDIS8	Disable UC 8 0b - Enable 1b - Disable
7 UCDIS7	Disable UC 7 0b - Enable 1b - Disable
6 UCDIS6	Disable UC 6 0b - Enable 1b - Disable
5 UCDIS5	Disable UC 5 0b - Enable 1b - Disable
4 UCDIS4	Disable UC 4 0b - Enable 1b - Disable
3 UCDIS3	Disable UC 3 0b - Enable 1b - Disable
2 UCDIS2	Disable UC 2 0b - Enable 1b - Disable
1 UCDIS1	Disable UC 1

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Enable 1b - Disable
0 UCDIS0	Disable UC 0 0b - Enable 1b - Disable

63.8.6.6 UC A n (A0 - A23)

Offset

For n = 0 to 23:

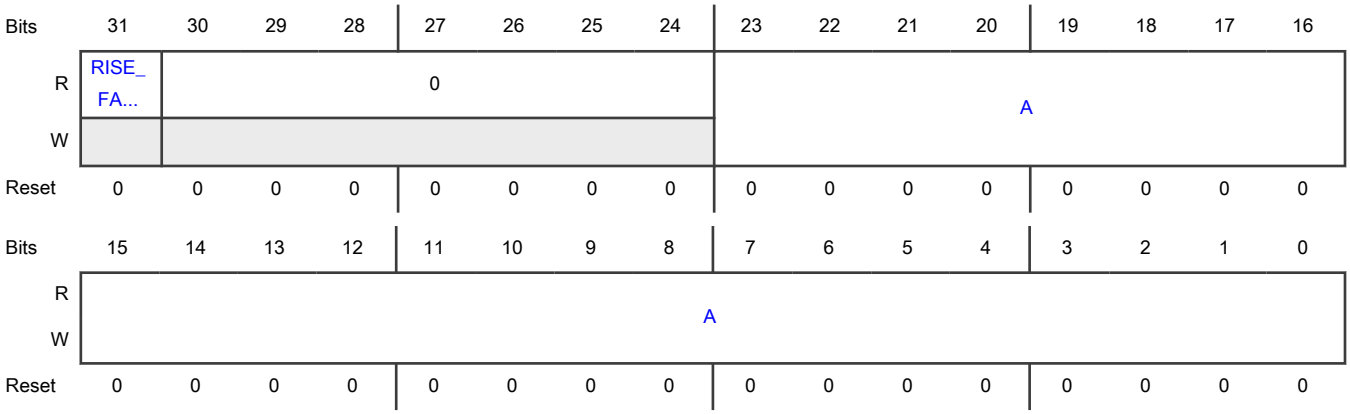
Register	Offset
An	20h + (n × 20h)

Function

Accesses the AS1 and AS2 shadow registers for each UC, depending on the selected UC mode. See AS1, AS2, BS1, and BS2 shadow registers.

For information about how you write to and read from A*n*, B*n*, and ALTA*n* for all UC modes, see Assignment of values for A*n*[A], B*n*[B], and ALTA*n*[ALTA].

Diagram



Fields

Field	Function
31 RISE_FALL	Rising and falling edge detection Indicates rising and falling edge in SAIC sub-mode.

Table continues on the next page...

Table continued from the previous page...

Field	Function
30-24 —	Reserved
23-0 A	A See the register description.

63.8.6.7 UC B n (B0 - B23)

Offset

For n = 0 to 23:

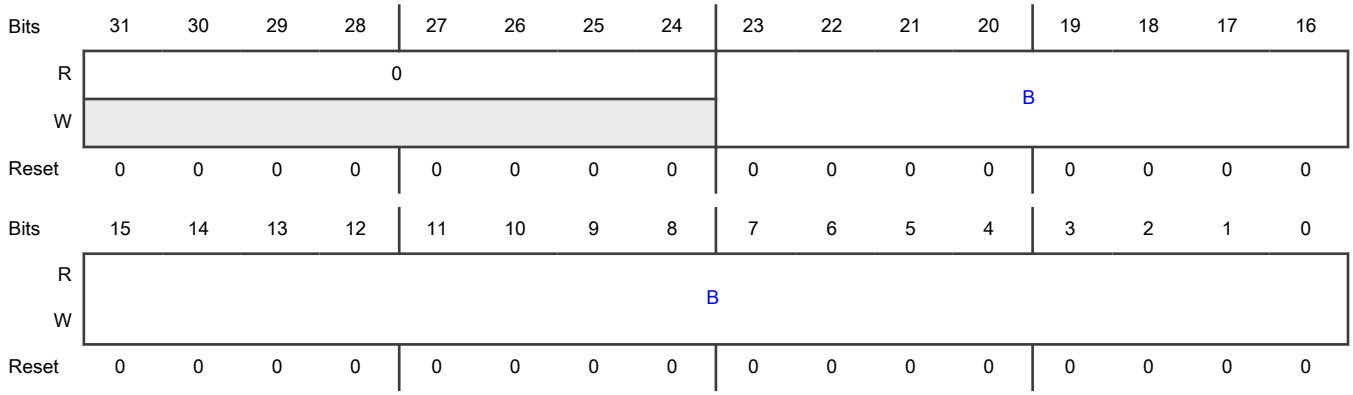
Register	Offset
Bn	24h + (n × 20h)

Function

Accesses the BS1 and BS2 shadow registers for each UC, depending on the selected UC mode. See AS1, AS2, BS1, and BS2 shadow registers.

For information about how you write to and read from A*n*, B*n*, and ALTA*n* for all UC modes, see Assignment of values for A*n*[A], B*n*[B], and ALTA*n*[ALTA].

Diagram



Fields

Field	Function
31-24 —	Reserved
23-0	B

Table continues on the next page...

Table continued from the previous page...

Field	Function
B	See the register description.

63.8.6.8 UC Counter n (CNT0 - CNT23)

Offset

Register	Offset
CNT0	28h
CNT1	48h
CNT2	68h
CNT3	88h
CNT4	A8h
CNT5	C8h
CNT6	E8h
CNT7	108h
CNT8	128h
CNT16	228h
CNT22	2E8h
CNT23	308h

Function

Indicates the value of the UC internal counter.

When eMIOS is in GPIO mode or the UC is frozen, this register is read-write. For all other modes, this register is read-only. When entering some UC modes, eMIOS automatically clears this register. See Unified channels (UC) for details.

The following modes use the UC counter:

- Output Pulse Width and Frequency Modulation Buffered (OPWFMB) mode
- Center Aligned Output PWM with Dead Time Insertion Buffered (OPWMCB) mode
- Pulse Edge Counting (PEC) mode
- Modulus Counter (MC) mode
- Modulus Counter Buffered (MCB) mode

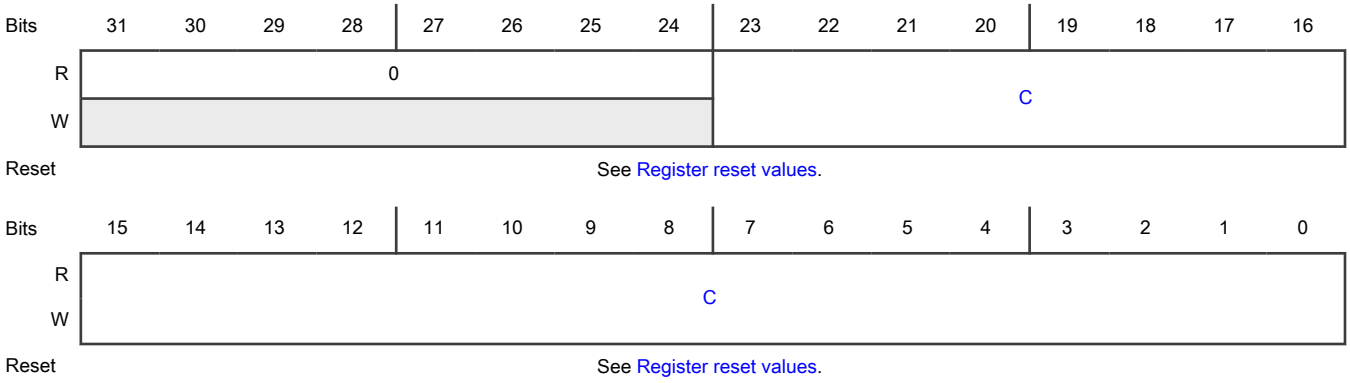
All other modes not in this list do not use the internal counter.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
eMIOS_0	CNT0–CNT8 CNT16 CNT22–CNT23	—
eMIOS_1	CNT0 CNT8 CNT16 CNT22–CNT23	CNT1–CNT7
eMIOS_2	CNT0 CNT8 CNT16 CNT22–CNT23	CNT1–CNT7

Diagram



Register reset values

Register	Reset value
CNT0	eMIOS_0–eMIOS_2: 0000_0000h
CNT1–CNT7	0000_0000h
CNT8–CNT23	eMIOS_0–eMIOS_2: 0000_0000h
CNT9–CNT15	—
CNT17–CNT21	—

Fields

Field	Function
31-24 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
23-0 C	Internal Counter Value Indicates the value of the internal counter.

63.8.6.9 UC Control n (C0 - C23)

Offset

For n = 0 to 23:

Register	Offset
Cn	2Ch + (n × 20h)

Function

Controls UC operation.

Table 420. MODE field values

UC mode	MODE (binary)
GPIO (input)	000_0000
GPIO (output)	000_0001
SAIC	000_0010
SAIC with edge capturing	100_0010
SAOC	000_0011
IPWM	000_0100
IPM	000_0101
DAOC (with FLAG = 1 on B match)	000_0110
DAOC (with FLAG = 1 on A and B match)	000_0111
PEC (continuous)	000_1010
PEC (single shot)	000_1011
Reserved	000_1111
MC (up counter with clear on match start)	001_000p[1]
MC (up counter with clear on match end)	001_001p[1]
MC (up/down counter)	001_01pp[1]
Reserved	010_0100–010_0101
OPWMT	010_0110
Reserved	010_0111–100_1111

Table continues on the next page...

Table 420. MODE field values (continued)

UC mode	MODE (binary)
MCB (up counter)	101_000p[1]
Reserved	101_001p
MCB (up or down counter)	101_01pp[1]
OPWFMB	101_10p0[1]
Reserved	101_10p1
OPWMCB (with trailing edge dead time)	101_11p0[1]
OPWMCB (with leading edge dead time)	101_11p1[1]
OPWMB	110_00p0[1]
Reserved	110_0001–111_1111

1. p = Adjust parameters for the mode of operation. See Unified channels (UC) for details.

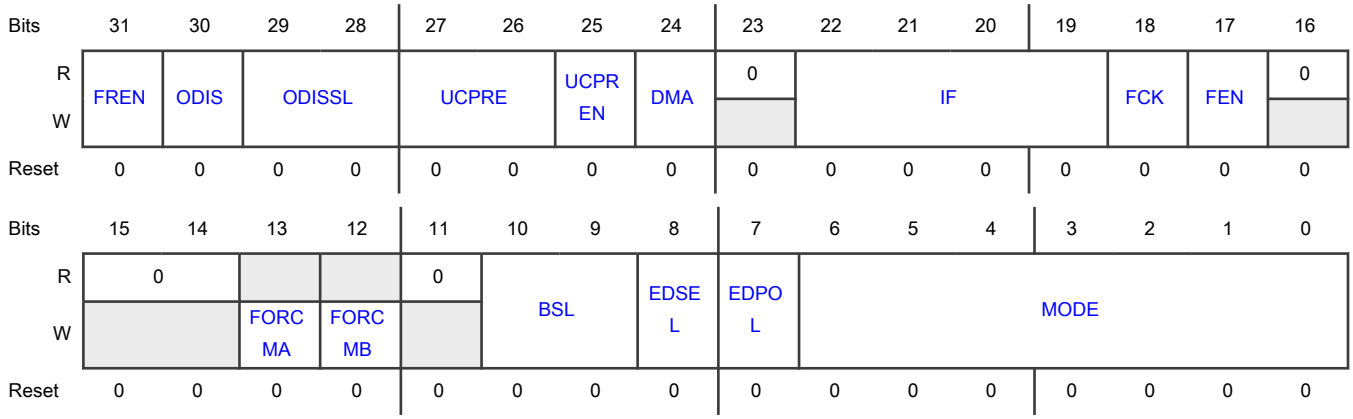
Table 421. Edge Polarity (EDPOL) settings

For these modes:	EDPOL does this:	and EDPOL values mean:
Input	Selects which edge triggers the internal counter, an input capture, or the input capture flag. If EDPOL does not appear in the UC mode description, it has no effect on UC operation.	0b - Trigger on a falling edge 1b - Trigger on a rising edge
Output	Selects the logic level on the output pin.	0b - A match on comparator A deasserts the output flip-flop, while a match on comparator B sets it 1b - A match on comparator A asserts the output flip-flop, while a match on comparator B clears it

Table 422. Edge Select (EDSEL) settings

For these modes:	EDSEL does this:	and EDSEL values mean:
Input	Selects whether the internal counter is triggered by both edges of a pulse or by a single edge as defined by EDPOL. If EDSEL does not appear in the UC mode description, it has no effect on UC operation.	0b - Single-edge triggering on edge specified in EDPOL 1b - Both-edges triggering
GPIO in	Selects whether an edge asserts the input capture flag (S*n*[FLAG]).	0b - Sets S*n*[FLAG] on the edge specified in EDPOL 1b - Does not set S*n*[FLAG]
SAOC	Selects the behavior of the output flip-flop at each match.	0b - Drive the EDPOL value on the output flip-flop 1b - Toggle output flip-flop

Diagram



Fields

Field	Function
31 FREN	Freeze Enable Enables putting the UC in Freeze state. See Freeze and chip Debug mode. 0b - Disable 1b - Enable
30 ODIS	Output Disable Disables the output pin when running any of the output modes except GPIO. When you assert ODIS: • If the selected Output Disable Input signal asserts, the output pin reflects: — For OPWFMB and OPWMB modes, EDPOL (see Output Pulse Width and Frequency Modulation Buffered (OPWFMB) mode and Output PWM Buffered (OPWMB) mode). — For all other modes, the complement of EDPOL, but the UC continues to operate normally—it continues to generate flags and matches. • If the selected Output Disable Input signal deasserts, the output pin operates normally. 0b - Enables output pin (the output pin operates normally) 1b - Disables output pin
29-28 ODISSL	Output Disable Select Selects one of four output disable input signals. 00b - 0 01b - 1 10b - 2 11b - 3
27-26	Prescaler Selects the clock divider value for the UC internal prescaler.

Table continues on the next page...

Table continued from the previous page...

Field	Function
UCPRE	**NOTE** The two least-significant bits of C2_*n*[UCEXTPRE] mirror UCPRE. Any write to UCPRE also affects UCEXTPRE. 00b - 1 01b - 2 10b - 3 11b - 4
25 UCPREN	Prescaler Enable Enables the prescaler counter. 0b - Disable (no clock) 1b - Enable
24 DMA	Direct Memory Access Selects whether the input capture flag (S*n*[FLAG]) or overrun (S*n*[OVR]) triggers an interrupt or DMA request. 0b - Interrupt request 1b - DMA request
23 —	Reserved
22-19 IF	Input Filter Selects the minimum input pulse width, in filter clock cycles, that can pass through the input filter. This field does not apply to output modes. The filter latency—the difference in time between the input and the response—is three clock edges. 0000b - Bypassed; the input signal is synchronized before arriving at the digital filter 0001b - 2 cycles 0010b - 4 cycles 0100b - 8 cycles 1000b - 16 cycles All other values are reserved.
18 FCK	Filter Clock Select Selects the clock source for the programmable input filter. 0b - Prescaled clock 1b - eMIOS module clock

Table continues on the next page...

Table continued from the previous page...

Field	Function
17 FEN	Flag Enable Enables the input capture flag (S*n*[FLAG]) to generate an interrupt request or a DMA request, as selected in DMA. 0b - Disable 1b - Enable
16-14 —	Reserved
13 FORCMA	Force Match A For output modes, asserting Force Match A causes the comparator to report a successful comparison on comparator A regardless of whether a comparison actually occurred. It does not set the input capture flag (S*n*[FLAG]). This field always reads 0. This bit is meaningful for every output operation mode that uses comparator A. Otherwise it has no effect. 0b - No effect 1b - Force a match at comparator A
12 FORCMB	Force Match B For output modes, asserting Force Match B causes the comparator to report a successful comparison on comparator B regardless of whether a comparison actually occurred. It does not set the input capture flag (S*n*[FLAG] does not become 1). This field always reads 0. This field applies only if the UC is in one of the following output modes. • Single Action Output Capture (SAOC) mode • Double Action Output Compare (DAOC) mode • Output Pulse Width and Frequency Modulation Buffered (OPWFMB) mode • Center Aligned Output PWM with Dead Time Insertion Buffered (OPWMCB) mode • Output PWM Buffered (OPWMB) mode • Output PWM with Trigger (OPWMT) mode This field has no effect in all other modes. 0b - No effect 1b - Force a match at comparator B
11 —	Reserved
10-9 BSL	Bus Select Selects one of the counter buses or the internal counter for the UC. Not all chips support all counter buses. See the chip-specific eMIOS information for details. 00b -

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• Channels 0-22: counter bus A • Channel 23: reserved 01b - • Channels 1-7: counter bus B • Channels 9-15: counter bus C • Channels 17-23: counter bus D • Channels 25-31: counter bus E • Channels 0, 8, 16, 24: reserved 10b - • Channels 0-21, 23: counter bus F • Channel 22: reserved 11b - Internal counter for all channels
8 EDSEL	Edge Selection Controls several functions as shown in Table 422.
7 EDPOL	Edge Polarity Controls several functions as shown in Table 421.
6-0 MODE	Mode Selection Selects the UC mode as shown in Table 420. **NOTE** If you write a reserved value to this field, the results are unpredictable. When you change the value of MODE, you must first enter GPIO mode to reset the UC's internal functions. Failure to do this can result in invalid and unexpected output compare or input capture results or incorrectly set input capture flags (S*n*[FLAG]).

63.8.6.10 UC Status n (S0 - S23)

Offset

For n = 0 to 23:

Register	Offset
Sn	30h + (n × 20h)

Function

Indicates the status of the UC input-output signals and the overflow condition of the internal counter.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	OVR	0														
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	OVFL	0												UCIN	UCOUT	FLAG
W	W1C															W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	u[1]	0	0

1. Reset value is undefined.

Fields

Field	Function
31 OVR	Overrun Indicates that FLAG generation occurred when FLAG was already 1. Return OVR to 0 by writing 1 to either OVR or FLAG. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No overrun 1b - Overrun When writing 0b - No overrun 1b - Return OVR to 0
30-16 —	Reserved
15 OVFL	Overflow Indicates that an overflow has occurred in the internal counter. **NOTE** This field is not supported in every instance. The following table includes only supported registers.

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>eMIOS_0</td><td>S0–S8
S16
S22–S23</td><td>S9–S15
S17–S21</td></tr>
<tr><td>eMIOS_1</td><td>S0
S8
S16
S22–S23</td><td>S1–S7
S9–S15
S17–S21</td></tr>
<tr><td>eMIOS_2</td><td>S0
S8
S16
S22–S23</td><td>S1–S7
S9–S15
S17–S21</td></tr>
</table>
NOTE
This field behaves differently for register reads and writes.
When reading
0b - No overflow
1b - Overflow
When writing
0b - No overflow
1b - Return OVFL to 0</td></tr>
<tr><td>14-3
—</td><td>Reserved</td></tr>
<tr><td>2
UCIN</td><td>UC Input Pin
Indicates the input pin state after the input pin has been filtered and synchronized.
0b - Negated
1b - Asserted</td></tr>
<tr><td>1
UCOUT</td><td>UC Output Pin
Indicates the output pin state.
0b - Negated
1b - Asserted</td></tr>
<tr><td>0
FLAG</td><td>Flag
Indicates that an input capture or a match event in the comparators has occurred.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field behaves differently for register reads and writes. When reading 0b - No event 1b - Event has occurred When writing 0b - No event 1b - Clear the flag (FLAG returns to 0)

63.8.6.11 Alternate Address n (ALTA0 - ALTA23)

Offset

For n = 0 to 23:

Register	Offset
ALTAn	34h + (n × 20h)

Function

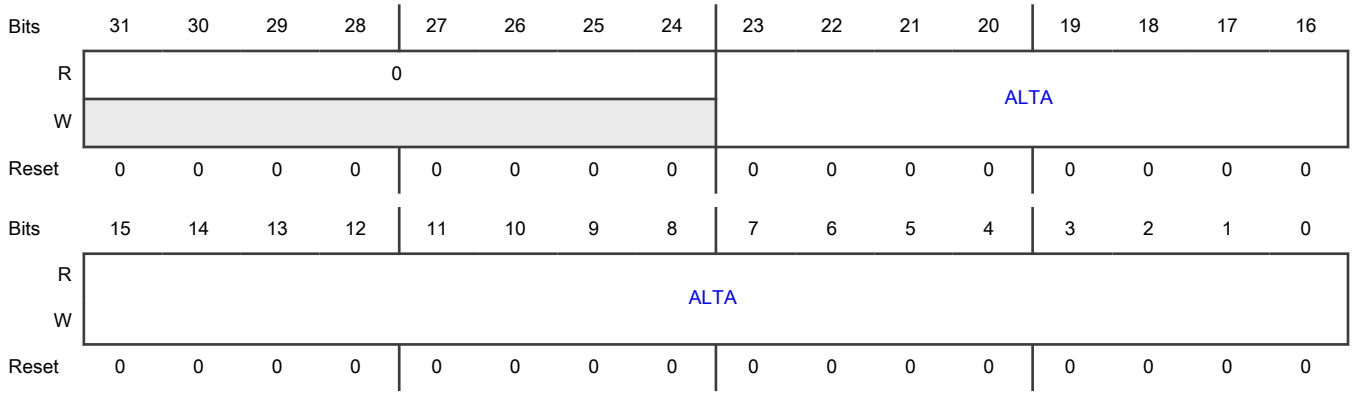
Provides an alternate address to access AS2 shadow registers in these restricted modes:

- General-Purpose Input and Output (GPIO) mode
- Pulse Edge Counting (PEC) mode
- Output PWM with Trigger (OPWMT) mode

In these modes, you can use A*n*[A] together with ALTA*n*[ALTA] to access both AS1 and AS2. Assignment of values for A*n*[A], B*n*[B], and ALTA*n*[ALTA] summarizes the ALTA*n* writing and reading accesses for all ALTA*n* operation modes.

In modes that do not require this register, any write access is ignored and any read access returns unspecified data.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-0 ALTA	Alternate Address See the register description.

63.8.6.12 UC Control 2 n (C2_0 - C2_23)

Offset

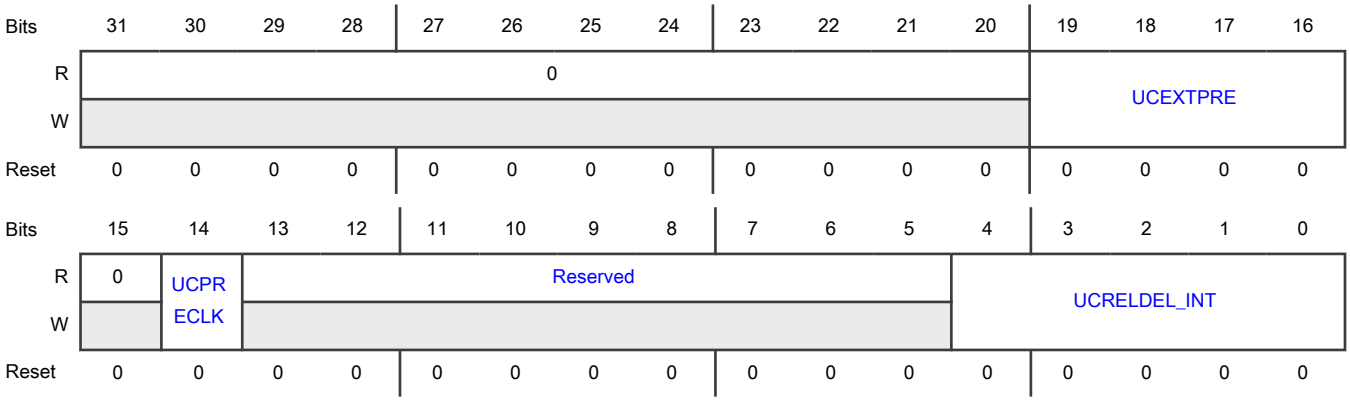
For n = 0 to 23:

Register	Offset
C2_n	38h + (n × 20h)

Function

Provides additional controls for the UC.

Diagram



Fields

Field	Function
31-20 —	Reserved
19-16 UCEXTPRE	Extended Prescaler Specifies the clock divider for the internal UC prescaler. Write the desired divider value minus 1. For example, for a divider of 1, write 0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE The two least-significant bits of UCEXTPRE mirror C*n*[UCPRE]. Any write operation to UCEXTPRE also affects UCPRE.
15 —	Reserved
14 UCPRECLK	Prescaler Clock Source Selects the clock source for the internal prescaler. 0b - Prescaled clock 1b - eMIOS module clock
13-5 —	Reserved
4-0 UCRELDEL_INT	Reload Signal Output Delay Interval Specifies the delay interval, in counter bus reload events, between each assertion of AS1-BS1 reload in MC and MCB modes. 00000b - Every event 00001b - Every 2nd event 00010b - Every 3rd event . . . 11111b - Every 32nd event

63.9 Glossary

BIU	IP bus interface unit
CP	Clock prescaler
GCP	Global clock prescaler
IIB	Internal interface bus
PIF	Programmable input filter
UC	Unified channel

Chapter 64
Body Cross-triggering Unit (BCTU)

64.1 Chip-specific BCTU information

64.1.1 BCTU configuration

The following table lists the BCTU configuration.

Table 423. BCTU configuration

Feature	Configuration
FIFO1	16 words
FIFO2	8 words

64.1.2 Instances and trigger sources

This chip has one instance of BCTU module.

The following table lists the BCTU trigger sources.

Table 424. BCTU Trigger sources

BCTU Trigger Number	Module	Source	Applicability
0	eMIOS_0	Channel_0	All
1	eMIOS_0	Channel_1	All
2	eMIOS_0	Channel_2	All
3	eMIOS_0	Channel_3	All
4	eMIOS_0	Channel_4	All
5	eMIOS_0	Channel_5	All
6	eMIOS_0	Channel_6	All
7	eMIOS_0	Channel_7	All
8	eMIOS_0	Channel_8	All
9	eMIOS_0	Channel_9	All
10	eMIOS_0	Channel_10	All
11	eMIOS_0	Channel_11	All
12	eMIOS_0	Channel_12	All
13	eMIOS_0	Channel_13	All
14	eMIOS_0	Channel_14	All
15	eMIOS_0	Channel_15	All
16	eMIOS_0	Channel_16	All
17	eMIOS_0	Channel_17	All

Table continues on the next page...

Table 424. BCTU Trigger sources (continued)

BCTU Trigger Number	Module	Source	Applicability
18	eMIOS_0	Channel_18	All
19	eMIOS_0	Channel_19	All
20	eMIOS_0	Channel_20	All
21	eMIOS_0	Channel_21	All
22	eMIOS_0	Channel_22	All
23	Any of the TRGMUX inp	BCTU_Trg23	All
24	eMIOS_1	Channel_0	All
25	eMIOS_1	Channel_1	All
26	eMIOS_1	Channel_2	All
27	eMIOS_1	Channel_3	All
28	eMIOS_1	Channel_4	All
29	eMIOS_1	Channel_5	All
30	eMIOS_1	Channel_6	All
31	eMIOS_1	Channel_7	All
32	eMIOS_1	Channel_8	All
33	eMIOS_1	Channel_9	All
34	eMIOS_1	Channel_10	All
35	eMIOS_1	Channel_11	All
36	eMIOS_1	Channel_12	All
37	eMIOS_1	Channel_13	All
38	eMIOS_1	Channel_14	All
39	eMIOS_1	Channel_15	All
40	eMIOS_1	Channel_16	All
41	eMIOS_1	Channel_17	All
42	eMIOS_1	Channel_18	All
43	eMIOS_1	Channel_19	All
44	eMIOS_1	Channel_20	All
45	eMIOS_1	Channel_21	All
46	eMIOS_1	Channel_22	All
47	Any of the TRGMUX inp	BCTU_Trg47	All
48	eMIOS_2	Channel_0	S32K388/S32K389/ S32K358/S32K356/S32K348/

Table continues on the next page...

Table 424. BCTU Trigger sources (continued)

BCTU Trigger Number	Module	Source	Applicability
			S32K338/S32K328/S32K344/ S32K324/S32K314
49	eMIOS_2	Channel_1	S32K388/S32K389/ S32K358/S32K356/S32K348/ S32K338/S32K328/S32K344/ S32K324/S32K314
50	eMIOS_2	Channel_2	S32K388/S32K389/ S32K358/S32K356/S32K348/ S32K338/S32K328/S32K344/ S32K324/S32K314
51	eMIOS_2	Channel_3	S32K388/S32K389/ S32K358/S32K356/S32K348/ S32K338/S32K328/S32K344/ S32K324/S32K314
52	eMIOS_2	Channel_4	S32K388/S32K389/ S32K358/S32K356/S32K348/ S32K338/S32K328/S32K344/ S32K324/S32K314
53	eMIOS_2	Channel_5	S32K388/S32K389/ S32K358/S32K356/S32K348/ S32K338/S32K328/S32K344/ S32K324/S32K314
54	eMIOS_2	Channel_6	S32K388/S32K389/ S32K358/S32K356/S32K348/ S32K338/S32K328/S32K344/ S32K324/S32K314
55	eMIOS_2	Channel_7	S32K388/S32K389/ S32K358/S32K356/S32K348/ S32K338/S32K328/S32K344/ S32K324/S32K314
56	eMIOS_2	Channel_8	S32K388/S32K389/ S32K358/S32K356/S32K348/ S32K338/S32K328/S32K344/ S32K324/S32K314
57	eMIOS_2	Channel_9	S32K388/S32K389/ S32K358/S32K356/S32K348/ S32K338/S32K328/S32K344/ S32K324/S32K314
58	eMIOS_2	Channel_10	S32K388/S32K389/ S32K358/S32K356/S32K348/ S32K338/S32K328/S32K344/ S32K324/S32K314
59	eMIOS_2	Channel_11	S32K388/S32K389/ S32K358/S32K356/S32K348/

Table continues on the next page...

Table 424. BCTU Trigger sources (continued)

BCTU Trigger Number	Module	Source	Applicability
			S32K338/S32K328/S32K344/ S32K324/S32K314
60	eMIOS_2	Channel_12	S32K388/S32K389/ S32K358/S32K356/S32K348/ S32K338/S32K328/S32K344/ S32K324/S32K314
61	eMIOS_2	Channel_13	S32K388/S32K389/ S32K358/S32K356/S32K348/ S32K338/S32K328/S32K344/ S32K324/S32K314
62	eMIOS_2	Channel_14	S32K388/S32K389/ S32K358/S32K356/S32K348/ S32K338/S32K328/S32K344/ S32K324/S32K314
63	eMIOS_2	Channel_15	S32K388/S32K389/ S32K358/S32K356/S32K348/ S32K338/S32K328/S32K344/ S32K324/S32K314
64	eMIOS_2	Channel_16	S32K388/S32K389/ S32K358/S32K356/S32K348/ S32K338/S32K328/S32K344/ S32K324/S32K314
65	eMIOS_2	Channel_17	S32K388/S32K389/ S32K358/S32K356/S32K348/ S32K338/S32K328/S32K344/ S32K324/S32K314
66	eMIOS_2	Channel_18	S32K388/S32K389/ S32K358/S32K356/S32K348/ S32K338/S32K328/S32K344/ S32K324/S32K314
67	eMIOS_2	Channel_19	S32K388/S32K389/ S32K358/S32K356/S32K348/ S32K338/S32K328/S32K344/ S32K324/S32K314
68	eMIOS_2	Channel_20	S32K388/S32K389/ S32K358/S32K356/S32K348/ S32K338/S32K328/S32K344/ S32K324/S32K314
69	eMIOS_2	Channel_21	S32K388/S32K389/ S32K358/S32K356/S32K348/ S32K338/S32K328/S32K344/ S32K324/S32K314
70	eMIOS_2	Channel_22	S32K388/S32K389/ S32K358/S32K356/S32K348/

Table continues on the next page...

Table 424. BCTU Trigger sources (continued)

BCTU Trigger Number	Module	Source	Applicability
			S32K338/S32K328/S32K344/ S32K324/S32K314
71	Any of the TRGMUX inp	BCTU_Trg71	S32K388/S32K389/ S32K358/S32K356/S32K348/ S32K338/S32K328/S32K344/ S32K324/S32K314

NOTE

You must clear triggers to BCTU after TRGMUX programming using TRGCFG_n[TRG_FLAG] bit.

64.2 Overview

BCTU accepts ADC conversion-request trigger inputs and routes those requests to one or more ADCs.

BCTU accepts the following trigger sources:

- Hardware triggers (outputs from on-chip timer modules)
- Software triggers through the SFTRGRn register

BCTU maps each trigger input to a Trigger Configuration (TRGCFG_0 - TRGCFG_71) register, which specifies the target ADC channel and other control data for the requested conversion.

You can configure a list of ADC channels, the CL, that processes a sequence of conversions in response to a single trigger input. See the Conversion list (CL) section.

BCTU stores each ADC conversion result for CPU access or routing to memory through DMA.

If the same ADC is targeted by multiple triggers, priority logic selects the next trigger to process from all active trigger inputs. See Priority of simultaneous triggers.

BCTU input trigger signals are level-sensitive; thus, if a logical level 1 is detected, an event is captured by the BCTU.

64.2.1 Block diagram

The following figure shows the top-level block diagram of BCTU.

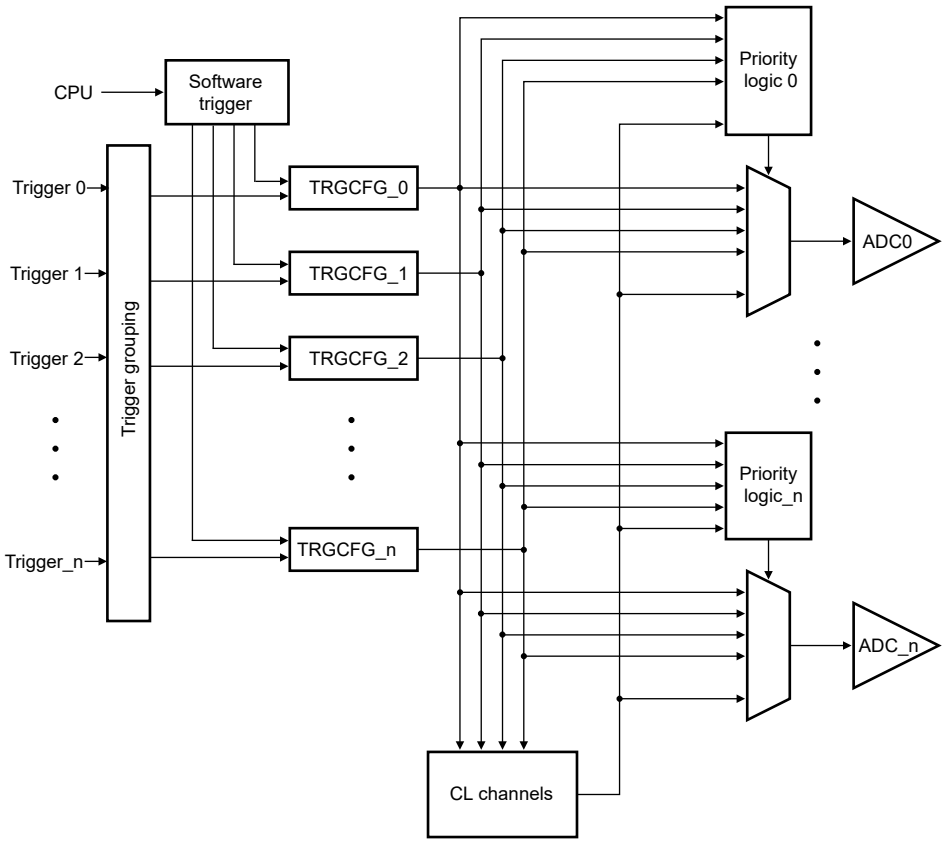


Figure 370. BCTU block diagram

64.2.2 Features

BCTU supports:

- 72 Trigger inputs, configured with the Trigger Configuration (TRGCFG_0 - TRGCFG_71) registers
- Trigger outputs to 3 ADCs
- BCTU to ADC0 and ADC1
- One priority selection logic block per ADC
- Selection multiplexers to select one trigger at a time for each ADC
- A CL of 32 ADC channels
- A trigger grouping block to latch the hardware triggers. BCTU clears these triggers when the ADC sends an acknowledgement for conversion completion.

64.2.3 Interaction with other peripherals

BCTU works together with timer and ADC modules on the chip. Use the trigger configuration registers (TRGCFG_*n*) to map BCTU triggers to ADC inputs. When a timer output asserts a BCTU trigger, BCTU passes the trigger to an ADC input and stores the trigger assertion until the corresponding ADC begins the requested conversion. When the ADC completes the conversion, BCTU asserts a trigger clear signal to clear both the internally stored trigger and the external trigger source if they are still asserted. Implementation illustrates the modules that interact with BCTU.

NOTE

A timer trigger can remain active until the corresponding ADC initiates conversion. Therefore, another match may occur in the timer before the ADC conversion begins. BCTU does not detect this trigger overrun. Your application must look for and respond to such conditions.

64.2.4 Implementation

The following figure shows the implementation structure of the BCTU.

NOTE

For more details of the actual trigger sources implemented in this device, see the chip-specific information section.

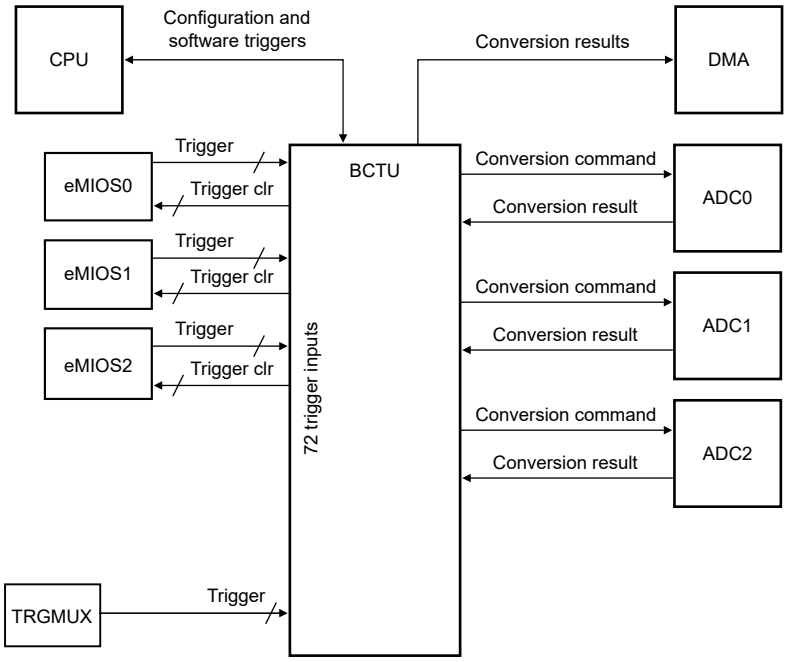


Figure 371. Implementation

64.3 Functional description

64.3.1 Triggers

64.3.1.1 Overview

BCTU prioritizes ADC trigger inputs based on the input number: the lower the number, the higher the priority associated with it. Therefore, trigger 0 is the highest priority trigger and trigger 71 is the lowest priority. The trigger input priority scheme determines which trigger takes precedence when more than one trigger occurs at the same time, or when the triggered ADC is busy with a conversion. Therefore, there are two possible scenarios for the trigger priority selection:

- When there are no ongoing conversions
- When a triggered ADC is executing a conversion

BCTU supports two types of trigger sources:

- Hardware triggers from timer modules
- Software triggers (SFTRGR*n*)

64.3.1.2 Hardware triggers

To request an ADC conversion through the BCTU, a timer module such as eMIOS asserts its output signal, which is connected to a BCTU trigger input. When BCTU detects the asserted trigger, it routes that request to the ADCs selected in the corresponding Trigger Configuration (TRGCFG_*n*[ADC_SEL*m*]). The trigger remains asserted until the requested conversion begins, at which time BCTU clears the trigger. If the timer asserts the trigger input when the trigger is already set, BCTU ignores the new trigger. If BCTU receives a new trigger input at the same time it is clearing the trigger, then the trigger remains asserted and BCTU initiates a new conversion.

If you configure the trigger for CL, trigger assertion begins a sequence of conversions controlled by the CL Channel Address (LISTCHR_0 - LISTCHR_15) registers. In this event, BCTU clears the trigger when the last conversion in the CL begins.

NOTE

Any new trigger input participates in the trigger priority selection scheme and therefore may not trigger the next conversion.

64.3.1.3 Software triggers

You assert a software trigger by writing 1 to the corresponding field of the appropriate software trigger (SFTRGR*n*) register. Similar to the hardware triggers, BCTU changes the software trigger field to 0 when the requested conversion begins. If you write 1 to the software trigger field while the conversion is still in progress, BCTU ignores the request. If you write 1 to the field at the same time BCTU is attempting to clear the trigger (change the field to 0), the trigger remains asserted and BCTU triggers a new conversion following the trigger priority criteria.

64.3.1.4 Configure all triggers

Global trigger enable (MCR[GTRGEN]) allows you to configure the BCTU trigger inputs and then activate them all at one time.

To configure and then activate all BCTU triggers:

1. Clear any currently active and asserted trigger inputs (write 0 to TRGFG_*n*[TRG_FLAG]), which also causes BCTU to assert trigger clear to the trigger sources.
2. Disable global triggers (MCR[GTRGEN]). This action deactivates all trigger inputs and prevents BCTU from generating any ADC conversion requests.
3. Configure the individual trigger inputs (see Configure individual triggers).
4. Enable global triggers (MCR[GTRGEN]). This action activates all individually enabled triggers at the same time.

64.3.1.5 Configure individual triggers

Using Trigger Configuration (TRGCFG_0 - TRGCFG_71), for each trigger:

- Enable the trigger (TRIGEN).
- Specify the target ADC (ADC_SEL*n*).
- Specify the ADC channel (CHANNEL_VALUE_OR_LADDR).
- Specify whether the trigger initiates a single conversion or a CL (TRS).
- For a CL trigger, specify a pointer to the CL element (TRGCFG_x[CHANNEL_VALUE_OR_LADDR]).

64.3.1.6 Reconfigure triggers on-the-fly

NOTE

Reconfigure BCTU triggers only when the individual trigger is not active (TRG_FLAG). If TRG_FLAG is 1, the requested conversion is still in progress and it is not safe to modify the trigger configuration.

To reconfigure a trigger on-the-fly (while BCTU is in full operation):

1. Disable the trigger (TRIGEN) to prevent it from requesting a new conversion.
2. Check the current state of the trigger (TRG_FLAG) to be sure no conversion or CL is in progress.
3. When any in-progress conversion or CL completes (TRG_FLAG == 0), clear the trigger source by writing 1 to TRG_FLAG. This action ensures that only trigger inputs that assert after you reconfigure and re-enable the trigger are able to request a new conversion or CL.
4. Reconfigure the trigger (see Configure individual triggers).
5. Enable the trigger (TRIGEN).

64.3.1.7 Enable software triggers

To allow software triggers (SFTRGRn) to request conversions, you must enable global trigger enable (MCR[GTRGEN]). If you disable global trigger enable, software triggers do not initiate a conversion but asserted triggers remain asserted. However, when you enable global trigger enable, any asserted software triggers initiate conversions following the priority scheme of lowest to highest trigger number.

Software triggers operate regardless of individual trigger enable (TRGCFG_x[TRIGEN]). That is, writing 1 to any Software Trigger (SFTRGRn) initiates a conversion (following the priority scheme) regardless of the value of trigger enable. This software trigger independence supports the elimination of conflict between software and hardware triggers.

64.3.1.8 Trigger timing

There are four clock cycles from assertion of a BCTU trigger input to assertion of the trigger output to the ADC. For single conversions, BCTU asserts trigger clear back to the trigger source simultaneous to asserting the trigger to the ADC.

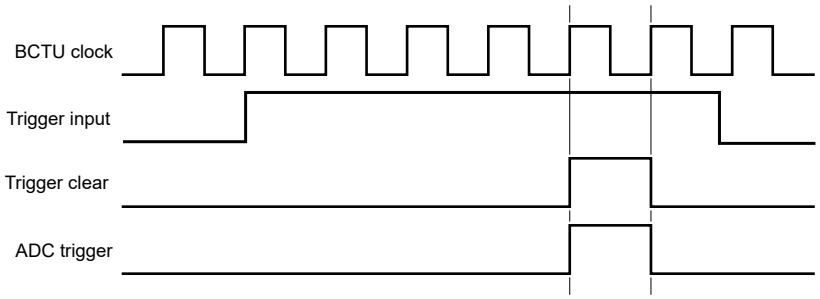


Figure 372. Trigger timing

64.3.1.9 Priority of simultaneous triggers

When more than one trigger input asserts a conversion request to the same ADC at the same time—that is, in the same BCTU clock cycle—BCTU employs a priority scheme to determine which asserted trigger to process next. BCTU prioritizes trigger inputs based on the trigger number. The lowest number, trigger 0, has the highest priority, and the highest number, trigger 71, has the lowest priority.

If two triggers assert in the same clock cycle, BCTU gives priority to the lower numbered trigger. For example, if trigger 0 and trigger 1 both assert in the same cycle, BCTU selects trigger 0 for trigger generation. Trigger 1 remains active (pending) until the conversion requested by trigger 0 completes. See Simultaneous trigger priority management.

If trigger 0 asserts again before the next trigger selection opportunity at the end of the current conversion, then BCTU selects it again and trigger 1 remains asserted (pending) during the next ADC conversion time frame.

As illustrated in Trigger timing, BCTU requires four clock cycles to transmit a conversion request to an ADC if that ADC is not already executing a conversion. However, if the ADC is busy at the time of trigger assertion, BCTU requires only three clock cycles to generate the conversion request for a subsequent conversion.

Triggers that assert in the same clock cycle but target different ADCs are not in conflict, and BCTU processes both triggers at the same time.

64.3.1.10 Simultaneous trigger priority management

The following figure shows the simultaneous triggers for priority management.

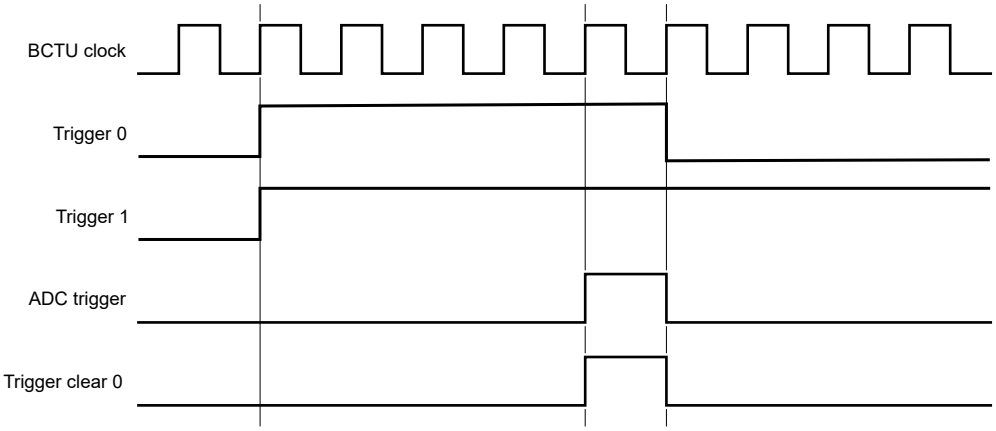


Figure 373. Simultaneous trigger priority management

64.3.1.11 Trigger management when ADC is busy

When a trigger input asserts a conversion request to an ADC that is already processing a conversion requested by a different trigger, the new trigger remains active (pending) until the conversion in progress completes. This behavior is the same as if the two triggers had been simultaneous (occurred in the same clock cycle). When the in-progress conversion completes, the pending trigger then competes for priority with any other active trigger for that ADC.

64.3.1.12 Trigger clearing mechanism

There are two sources of ADC conversion triggers: hardware trigger, from the trigger inputs and software triggers, from SFTRGRn register.

A logical level 1 is detected in the input and captured into an input trigger register. This register remains set until the related conversion starts. At this time, the input trigger register is cleared. If another logical level 1 is detected while the trigger register is already set, the new level is not considered by the BCTU hardware. If a new logical level 1 occurs at the same time the trigger register is being cleared, then the register remains set and a new conversion is triggered by this input.

NOTE

The new triggered conversion will participate in the trigger priority selection thus may not trigger the next conversion.

The software trigger behaves similarly to the hardware trigger. The difference is that the trigger register is set by a CPU write operation. The trigger register is cleared by the BCTU hardware when the related conversion starts. If the CPU tries to set the trigger register by writing to the SFTRGR register corresponding bit, while the register is already set, the new attempt to set the register is not considered by the BCTU hardware. If a set occurs at the same time the hardware is attempting to clear the same trigger register, then the set has precedence and the register remains set. A new conversion will be triggered following the trigger priority criteria implemented in the hardware.

In case a LIST of conversions is triggered, instead of a single conversion, the trigger register is cleared at the start of the last conversion in the LIST.

64.3.2 Conversion list (CL)

64.3.2.1 CL overview

A CL executes a sequence of conversions on one or more target ADCs. See CL architecture and operation. It also stores information from the last conversion executed in a sequence except for the target ADC, which is stored in ADC select n (TRGCFG_*m*[ADC_SEL*n*]).

If the channel or CL address (TRGCFG_x[CHANNEL_VALUE_OR_LADDR]) points to a CL element containing one or more entries:

1. The trigger initiates a sequence of conversions starting with the target CL element.
2. The sequence executes until the LAST channel (LISTCHR_*n*[LAST_y] is 1), and stops after that channel is converted.

When you configure the CL element for next channel wait for trigger (write 1 to LISTCHR_*n*[NEXT_CH_WAIT_ON_TRIG_y]):

1. BCTU clears the trigger source.
2. CL execution pauses after the current conversion is completed.
3. CL execution resumes when the same trigger asserts again.

BCTU determines trigger priority before executing any CL. See Priority of simultaneous triggers.

64.3.2.2 CL architecture and operation

The following figure shows the CL architecture and operation.

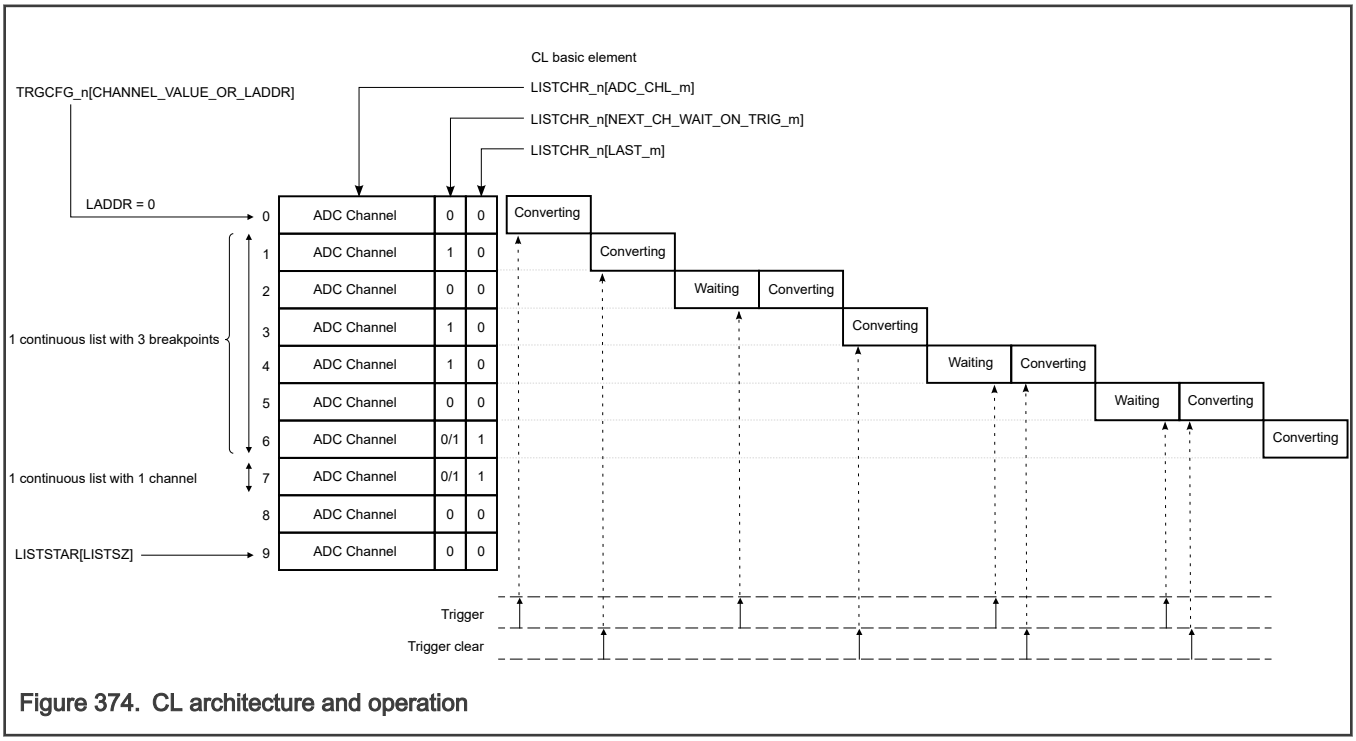


Figure 374. CL architecture and operation

64.3.2.3 CL channel rules

CL conversion obeys the following rules:

- The sequence of conversions executes until the Last Channel (LISTCHR_*n*[LAST_y] is 1).
- A CL conversion sequence precludes other conversions on the same ADC. If the CL is waiting (LISTCHR_*n*[NEXT_CH_WAIT_ON_TRIG_y] is 1), BCTU cannot route a non-CL conversion to that ADC (although that ADC remains available for normal and injected conversions).
- Two or more triggers for different ADCs may point to the same CL. This CL can drive parallel execution on the selected ADCs.

- If the Channel or CL Address (TRGCFG_x[CHANNEL_VALUE_OR_LADDR]) does not point to a valid list address, BCTU uses address 0 by default.
- When the CL pointer reaches the last available list address (LISTSTAR[LISTSZ] minus 1), the pointer wraps to list address 0.
- You can select more than one ADC per trigger using the ADC Select n (TRGCFG_x[ADC_SELn]) fields, but selecting more than one ADC is valid only for multiple parallel CL operation. If you select more than one ADC for a single-conversion trigger, BCTU ignores the trigger.

64.3.3 ADC conversion results access

64.3.3.1 Overview

The value of DATA_DEST determines access to ADC results.

- DATA_DEST = 000: ADC results are available in the ADC data registers.
- DATA_DEST = 001: ADC results are available in FIFO 1 Result Data (FIFO Result Data (FIFO1DR - FIFO2DR)). See Route conversion data to FIFO.
- DATA_DEST = 010: ADC results are available in FIFO 2 Result Data (FIFO Result Data (FIFO1DR - FIFO2DR)). See Route conversion data to FIFO.

You can access ADC results using flags and interrupt requests (see Access on interrupt request) or DMA (see Access on DMA request). Interrupt requests are based on Module Status (MSR) flags. ADC*n*DR provides the following information, with one register used for each ADC:

- The conversion results
- The channel that was used for the conversion
- The trigger input that initiated the conversion
- Whether a CL element generated the conversion
- Whether the data was the last conversion of a CL

64.3.3.2 Access on interrupt request

If interrupts are enabled for ADC*n* Result Data (MCR[IEN*n*] is 1) and DMAs are disabled for ADC*n* Result Data (MCR[DMA*n*] is 0), BCTU generates an interrupt request for each available conversion data in ADC*n*DR. Because BCTU executes conversions on different ADCs in parallel, conversion results may be available on multiple ADCs at the same time. New Data (MSR[NDATAn] is 1) indicates that new conversion data is available in the corresponding ADC Data Result (ADC*n*DR). To clear the New Data flag, write 1 to MSR[NDATAn_CLR].

64.3.3.3 Access on DMA request

If you enable DMA requests (MCR[DMA*n*]):

- You must select one channel for DMA per ADC.
- New conversion result data (ADC*n*DR) generates a DMA request.
- The DMA controller deasserts the DMA request and writes 1 to MSR[NDATAn_CLR] to clear the flag.

When an ADC generates new data (MSR[NDATAn]), BCTU processes the DMA request. See Clear a DMA request.

64.3.3.4 Clear an interrupt request

BCTU clears an interrupt request when:

- The number of valid entries is less than or equal to the FIFOn watermark level (FIFOWM[WM_FIFO*n*]).
- You write 1 to FIFO watermark interrupt status (FIFOERR[WM_INT_FIFO*n*]).

64.3.3.5 Clear a DMA request

BCTU clears a DMA request when:

- The number of valid entries is less than or equal to the FIFOn watermark level (FIFOWM[WM_FIFO*n*]).
- The DMA controller sends acknowledgment to BCTU.

64.3.3.6 Clear an error-generated interrupt request

Clear an interrupt generated because of an error condition by writing 1 to the appropriate error flag:

- FIFOERR[OVR_ERR_FIFO1]
- FIFOERR[UNDR_ERR_FIFO1]
- FIFOERR[OVR_ERR_FIFO2]
- FIFOERR[UNDR_ERR_FIFO2]

64.3.3.7 Conversion data overrun

If new ADC conversion data becomes available before you read the previous data (ADC*n*DR), then BCTU indicates a data overrun (MSR[DATAOVRn] is 1).

If you enable interrupts for the ADC (MCR[IEN*n*]), a data overrun also generates an interrupt request, regardless of the DMA request configuration.

To clear a data overrun (MSR[DATAOVRn]), write 1 to data overrun clear (MSR[DATAOVRn_CLR]). See Clear an error-generated interrupt request.

64.3.3.8 Route conversion data to FIFO

BCTU allows you to route multiple ADC conversion results to one of 2 internal data FIFOs. Any trigger source can route results to an available FIFO based on the trigger index. Each FIFO has a dedicated counter (FIFOCNTR[CNTR_FIFO*n*]) to indicate the number of entries in the FIFO.

Each FIFO also has its own result data register (FIFO*n*DR) that contains the following information about the last conversion result routed to the FIFO.

- ADC result data
- Trigger source index
- ADC number
- ADC channel number

64.3.3.9 Read ADC result from FIFO

Each FIFO has DMA and interrupt interfaces. The DMA interface reads data. The interrupt interface indicates watermark and error conditions:

- An overrun error (FIFOERR[OVR_ERR_FIFO*n*]) occurs when you attempt to write data to an already full FIFO.
- An underrun error (FIFOERR[UNDR_ERR_FIFO*n*]) occurs when you attempt to read data from an empty FIFO.
- BCTU asserts a watermark interrupt request if the number of valid FIFO entries exceeds the watermark level (FIFOWM[WM_FIFO*n*]).

You can read a FIFO:

- by enabling DMA requests (FIFOCR[DMA_EN_FIFO*n*])
- by polling the FIFO full status (FIFOSR[FULL_FIFO*n*])

See Configuration of DMA and interrupt behavior.

64.3.3.10 Configuration of DMA and interrupt behavior

Table 425. Configuration of DMA and interrupt behavior

FIFOCR[DMA_EN_FIFO*n*]	FIFOCR[IEN_FIFO*n*]	Condition
0	0	No DMA request; no interrupt request
0	1	Interrupt request when FIFO exceeds watermark level; interrupt request on error
1	0	DMA request when FIFO exceeds watermark level
1	1	DMA request when FIFO exceeds watermark level; interrupt request on error

64.3.4 Multiple parallel conversions (MPC)

64.3.4.1 Overview

The MPC mechanism allows you to execute a series of simultaneous ADC conversions by selecting more than one ADC for any CL-configured trigger.

64.3.4.2 Configure MPC

To configure a CL to perform MPC:

1. Select CL for the trigger resolution (write 1 to TRGCFG_*m*[TRS]).
2. Select more than one ADC for the trigger (TRGCFG_*m*[ADC_SEL*n*]).
3. Specify the CL address for the first CL element (TRGCFG_*m*[CHANNEL_VALUE_OR_LADDR]).

64.3.4.3 CL address selection for MPC

When the trigger input for the CL asserts, BCTU uses the channel or list address (CHANNEL_VALUE_OR_LADDR) as an index into the CL for the lowest numbered ADC selected (ADC_SEL*n*), and each subsequent element of the CL for each remaining selected ADC. The set of selected ADC channels perform their conversions in parallel. When that set of conversions is complete, the next set of CL elements determine the next set of ADC channels to perform parallel conversions, as illustrated in:

- Example: MPC with two ADCs
- Example: MPC with three ADCs

64.3.4.4 Example: MPC with two ADCs

The following figure shows the example of MPC with two ADCs.

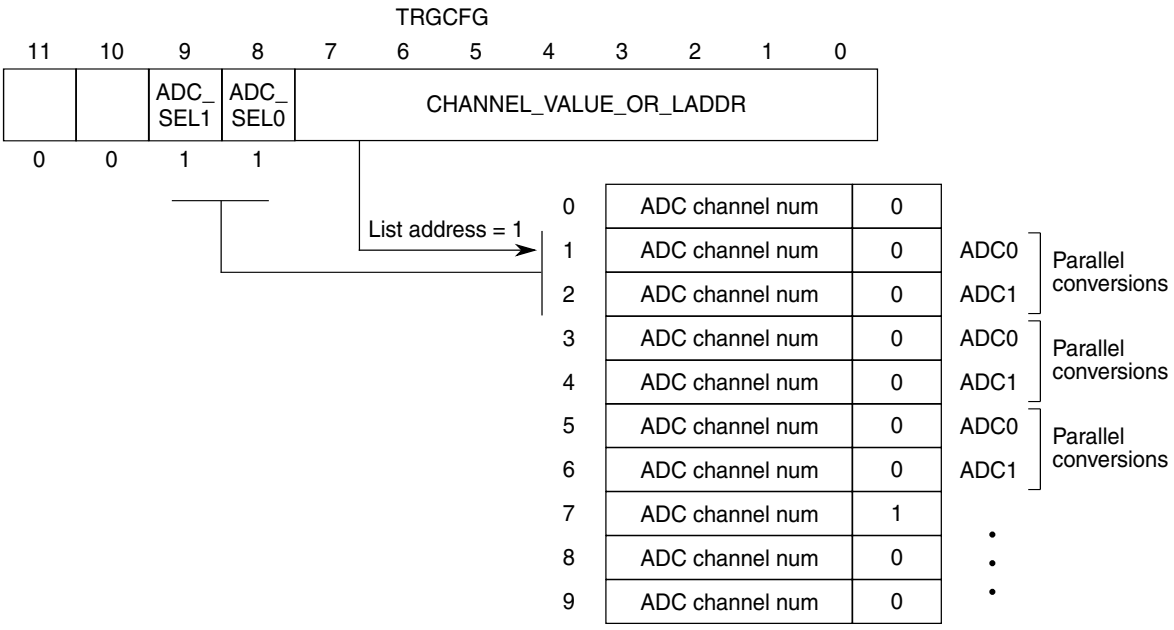


Figure 375. Example: MPC with two ADCs

64.3.4.5 Example: MPC with three ADCs

The following figure shows the example of MPC with three ADCs.

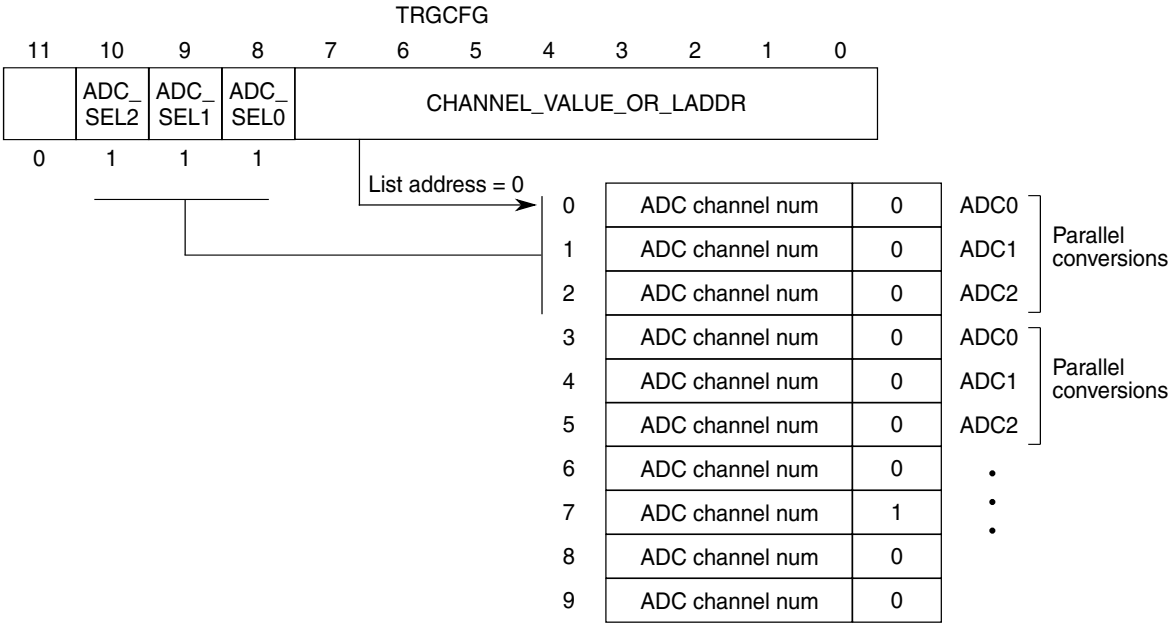


Figure 376. Example: MPC with three ADCs

64.3.4.6 MPC priority rules

MPC follows the same priority rules as for single conversions, which is based on the trigger input number: the lower the trigger, the higher the priority. However, MPC cannot start until all ADCs selected for the CL are available; that is, they are not currently

engaged in conversion. For example, if a trigger input asserts for a CL using ADC0 and ADC1, BCTU does not initiate the CL until ADC0 and ADC1 are both available.

Also, if ADC0 is available but there is a higher priority trigger for ADC1, the MPC does not start. If trigger inputs assert for two MPCs that have at least one ADC in common, the lower trigger number has higher priority. A priority conflict occurs only when separate requested CLs have at least one ADC in common. If the MPCs use different ADCs, there is no conflict and the MPCs execute at the same time.

64.3.5 Modes

64.3.5.1 Module Disable (Low Power state)

Module disable (MCR[MDIS]) stops the BCTU clock, resulting in reduced power consumption.

When you disable BCTU (write 1 to MDIS), it:

- Waits for the completion of any in-progress conversions before stopping the clock.
- Ignores all trigger inputs.
- Does not trigger any ADC conversion requests.
- Maintains any asserted trigger inputs, which you can clear if necessary.

When you re-enable BCTU (write 0 to MDIS) and global triggers are enabled (MCR[GTRGEN]), BCTU processes all active asserted trigger inputs and sends conversion requests to ADCs according to the priority scheme.

64.3.6 Clocking

This module has two clock ports.

1. ipg_clk_bctu - is the main clock of complete BCTU IP.
2. ipg_clk_s - is the gated IPG clock to be used in BCTU registers only.

64.3.7 Software reset

You can reinitialize all BCTU registers to their reset values with software reset (MCR[Software_Reset]).

Software reset does not clear any active trigger inputs, but it does put the ADC interface into the Reset state. Therefore, to avoid data incoherency, you must terminate any BCTU-triggered conversions or wait until they are complete before asserting software reset.

64.3.8 Interrupts and DMA requests

Table 426. Interrupts and DMA requests

Event	Enable interrupt	Enable DMA request	Description
Trigger Flag	MCR[TRGEN]	—	Interrupt request on assertion of trigger flag (MSR[TRGF])
ADC Data Ready	MCR[IEN*n*]	MCR[DMA*n*]	Interrupt or DMA request when ADC result data is available (ADC*n*DR)
CL Last	MCR[LIST_IEN]	—	Interrupt request when the last conversion of any CL executes
FIFO Watermark	FIFOCR[IEN_FIFO*n*]	FIFOCR[DMA_EN_FIFO*n*]	Interrupt or DMA request when the number of valid FIFO entries exceeds the FIFO

64.4 BCTU register descriptions

All BCTU registers are 32 bits wide. 8-bit write operations are allowed. The BCTU module does not include any coherence mechanism, so application software must ensure coherency.

NOTE

- You must always write 0 to reserved writable registers and fields. Read accesses of reserved registers and fields do not return meaningful data.
- For registers with write protection, if protection is active and you attempt to write to those registers, register content is unaffected and BCTU issues a transfer error.

64.4.1 BCTU memory map

BCTU base address: 4008_4000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Module Configuration (MCR)	32	RW	0000_0000h
8h	Module Status (MSR)	32	RW	0000_0000h
18h - 134h	Trigger Configuration (TRGCFG_0 - TRGCFG_71)	32	RW	0000_0000h
228h	Write Protection (WRPROT)	32	RW	0000_0000h
22Ch	Software Trigger 1 (SFTRGR1)	32	RW	0000_0000h
230h	Software Trigger 2 (SFTRGR2)	32	RW	0000_0000h
234h	Software Trigger 3 (SFTRGR3)	32	RW	0000_0000h
23Ch - 244h	ADCn Result Data (ADC0DR - ADC2DR)	32	R	0000_0000h
24Ch	CL Size Status (LISTSTAR)	32	R	0000_0020h
250h - 28Ch	CL Channel Address (LISTCHR_0 - LISTCHR_15)	32	RW	0000_0000h
450h - 454h	FIFO Result Data (FIFO1DR - FIFO2DR)	32	R	0000_0000h
460h	FIFO Control (FIFOCR)	32	RW	0000_0000h
464h	FIFO Watermark Configuration (FIFOWM)	32	RW	0000_0000h
468h	FIFO Error/Status (FIFOERR)	32	RW	0000_0000h
46Ch	FIFO Status (FIFOSR)	32	R	0000_0000h
470h	FIFO Counter (FIFOCNTR)	32	R	0000_0000h

64.4.2 Module Configuration (MCR)

Offset

Register	Offset
MCR	0h

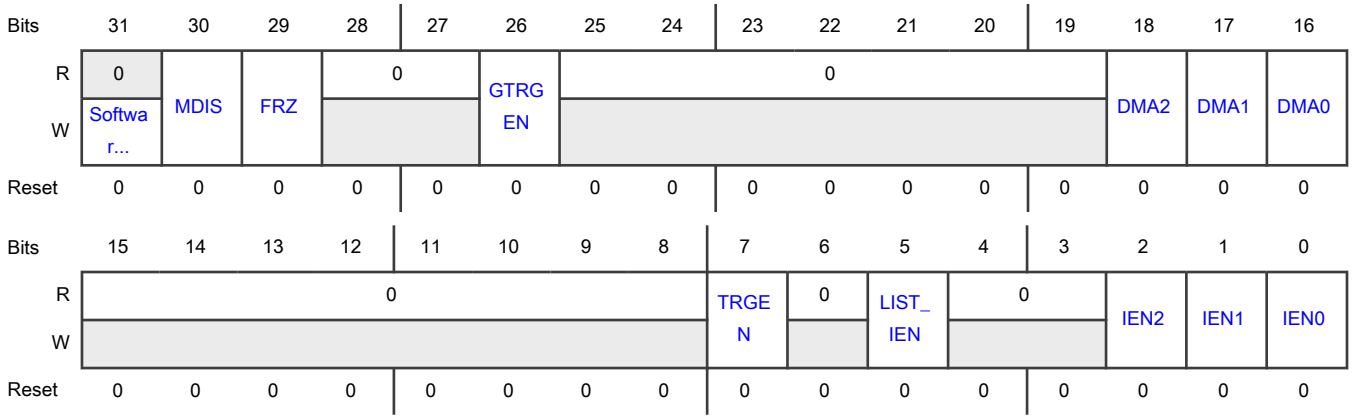
Function

Configures the following aspects of BCTU behavior:

- Low power
- Debug
- Trigger control
- DMA control
- Interrupt control

Access: User read/write

Diagram



Fields

Field	Function
31 Software_Reset	Software Reset Asserts soft reset, which initializes all BCTU registers and internal state machines to their reset values. 0b - Deasserts 1b - Asserts
30 MDIS	Module Disable Disables BCTU by putting it in Low Power state (see Module Disable (Low Power state)). 0b - Enable (normal operation) 1b - Disable (low-power operation)
29 FRZ	Debug Freeze Disables all hardware trigger inputs but leaves the software trigger enabled. Debug Freeze isolates BCTU from external triggers and allows you to manually trigger a conversion and read the conversion result. 0b - Disables 1b - Enables

Table continues on the next page...

Table continued from the previous page...

Field	Function
28-27 —	Reserved
26 GTRGEN	Global Trigger Enable Enables all individually enabled trigger inputs at the same time. For any given input trigger to be active, you must both enable the individual trigger (TRGCFG_*n*[TRIGEN]) and Global Trigger Enable. See Configure all triggers. 0b - Disable 1b - Enable
25-19 —	Reserved
18-16 DMAn	Enable ADCnDR DMA Enables DMA operation when new data is available in ADC*n*DR. 0b - Disable 1b - Enable
15-8 —	Reserved
7 TRGEN	Trigger Interrupt Request Enable Enables an interrupt request on a Trigger Flag (MSR[TRGF]). 0b - Disable 1b - Enable
6 —	Reserved
5 LIST_IEN	CL Interrupt Enable Enables an interrupt request for the last conversion in a CL (both ADC*n*DR[LIST] and ADC*n*DR[LAST] are 1). 0b - Disable 1b - Enable
4-3 —	Reserved
2-0 IENn	Interrupt Enable For ADCnDR

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Enables an interrupt request when BCTU writes a new conversion result to ADC Data (ADC*n*DR). Also enables an interrupt request on an overrun in ADC*n*DR, which indicates the previous data has been overwritten by a new conversion result. See also MSR[NDATA*n*] and MSR[DATAOVR*n*]]. Enabling ADC*n*DR DMA (DMA*n* field) disables the interrupt for new data in ADC*n*DR, but the interrupt for ADC*n*DR data overrun remains enabled. 0b - Disable 1b - Enable

64.4.3 Module Status (MSR)

Offset

Register	Offset
MSR	8h

Function

Indicates the status of various BCTU functions, including:

- Whether conversion data is available for each ADC
- CL operation, such as the last conversion command executed in a CL
- Whether BCTU triggered any of the ADCs

All flag fields become 0 upon assertion of a low-power (software) or clock-stop (hardware) request. You can turn off individual flags by writing 1 to the field.

Access: User read/write

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0				0	0	0	0	0	0	0	0	0	0	0
W	TRGF_CLR					LIST2_L...	LIST1_L...	LIST0_L...		DATAOVR...	DATAOVR...	DATAOVR...		NDATA2_...	NDATA1_...	NDATA0_...
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TRGF	0				LIST2_L...	LIST1_L...	LIST0_L...	0	DATAOVR2	DATAOVR1	DATAOVR0	0	NDATA2	NDATA1	NDATA0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 TRGF_CLR	TRGF Clear Changes Trigger Flag (TRGF) to 0. 0b - No action 1b - Changes to 0
30-27 —	Reserved
26-24 LISTn_Last_CLR	CL n Last Clear Changes CL *n* Last (LIST*n*_Last) to 0. 0b - No action 1b - Changes to 0
23 —	Reserved
22-20 DATAOVRn_CLR	DATAOVRn Clear Changes Data Overrun *n* (DATAOVR*n*) to 0. 0b - No action 1b - Changes to 0
19 —	Reserved
18-16 NDATAn_CLR	New Data Clear Changes New Data *n* (NDATA*n*) to 0. 0b - No action 1b - Changes to 0
15 TRGF	Trigger Flag Indicates at least one ADC was triggered. 0b - No ADC triggered 1b - An ADC was triggered
14-11 —	Reserved
10-8 LISTn_Last	CL n Last Conversion Indicates that ADCn has executed the last conversion in a CL.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Last conversion not complete 1b - Last conversion complete
7 —	Reserved
6-4 DATAOVRn	Data Overrun n Indicates that the data in ADCn has been overwritten with new data. 0b - Data not overwritten 1b - Data overwritten
3 —	Reserved
2-0 NDATAn	New Data n Indicates that new conversion data is available for ADCn. 0b - Not available 1b - Available

64.4.4 Trigger Configuration (TRGCFG_0 - TRGCFG_71)

Offset

For a = 0 to 71:

Register	Offset
TRGCFG_a	18h + (a × 4h)

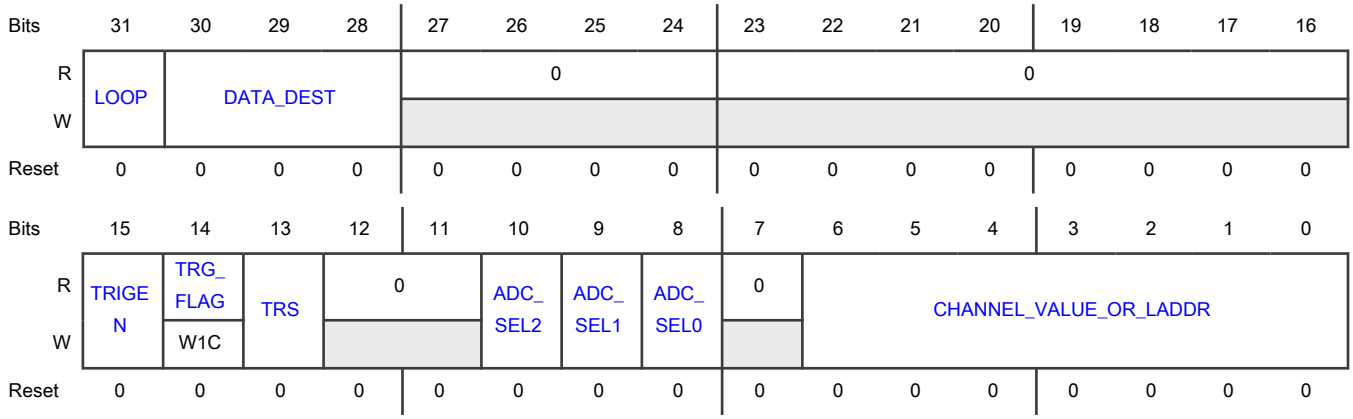
Function

Configures the corresponding trigger:

- Specifies the target ADC and ADC channel to use for conversion.
- Specifies whether to use a CL or a single ADC conversion command.

Access: User read/write

Diagram



Fields

Field	Function
31 LOOP	Loop Specifies that the current trigger executes in a loop; that is, the triggered conversion repeats until you disable Loop. Each loop-triggered conversion wins the priority evaluation among all pending triggers, and executes only if it is the highest-priority triggered conversion at the moment the conversion starts. 0b - Disable 1b - Enable
30-28 DATA_DEST	Data Destination Specifies the destination for storing the conversion results. You can route each ADC's conversion result to either of the following: • ADC-specific data registers • One of the available FIFOs BCTU routes conversion results to different destinations based on the trigger index that requested the conversion. You cannot duplicate the same data in more than one destination. 000b - ADC-specific data registers 001b - FIFO1 010b - FIFO2 011b - Reserved All other values are reserved.
27-24 —	Reserved
23-16 —	Reserved
15	Trigger Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
TRIGEN	Enables hardware input triggers to initiate ADC conversions. This field does not affect software triggers. 0b - Disable 1b - Enable
14 TRG_FLAG	Trigger Flag Indicates the trigger input is asserted and an ADC conversion is in progress. When the conversion completes, BCTU changes this field to 0. Writing 1 to this field causes the input trigger event to clear when the in-progress conversion completes, and prevents this trigger from initiating a new ADC conversion until the input trigger is asserted again. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No ongoing ADC conversion initiated 1b - Ongoing ADC conversion initiated When writing 0b - No action 1b - Changes to 0
13 TRS	Trigger Resolution Selects single conversion or CL conversions. 0b - Single conversion 1b - CL conversions
12-11 —	Reserved
10-8 ADC_SELn	ADC Select n Selects ADC *n* for conversion. 0b - Deselects 1b - Selects
7 —	Reserved
6-0 CHANNEL_VALUE_OR_LADDR	Channel or CL Address Depending on the value of TRS: • 0: Specifies the ADC channel for conversion • 1: Specifies the address of the first CL position to execute

64.4.5 Write Protection (WRPROT)

Offset

Register	Offset
WRPROT	228h

Function

Enables write protection at reset exit. To allow writes to protected registers, you must write a special code into PROTEC_CODE. There are two codes to remove protection: one that permanently disables the protection; one that disables protection during one write cycle to a protected register, after which the protection code returns to the initial (protected) value of 0000b.

Access: User read/write

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												PROTEC_CODE			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-4 —	Reserved
3-0 PROTEC_CODE	Protection Code Controls the protection of write-protected registers. **NOTE** You can write this field only in Normal mode (MCR[MDIS] is 0). 0000b - Enable protection 1001b - Disable protection for one write cycle 1010b - Disable protection permanently

64.4.6 Software Trigger 1 (SFTRGR1)

Offset

Register	Offset
SFTRGR1	22Ch

Function

Allows you to trigger conversions on the associated triggers.

To use a software trigger, you must first disable the corresponding hardware trigger (TRGCFG_x[TRIGEN]).

BCTU writes 0 to a software trigger field on completion of the triggered conversion.

Access: User read/write; this register is write protected by Write Protection (WRPROT). If protection is enabled, writes to this register generate a transfer error and do not modify its content.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SFTRG31	SFTRG30	SFTRG29	SFTRG28	SFTRG27	SFTRG26	SFTRG25	SFTRG24	SFTRG23	SFTRG22	SFTRG21	SFTRG20	SFTRG19	SFTRG18	SFTRG17	SFTRG16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SFTRG15	SFTRG14	SFTRG13	SFTRG12	SFTRG11	SFTRG10	SFTRG9	SFTRG8	SFTRG7	SFTRG6	SFTRG5	SFTRG4	SFTRG3	SFTRG2	SFTRG1	SFTRG0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 SFTRGm	Software Trigger m Starts an ADC conversion. 0b - No effect 1b - Trigger conversion

64.4.7 Software Trigger 2 (SFTRGR2)

Offset

Register	Offset
SFTRGR2	230h

Function

Allows you to trigger conversions on the associated triggers.

To use a software trigger, you must first disable the corresponding hardware trigger (TRGCFG_*n*[TRIGEN]).

BCTU writes 0 to a software trigger field on completion of the triggered conversion.

Access: User read/write; this register is write protected by Write Protection (WRPROT). If protection is enabled, writes to this register generate a transfer error and do not modify its content.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SFTRG63	SFTRG62	SFTRG61	SFTRG60	SFTRG59	SFTRG58	SFTRG57	SFTRG56	SFTRG55	SFTRG54	SFTRG53	SFTRG52	SFTRG51	SFTRG50	SFTRG49	SFTRG48
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SFTRG47	SFTRG46	SFTRG45	SFTRG44	SFTRG43	SFTRG42	SFTRG41	SFTRG40	SFTRG39	SFTRG38	SFTRG37	SFTRG36	SFTRG35	SFTRG34	SFTRG33	SFTRG32
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 SFTRGm	Software Trigger m Starts an ADC conversion. 0b - No effect 1b - Trigger conversion

64.4.8 Software Trigger 3 (SFTRGR3)

Offset

Register	Offset
SFTRGR3	234h

Function

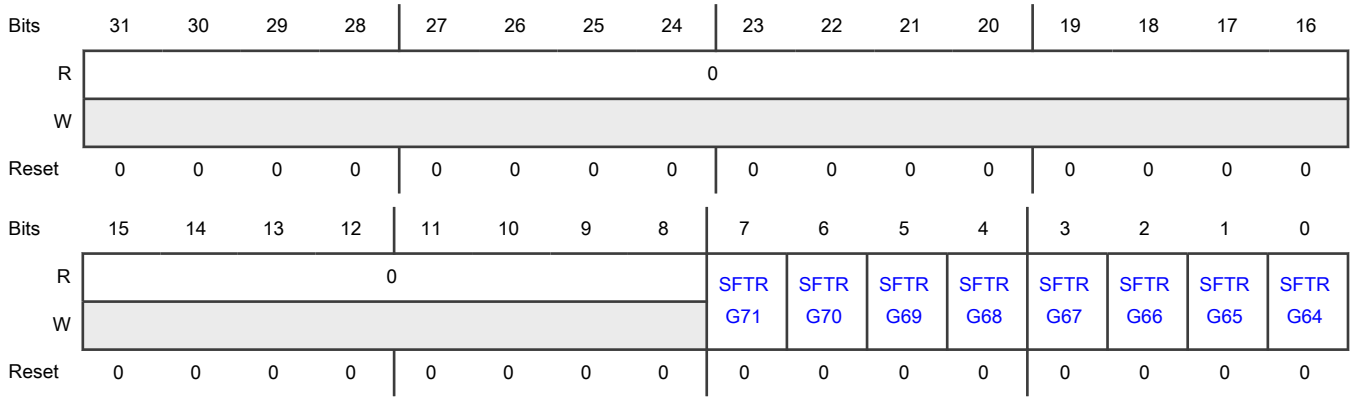
Allows you to trigger conversions on the associated triggers.

To use a software trigger, you must first disable the corresponding hardware trigger (TRGCFG_*n*[TRIGEN]).

BCTU writes 0 to a software trigger field on completion of the triggered conversion.

Access: User read/write; this register is write protected by Write Protection (WRPROT). If protection is enabled, writes to this register generate a transfer error and do not modify its content.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 SFTRGm	Software Trigger m Starts an ADC conversion. 0b - No effect 1b - Trigger conversion

64.4.9 ADCn Result Data (ADC0DR - ADC2DR)

Offset

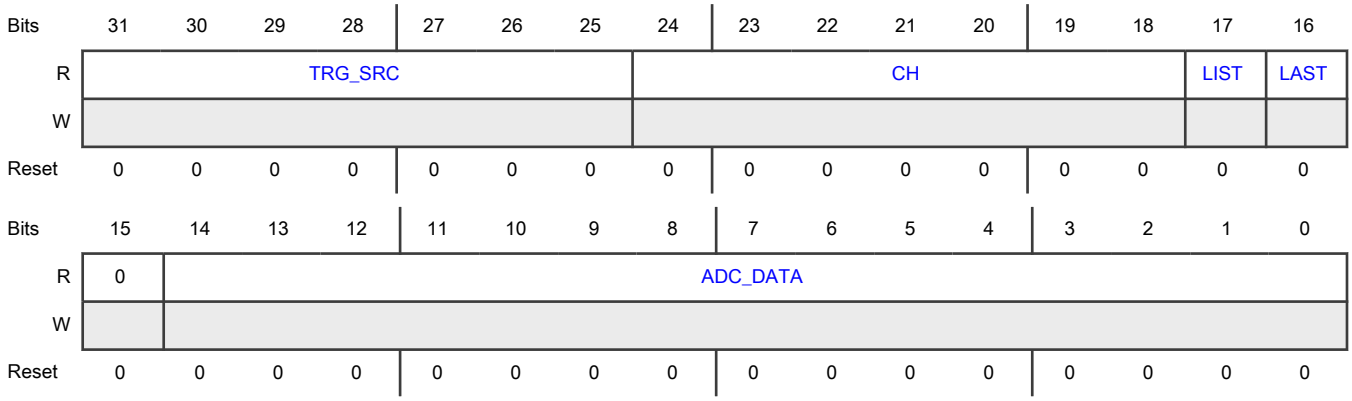
Register	Offset
ADC0DR	23Ch
ADC1DR	240h
ADC2DR	244h

Function

Provides ADC conversion result data. Also indicates the ADC channel used in that conversion, the trigger source that initiated it, and whether conversion was initiated from a CL.

Access: User read

Diagram



Fields

Field	Function
31-25 TRG_SRC	Trigger Source Contains the trigger number used to trigger the conversion.
24-18 CH	Channel Contains the ADC channel used for the conversion.
17 LIST	List Indicates whether the conversion was part of a CL or a single conversion trigger. 0b - Single conversion 1b - CL
16 LAST	Last For CL conversions, indicates this conversion result is the last conversion of the CL. 0b - Not the last conversion of a CL, or not a CL conversion 1b - Last conversion of a CL
15 —	Reserved
14-0 ADC_DATA	ADC Data Contains the data from the conversion.

64.4.10 CL Size Status (LISTSTAR)

Offset

Register	Offset
LISTSTAR	24Ch

Function

Indicates the size of the CL. Each element of the CL contains the channel number used in one conversion.

Access: User read

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								LISTSZ							
W																
Reset	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

Fields

Field	Function
31-8 —	Reserved
7-0 LISTSZ	CL Size Indicates the size of the CL in number of elements.

64.4.11 CL Channel Address (LISTCHR_0 - LISTCHR_15)

Offset

For m = 0 to 15:

Register	Offset
LISTCHR_m	250h + (m × 4h)

Function

Configures these attributes for two CL elements:

- ADC channel number to use for conversion
- Whether the element is the last element of a CL

Access: User read/write

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LAST_y	NEXT_CH...	0		0				0	ADC_CH_y						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LAST_y_...	NEXT_CH...	0		0				0	ADC_CHL_y_plus_1						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 LAST_y	Last Channel Specifies this element as the last channel in the CL. 0b - Not last 1b - Last channel in CL
30 NEXT_CH_WAIT_ON_TRIG_y	Next Channel Wait For Trigger Instructs CL execution to stop after executing the current ADC channel. Execution begins again when the same trigger, which started the CL, reoccurs. **NOTE** If this is the last channel in the CL (LAST_y), you must write 0 to this field. 0b - CL executes continuously 1b - CL stops executing
29-28 —	Reserved
27-24 —	Reserved
23 —	Reserved
22-16 ADC_CH_y	ADC Channel Selection Specifies the ADC channel to use for the *y* element of the CL, where *y* equals (2 x *m*).
15 LAST_y_plus_1	Last Channel Plus 1 Specifies this element as the next-to-last channel in the CL.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Not next-to-last 1b - Next-to-last channel in CL
14 NEXT_CH_WAIT_ON_TRIG_y_plus_1	Next Channel Wait For Trigger Plus 1 Instructs CL execution to stop after executing the current ADC channel. Execution begins again when the same trigger, which started the CL, reoccurs. **NOTE** If this is the last channel in the CL (LAST_y_plus_1), you must write 0 to this field. 0b - CL executes continuously 1b - CL stops executing
13-12 —	Reserved
11-8 —	Reserved
7 —	Reserved
6-0 ADC_CHL_y_plus_1	ADC Channel Selection Plus 1 Specifies the ADC channel to use for the (*y* + 1) element of the CL, where *y* equals (2 x *m*).

64.4.12 FIFO Result Data (FIFO1DR - FIFO2DR)

Offset

Register	Offset
FIFO1DR	450h
FIFO2DR	454h

Function

Contains the ADC conversion result and information about the conversion:

- ADC channel
- Trigger source
- ADC number

Access: User read

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TRG_SRC								CH						ADC_NUM	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	ADC_DATA														
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-25 TRG_SRC	Trigger Source Indicates the trigger number used to trigger the conversion.
24-18 CH	Channel Indicates the ADC channel used for the conversion.
17-16 ADC_NUM	ADC Number Indicates the ADC used for conversion. 00b - ADC 0 01b - ADC 1 10b - ADC 2 11b - ADC 3
15 —	Reserved
14-0 ADC_DATA	ADC Data Contains the conversion data.

64.4.13 FIFO Control (FIFOCR)

Offset

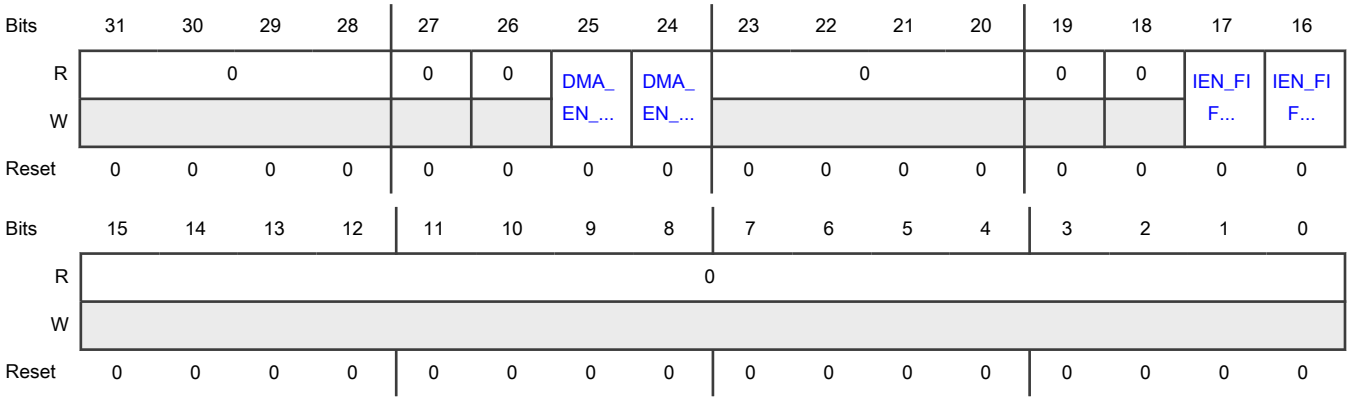
Register	Offset
FIFOCR	460h

Function

Provides control of FIFO-specific DMA, interrupts, and error interrupts.

Access: User read/write

Diagram



Fields

Field	Function
31-28 —	Reserved
27 —	Reserved
26 —	Reserved
25-24 DMA_EN_FIFOn	FIFOn DMA Enable Enables a DMA request when the number of valid FIFO entries is greater than the watermark level for that FIFO. 0b - Disable 1b - Enable
23-20 —	Reserved
19 —	Reserved
18 —	Reserved
17-16 IEN_FIFOn	FIFOn Interrupt Enable Enables an interrupt request if the number of valid FIFO entries is greater than the watermark level for the FIFO.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disable 1b - Enable
15-0 —	Reserved

64.4.14 FIFO Watermark Configuration (FIFOWM)

Offset

Register	Offset
FIFOWM	464h

Function

Specifies the FIFO watermark levels. If the number of active FIFO entries exceeds the watermark level, a DMA or interrupt request can be generated. See FIFO Control (FIFOCR).

Access: User read/write

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				WM_FIFO2				0				WM_FIFO1			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-16 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-11 —	Reserved
10-8 WM_FIFO2	FIFO2 Watermark Level Specifies the watermark level for FIFO2.
7-4 —	Reserved
3-0 WM_FIFO1	FIFO1 Watermark Level Specifies the watermark level for FIFO1.

64.4.15 FIFO Error/Status (FIFOERR)

Offset

Register	Offset
FIFOERR	468h

Function

Allows you to clear FIFO error and status flags. Each FIFO has its own error and status flags that can be configured to generate an interrupt. FIFO Status (FIFOSR) maintains the status of these flags. BCTU writes a 1 to these flags, and you must then clear them using FIFOERR.

Access: User read/write

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		0		UNDR_ER...	OVR_ERR...	UNDR_ER...	OVR_ERR...	0				0		WM_INT_...	WM_INT_...
W					W1C	W1C	W1C	W1C							W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29-28 —	Reserved
27 UNDR_ERR_FIFO2	Underrun Error Flag Indicates you have attempted to read from an empty FIFO. 0b - No underrun 1b - Underrun
26 OVR_ERR_FIFO2	Overrun Error Flag Indicates you have attempted to write to a full FIFO. 0b - No overrun 1b - Overrun
25 UNDR_ERR_FIFO1	Underrun Error Flag Indicates you have attempted to read from an empty FIFO. 0b - No underrun 1b - Underrun
24 OVR_ERR_FIFO1	Overrun Error Flag Indicates you have attempted to write to a full FIFO. 0b - No overrun 1b - Overrun
23-20 —	Reserved
19-18 —	Reserved
17-16 WM_INT_FIFOn	FIFO Watermark Interrupt Status Indicates the number of active entries in FIFOn exceeds the watermark level. 0b - Does not exceed watermark 1b - Exceeds watermark
15-0 —	Reserved

64.4.16 FIFO Status (FIFOSR)

Offset

Register	Offset
FIFOSR	46Ch

Function

Contains the FIFO-full flags.

Access: User read

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0														FULL_FI...	FULL_FI...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-2 —	Reserved
1-0 FULL_FIFOn	FIFO Full Indicates the FIFO is full. 0b - Not full 1b - Full

64.4.17 FIFO Counter (FIFOCNTR)

Offset

Register	Offset
FIFOCNTR	470h

Function

Indicates the number of active entries in each FIFO.

Access: User read

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				CNTR_FIFO2				0				CNTR_FIFO1			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-16 —	Reserved
15-12 —	Reserved
11-8 CNTR_FIFO2	FIFO2 Counter Indicates the number of active entries in FIFO2. See the chip-specific BCTU information for BCTU configuration.
7-5 —	Reserved
4-0 CNTR_FIFO1	FIFO1 Counter Indicates the number of active entries in FIFO1. See the chip-specific BCTU information for BCTU configuration.

64.5 Glossary

CL Conversion list. The set of ADC channels, configured through the CL Channel Address (LISTCHR_0 - LISTCHR_15) registers, used for performing multiple sequential conversions initiated by a single trigger.

MPC Multiple parallel conversions

PSI Parallel side interface

RWB Read/write bus

WDATA Write Data bus

Chapter 65
Trigger MUX (TRGMUX)

65.1 Chip-specific TRGMUX information

65.1.1 TRGMUX input output configuration and instances

This chip has one instance of TRGMUX module.

The device supports the triggering scheme between peripherals. For the supported trigger sources and destination, see the TRGMUX connectivity file attached to this document.

This device has 16 pads (SIUL2) mapped from TRGMUX inputs and TRGMUX outputs are mapped to the eMIOS channels, hence two timers channels can use a single pin of the device to do input capture.

While using TRGMUX, below points need to be taken care:

- User must ensure the minimum pulse of 100 ns on SIUL2 pads when using them as trigger source on TRGMUX.
- Pulses which are visible on pads depends on the pad type. Different pad supports different frequencies. For more details on pad types and their respective bandwidth, see section "IO signal table" in "Signal Multiplexing" chapter.
- End of conversion (EOC) signals of ADC modules are mapped on TRGMUX inputs as trigger sources. The EOC signal is asserted after ADC conversion regardless whether conversion is signaled by polling flags, interrupt or DMA transfer. The signal shall not be used to start injected conversion on same ADC channel as it will overwrite current result register.
- The minimum pulse length requirement is 1.5X of destination clock, so that it gets properly sampled at the destination IP connected to TRGMUX outputs, otherwise there are chances of missing the triggers generated from source. For example, the trigger generated from PAD for ADC conversion should be kept high/low for at least 1.5X of ADC clock so that it gets sampled in ADC clock domain. To ensure this, pulse strechers have been placed before some of the hardware modules mapped on TRGMUX outputs.
- Some PADS are being shared by both ADC and TRGMUX. It is recommended that trigger initiated from such PADS should not be used to trigger a conversion on the ADC channel mapped on same PAD. Failing to do this will cause congestion on same PAD.
- Out of all the pads mapped on TRGMUX, first four pads have glitch filters. For details, see the TRGMUX connectivity file attached to this document. The trigger pulse width should honour the pulse width requirement as per Glitch Filter specifications. The same signal can be observed at output pin if the pulse width of the input data signal is more than 400ns and no output signal should be observed if the pulse width of the input data signal is less than 20ns. A signal with a pulse width that is between 20ns and 400 ns should not be applied as the behavior is not guaranteed.
- Trigger outputs are grouped peripheral-wise and have a common lock bit based on TRGMUX REGx. For instance, normal_trigger, injected_trigger and external_sync of ADC_0 are grouped onto TRGMUX_REG0 and have a common lock bit.

65.1.2 Pulse strechers in TRGMUX

TRGMUX has some hardware modules on its input side running at faster clock than some of the IPs present on output side. For instance, eMIOS reload outputs running at 160 MHz can trigger LPI2C trigger input clocked at 40 MHz, in that case there is a high chance that trigger from eMIOS will be missed. Following Pulse strechers are added for the IPs on output side of TRGMUX.

Table 427. Pulse strechers in TRGMUX

Evaluation Parameter	ADC_0/1/2 external trigger to sync the start pulse	BCTU trigger 23/47/71	FlexIO trigger input_0/1/2/3[1]	LPI2C_0 Trigger input	LPSPI_0/1/2 Trigger input	CM7_0/1/2/3 RXEV
IP expects	Synchronized single cycle pulse	Synchronized pulse	IP requirement is to have 2 cycle pulse of flexio_clk	IP requirement is to have 2 cycle pulse of lpi2c_clk	IP requirement is to have 2 cycle pulse of lpspi_clk	Single cycle pulse
Frequency	CORE_CLK	CORE_CLK	CORE_CLK	AIPS_SLOW_CLK	LPSPI_0 - PLAT_AIPS_CLK LPSPI_1/2/3 - AIPS_SLOW_CLK	CORE_CLK
Inside IP	Used as combo signal	Flopped inside IP and clear after ADC conversion completed	Synchronized inside IP	Synchronized inside IP	Synchronized inside IP	-
At SoC	Since it an ASYNC signal and IP required synchronized single cycle pulse, so a pulse stretcher is added to convert pulse from any frequency domain into a single cycle pulse of CORE_CLK	Since slow IP such as LPCMP or from PAD can also trigger BCTU and BCTU wants that trigger should clear after the ADC conversion. So a pulse stretcher is added which convert any size of pulse to a single cycle pulse of BCTU clock (CORE_CLK)	Pulse stretcher is added to convert any pulse into a two cycle pulse of FLEXIO_CLK	Pulse stretcher is added to convert any pulse into a two cycle pulse of LPI2C_CLK	Pulse stretcher is added to convert any pulse into a two cycle pulse of LPSPI_CLK	Pulse stretcher is added to convert any pulse into a single cycle pulse of CORE_CLK
Output slot at which pulse stretcher is added	2, 6, 10	24, 25, 26	64, 65, 66, 67	84	88, 92, 96	156, 157, 158[2], 159[3]

1. These pulse stretchers are available in S32K314, S32K324 and S32K344 variants only.
2. CM7_2 is applicable for S32K358/S32K356/S32K348/S32K338/S32K328 only.
3. CM7_3 is applicable for S32K388/S32K389 only.

NOTE

The trigger outputs which have pulse stretcher before them, there should be atleast a gap of 5 cycle of destination clock for back to back trigger.

65.1.3 Trigger monitor

A trigger monitor is added before the TRGMUX inputs ADC12_0_EOC, ADC12_1_EOC and ADC12_2_EOC. You can configure the UDR registers in SIUL2 to select which trigger needs to propagate to the above TRGMUX inputs. By default after coming out of reset, all the enables are 0. Hence no trigger will be coming to these 3 TRGMUX inputs.

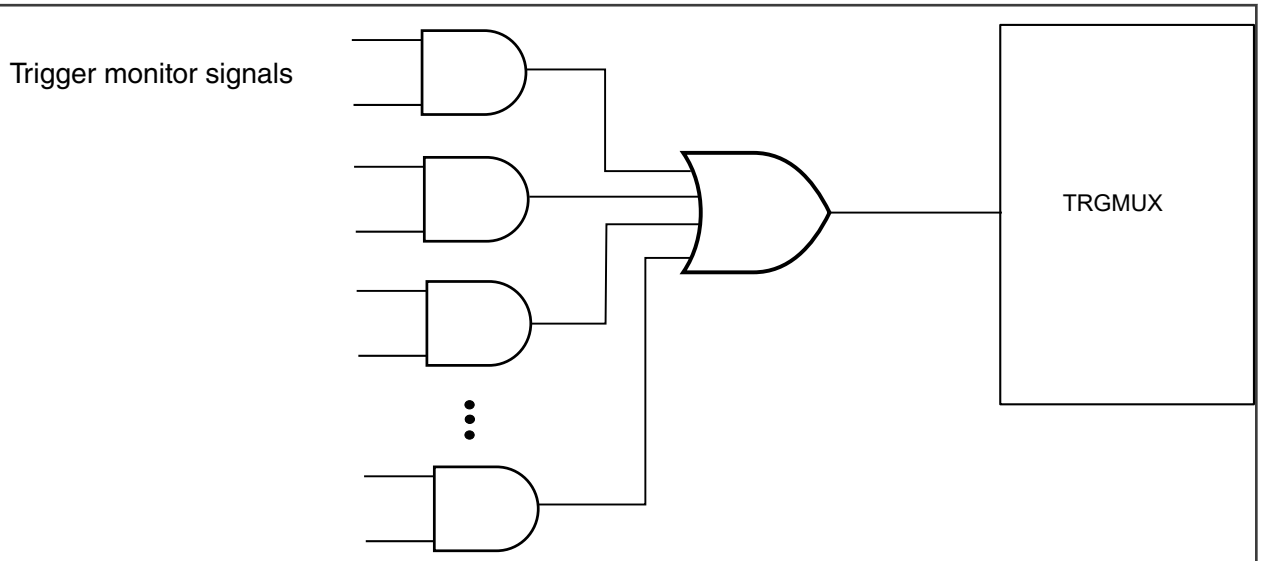


Figure 377. Trigger monitor

NOTE

Trigger monitor is not present in S32K314, S32K324 and S32K344.

65.2 Overview

TRGMUX allows you to configure the trigger inputs for various peripherals.

65.2.1 Block diagram

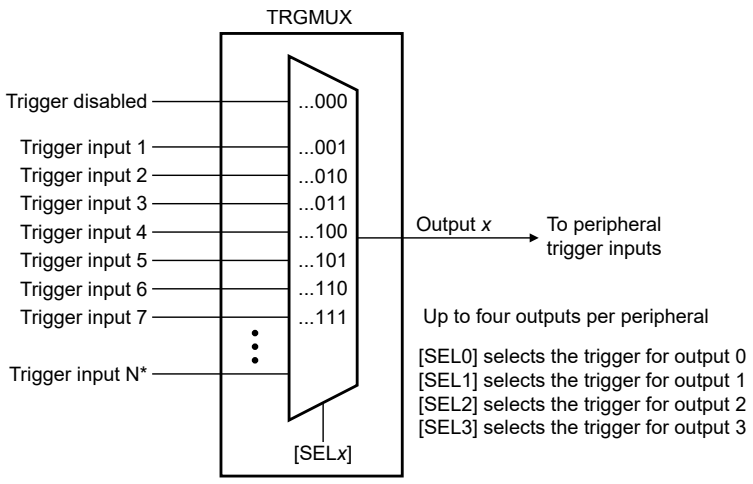


Figure 378. Block diagram

NOTE

Up to 255 trigger inputs may be available for SEL0, SEL1, and SEL2. For SEL3, up to 127 trigger inputs may be available. When the number of trigger inputs is 255, SEL3 is not available and becomes reserved. See the chip-specific TRGMUX information for details about trigger input and output configuration on this chip.

65.2.2 Features

- Configurable trigger sources for peripherals
- Dedicated TRGMUX register for each peripheral

65.3 Functional description

65.3.1 Clocking

This module has no clocking considerations.

65.3.2 Interrupts

This module has no interrupts.

65.4 External signals

This module has no external signals.

65.5 Initialization

This module does not require initialization.

65.6 TRGMUX register descriptions

65.6.1 TRGMUX memory map

You can only write to TRGMUX registers in Supervisor mode.

Table 428. Select bit fields

Field	Description
SEL*x*	Specifies the MUX select for the peripheral trigger inputs. Use this field to select the trigger sources for peripheral modules. 0h - LOGIC 0 (VSS) 1h - LOGIC 1 (VDD) 2h - ADC12_0_EOC 3h - ADC12_1_EOC 4h - ADC12_2_EOC 5h - LPCMP_0_COUT output 6h - LPCMP_1_COUT output 7h - LPCMP_2_COUT output 8h - eDMA_eDMA_0 DONE

Table continues on the next page...

Table 428. Select bit fields

Field	Description
	9h - eDMA_eDMA_1 DONE Ah - eDMA_eDMA_16 DONE Bh - eDMA_eDMA_17 DONE Ch - eMIOS_0_RELOAD_OUT_CH[23] Dh - eMIOS_0_RELOAD_OUT_CH[22] Eh - eMIOS_0_RELOAD_OUT_CH[8] Fh - eMIOS_0_RELOAD_OUT_CH[0] 10h - eMIOS_0_IPP_DO_eMIOS_CH[0] 11h - eMIOS_0_IPP_DO_eMIOS_CH[1] 12h - eMIOS_0_IPP_DO_eMIOS_CH[2] 13h - eMIOS_0_IPP_DO_eMIOS_CH[3] 14h - eMIOS_0_IPP_DO_eMIOS_CH[4] 15h - eMIOS_0_IPP_DO_eMIOS_CH[5] 16h - eMIOS_0_IPP_DO_eMIOS_CH[6] 17h - eMIOS_0_IPP_DO_eMIOS_CH[7] 18h - eMIOS_0_IPP_DO_eMIOS_CH[8] 19h - eMIOS_0_IPP_DO_eMIOS_CH[9] 1Ah - eMIOS_0_IPP_DO_eMIOS_CH[10] 1Bh - eMIOS_0_IPP_DO_eMIOS_CH[11] 1Ch - eMIOS_0_IPP_DO_eMIOS_CH[12] 1Dh - eMIOS_0_IPP_DO_eMIOS_CH[13] 1Eh - eMIOS_0_IPP_DO_eMIOS_CH[14] 1Fh - eMIOS_0_IPP_DO_eMIOS_CH[15] 20h - eMIOS_0_IPP_DO_eMIOS_CH[22] 21h - eMIOS_0_IPP_DO_eMIOS_CH[23] 22h - eMIOS_1_RELOAD_OUT_CH[23] 23h - eMIOS_1_RELOAD_OUT_CH[22] 24h - eMIOS_1_RELOAD_OUT_CH[8] 25h - eMIOS_1_RELOAD_OUT_CH[0] 26h - eMIOS_1_IPP_DO_eMIOS_CH[0] 27h - eMIOS_1_IPP_DO_eMIOS_CH[1] 28h - eMIOS_1_IPP_DO_eMIOS_CH[2] 29h - eMIOS_1_IPP_DO_eMIOS_CH[3] 2Ah - eMIOS_1_IPP_DO_eMIOS_CH[4]

Table continues on the next page...

Table 428. Select bit fields

Field	Description
	2Bh - eMIOS_1_IPP_DO_eMIOS_CH[5] 2Ch - eMIOS_1_IPP_DO_eMIOS_CH[6] 2Dh - eMIOS_1_IPP_DO_eMIOS_CH[7] 2Eh - eMIOS_1_IPP_DO_eMIOS_CH[8] 2Fh - eMIOS_1_IPP_DO_eMIOS_CH[9] 30h - eMIOS_1_IPP_DO_eMIOS_CH[10] 31h - eMIOS_1_IPP_DO_eMIOS_CH[11] 32h - eMIOS_1_IPP_DO_eMIOS_CH[12] 33h - eMIOS_1_IPP_DO_eMIOS_CH[13] 34h - eMIOS_1_IPP_DO_eMIOS_CH[14] 35h - eMIOS_1_IPP_DO_eMIOS_CH[15] 36h - eMIOS_1_IPP_DO_eMIOS_CH[22] 37h - eMIOS_1_IPP_DO_eMIOS_CH[23] 38h - FlexIO_External Output Trigger 0 39h - FlexIO_External Output Trigger 1 3Ah - FlexIO_External Output Trigger 2 3Bh - FlexIO_External Output Trigger 3 3Ch - SIUL_TRGMUX_IN0 3Dh - SIUL_TRGMUX_IN1 3Eh - SIUL_TRGMUX_IN2 3Fh - SIUL_TRGMUX_IN3 40h - SIUL_TRGMUX_IN4 41h - SIUL_TRGMUX_IN5 42h - SIUL_TRGMUX_IN6 43h - SIUL_TRGMUX_IN7 44h - SIUL_TRGMUX_IN8 45h - SIUL_TRGMUX_IN9 46h - SIUL_TRGMUX_IN10 47h - SIUL_TRGMUX_IN11 48h - SIUL_TRGMUX_IN12 49h - SIUL_TRGMUX_IN13 4Ah - SIUL_TRGMUX_IN14 4Bh - SIUL_TRGMUX_IN15 4Ch - LPI2C_0_Master trigger output

Table continues on the next page...

Table 428. Select bit fields

Field	Description
	4Dh - LPI2C_0_Slave trigger output 4Eh - LPSPI_0_End of frame trigger 4Fh - LPSPI_0_Receive data trigger 50h - LPSPI_1_End of frame trigger 51h - LPSPI_1_Receive data trigger 52h - LPSPI_2_End of frame trigger 53h - LPSPI_2_Receive data trigger 54h - LPUART_0_trg_txword 55h - LPUART_0_trg_rxword 56h - LPUART_0_trg_rxidle 57h - LPUART_1_trg_txword 58h - LPUART_1_trg_rxword 59h - LPUART_1_trg_rxidle 5Ah - LPUART_2_trg_txword 5Bh - LPUART_2_trg_rxword 5Ch - LPUART_2_trg_rxidle 5Dh - LCU_0_LC1_out_i1 5Eh - LCU_0_LC1_out_i2 5Fh - LCU_0_LC1_out_i3 60h - LCU_0_LC1_out_i4 61h - LCU_0_LC2_out_i1 62h - LCU_0_LC2_out_i2 63h - LCU_0_LC2_out_i3 64h - LCU_0_LC2_out_i4 65h - LCU_0_LC3_out_i1 66h - LCU_0_LC3_out_i2 67h - LCU_0_LC3_out_i3 68h - LCU_0_LC3_out_i4 69h - LCU_1_LC1_out_i1 6Ah - LCU_1_LC1_out_i2 6Bh - LCU_1_LC1_out_i3 6Ch - LCU_1_LC1_out_i4 6Dh - LCU_1_LC2_out_i1 6Eh - LCU_1_LC2_out_i2

Table continues on the next page...

Table 428. Select bit fields

Field	Description
	6Fh - LCU_1_LC2_out_i3 70h - LCU_1_LC2_out_i4 71h - LCU_1_LC3_out_i1 72h - LCU_1_LC3_out_i2 73h - LCU_1_LC3_out_i3 74h - LCU_1_LC3_out_i4 75h - PIT0_PIT0 CH0 76h - PIT0_PIT0 CH1 77h - PIT0_PIT0 CH2 78h - PIT0_PIT0 CH3 79h - PIT0_PIT0 CH4 RTI 7Ah - PIT1_PIT1 CH0 7Bh - PIT1_PIT1 CH1 7Ch - PIT1_PIT1 CH2 7Dh - PIT1_PIT1 CH3 7Eh - CM7_0_TXEV 7Fh - CM7_1_TXEV

TRGMUX base address: 4008_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	TRGMUX ADC12_0 (ADC12_0)	32	RW	0000_0000h
4h	TRGMUX ADC12_1 (ADC12_1)	32	RW	0000_0000h
8h	TRGMUX ADC12_2 (ADC12_2)	32	RW	0000_0000h
Ch	TRGMUX LPCMP_0 (LPCMP_0)	32	RW	0000_0000h
10h	TRGMUX LPCMP_1 (LPCMP_1)	32	RW	0000_0000h
14h	TRGMUX LPCMP_2 (LPCMP_2)	32	RW	0000_0000h
18h	TRGMUX BCTU (BCTU)	32	RW	0000_0000h
1Ch	TRGMUX eMIOS012_ODIS (eMIOS012_ODIS)	32	RW	0000_0000h
20h	TRGMUX eMIOS0_0 (eMIOS0_0)	32	RW	0000_0000h
24h	TRGMUX eMIOS0_1 (eMIOS0_1)	32	RW	0000_0000h
28h	TRGMUX eMIOS0_2 (eMIOS0_2)	32	RW	0000_0000h
2Ch	TRGMUX eMIOS0_3 (eMIOS0_3)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
30h	TRGMUX eMIOS1_0 (eMIOS1_0)	32	RW	0000_0000h
34h	TRGMUX eMIOS1_1 (eMIOS1_1)	32	RW	0000_0000h
38h	TRGMUX eMIOS1_2 (eMIOS1_2)	32	RW	0000_0000h
3Ch	TRGMUX eMIOS1_3 (eMIOS1_3)	32	RW	0000_0000h
40h	TRGMUX FlexIO (FlexIO)	32	RW	0000_0000h
44h	TRGMUX SIUL_OUT0 (SIUL_OUT0)	32	RW	0000_0000h
48h	TRGMUX SIUL_OUT1 (SIUL_OUT1)	32	RW	0000_0000h
4Ch	TRGMUX SIUL_OUT2 (SIUL_OUT2)	32	RW	0000_0000h
50h	TRGMUX SIUL_OUT3 (SIUL_OUT3)	32	RW	0000_0000h
54h	TRGMUX LPI2C_0 (LPI2C_0)	32	RW	0000_0000h
58h	TRGMUX LPSPI_0 (LPSPI_0)	32	RW	0000_0000h
5Ch	TRGMUX LPSPI_1 (LPSPI_1)	32	RW	0000_0000h
60h	TRGMUX LPSPI_2 (LPSPI_2)	32	RW	0000_0000h
64h	TRGMUX LPUART_0 (LPUART_0)	32	RW	0000_0000h
68h	TRGMUX LPUART_1 (LPUART_1)	32	RW	0000_0000h
6Ch	TRGMUX LPUART_2 (LPUART_2)	32	RW	0000_0000h
70h	TRGMUX LPUART_3 (LPUART_3)	32	RW	0000_0000h
74h	TRGMUX LCU0_SYNC (LCU0_SYNC)	32	RW	0000_0000h
78h	TRGMUX LCU0_FORCE (LCU0_FORCE)	32	RW	0000_0000h
7Ch	TRGMUX LCU0_0 (LCU0_0)	32	RW	0000_0000h
80h	TRGMUX LCU0_1 (LCU0_1)	32	RW	0000_0000h
84h	TRGMUX LCU0_2 (LCU0_2)	32	RW	0000_0000h
88h	TRGMUX LCU1_SYNC (LCU1_SYNC)	32	RW	0000_0000h
8Ch	TRGMUX LCU1_FORCE (LCU1_FORCE)	32	RW	0000_0000h
90h	TRGMUX LCU1_0 (LCU1_0)	32	RW	0000_0000h
94h	TRGMUX LCU1_1 (LCU1_1)	32	RW	0000_0000h
98h	TRGMUX LCU1_2 (LCU1_2)	32	RW	0000_0000h
9Ch	TRGMUX CM7_RXEV (CM7_RXEV)	32	RW	0000_0000h

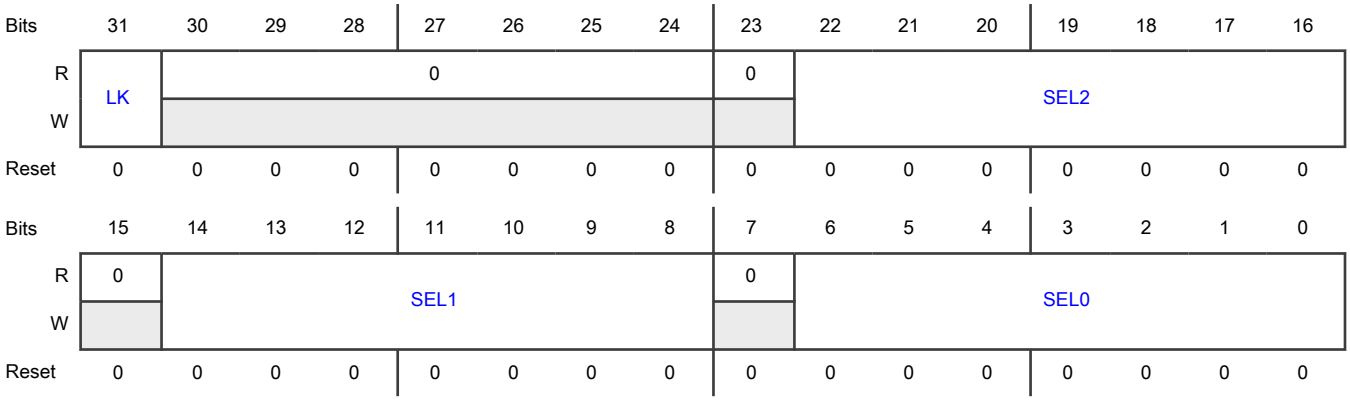
65.6.2 TRGMUX ADC12_0 (ADC12_0)

Offset

Register	Offset
ADC12_0	0h

Function
Configures the ADC12_0 module.

Diagram



Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable 1b - Register is not writable until the next system reset
30-24 —	Reserved
23 —	Reserved
22-16 SEL2	TRGMUX Source Select 2 Specifies the source select for output 2. See Table 428 for field values.
15 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
14-8 SEL1	TRGMUX Source Select 1 Specifies the source select for output 1. See Table 428 for field values.
7 —	Reserved
6-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 428 for field values.

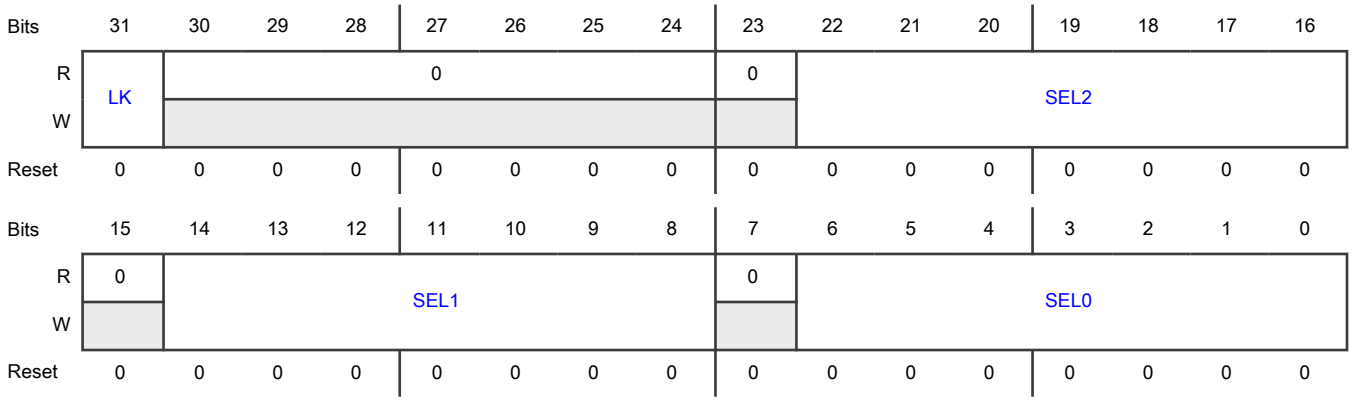
65.6.3 TRGMUX ADC12_1 (ADC12_1)

Offset

Register	Offset
ADC12_1	4h

Function
Configures the ADC12_1 module.

Diagram



Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Register is not writable until the next system reset
30-24 —	Reserved
23 —	Reserved
22-16 SEL2	TRGMUX Source Select 2 Specifies the source select for output 2. See Table 428 for field values.
15 —	Reserved
14-8 SEL1	TRGMUX Source Select 1 Specifies the source select for output 1. See Table 428 for field values.
7 —	Reserved
6-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 428 for field values.

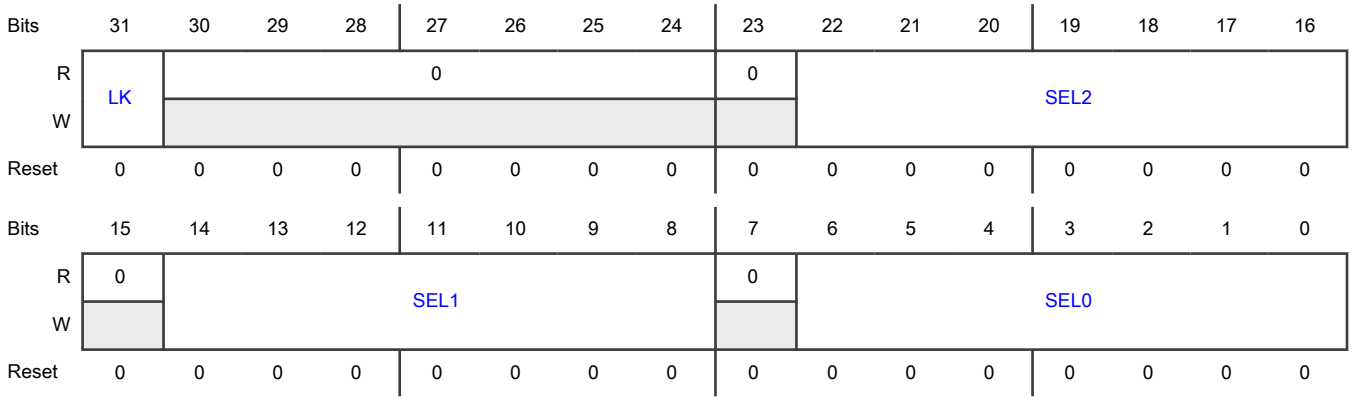
65.6.4 TRGMUX ADC12_2 (ADC12_2)

Offset

Register	Offset
ADC12_2	8h

Function
Configures the ADC12_2 module.

Diagram



Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable 1b - Register is not writable until the next system reset
30-24 —	Reserved
23 —	Reserved
22-16 SEL2	TRGMUX Source Select 2 Specifies the source select for output 2. See Table 428 for field values.
15 —	Reserved
14-8 SEL1	TRGMUX Source Select 1 Specifies the source select for output 1. See Table 428 for field values.
7 —	Reserved
6-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 428 for field values.

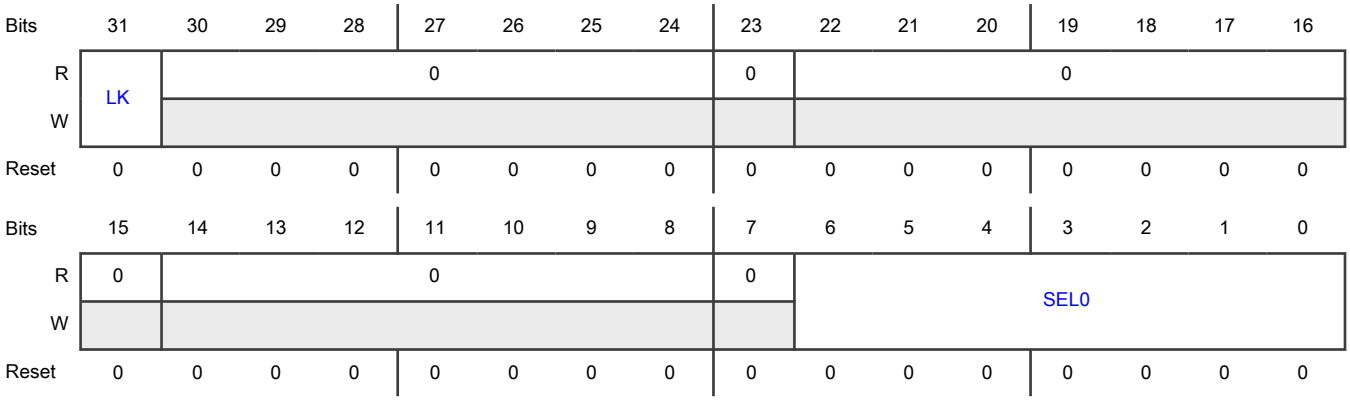
65.6.5 TRGMUX LPCMP_0 (LPCMP_0)

Offset

Register	Offset
LPCMP_0	Ch

Function
Configures the LPCMP_0 module.

Diagram



Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable 1b - Register is not writable until the next system reset
30-24 —	Reserved
23 —	Reserved
22-16 —	Reserved
15 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
14-8 —	Reserved
7 —	Reserved
6-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 428 for field values.

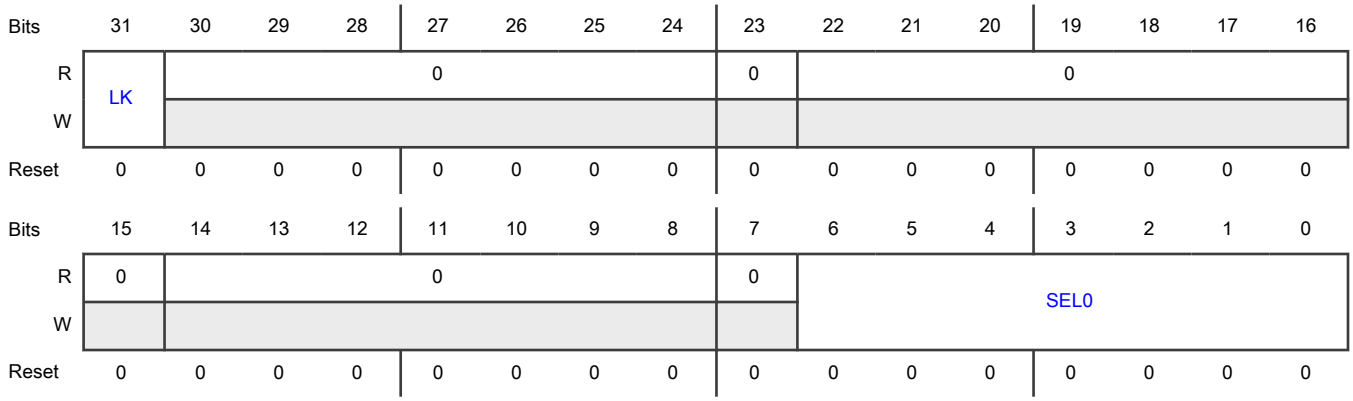
65.6.6 TRGMUX LPCMP_1 (LPCMP_1)

Offset

Register	Offset
LPCMP_1	10h

Function
Configures the LPCMP_1 module.

Diagram



Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Register is not writable until the next system reset
30-24 —	Reserved
23 —	Reserved
22-16 —	Reserved
15 —	Reserved
14-8 —	Reserved
7 —	Reserved
6-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 428 for field values.

65.6.7 TRGMUX LPCMP_2 (LPCMP_2)

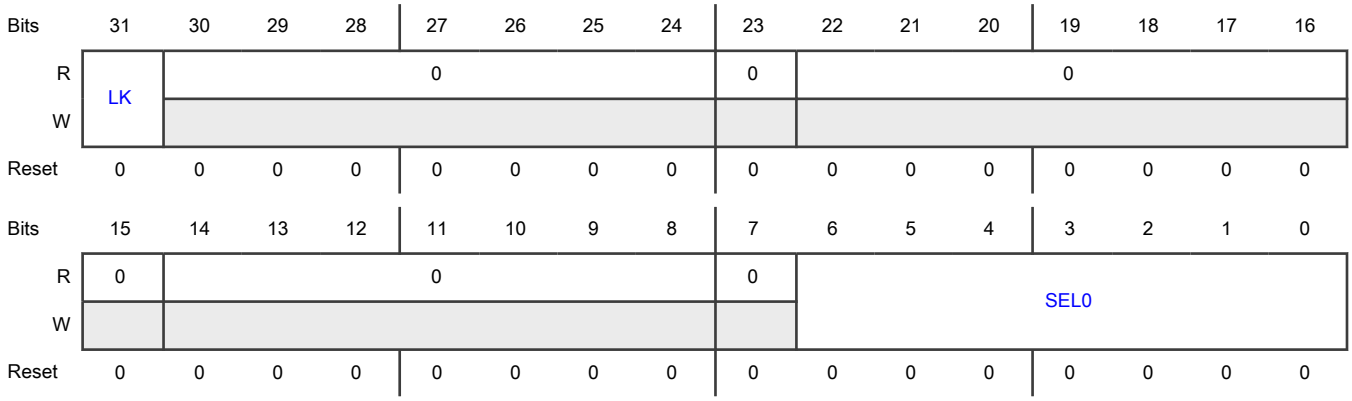
Offset

Register	Offset
LPCMP_2	14h

Function

Configures the LPCMP_2 module.

Diagram



Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable 1b - Register is not writable until the next system reset
30-24 —	Reserved
23 —	Reserved
22-16 —	Reserved
15 —	Reserved
14-8 —	Reserved
7 —	Reserved
6-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 428 for field values.

65.6.8 TRGMUX BCTU (BCTU)

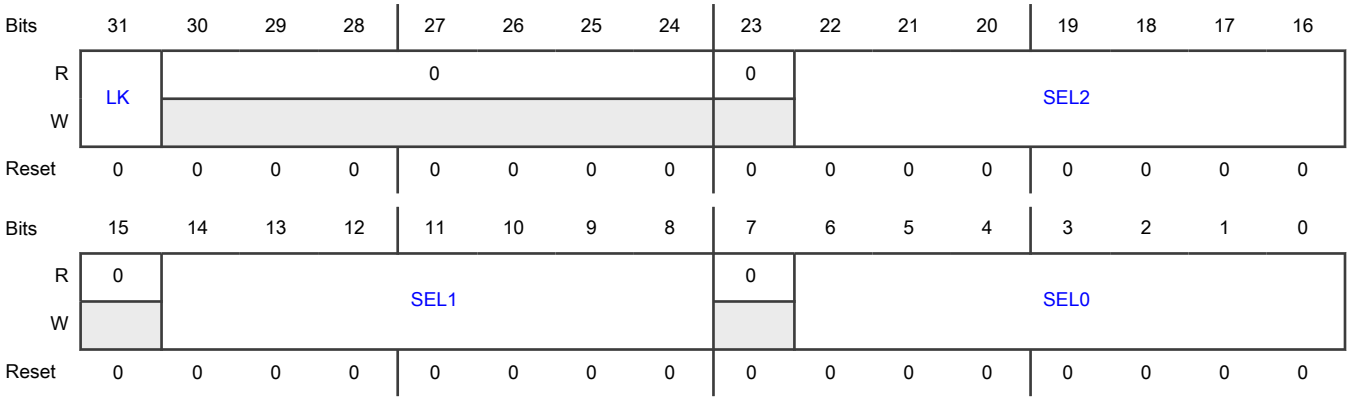
Offset

Register	Offset
BCTU	18h

Function

Configures the BCTU module.

Diagram



Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable 1b - Register is not writable until the next system reset
30-24 —	Reserved
23 —	Reserved
22-16 SEL2	TRGMUX Source Select 2 Specifies the source select for output 2. See Table 428 for field values.
15 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
14-8 SEL1	TRGMUX Source Select 1 Specifies the source select for output 1. See Table 428 for field values.
7 —	Reserved
6-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 428 for field values.

65.6.9 TRGMUX eMIOS012_ODIS (eMIOS012_ODIS)

Offset

Register	Offset
eMIOS012_ODIS	1Ch

Function
Configures the eMIOS012_ODIS module.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	SEL3							0	SEL2						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	SEL1							0	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Register is not writable until the next system reset
30-24 SEL3	TRGMUX Source Select 3 Specifies the source select for output 3. See Table 428 for field values.
23 —	Reserved
22-16 SEL2	TRGMUX Source Select 2 Specifies the source select for output 2. See Table 428 for field values.
15 —	Reserved
14-8 SEL1	TRGMUX Source Select 1 Specifies the source select for output 1. See Table 428 for field values.
7 —	Reserved
6-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 428 for field values.

65.6.10 TRGMUX eMIOS0_0 (eMIOS0_0)

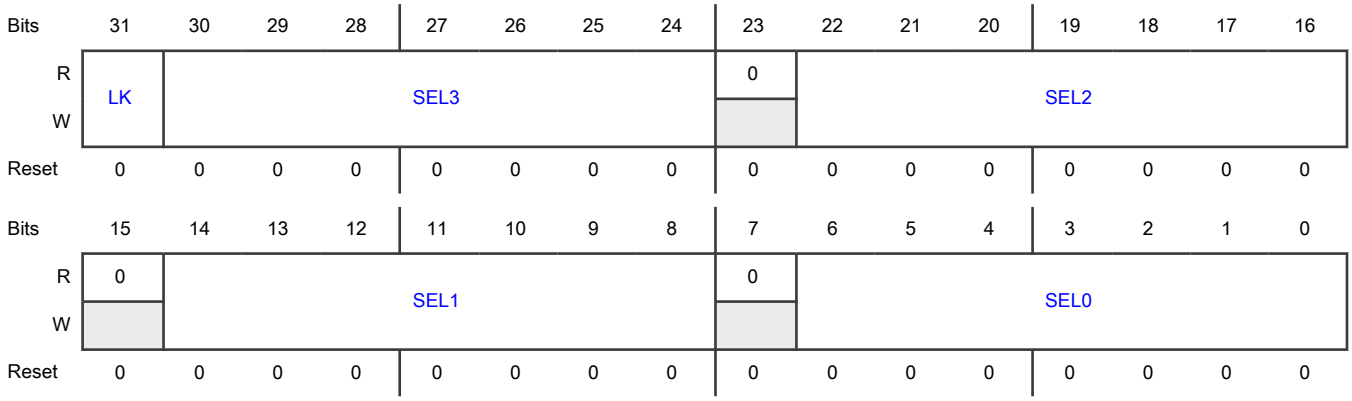
Offset

Register	Offset
eMIOS0_0	20h

Function

Configures the eMIOS0_0 module.

Diagram



Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable 1b - Register is not writable until the next system reset
30-24 SEL3	TRGMUX Source Select 3 Specifies the source select for output 3. See Table 428 for field values.
23 —	Reserved
22-16 SEL2	TRGMUX Source Select 2 Specifies the source select for output 2. See Table 428 for field values.
15 —	Reserved
14-8 SEL1	TRGMUX Source Select 1 Specifies the source select for output 1. See Table 428 for field values.
7 —	Reserved
6-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 428 for field values.

65.6.11 TRGMUX eMIOS0_1 (eMIOS0_1)

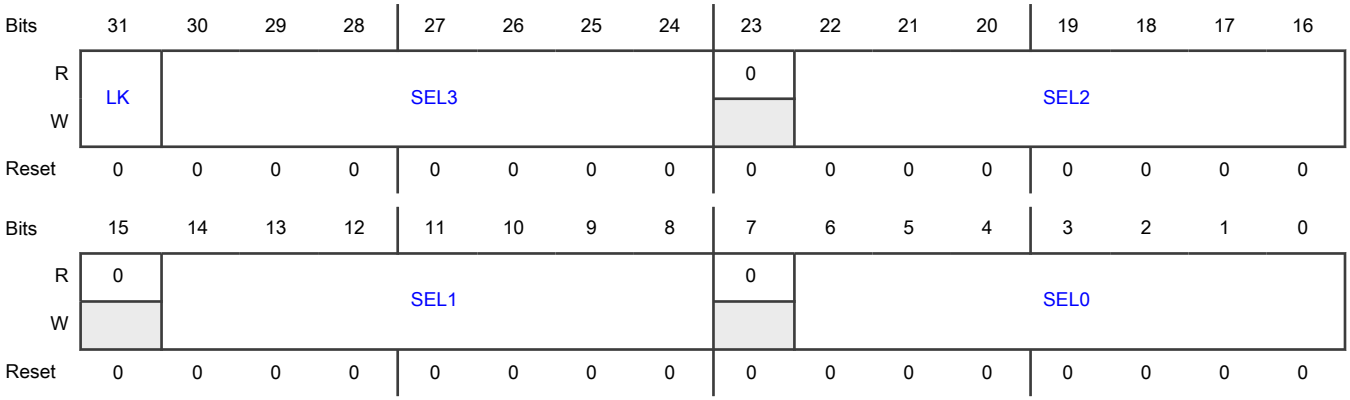
Offset

Register	Offset
eMIOS0_1	24h

Function

Configures the eMIOS0_1 module.

Diagram



Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable 1b - Register is not writable until the next system reset
30-24 SEL3	TRGMUX Source Select 3 Specifies the source select for output 3. See Table 428 for field values.
23 —	Reserved
22-16 SEL2	TRGMUX Source Select 2 Specifies the source select for output 2. See Table 428 for field values.
15 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
14-8 SEL1	TRGMUX Source Select 1 Specifies the source select for output 1. See Table 428 for field values.
7 —	Reserved
6-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 428 for field values.

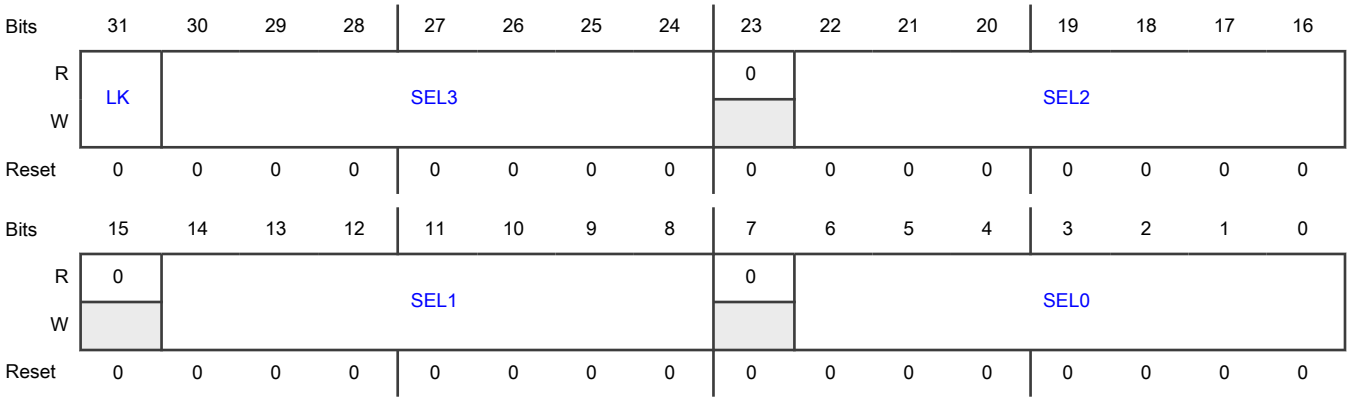
65.6.12 TRGMUX eMIOS0_2 (eMIOS0_2)

Offset

Register	Offset
eMIOS0_2	28h

Function
Configures the eMIOS0_2 module.

Diagram



Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Register is not writable until the next system reset
30-24 SEL3	TRGMUX Source Select 3 Specifies the source select for output 3. See Table 428 for field values.
23 —	Reserved
22-16 SEL2	TRGMUX Source Select 2 Specifies the source select for output 2. See Table 428 for field values.
15 —	Reserved
14-8 SEL1	TRGMUX Source Select 1 Specifies the source select for output 1. See Table 428 for field values.
7 —	Reserved
6-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 428 for field values.

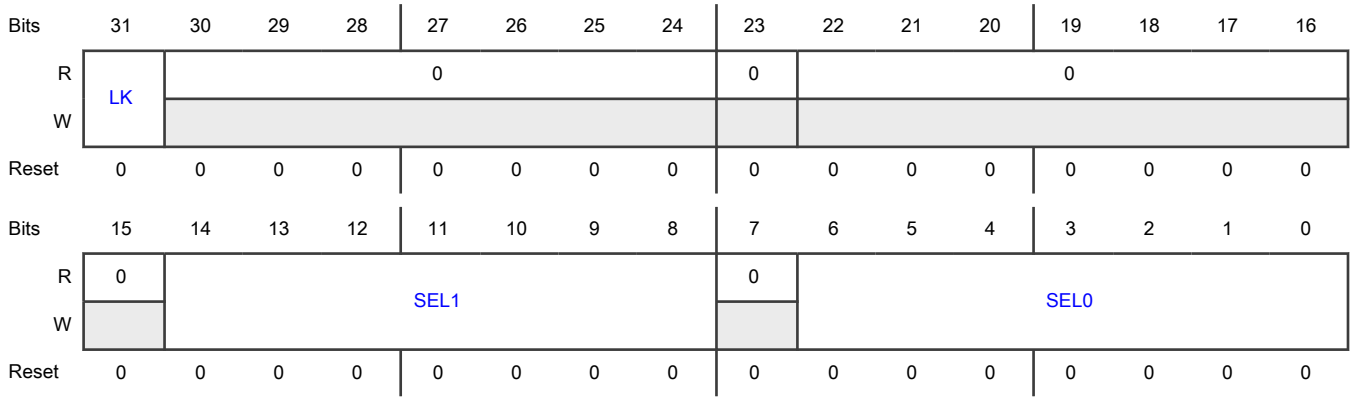
65.6.13 TRGMUX eMIOS0_3 (eMIOS0_3)

Offset

Register	Offset
eMIOS0_3	2Ch

Function
Configures the eMIOS0_3 module.

Diagram



Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable 1b - Register is not writable until the next system reset
30-24 —	Reserved
23 —	Reserved
22-16 —	Reserved
15 —	Reserved
14-8 SEL1	TRGMUX Source Select 1 Specifies the source select for output 1. See Table 428 for field values.
7 —	Reserved
6-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 428 for field values.

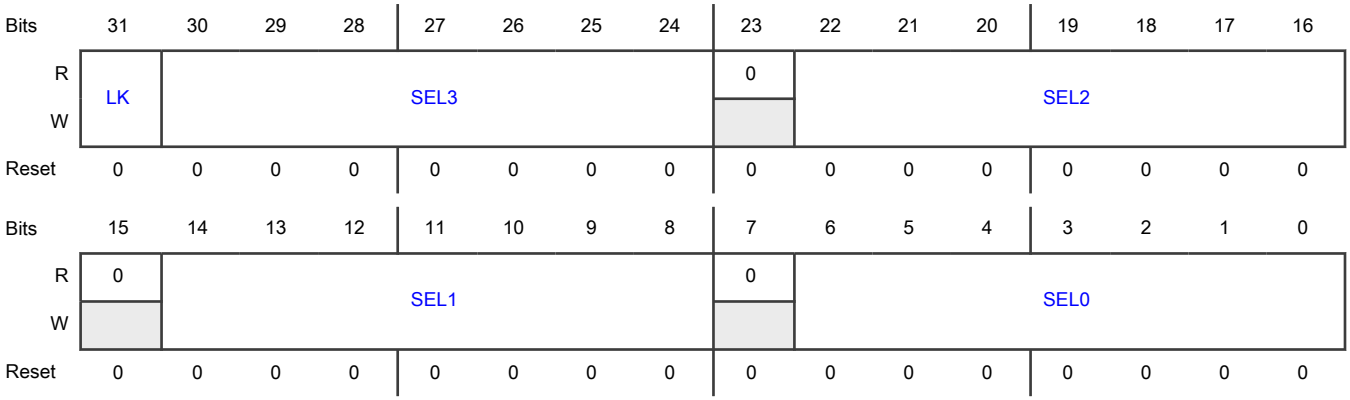
65.6.14 TRGMUX eMIOS1_0 (eMIOS1_0)

Offset

Register	Offset
eMIOS1_0	30h

Function
Configures the eMIOS1_0 module.

Diagram



Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable 1b - Register is not writable until the next system reset
30-24 SEL3	TRGMUX Source Select 3 Specifies the source select for output 3. See Table 428 for field values.
23 —	Reserved
22-16 SEL2	TRGMUX Source Select 2 Specifies the source select for output 2. See Table 428 for field values.
15 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
14-8 SEL1	TRGMUX Source Select 1 Specifies the source select for output 1. See Table 428 for field values.
7 —	Reserved
6-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 428 for field values.

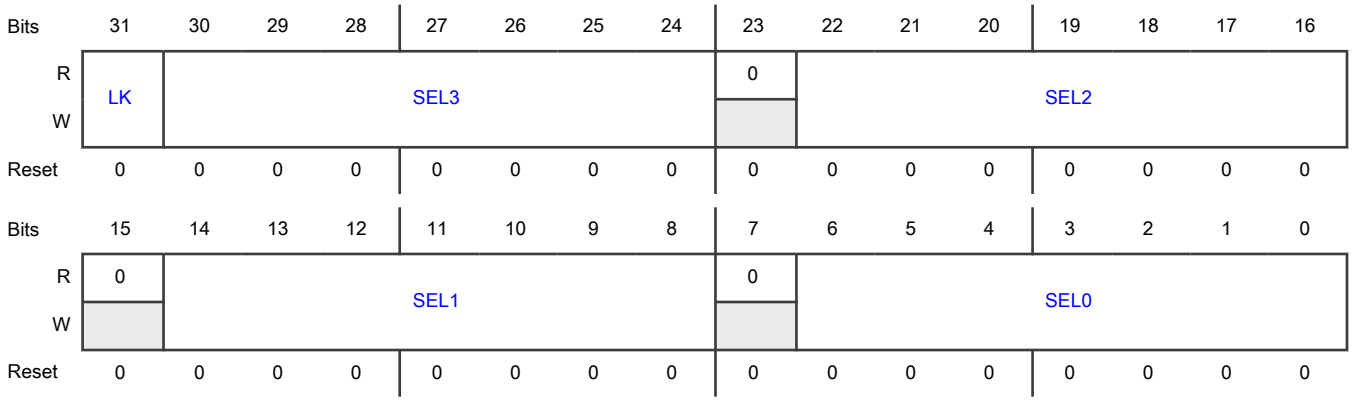
65.6.15 TRGMUX eMIOS1_1 (eMIOS1_1)

Offset

Register	Offset
eMIOS1_1	34h

Function
Configures the eMIOS1_1 module.

Diagram



Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Register is not writable until the next system reset
30-24 SEL3	TRGMUX Source Select 3 Specifies the source select for output 3. See Table 428 for field values.
23 —	Reserved
22-16 SEL2	TRGMUX Source Select 2 Specifies the source select for output 2. See Table 428 for field values.
15 —	Reserved
14-8 SEL1	TRGMUX Source Select 1 Specifies the source select for output 1. See Table 428 for field values.
7 —	Reserved
6-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 428 for field values.

65.6.16 TRGMUX eMIOS1_2 (eMIOS1_2)

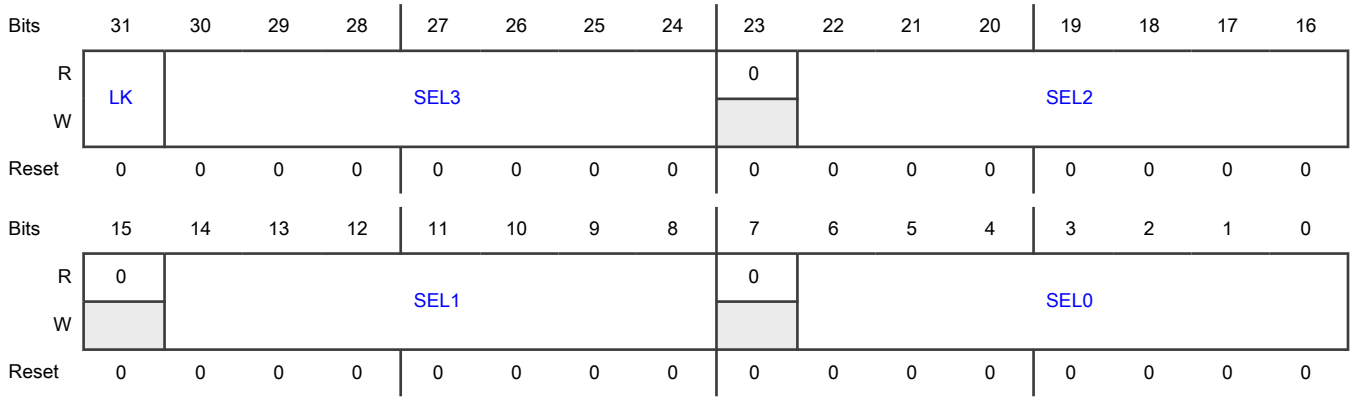
Offset

Register	Offset
eMIOS1_2	38h

Function

Configures the eMIOS1_2 module.

Diagram



Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable 1b - Register is not writable until the next system reset
30-24 SEL3	TRGMUX Source Select 3 Specifies the source select for output 3. See Table 428 for field values.
23 —	Reserved
22-16 SEL2	TRGMUX Source Select 2 Specifies the source select for output 2. See Table 428 for field values.
15 —	Reserved
14-8 SEL1	TRGMUX Source Select 1 Specifies the source select for output 1. See Table 428 for field values.
7 —	Reserved
6-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 428 for field values.

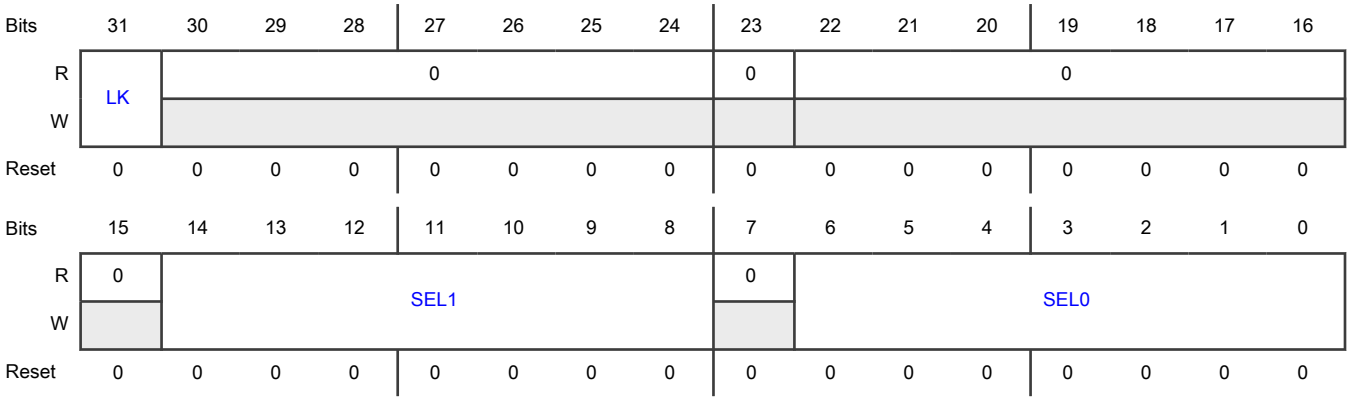
65.6.17 TRGMUX eMIOS1_3 (eMIOS1_3)

Offset

Register	Offset
eMIOS1_3	3Ch

Function
Configures the eMIOS1_3 module.

Diagram



Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable 1b - Register is not writable until the next system reset
30-24 —	Reserved
23 —	Reserved
22-16 —	Reserved
15 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
14-8 SEL1	TRGMUX Source Select 1 Specifies the source select for output 1. See Table 428 for field values.
7 —	Reserved
6-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 428 for field values.

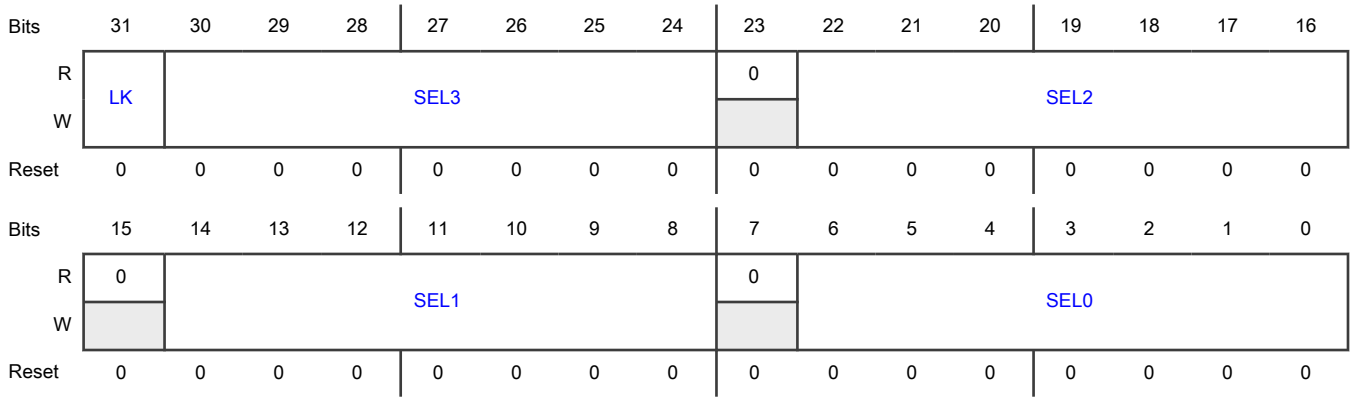
65.6.18 TRGMUX FlexIO (FlexIO)

Offset

Register	Offset
FlexIO	40h

Function
Configures the FlexIO module.

Diagram



Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Register is not writable until the next system reset
30-24 SEL3	TRGMUX Source Select 3 Specifies the source select for output 3. See Table 428 for field values.
23 —	Reserved
22-16 SEL2	TRGMUX Source Select 2 Specifies the source select for output 2. See Table 428 for field values.
15 —	Reserved
14-8 SEL1	TRGMUX Source Select 1 Specifies the source select for output 1. See Table 428 for field values.
7 —	Reserved
6-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 428 for field values.

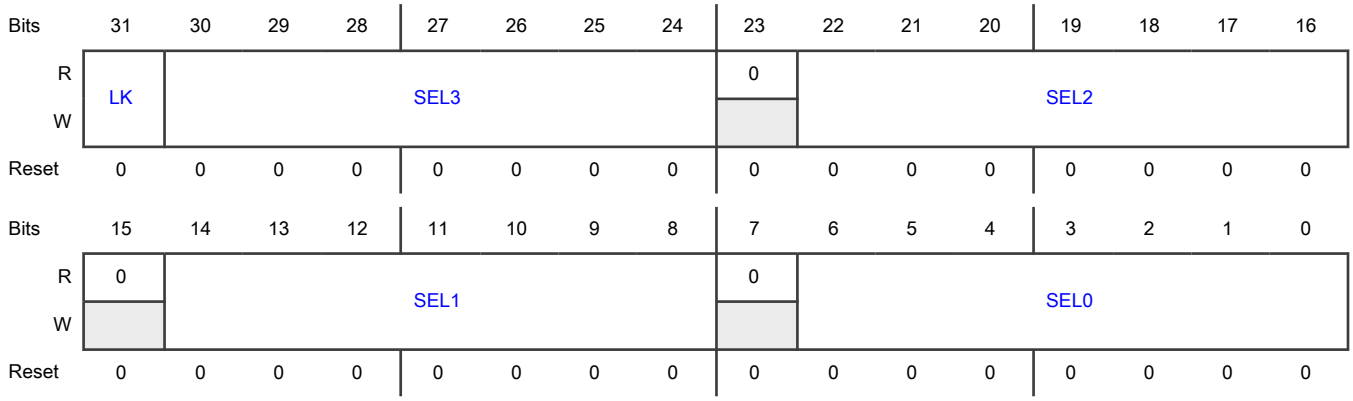
65.6.19 TRGMUX SIUL_OUT0 (SIUL_OUT0)

Offset

Register	Offset
SIUL_OUT0	44h

Function
Configures the SIUL_OUT0 module.

Diagram



Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable 1b - Register is not writable until the next system reset
30-24 SEL3	TRGMUX Source Select 3 Specifies the source select for output 3. See Table 428 for field values.
23 —	Reserved
22-16 SEL2	TRGMUX Source Select 2 Specifies the source select for output 2. See Table 428 for field values.
15 —	Reserved
14-8 SEL1	TRGMUX Source Select 1 Specifies the source select for output 1. See Table 428 for field values.
7 —	Reserved
6-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 428 for field values.

65.6.20 TRGMUX SIUL_OUT1 (SIUL_OUT1)

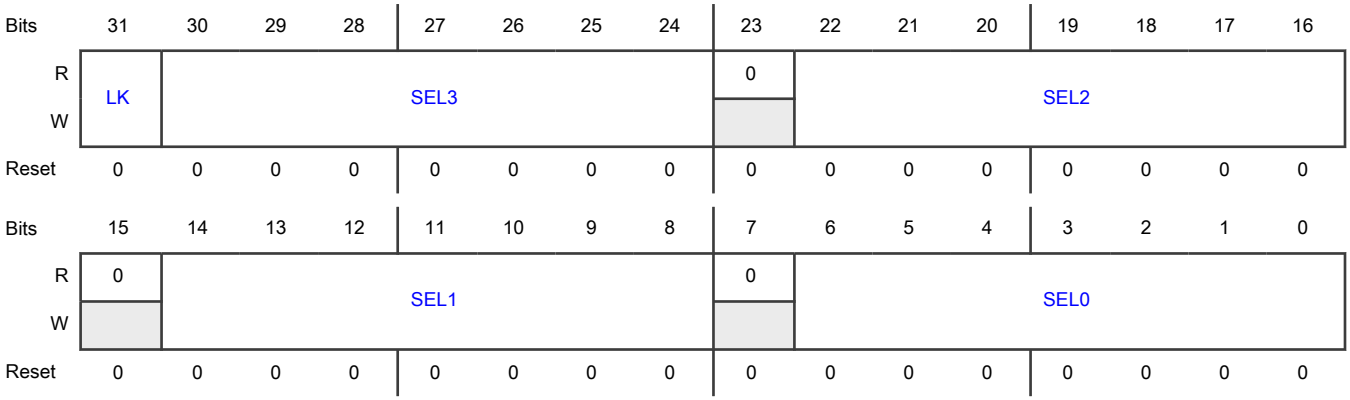
Offset

Register	Offset
SIUL_OUT1	48h

Function

Configures the SIUL_OUT1 module.

Diagram



Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable 1b - Register is not writable until the next system reset
30-24 SEL3	TRGMUX Source Select 3 Specifies the source select for output 3. See Table 428 for field values.
23 —	Reserved
22-16 SEL2	TRGMUX Source Select 2 Specifies the source select for output 2. See Table 428 for field values.
15 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
14-8 SEL1	TRGMUX Source Select 1 Specifies the source select for output 1. See Table 428 for field values.
7 —	Reserved
6-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 428 for field values.

65.6.21 TRGMUX SIUL_OUT2 (SIUL_OUT2)

Offset

Register	Offset
SIUL_OUT2	4Ch

Function
Configures the SIUL_OUT2 module.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	SEL3							0	SEL2						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	SEL1							0	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Register is not writable until the next system reset
30-24 SEL3	TRGMUX Source Select 3 Specifies the source select for output 3. See Table 428 for field values.
23 —	Reserved
22-16 SEL2	TRGMUX Source Select 2 Specifies the source select for output 2. See Table 428 for field values.
15 —	Reserved
14-8 SEL1	TRGMUX Source Select 1 Specifies the source select for output 1. See Table 428 for field values.
7 —	Reserved
6-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 428 for field values.

65.6.22 TRGMUX SIUL_OUT3 (SIUL_OUT3)

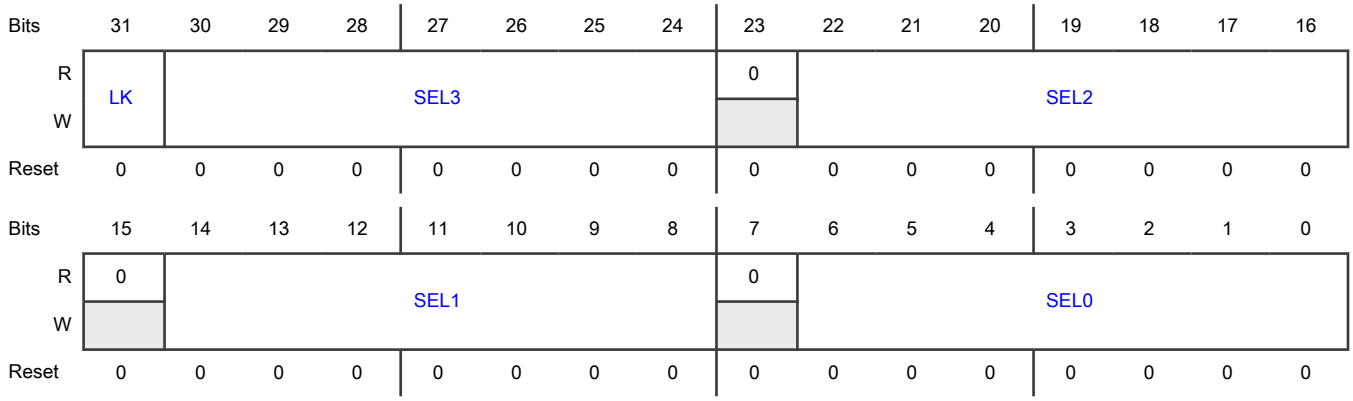
Offset

Register	Offset
SIUL_OUT3	50h

Function

Configures the SIUL_OUT3 module.

Diagram



Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable 1b - Register is not writable until the next system reset
30-24 SEL3	TRGMUX Source Select 3 Specifies the source select for output 3. See Table 428 for field values.
23 —	Reserved
22-16 SEL2	TRGMUX Source Select 2 Specifies the source select for output 2. See Table 428 for field values.
15 —	Reserved
14-8 SEL1	TRGMUX Source Select 1 Specifies the source select for output 1. See Table 428 for field values.
7 —	Reserved
6-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 428 for field values.

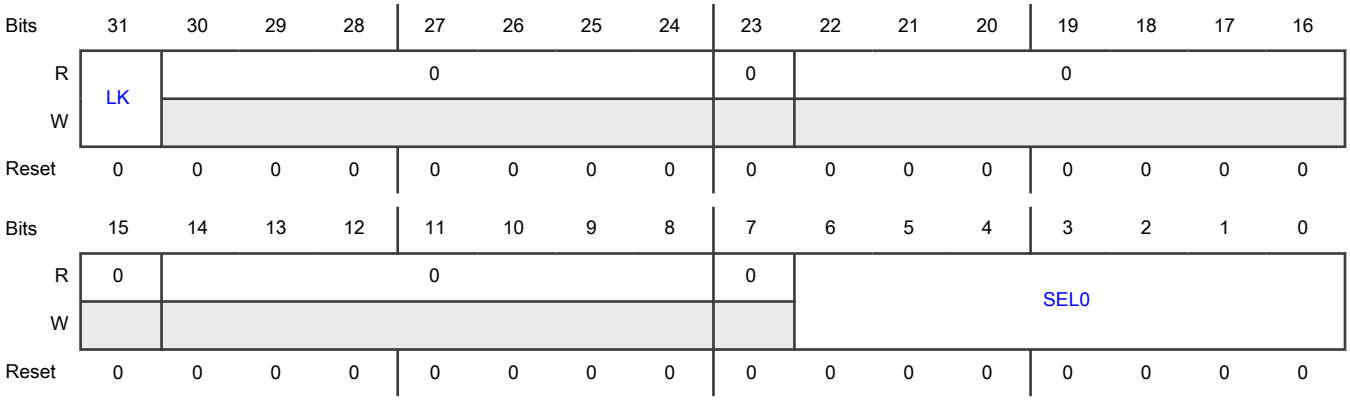
65.6.23 TRGMUX LPI2C_0 (LPI2C_0)

Offset

Register	Offset
LPI2C_0	54h

Function
Configures the LPI2C_0 module.

Diagram



Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable 1b - Register is not writable until the next system reset
30-24 —	Reserved
23 —	Reserved
22-16 —	Reserved
15 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
14-8 —	Reserved
7 —	Reserved
6-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 428 for field values.

65.6.24 TRGMUX LPSPI_0 (LPSPI_0)

Offset

Register	Offset
LPSPI_0	58h

Function
Configures the LPSPI_0 module.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0							0	0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0							0	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Register is not writable until the next system reset
30-24 —	Reserved
23 —	Reserved
22-16 —	Reserved
15 —	Reserved
14-8 —	Reserved
7 —	Reserved
6-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 428 for field values.

65.6.25 TRGMUX LPSPI_1 (LPSPI_1)

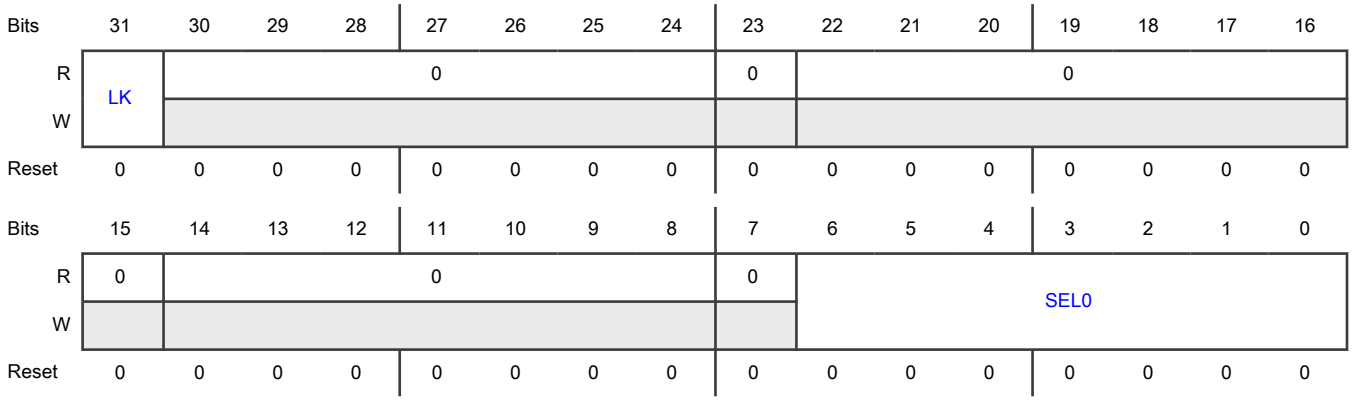
Offset

Register	Offset
LPSPI_1	5Ch

Function

Configures the LPSPI_1 module.

Diagram



Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable 1b - Register is not writable until the next system reset
30-24 —	Reserved
23 —	Reserved
22-16 —	Reserved
15 —	Reserved
14-8 —	Reserved
7 —	Reserved
6-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 428 for field values.

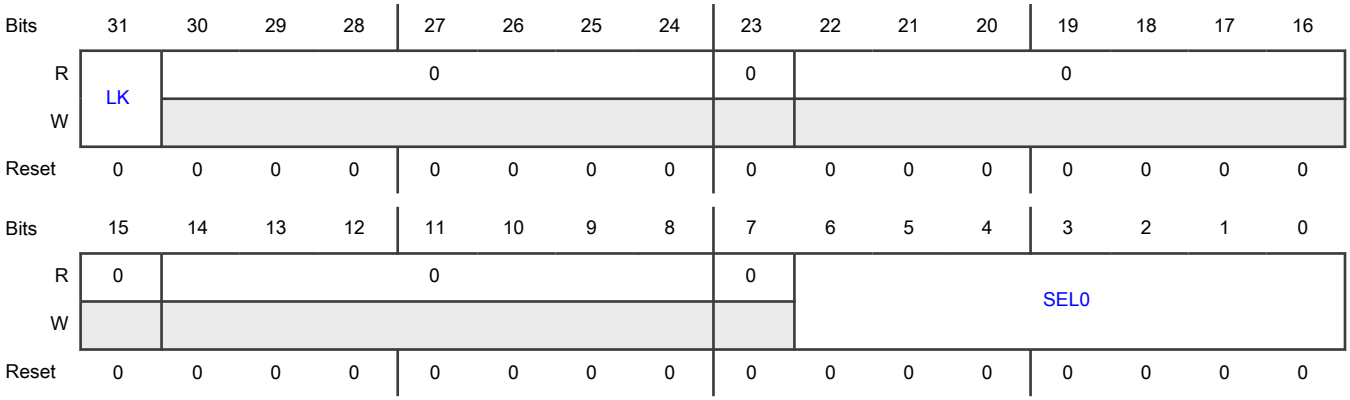
65.6.26 TRGMUX LPSPI_2 (LPSPI_2)

Offset

Register	Offset
LPSPI_2	60h

Function
Configures the LPSPI_2 module.

Diagram



Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable 1b - Register is not writable until the next system reset
30-24 —	Reserved
23 —	Reserved
22-16 —	Reserved
15 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
14-8 —	Reserved
7 —	Reserved
6-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 428 for field values.

65.6.27 TRGMUX LPUART_0 (LPUART_0)

Offset

Register	Offset
LPUART_0	64h

Function

Configures the LPUART_0 module.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0							0	0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0							0	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Register is not writable until the next system reset
30-24 —	Reserved
23 —	Reserved
22-16 —	Reserved
15 —	Reserved
14-8 —	Reserved
7 —	Reserved
6-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 428 for field values.

65.6.28 TRGMUX LPUART_1 (LPUART_1)

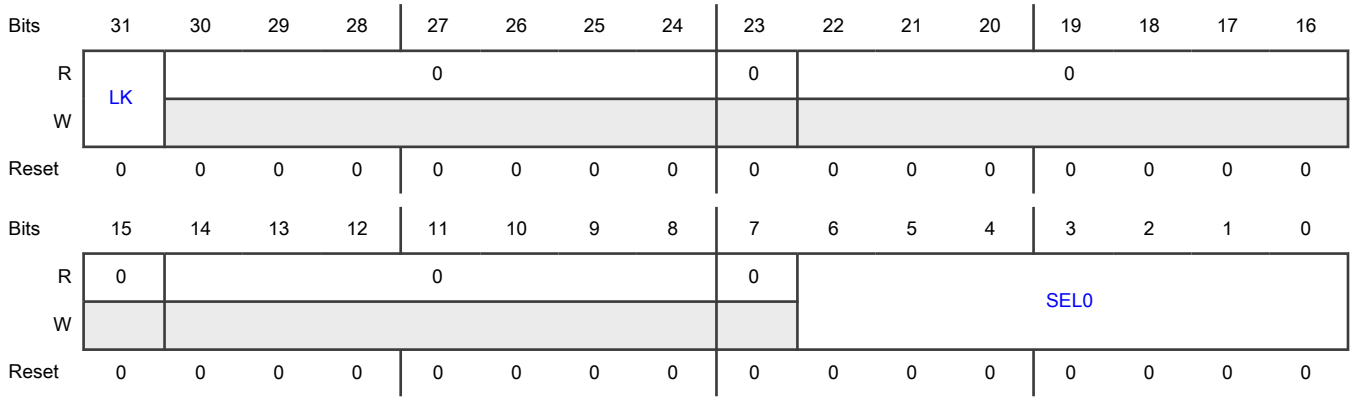
Offset

Register	Offset
LPUART_1	68h

Function

Configures the LPUART_1 module.

Diagram



Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable 1b - Register is not writable until the next system reset
30-24 —	Reserved
23 —	Reserved
22-16 —	Reserved
15 —	Reserved
14-8 —	Reserved
7 —	Reserved
6-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 428 for field values.

65.6.29 TRGMUX LPUART_2 (LPUART_2)

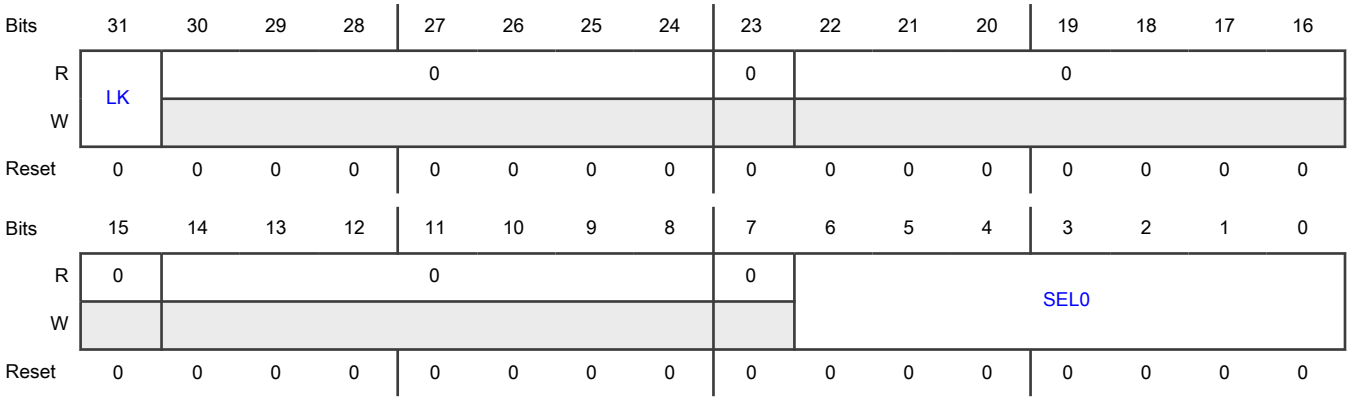
Offset

Register	Offset
LPUART_2	6Ch

Function

Configures the LPUART_2 module.

Diagram



Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable 1b - Register is not writable until the next system reset
30-24 —	Reserved
23 —	Reserved
22-16 —	Reserved
15 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
14-8 —	Reserved
7 —	Reserved
6-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 428 for field values.

65.6.30 TRGMUX LPUART_3 (LPUART_3)

Offset

Register	Offset
LPUART_3	70h

Function
Configures the LPUART_3 module.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0							0	0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0							0	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Register is not writable until the next system reset
30-24 —	Reserved
23 —	Reserved
22-16 —	Reserved
15 —	Reserved
14-8 —	Reserved
7 —	Reserved
6-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 428 for field values.

65.6.31 TRGMUX LCU0_SYNC (LCU0_SYNC)

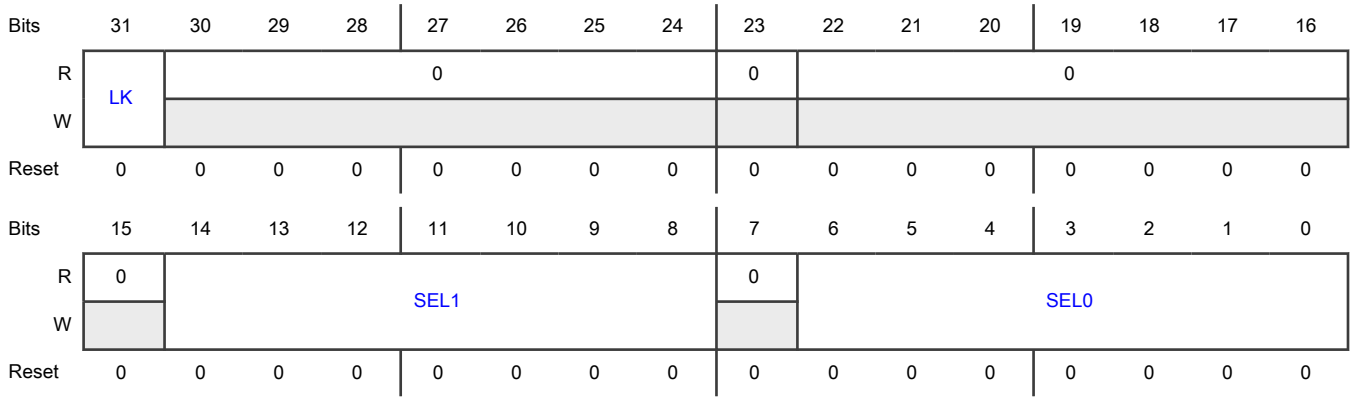
Offset

Register	Offset
LCU0_SYNC	74h

Function

Configures the LCU0_SYNC module.

Diagram



Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable 1b - Register is not writable until the next system reset
30-24 —	Reserved
23 —	Reserved
22-16 —	Reserved
15 —	Reserved
14-8 SEL1	TRGMUX Source Select 1 Specifies the source select for output 1. See Table 428 for field values.
7 —	Reserved
6-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 428 for field values.

65.6.32 TRGMUX LCU0_FORCE (LCU0_FORCE)

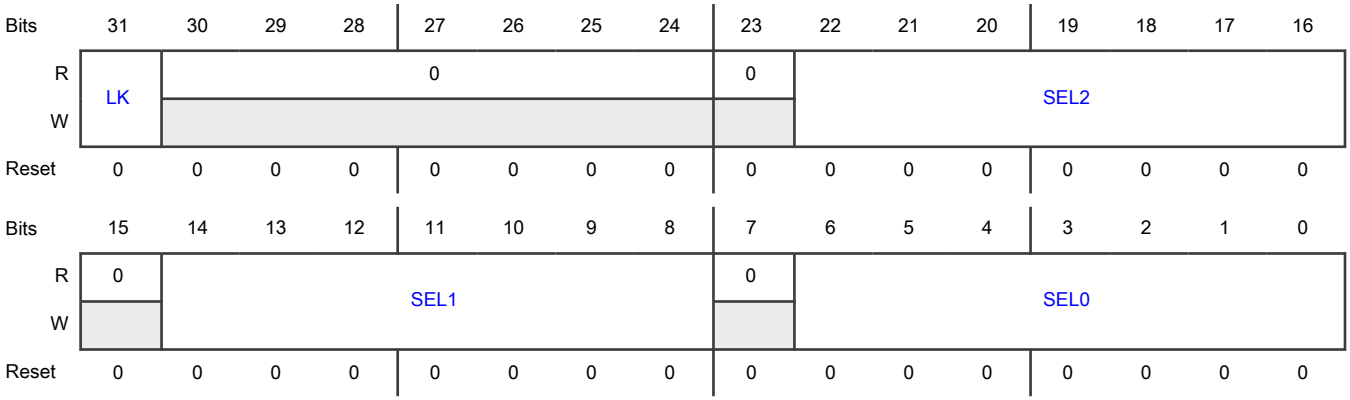
Offset

Register	Offset
LCU0_FORCE	78h

Function

Configures the LCU0_FORCE module.

Diagram



Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable 1b - Register is not writable until the next system reset
30-24 —	Reserved
23 —	Reserved
22-16 SEL2	TRGMUX Source Select 2 Specifies the source select for output 2. See Table 428 for field values.
15 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
14-8 SEL1	TRGMUX Source Select 1 Specifies the source select for output 1. See Table 428 for field values.
7 —	Reserved
6-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 428 for field values.

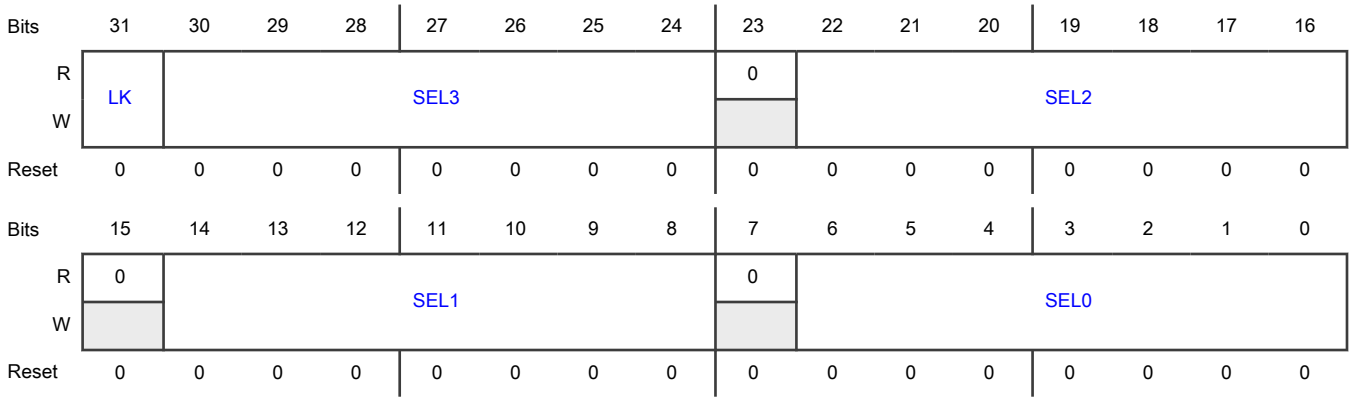
65.6.33 TRGMUX LCU0_0 (LCU0_0)

Offset

Register	Offset
LCU0_0	7Ch

Function
Configures the LCU0_0 module.

Diagram



Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Register is not writable until the next system reset
30-24 SEL3	TRGMUX Source Select 3 Specifies the source select for output 3. See Table 428 for field values.
23 —	Reserved
22-16 SEL2	TRGMUX Source Select 2 Specifies the source select for output 2. See Table 428 for field values.
15 —	Reserved
14-8 SEL1	TRGMUX Source Select 1 Specifies the source select for output 1. See Table 428 for field values.
7 —	Reserved
6-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 428 for field values.

65.6.34 TRGMUX LCU0_1 (LCU0_1)

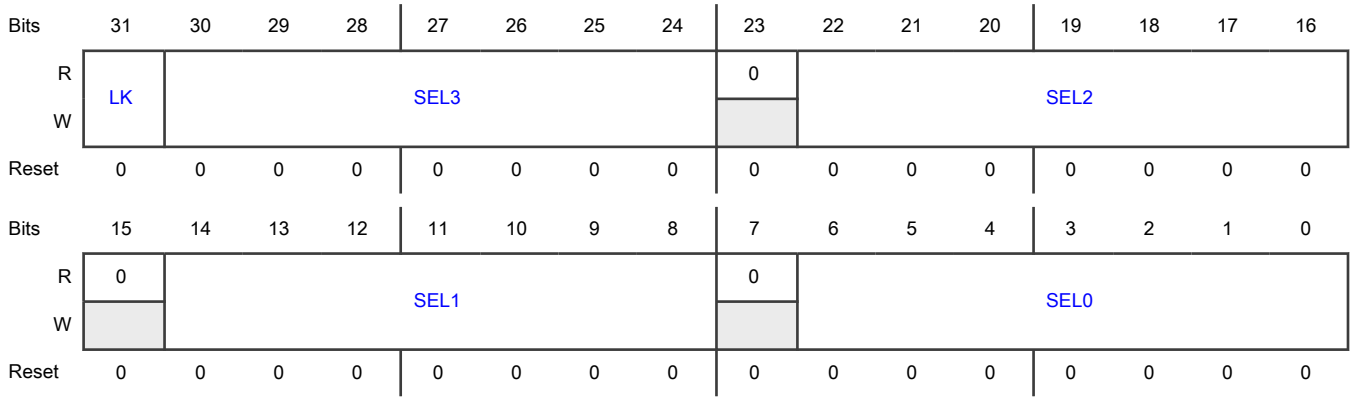
Offset

Register	Offset
LCU0_1	80h

Function

Configures the LCU0_1 module.

Diagram



Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable 1b - Register is not writable until the next system reset
30-24 SEL3	TRGMUX Source Select 3 Specifies the source select for output 3. See Table 428 for field values.
23 —	Reserved
22-16 SEL2	TRGMUX Source Select 2 Specifies the source select for output 2. See Table 428 for field values.
15 —	Reserved
14-8 SEL1	TRGMUX Source Select 1 Specifies the source select for output 1. See Table 428 for field values.
7 —	Reserved
6-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 428 for field values.

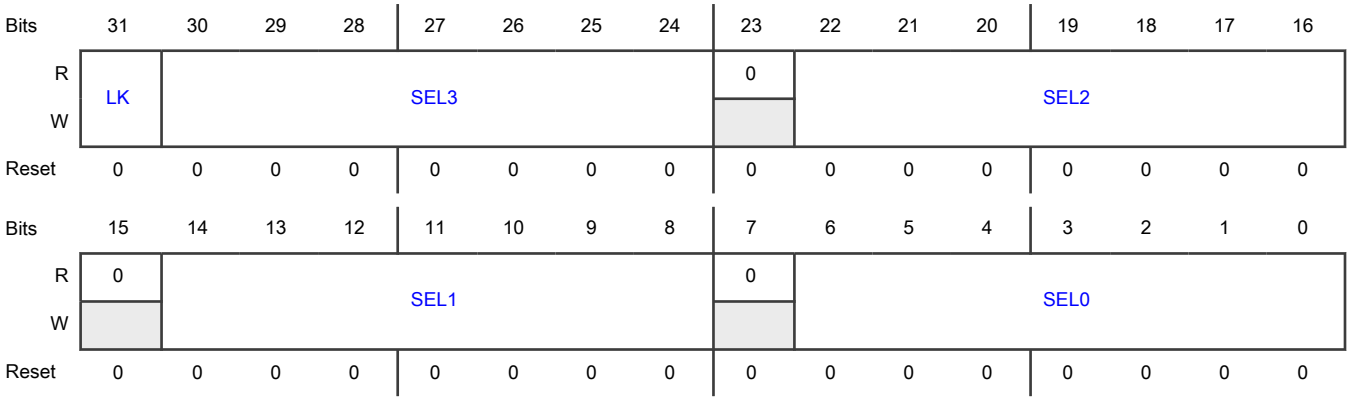
65.6.35 TRGMUX LCU0_2 (LCU0_2)

Offset

Register	Offset
LCU0_2	84h

Function
Configures the LCU0_2 module.

Diagram



Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable 1b - Register is not writable until the next system reset
30-24 SEL3	TRGMUX Source Select 3 Specifies the source select for output 3. See Table 428 for field values.
23 —	Reserved
22-16 SEL2	TRGMUX Source Select 2 Specifies the source select for output 2. See Table 428 for field values.
15 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
14-8 SEL1	TRGMUX Source Select 1 Specifies the source select for output 1. See Table 428 for field values.
7 —	Reserved
6-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 428 for field values.

65.6.36 TRGMUX LCU1_SYNC (LCU1_SYNC)

Offset

Register	Offset
LCU1_SYNC	88h

Function

Configures the LCU1_SYNC module.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0							0	0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	SEL1							0	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Register is not writable until the next system reset
30-24 —	Reserved
23 —	Reserved
22-16 —	Reserved
15 —	Reserved
14-8 SEL1	TRGMUX Source Select 1 Specifies the source select for output 1. See Table 428 for field values.
7 —	Reserved
6-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 428 for field values.

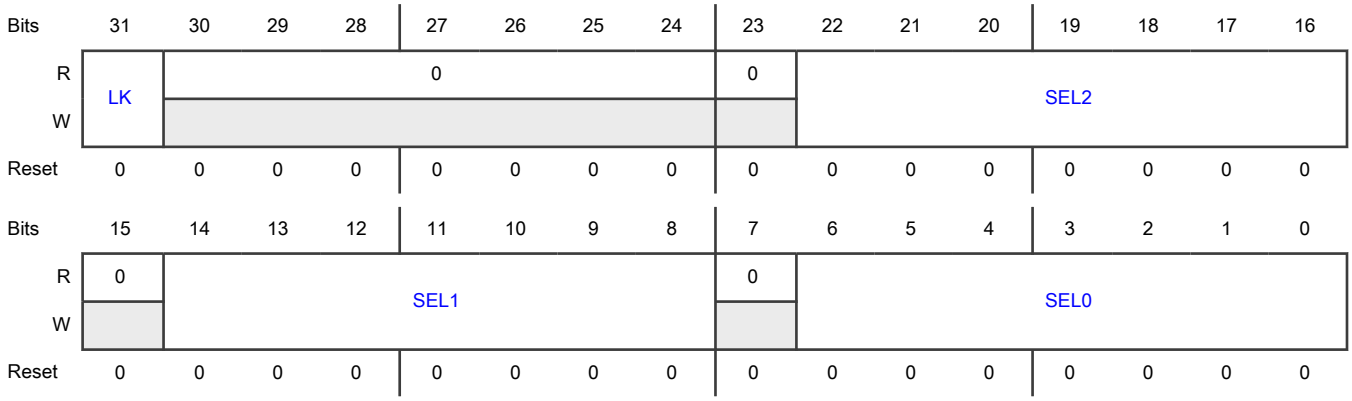
65.6.37 TRGMUX LCU1_FORCE (LCU1_FORCE)

Offset

Register	Offset
LCU1_FORCE	8Ch

Function
Configures the LCU1_FORCE module.

Diagram



Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable 1b - Register is not writable until the next system reset
30-24 —	Reserved
23 —	Reserved
22-16 SEL2	TRGMUX Source Select 2 Specifies the source select for output 2. See Table 428 for field values.
15 —	Reserved
14-8 SEL1	TRGMUX Source Select 1 Specifies the source select for output 1. See Table 428 for field values.
7 —	Reserved
6-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 428 for field values.

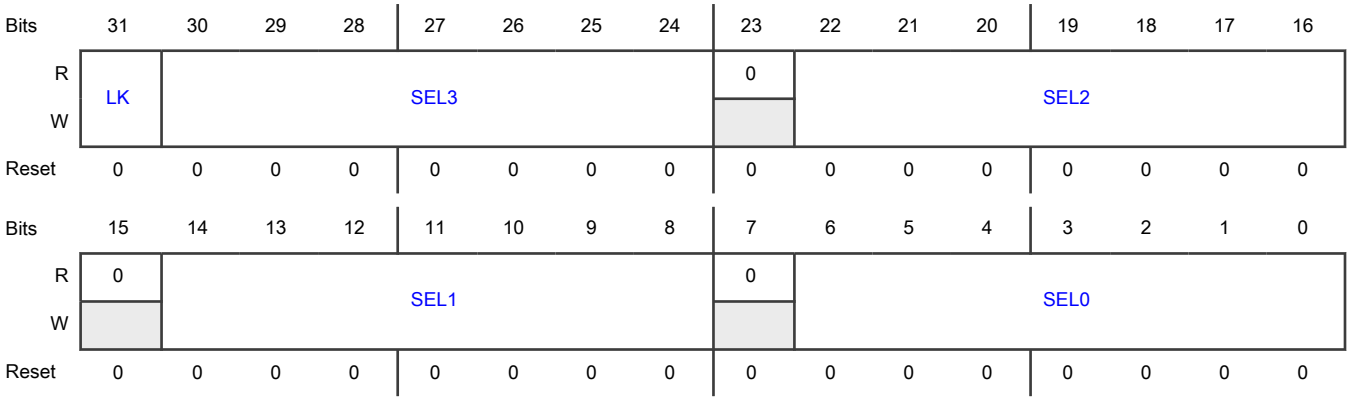
65.6.38 TRGMUX LCU1_0 (LCU1_0)

Offset

Register	Offset
LCU1_0	90h

Function
Configures the LCU1_0 module.

Diagram



Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable 1b - Register is not writable until the next system reset
30-24 SEL3	TRGMUX Source Select 3 Specifies the source select for output 3. See Table 428 for field values.
23 —	Reserved
22-16 SEL2	TRGMUX Source Select 2 Specifies the source select for output 2. See Table 428 for field values.
15 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
14-8 SEL1	TRGMUX Source Select 1 Specifies the source select for output 1. See Table 428 for field values.
7 —	Reserved
6-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 428 for field values.

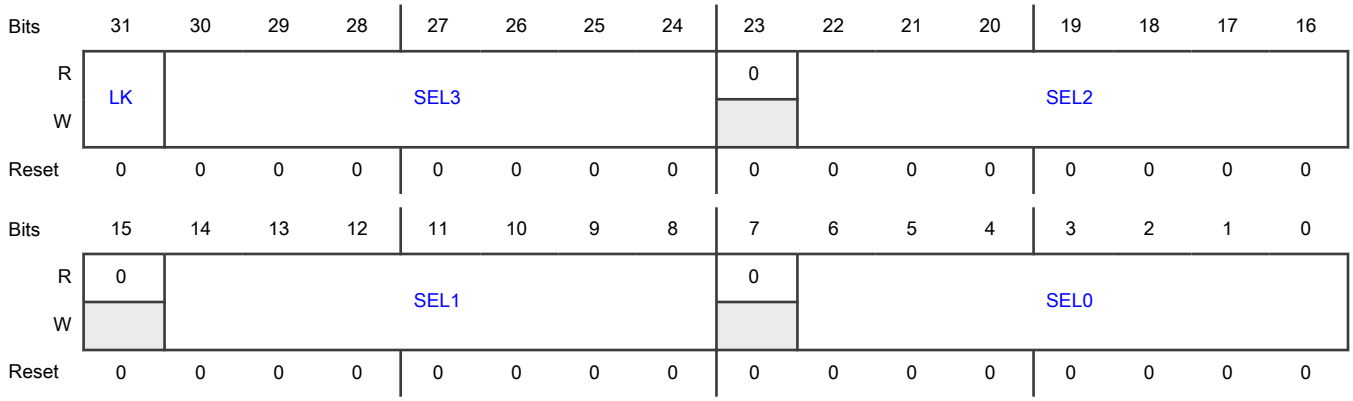
65.6.39 TRGMUX LCU1_1 (LCU1_1)

Offset

Register	Offset
LCU1_1	94h

Function
Configures the LCU1_1 module.

Diagram



Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Register is not writable until the next system reset
30-24 SEL3	TRGMUX Source Select 3 Specifies the source select for output 3. See Table 428 for field values.
23 —	Reserved
22-16 SEL2	TRGMUX Source Select 2 Specifies the source select for output 2. See Table 428 for field values.
15 —	Reserved
14-8 SEL1	TRGMUX Source Select 1 Specifies the source select for output 1. See Table 428 for field values.
7 —	Reserved
6-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 428 for field values.

65.6.40 TRGMUX LCU1_2 (LCU1_2)

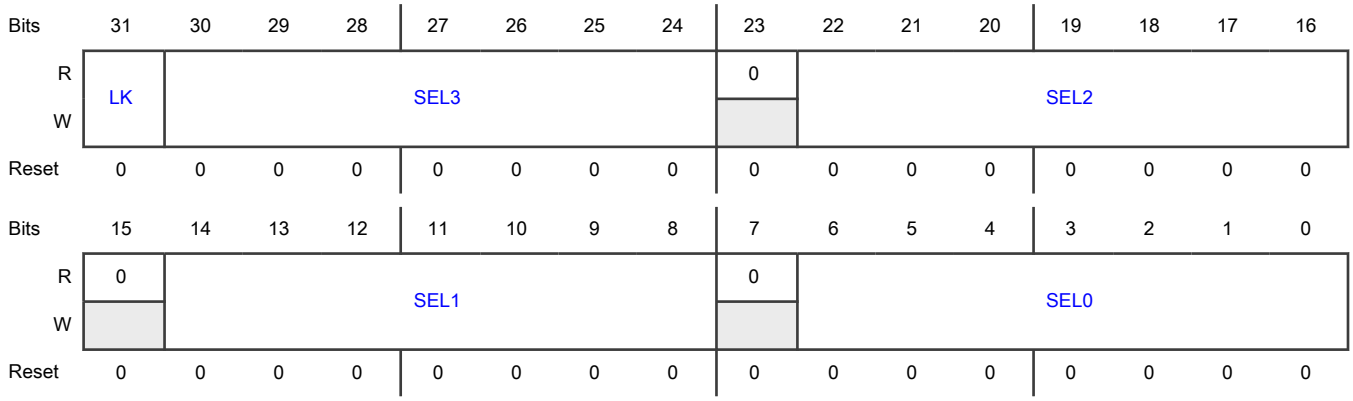
Offset

Register	Offset
LCU1_2	98h

Function

Configures the LCU1_2 module.

Diagram



Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable 1b - Register is not writable until the next system reset
30-24 SEL3	TRGMUX Source Select 3 Specifies the source select for output 3. See Table 428 for field values.
23 —	Reserved
22-16 SEL2	TRGMUX Source Select 2 Specifies the source select for output 2. See Table 428 for field values.
15 —	Reserved
14-8 SEL1	TRGMUX Source Select 1 Specifies the source select for output 1. See Table 428 for field values.
7 —	Reserved
6-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 428 for field values.

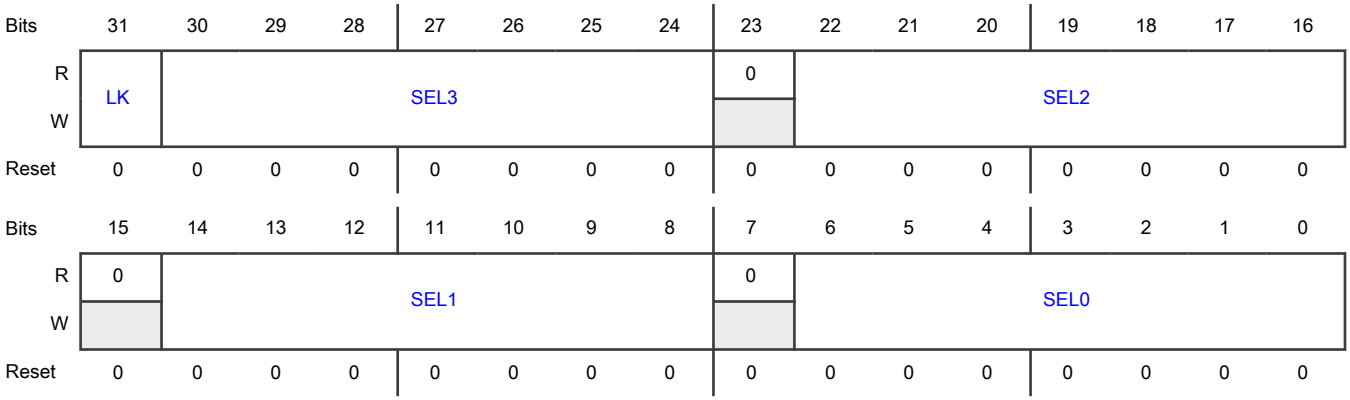
65.6.41 TRGMUX CM7_RXEV (CM7_RXEV)

Offset

Register	Offset
CM7_RXEV	9Ch

Function
Configures the CM7_RXEV module.

Diagram



Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable 1b - Register is not writable until the next system reset
30-24 SEL3	TRGMUX Source Select 3 Specifies the source select for output 3. See Table 428 for field values.
23 —	Reserved
22-16 SEL2	TRGMUX Source Select 2 Specifies the source select for output 2. See Table 428 for field values.
15 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
14-8 SEL1	TRGMUX Source Select 1 Specifies the source select for output 1. See Table 428 for field values.
7 —	Reserved
6-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 428 for field values.

Chapter 66
Software Watchdog Timer (SWT)

66.1 Chip-specific SWT information

66.1.1 SWT instances and configuration

This chip supports up to four instances of SWT, one for each Cortex-M7 core.

Table 429. SWT instances

Instance	S32K388/ S32K389	S32K358/ S32K356	S32K348	S32K338	S32K328	S32K322/ S32K324	S32K310/S32K311/ S32K312/S32K314/ S32K342/ S32K341/S32K344
SWT_0	Yes	Yes	Yes	Yes	Yes	Yes	Yes
SWT_1	Yes	Yes	No	Yes	Yes	Yes	No
SWT_2	Yes	No	No	Yes	No	No	No
SWT_3	Yes	No	No	No	No	No	No

Table 430. SWT configuration

Instance	Core	Reset type	Clock Source	SWT_MIN_TO (100µs)	SWT_TO_RST (25ms)
SWT_0[1]	Cortex-M7_0	Functional Reset - RAM retention	SIRC (32K)	3	320h
SWT_1	Cortex-M7_1	Functional Reset - RAM retention	SIRC (32K)	3	320h
SWT_2	Cortex-M7_2	Functional Reset - RAM retention	SIRC (32K)	3	320h
SWT_3	Cortex-M7_3	Functional Reset - RAM retention	SIRC (32K)	3	320h

1. SWT_0 supports operation in STANDBY mode.

NOTE

When using SWT as a wakeup source from standby, software should ensure proper value of timeout such that SWT timeout doesn't get expired again (after wakeup till standby exit is completed). The timeout should be programmed to a value greater than the time till standby exit is complete (sum of standby exit time and the time till software disables padkeeping).

NOTE

The "Reset only the SWT" feature using SWT_RRR[RRF] is supported only when SWT reset reaction is demoted to interrupt within MC_RGM.

Protect SWT from service by non-core initiators

You must program protection for the SWT registers so that only core initiators can service SWT.

When an SWT timeout takes place, the logic generating a reset to the system runs off the slow internal RC oscillator (32KHz). It could therefore take the SWT almost 31.25µs to reset the system, after timeout, leaving more time for runaway code to execute. The register interface clock of SWT is running from AIPS_SLOW_CLK, see Clocking chapter for more details. The register interface clock is responsible for generating the abort for invalid access while the counter clock is responsible for generating the SWT reset request. Since there is a synchronization happening between the register interface clock and counter clock domains before generating the SWT reset request, thus the generation of the reset request can take a number of register interface clock cycles after an invalid access is done.

66.2 Overview

The Software Watchdog Timer (SWT) is a 32-bit window watchdog timer that enables the system to recover from situations such as:

- Software trapped in a loop
- A bus transaction failing to terminate

In regular operation, SWT requires periodic execution of a watchdog servicing operation. The servicing operation resets the timer to a specified timeout period. If this servicing action does not occur before the timer expires, SWT generates an interrupt or a hardware reset request.

You can configure SWT to generate a reset request or an interrupt on the initial timeout. SWT always generates a reset request on a second consecutive timeout.

NOTE

Terminology in this chapter has been updated as follows:

Table 431. Updated terms

Updated terms	Deprecated term
Initiator	Master

66.2.1 Block diagram

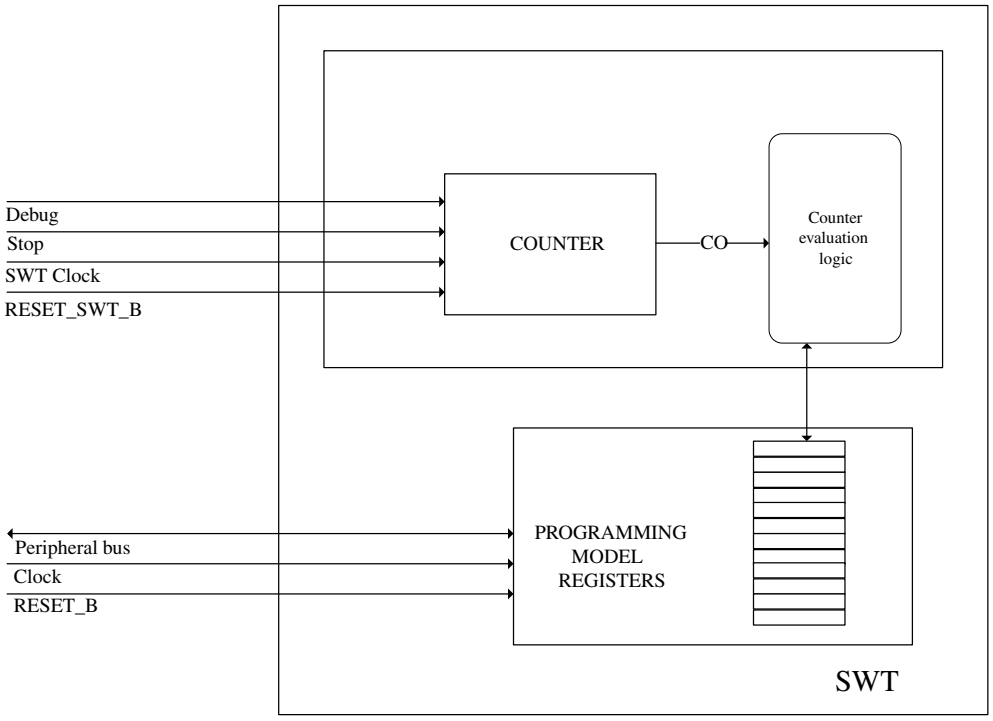


Figure 379. SWT block diagram

66.2.2 Features

SWT has the following features:

- 32-bit countdown timer
- Selection of regular or window servicing
- Selection of reset request or interrupt on an initial timeout
- Selection of fixed or keyed service sequence
- Initiator access protection
- Hard and soft configuration lock bits

66.3 Functional description

66.3.1 Behavior in different chip and core modes

SWT supports the core modes of operation as follows:

Core mode	SWT behavior
Normal	When SWT is enabled (CR[WEN] = 1), the SWT timer counts down continuously.
Debug	If CR[FRZ] = 1, SWT stops the timer; otherwise, the timer continues to run.

66.3.2 Register access latency

Accesses to SWT registers occur with no peripheral bus wait states. The peripheral bus bridge may add 1 or more system wait states. However, due to synchronization logic in the SWT design, recognition of the service sequence or configuration changes may require up to three system clock cycles plus seven counter clock cycles.

66.3.3 Service key generation

SWT generates service keys with the pseudorandom key generator defined by the following equation.

$$SK_{n+1} = (17 \times SK_n + 3) \bmod 2^{16}$$

Figure 380. Pseudorandom key generator

This algorithm generates a sequence of 2^{16} different key values before repeating. In keyed service sequence mode, each time you write a valid key to SR[WSC], SWT updates SK[SK] with the next key.

For example, if the previous service key (SK[SK]) is 100h, then the next service sequence keys are 1103h followed by 2136h.

66.3.4 Clocking

The SWT has two clock domains, one for the IPS interface and the other for the counter and reset request logic. The two clocks can be asynchronous to each other and have independent resets.

See the chip-specific SWT information for the clock source of the counter and reset logic.

See the chip-specific SWT information to find the clock source that drives the countdown timer.

66.3.5 Reset

The following table describes the SWT resets:

Table 432. SWT resets

Reset	Description
RESET_B	It is used to reset the programming model registers.
RESET_SWT_B	It is used to reset the functional domain of the SWT.

66.3.6 Interrupts

The SWT generates one interrupt:

- Timeout interrupt: it is asserted when the watchdog timer count exceeds the timeout period in the TO register.

66.3.7 Reset only the SWT

When a timeout triggers a reset request and the application does not require a system reset, you can make the SWT instance reset itself rather than the whole system.

To make a timeout reset request trigger only an SWT reset (rather than the whole system), write 1 to the Reset Request Flag (RRR[RRF]).

This action causes SWT to:

- Clear the reset request.
- If an interrupt request also occurred, clear the interrupt request (IR[TIF]).
- Reset and restart the SWT.

For systems with multiple SWT instances, this feature allows you to reset an individual SWT instance, avoiding the widespread impact of a system reset.

66.3.8 Test SWT operation

When you disable SWT, it loads the current countdown timer into CO[CNT]. When you enable SWT, it clears this register. You can use CO[CNT] to perform a software self-test of SWT.

To test SWT operation:

1. Enable the SWT (CR[WEN] = 1).
2. Do not service SWT for a fixed period of time that is less than the timeout value.
3. Disable the SWT (CR[WEN] = 0).
4. Read the value in CO[CNT] to determine whether the internal countdown timer is working properly.

NOTE

The value shown in CO[CNT] can lag behind the actual internal timer up to six system clock cycles plus eight counter clock cycles.

66.4 External signals

This module has no external signals.

66.5 Initialization

66.5.1 Initialize the SWT

To initialize the SWT, you must do the following:

- Set the timeout period.
- Control timeout behavior. There are two options:
  - Select reset request on timeout, or
  - Select interrupt on initial timeout.

You must initialize all registers before enabling SWT (CR[WEN]). You can initialize the registers in any sequence.

66.5.1.1 Set the timeout period

When you enable SWT, it loads the greater of the specified watchdog timeout period or the minimum timeout period into an internal 32-bit countdown timer every time you perform a valid service operation.

To set the watchdog timeout period:

- Write the desired timeout period to TO[WTO].

The minimum timeout period is 3 clock cycles.

66.5.1.2 Control timeout behavior

SWT can respond to a timeout in one of two ways:

- Generate an immediate reset request on any timeout.
- Generate an interrupt and load the countdown timer on an initial timeout, then generate a reset request on a subsequent timeout.

66.5.1.2.1 Select reset request on timeout

To configure SWT to generate an immediate reset request on any timeout:

- Write 0 to CR[ITR].

66.5.1.2.2 Select interrupt on initial timeout

To configure SWT to generate an interrupt on an initial timeout:

- Write 1 to CR[ITR].

In this configuration, on the initial timeout:

- SWT loads the countdown timer with the timeout period (TO[WTO]).
- SWT indicates the interrupt with the timeout interrupt flag (IR[TIF]).
- If a second consecutive timeout occurs before writing the service sequence for the first timeout, SWT generates a reset request.

You clear the interrupt flag by writing 1 to IR[TIF].

66.5.1.3 Configure locking and unlocking

You can lock the SWT configuration with either a hard lock or a soft lock. When either lock is in effect, CR, TO, WN, and SK are read-only.

66.5.1.3.1 Enable the hard lock

To enable the hard lock:

- Write 1 to CR[HLK].

Hard lock is disabled only by a reset.

66.5.1.3.2 Enable the soft lock

To enable the soft lock:

- Write 1 to CR[SLK].

66.5.1.3.3 Unlock the soft lock

To unlock the soft lock:

1. Write C520h to SR[WSC].
2. Write D928h to SR[WSC].

This unlock sequence:

- Ignores service sequence writes.
- Recognizes the unlock sequence regardless of previous writes.
- Recognizes the unlock sequence regardless of the time between writes.
- Does not require CR[WEN] to be 1.

You can write this unlock sequence at any time.

NOTE

It is possible for a keyed service sequence to unlock soft lock. See Avoid soft unlock for the procedure to handle this situation.

66.5.1.3.4 Avoid soft unlock

When SWT operates in keyed sequence service, the sequence of service keys generated by the pseudorandom generator includes the unlock keys for soft lock. If one or more service routines use both unlock keys in the proper order, 0xC520 followed

at any future time by 0xD928, SWT unlocks soft lock (CR[SLK] = 0). The unlock sequence logic ignores any service keys other than the unlock keys, so the unlock keys don't have to be inserted consecutively. If the second unlock key is also the second key of a service operation, unlock occurs before the service operation completes.

To avoid unlock:

If the service routine writes a key value of C520h to SR[WSC] as a key of the service operation, then do the following:

1. Complete the service operation.
2. Complete the unlock sequence by writing the second unlock key, D928h, to SR[WSC].
3. Reinitiate the soft lock (CR[SLK] = 1).

66.5.1.4 Select behavior on an invalid access

SWT can respond to an invalid access one of two ways:

- Generate a bus error.
- Generate a bus error and, if SWT is enabled, a reset request.

To select how SWT responds to an invalid access:

- Write the appropriate value to CR[RIA].

66.5.2 Initiate service operations

When enabled, SWT requires periodic execution of a servicing operation.

SWT can operate in one of the following service sequence modes:

- Fixed service sequence
- Keyed service sequence

66.5.2.1 Initiate a fixed service sequence

To initiate the fixed service sequence:

1. Select the fixed servicing mode (write 00b to CR[SMD]).
2. Write A602h to SR[WSC].
3. Write B480h to SR[WSC].

There is no timing requirement between the two writes. The service sequence logic ignores unlock-sequence writes.

66.5.2.2 Initiate a keyed service sequence

To initiate the keyed service sequence:

1. Select the keyed servicing mode (write 01b to CR[SMD]).
2. Read the initial service key from SK[SK].
3. Calculate the next service key using the provided equation.
4. Write that service key to SR[WSC].
5. Repeat steps 2 through 4 one time.

See Service key generation for information about service keys.

66.5.2.3 Select window service mode

In window service mode, you must write the service sequence only when the watchdog timer is less than the window start value (WN[WST]). If you write the service sequence outside of this window, the access is invalid and generates a bus error or reset request depending on the Reset on Invalid Access setting (CR[RIA]).

For example, if the timeout period is 5000 and the window start value is 1000, you must write the service sequence during the last 20% of the timeout period.

Synchronization logic in SWT causes a slight delay in the opening of the window. This delay can be up to three system clock cycles plus four counter clock cycles.

To select window service mode:

- Write 1 to (CR[WND]).

66.6 SWT register descriptions

The SWT programming model allows only 32-bit (word) accesses.

Any of the following attempted accesses are invalid:

- Non-32-bit accesses
- Writes to read-only registers
- Writes of incorrect values to SR when SWT is enabled
- Accesses to reserved addresses
- Accesses by initiators without permission

You control how SWT responds to an invalid access with CR[RIA].

66.6.1 SWT memory map

SWT_0 base address: 4027_0000h

SWT_1 base address: 4046_C000h

SWT_2 base address: 4047_0000h

SWT_3 base address: 4007_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Control (CR)	32	RW	FF00_010Ah
4h	Interrupt (IR)	32	RW	0000_0000h
8h	Timeout (TO)	32	RW	0000_0320h
Ch	Window (WN)	32	RW	0000_0000h
10h	Service (SR)	32	RW	0000_0000h
14h	Counter Output (CO)	32	R	0000_0000h
18h	Service Key (SK)	32	RW	0000_0000h
1Ch	Event Request (RRR)	32	RW	0000_0000h

66.6.2 Control (CR)

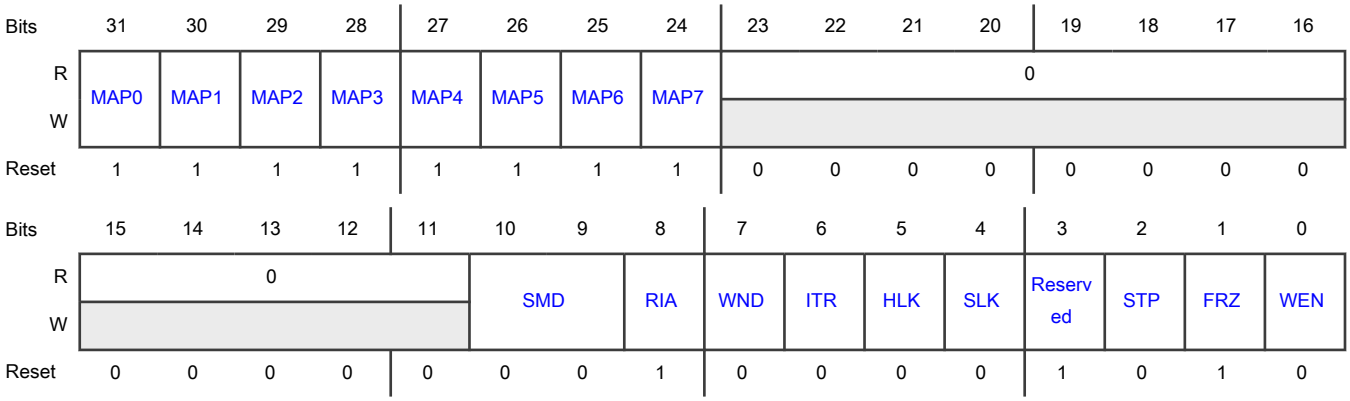
Offset

Register	Offset
CR	0h

Function

Contains fields for configuring and controlling SWT. The register is read-only if either hard lock or soft lock is enabled (either HLK or SLK is 1).

Diagram



Fields

Field	Function
31 MAP0	Initiator Access Protection 0 The platform bus initiator assignments are chip-specific. The number of this field corresponds to the XRDC DIDs of the bus initiator. MAP0 corresponds to XRDC DIDs 0 and 8. 0b - Access disabled 1b - Access enabled
30 MAP1	Initiator Access Protection 1 See the description for MAP0. MAP1 corresponds to XRDC DIDs 1 and 9. 0b - Access disabled 1b - Access enabled
29 MAP2	Initiator Access Protection 2 See the description for MAP0. MAP2 corresponds to XRDC DIDs 2 and 10. 0b - Access disabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Access enabled
28 MAP3	Initiator Access Protection 3 See the description for MAP0. MAP3 corresponds to XRDC DIDs 3 and 11. 0b - Access disabled 1b - Access enabled
27 MAP4	Initiator Access Protection 4 See the description for MAP0. MAP4 corresponds to XRDC DIDs 4 and 12. 0b - Access disabled 1b - Access enabled
26 MAP5	Initiator Access Protection 5 See the description for MAP0. MAP5 corresponds to XRDC DIDs 5 and 13. 0b - Access disabled 1b - Access enabled
25 MAP6	Initiator Access Protection 6 See the description for MAP0. MAP6 corresponds to XRDC DIDs 6 and 14. 0b - Access disabled 1b - Access enabled
24 MAP7	Initiator Access Protection 7 See the description for MAP0. MAP7 corresponds to XRDC DIDs 7 and 15. 0b - Access disabled 1b - Access enabled
23-11 —	Reserved
10-9 SMD	Service Mode 00b - Fixed Service Sequence. To service the watchdog, write the fixed sequence 0xA602, 0xB480 to SR. 01b - Keyed Service Sequence. To service the watchdog, write two pseudorandom key values to SR. 10b - Reserved. Do not use this value. Writing this value can cause the watchdog to not be serviced. 11b - Reserved. Do not use this value. Writing this value can cause the watchdog not to be serviced.

Table continues on the next page...

Table continued from the previous page...

Field	Function
8 RIA	Reset on Invalid Access Controls how SWT responds to an invalid access. **NOTE** For a description of how this chip implements SWT reset requests resulting from SWT invalid accesses, see the chip-specific SWT information. 0b - Generate a bus error 1b - Generate a bus error and reset request. If SWT is enabled (WEN = 1), generate a bus error and a reset request; otherwise, generate a bus error.
7 WND	Window Mode Specify regular or window mode operation. 0b - Regular mode. Can execute service sequence at any time 1b - Window mode. Can execute service sequence only when the timeout counter is less than the value in WN
6 ITR	Interrupt Then Reset Request **NOTE** For a description of how this chip implements SWT reset requests and interrupt requests resulting from SWT timeouts, see the chip-specific SWT information. 0b - Generate a reset request on a timeout 1b - Generate an interrupt on an initial timeout; generate a reset request on a second consecutive timeout
5 HLK	Hard Lock Indicates that the hard lock is enabled. You cannot directly write 0 to this field. This field becomes 0 only after a reset. 0b - CR, TO, WN, and SK are read/write registers if SLK is also 0 1b - CR, TO, WN, and SK are read-only registers
4 SLK	Soft Lock Indicates that the soft lock is enabled. You cannot directly write 0 to this field. Clear this field by writing the unlock sequence to the service register. 0b - CR, TO, WN, and SK are read/write registers if HLK is also 0 1b - CR, TO, WN, and SK are read-only registers
3 —	Reserved
2	Stop Mode Control

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>STP</td><td>Controls the watchdog timer when the chip enters Stop or Standby mode.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>SWT_0</td><td>CR</td><td>—</td></tr><tr><td>SWT_1</td><td>—</td><td>CR</td></tr><tr><td>SWT_2</td><td>—</td><td>CR</td></tr><tr><td>SWT_3</td><td>—</td><td>CR</td></tr></table>0b - Timer continues
1b - Timer stops</td></tr>
<tr><td>1
FRZ</td><td>Debug Mode Control
Controls the watchdog timer when the core enters Debug mode.
0b - Timer continues
1b - Timer stops</td></tr>
<tr><td>0
WEN</td><td>Watchdog Enable
Enables or disables SWT.
The reset value is 0. Therefore, after reset, you must enable SWT to start the countdown timer.
0b - Disabled
1b - Enabled</td></tr>
</table>

66.6.3 Interrupt (IR)

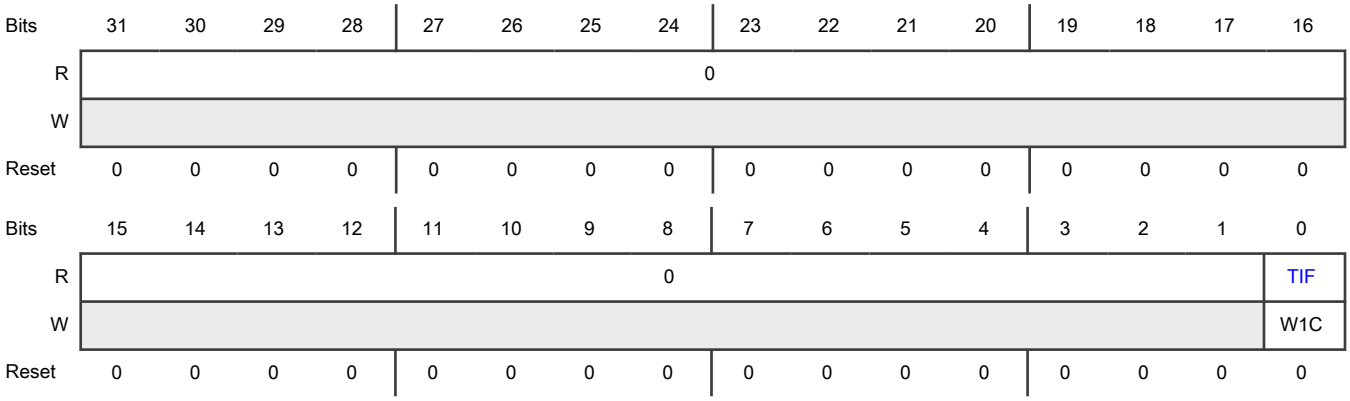
Offset

Register	Offset
IR	4h

Function

The timeout interrupt flag.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 TIF	Timeout Interrupt Flag Write 1 to this field to clear the flag and interrupt. Writing a 0 has no effect. 0b - No interrupt request 1b - Interrupt request due to an initial timeout

66.6.4 Timeout (TO)

Offset

Register	Offset
TO	8h

Function

Contains the 32-bit timeout period. The register is read-only if either hard lock or soft lock is enabled (CR[HLK] or CR[SLK] is 1).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	WTO															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	WTO															
Reset	0	0	0	0	0	0	1	1	0	0	1	0	0	0	0	0

Fields

Field	Function
31-0 WTO	Watchdog Timeout Watchdog timeout period in clock cycles. When software writes a service sequence or enables SWT, SWT loads the internal 32-bit countdown timer with this value or 3h, whichever is greater.

66.6.5 Window (WN)

Offset

Register	Offset
WN	Ch

Function

Contains the 32-bit window start value. SWT clears this register on reset. The register is read-only if either hard lock or soft lock is enabled (CR[HLK] or CR[SLK] is 1).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	WST															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	WST															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 WST	Window Start Value When you enable window mode (CR[WND]), you can write the service sequence only when the internal timer is less than this value.

66.6.6 Service (SR)

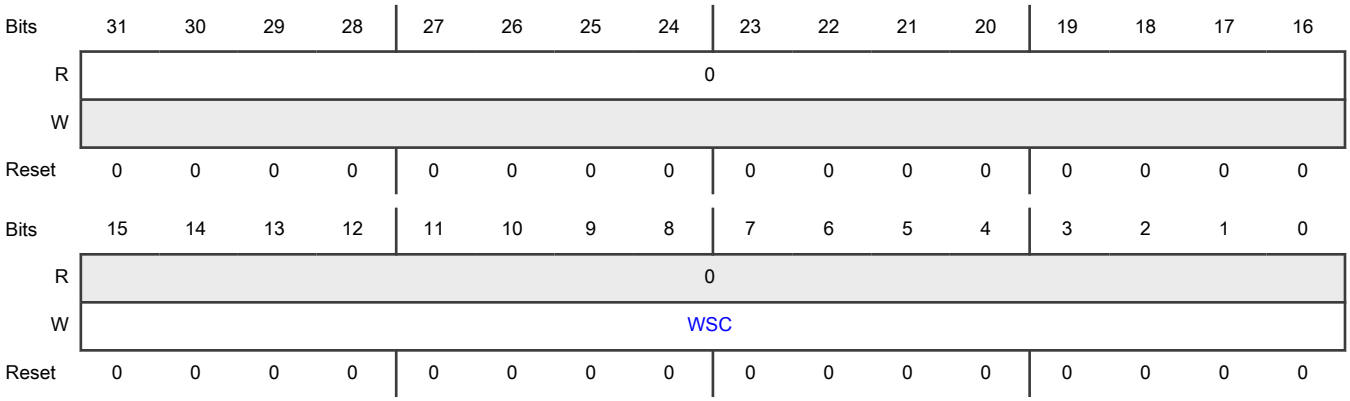
Offset

Register	Offset
SR	10h

Function

Initiates the service operation and resets the watchdog timer.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 WSC	Watchdog Service Code Use this field to service the watchdog and to unlock the Soft Lock (CR[SLK]). To service the watchdog: If SWT is in keyed service mode (CR[SMD]), write two pseudorandom key values to WSC (see Service key generation for details). Otherwise, write the following values to WSC, in the order shown:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1. A602h 2. B480h To unlock the Soft Lock (CR[SLK]), write the following values to WSC, in the order shown: 1. C520h 2. D928h When read, WSC always returns zero.

66.6.7 Counter Output (CO)

Offset

Register	Offset
CO	14h

Function

Shows the value of the internal timer when SWT is disabled.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CNT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CNT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 CNT	Watchdog Count When SWT is disabled (CR[WEN] is 0), CNT shows the value of the internal timer. When SWT is enabled (CR[WEN] is 1), it writes 0 to CNT. Values in this field can lag behind the internal timer value up to six system clock cycles plus eight counter clock cycles. Therefore, the CNT value that is read immediately after disabling SWT may be higher than the actual value of the internal timer.

66.6.8 Service Key (SK)

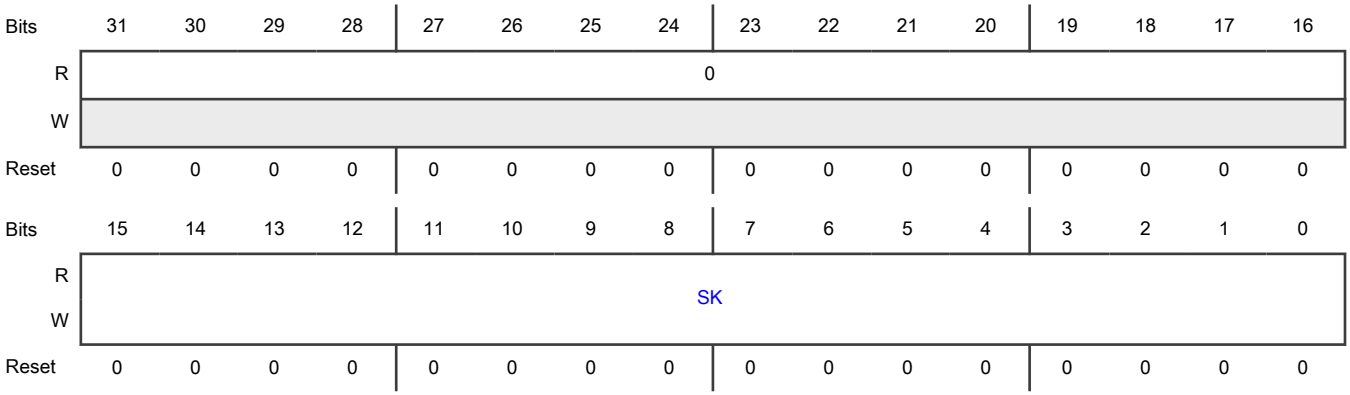
Offset

Register	Offset
SK	18h

Function

Holds the previous (or initial) service key value. This register is read-only if either hard lock or soft lock is enabled (CR[HLK] or CR[SLK] is 1).

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 SK	Service Key Holds the previous (or initial) service key value used in Initiate a keyed service sequence. If CR[SMD] is 01b, the next key value to write to SR is (17 * SK + 3) mod 2^{16}.

66.6.9 Event Request (RRR)

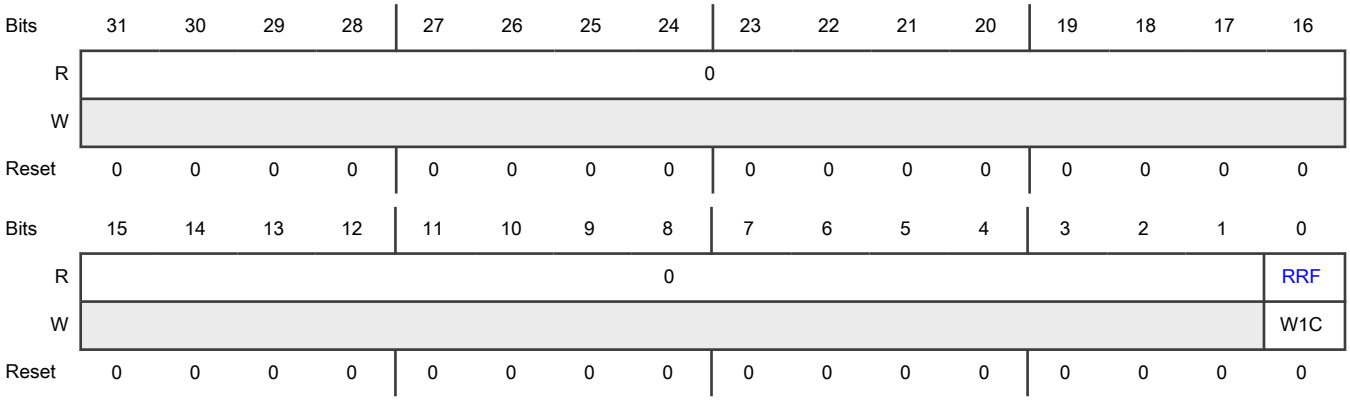
Offset

Register	Offset
RRR	1Ch

Function

Contains the timeout reset request flag. See the chip-specific information for the specific event associated with this flag.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 RRF	Reset Request Flag Write 1 to clear the flag and request. Writing 0 has no effect. 0b - No reset request 1b - Any reset request initiated

Chapter 67
System Timer Module (STM)

67.1 Chip-specific STM information

67.1.1 STM instances and configuration

This chip has up to four instances of STM, one for each Cortex-M7 core. The STM counter increments at the STM module clock frequency divided by a pre-scaled value.

Table 433. STM instances

Instance	S32K388/ S32K389	S32K358/ S32K356/ S32K348/ S32K338/ S32K328	S32K322/S32K342/S32K341/S32K314/ S32K324/S32K344	S32K310/S32K311/S32K312
STM_0	Yes	Yes	Yes	Yes
STM_1	Yes	Yes	Yes	No
STM_2	Yes	Yes	No	No
STM_3	Yes	No	No	No

Table 434. STM configuration

Instance	Description	Number of Channels	Number of Registers (32-bit)
STM_0	Intended for Cortex-M7_0 core	4	14
STM_1	Intended for Cortex-M7_1 core	4	14
STM_2	Intended for Cortex-M7_2 core	4	14
STM_3	Intended for Cortex-M7_3 core	4	14

67.2 Overview

STM supports commonly required system and application software timing functions. STM includes a 32-bit count-up timer and four 32-bit compare channels with a separate interrupt source for each channel. The timer is driven by the STM module clock divided by an 8-bit prescale value (1 to 256).

67.2.1 Block diagram

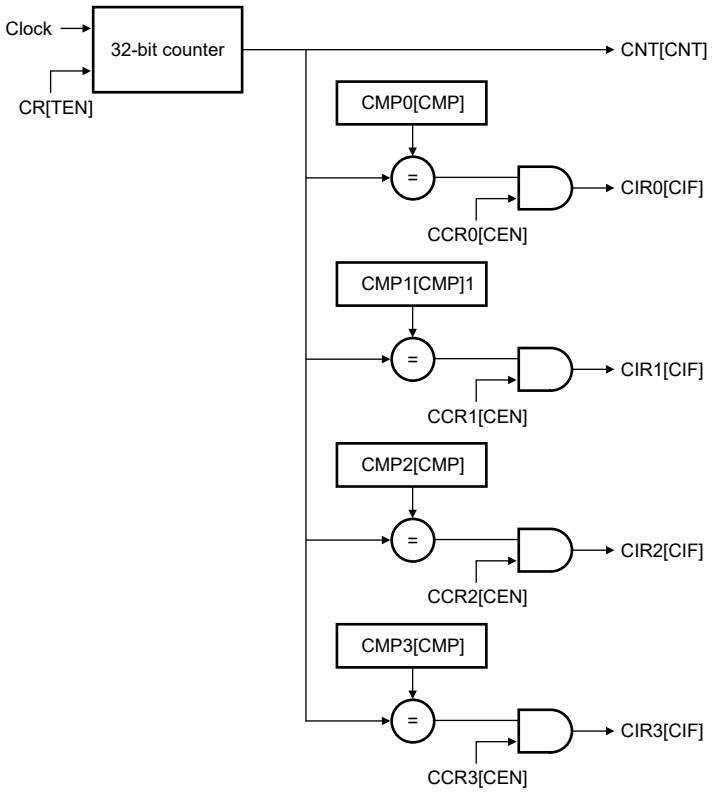


Figure 381. Block diagram

67.2.2 Features

STM has the following features:

- One 32-bit count-up timer with an 8-bit prescaler
- Four 32-bit compare channels
- An independent interrupt source for each channel
- Ability to stop the timer in Debug mode

67.3 Functional description

67.3.1 Count-up timer

STM has one 32-bit count-up timer that serves as the time base for four compare channels. When enabled, the counter increments at the module clock frequency divided by a prescaler value in the range from 1 to 256. When enabled in Normal mode, the timer increments continuously. The counter rolls over at FFFF_FFFFh to 0000_0000h with no restrictions at this boundary.

67.3.2 Compare channels

STM has four identical compare channels. Each channel includes a channel control register (CCR*n*), a channel interrupt register (CCR*n*), and a channel compare register (CMP*n*). When the channel is enabled and its channel compare value matches the timer count, STM sets the channel interrupt flag and generates an IRQ on that channel.

67.3.3 Behavior in different chip modes

STM supports the chip modes of operation as follows:

Chip mode	STM behavior
Normal	When the timer is enabled (CR[TEN] = 1), the timer counts up continuously.
Debug	If CR[FRZ] = 1, STM stops the timer. Otherwise, when the timer is enabled (CR[TEN] = 1), the timer counts up continuously.

67.3.4 Clocking

This module has no clocking considerations.

67.3.5 Interrupts

STM can generate a channel interrupt. For information, see:

- Compare channels
- Respond to compare channel events
- Channel Interrupt (CIR0 - CIR3)

67.4 External signals

This module has no external signals.

67.5 Initialization

This module does not require initialization.

67.6 Application information

67.6.1 Configure the timer

1. Set the initial timer count (CNT[CNT]).
2. Specify STM behavior in chip Debug mode (CNT[FRZ]).
3. Set the counter prescaler (CR[CPS]).
4. Start the timer (CR[TEN]).

67.6.2 Configure the compare channels

For each compare channel:

1. Set the channel compare value (CMP*n*[CMP]).
2. Enable the compare channel (CCR*n*[CEN]).

67.6.3 Respond to compare channel events

For each compare channel:

1. Check the channel interrupt flag (CIR*n*[CIF]).
2. If the channel interrupt flag is set, respond to the interrupt request.
3. When the channel interrupt has been handled, clear the channel interrupt flag (CIR*n*[CIF]).

67.7 STM register descriptions

The STM programming model allows only 32-bit (word) accesses. An attempted reference using a different size or to a reserved address generates a bus error termination.

67.7.1 STM memory map

STM_0 base address: 4027_4000h

STM_1 base address: 4047_4000h

STM_2 base address: 4047_8000h

STM_3 base address: 4047_C000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Control (CR)	32	RW	0000_0000h
4h	Count (CNT)	32	RW	0000_0000h
10h	Channel Control (CCR0)	32	RW	0000_0000h
14h	Channel Interrupt (CIR0)	32	RW	0000_0000h
18h	Channel Compare (CMP0)	32	RW	0000_0000h
20h	Channel Control (CCR1)	32	RW	0000_0000h
24h	Channel Interrupt (CIR1)	32	RW	0000_0000h
28h	Channel Compare (CMP1)	32	RW	0000_0000h
30h	Channel Control (CCR2)	32	RW	0000_0000h
34h	Channel Interrupt (CIR2)	32	RW	0000_0000h
38h	Channel Compare (CMP2)	32	RW	0000_0000h
40h	Channel Control (CCR3)	32	RW	0000_0000h
44h	Channel Interrupt (CIR3)	32	RW	0000_0000h
48h	Channel Compare (CMP3)	32	RW	0000_0000h

67.7.2 Control (CR)

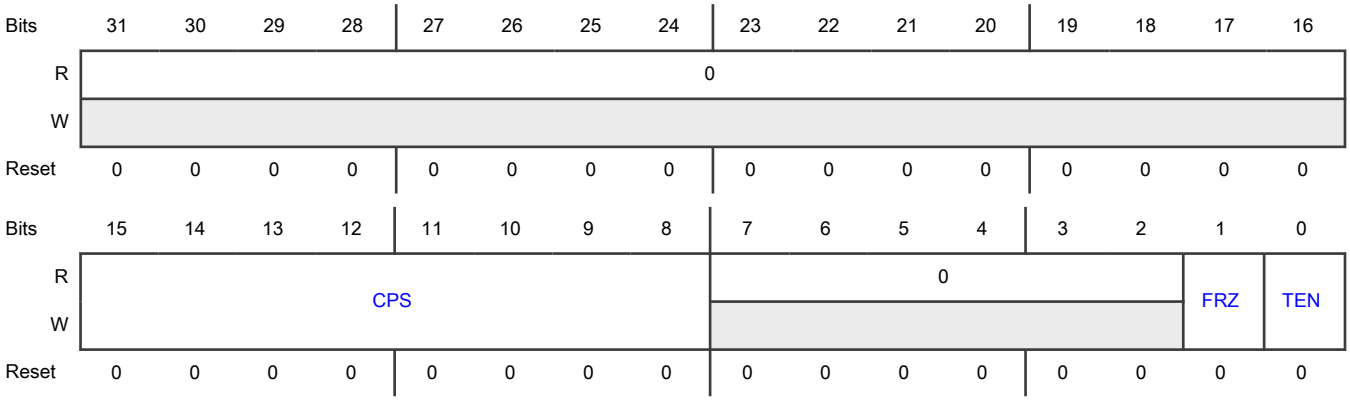
Offset

Register	Offset
CR	0h

Function

Contains fields for the prescale value, freeze control, and timer enable.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-8 CPS	Counter Prescaler Selects the module clock divide value for the prescaler (1–256). • 00h - Divide module clock by 1 • 01h - Divide module clock by 2 • ... • FFh - Divide module clock by 256
7-2 —	Reserved
1 FRZ	Freeze Stops the timer when the chip enters Debug mode. **NOTE** When the chip enters Debug mode, it notifies STM, which in turn uses this field to determine timer operation. 0b - Timer runs in Debug mode 1b - Timer stops in Debug mode
0 TEN	Timer Enable Enables the module timer. 0b - Disabled 1b - Enabled

67.7.3 Count (CNT)

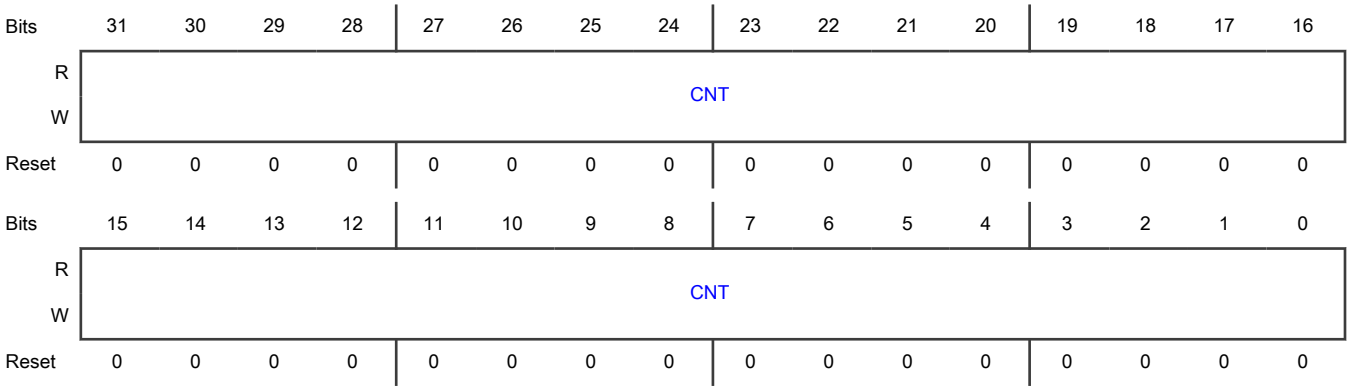
Offset

Register	Offset
CNT	4h

Function

Holds the timer count value.

Diagram



Fields

Field	Function
31-0 CNT	Timer Count The time base for all compare channels. When enabled, the timer count increments at the rate of the module clock divided by the prescale value.

67.7.4 Channel Control (CCR0 - CCR3)

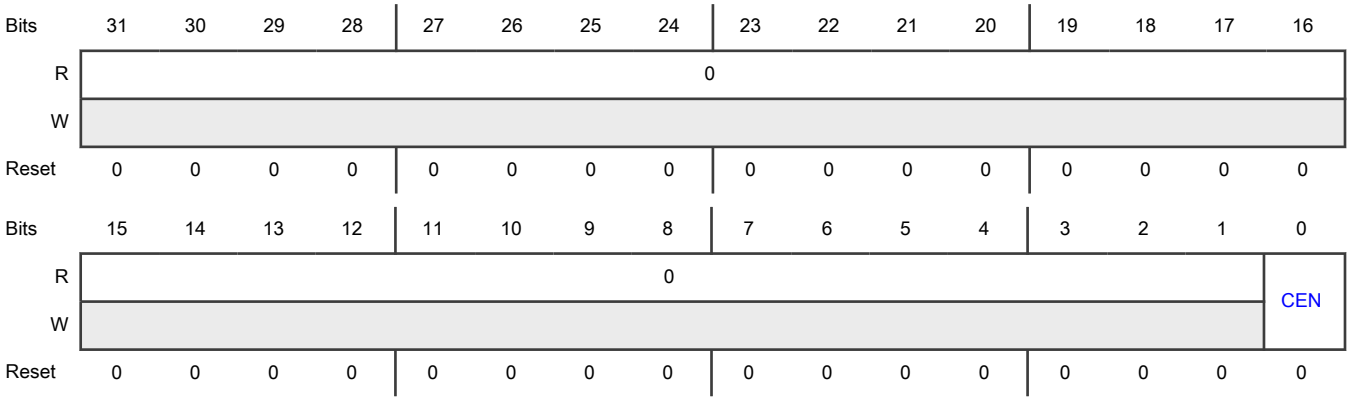
Offset

Register	Offset
CCR0	10h
CCR1	20h
CCR2	30h
CCR3	40h

Function

Enables channel n of the timer.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 CEN	Channel Enable 0b - Disabled 1b - Enabled

67.7.5 Channel Interrupt (CIR0 - CIR3)

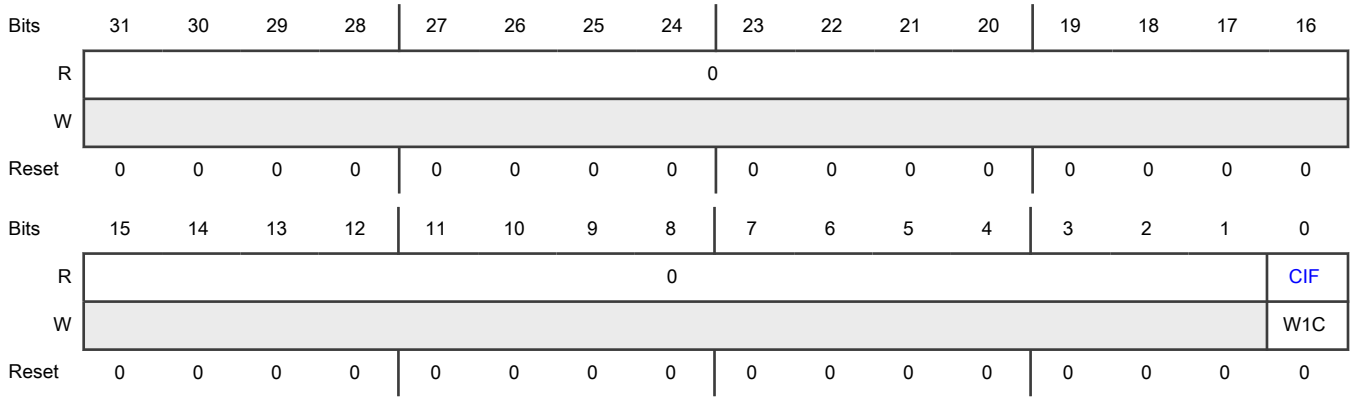
Offset

Register	Offset
CIR0	14h
CIR1	24h
CIR2	34h
CIR3	44h

Function

Indicates and clears the interrupt flag for channel n of the timer.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 CIF	Channel Interrupt Flag Indicates the channel IRQ is asserted due to a match on the channel. 0b - Read: IRQ is not asserted. Write: No effect. 1b - Read: IRQ is asserted. Write: Clear the flag.

67.7.6 Channel Compare (CMP0 - CMP3)

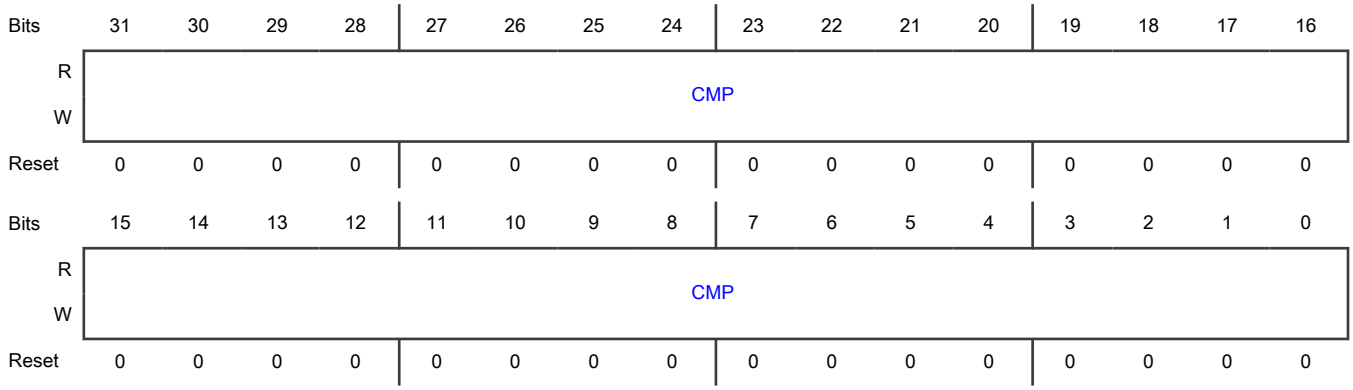
Offset

Register	Offset
CMP0	18h
CMP1	28h
CMP2	38h
CMP3	48h

Function

The compare value for channel n.

Diagram



Fields

Field	Function
31-0 CMP	Channel Compare If the channel is enabled (CCR*n*[CEN]), when the timer count (CNT) matches this value, STM asserts the channel IRQ and sets the channel interrupt flag (CIR*n*[CIF]).

67.8 Glossary

IRQ Interrupt request

Chapter 68
Periodic Interrupt Timer (PIT)

68.1 Chip-specific PIT information

68.1.1 PIT instances and configuration

This chip has up to four instances of the PIT each with 4 PIT timers/channels, each channel being 32 bits in length.

Table 435. PIT instances

Instance	S32K388/S32K389	S32K358/S32K356/ S32K348/S32K338/ S32K328/S32K324/ S32K344/S32K314/ S32K342/ S32K341/S32K322	S32K312	S32K311/S32K310
PIT_0	Yes	Yes	Yes	Yes
PIT_1	Yes	Yes	Yes	Yes
PIT_2	Yes	Yes	No	No
PIT_3	Yes	No	No	No

Table 436. PIT instances and features

Feature	PIT_0	PIT_1	PIT_2	PIT_3
Real time interrupt (RTI)	Yes (32-bit)[1]	No	No	No
Lifetime Timer	Yes	No	No	No
Timer chaining to create a 64-bit timer	Yes	No	No	No
Number of 32-bit timer channels	4	4	4	4
Timer value unchanged by a non-destructive reset	No	No	No	No

1. The RTI on PIT_0 supports operation in STANDBY mode

The RTI shall be possible to set the timer resolution to 1us independently to other timers. The interrupt can be generated by programming LDVAL0, LDVAL1, LDVAL2 and LDVAL3 to 48 in case of 48 MHz FIRC as system clock and 40 in case of 40 MHz system clock to generate an interrupt after 1us.

68.1.2 Input Output

All the PIT instances are capable of generating periodic triggers. PIT triggers are routed to motor control IPs such as eMIOS, LCU, BCTU, ADC etc. via TRGMUX. See the TRGMUX connectivity file attached to this document for details.

68.2 Overview

PIT includes:

- An array of timers that can be used to raise interrupts and trigger DMA channels.
- A dedicated RTI timer that runs on a separate oscillator clock and can be used for system wakeup

68.2.1 Block diagram

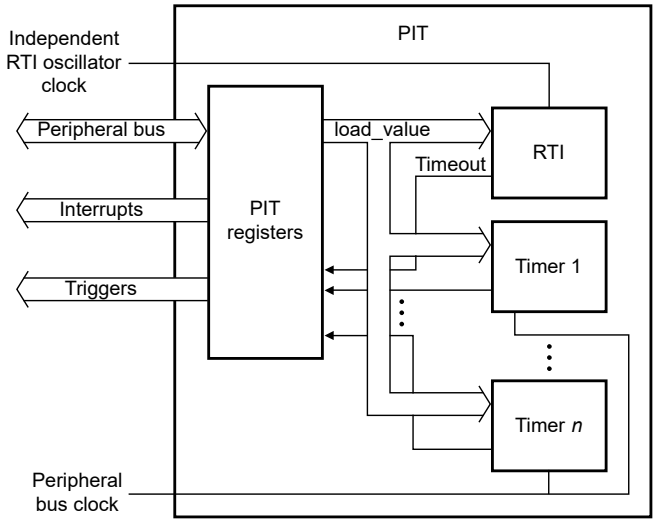


Figure 382. Block diagram

68.2.2 Features

The key features of the module are:

- 4 32-bit countdown timers
- Independent timeout periods for each timer
- Ability to chain two timers into a 64-bit lifetimer
- Ability of timers to generate interrupts
- Ability of timers to generate DMA trigger pulses
- Maskable interrupts
- One RTI countdown timer to wake up the CPU
- Option to generate the RTI even when the bus clock is off
- Power saving with a separate input clock for the RTI timer. All other timers share a common core clock.

68.3 Functional description

See the chip-specific PIT information for the number of PIT modules and their channel configuration on this chip.

68.3.1 Modes of operation

Mode	Description
Run	All functional parts of the PIT operate normally.
Debug	Timers run or stop depending on the Freeze setting (MCR[FRZ]). See Debug mode.

68.3.2 Timer operation

When you enable a timer (write 1 to TCTRL*n*[TEN] or RTI_TCTRL[TEN]), the timer counts down (one count per clock cycle) from its initial start value (LDVAL*n*[TSV] or RTI_LDVAL[TSV]). When the timer period expires (CVAL*n*[TVL] or RTI_CVAL[TVL] reaches 0), PIT sets the Timer Interrupt Flag (writes 1 to TFLG*n*[TIE] or RTI_TFLG[TIE]). It then reloads the timer start value and starts the timer counting down again.

If the timer interrupt is enabled (TCTRL*n*[TIE] or RTI_TCTRL[TIE] = 1), it generates an interrupt when it sets the Timer Interrupt Flag. The timer cannot generate a new interrupt until you clear the flag from the previous interrupt. Because clearing the RTI crosses clock domains, you should use a minimum RTI timer load value of 32.

You can read the current counter value of any timer (CVAL*n*[TVL] or RTI_CVAL[TVL]). Because the RTI timer is synchronized to the RTI clock domain, the value of the RTI counter may be several cycles old when read.

See these related topics:

- Stop and start a timer
- Change timer period
- Change timer period dynamically

NOTE

- If Freeze is enabled (MCR[FRZ] = 1) when the timer nears 0, the Debug mode command may not reach PIT before the timer expires and generates a trigger.
- If a timer is at zero when Debug mode is asserted, the interrupt trigger remains asserted until the device exits Debug mode. Clear the Timer Interrupt Flag (write 1 to TFLG*n*[TIF] or RTI_TFLG[TIF]) upon Debug exit.
- If Freeze is enabled (MCR[FRZ] = 1), sufficient time must be ensured for the IP to react.

68.3.2.1 Stop and start a timer

To restart a running timer:

1. Disable the timer (write 0 to TCTRL*n*[TEN] or RTI_TCTRL[TEN]).
2. Enable the timer (write 1 to TCTRL*n*[TEN] or RTI_TCTRL[TEN]).

   Enabling the timer reloads the start value and starts the timer counting down again. See Restart counter period.

68.3.2.2 Restart counter period

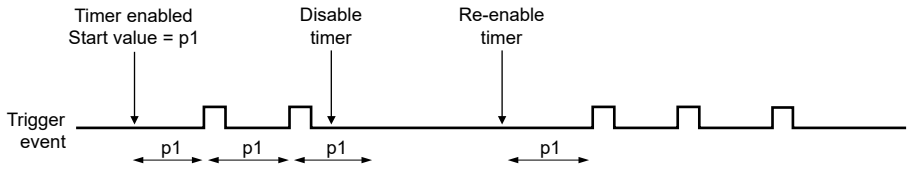


Figure 383. Restart counter period

68.3.2.3 Change timer period

To change the counter period of a running timer:

1. Disable the timer (write 0 to TCTRL*n*[TEN] or RTI_TCTRL[TEN]).
2. Specify a new start value (LDVAL*n*[TSV] or RTI_LDVAL[TSV]).
3. Enable the timer (write 1 to TCTRL*n*[TEN] or RTI_TCTRL[TEN]).

   Enabling the timer loads the new start value and starts the timer counting down again. See Change current timer period.

68.3.2.4 Change current timer period

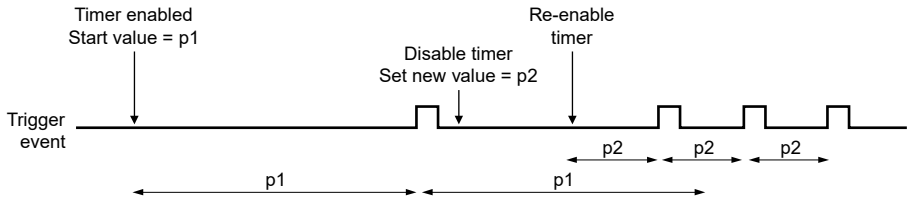


Figure 384. Change timer period dynamically timing

68.3.2.5 Change timer period dynamically

To change the counter period of a running timer:

- Specify a new start value (LDVAL*n*[TSV] or RTI_LDVAL[TSV])

  The next time the timer expires, it loads the new start value. See Dynamically setting a new timer period.

68.3.2.6 Dynamically setting a new timer period

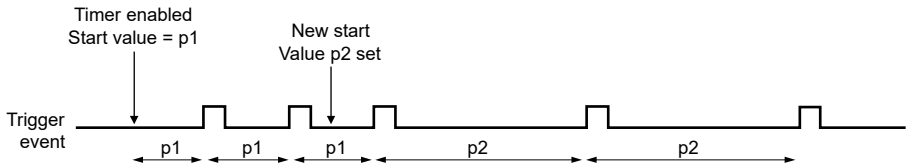


Figure 385. Dynamically setting a new timer period

68.3.3 Chained timers

PIT supports chaining two timers together to support timer periods greater than FFFFFFFFh. When a timer *n* has chain mode enabled, it decrements each time timer *n* - 1 expires. The chained timer sets its Timer Interrupt Flag when it expires (CVAL*n* = 0).

For example, assume you need to configure a timer period of 6,000,000,000 cycles. You could specify a start value of 10 for timer *n* and 599,999,999 for timer *n* - 1. With these values, the chained timer expires after timer *n* - 1 has expired ten times.

You cannot chain timer 0 or the RTI timer to another timer.

For an example configuration of a chained timer, see Example configuration using chained timers.

68.3.4 Lifetimer

PIT supports a 64-bit lifetimer. You configure the lifetimer by chaining timer 1 to timer 0 and populating the start values for both timers with the maximum value of FFFFFFFFh. You read the lifetimer in the LTMR64H and LTMR64L registers. LTMR64H returns the upper 32 bits of the lifetimer value and LMTR64L returns the lower 32 bits. When a timer 1 has chain mode enabled, it decrements each time timer 0 expires. The lifetimer expires when CVAL1 expires.

When you chain timer 1 to timer 0, timer 1 decrements every time timer 0 expires. To obtain the correct lifetimer value, read LTMR64H first and then read LTMR64L. When you read LTMR64H, it returns the value in CVAL1 (the chained timer value), and immediately latches the value from CVAL0 into LTMR64L. This latching eliminates the carryover effects of the running counter.

For an example configuration of the lifetimer, see Example configuration using the lifetimer.

68.3.5 Debug mode

If Freeze is enabled (MCR[FRZ] = 1), the timers stop if the device enters Debug mode. This freeze allows the software developer to halt the processor, investigate the current state of the system, including the PIT timer values, and then continue operation.

68.3.6 Clocking

The PIT general-purpose timers are driven by the peripheral bus clock. See the Clocking chapter for more information.

The RTI timer is driven by an independent oscillator clock. This clock input continues running when the device is in Stop mode. Driving the RTI timer with an independent clock allows the RTI to trigger periodic wake-up from Stop mode.

68.3.7 Interrupts

All of the timers support interrupt generation.

You enable interrupt generation for each individual timer (TCTRL*n*[TIE]). Each interrupt is available on a separate interrupt output.

When a timer expires (CVAL*n*[TVL] = 0), it sets the Timer Interrupt Flag (TFLG*n*[TIF]) regardless of whether the interrupt is enabled. To enable generation of an interrupt when the Timer Interrupt Flag is set, write 1 to TCTRL*n*[TIE]. To clear the Timer Interrupt Flag, write 1 to it.

When the RTI timer expires (RTI_CVAL*n*[TVL] = 0), PIT sets the RTI Timer Interrupt Flag (RTI_TFLG*n*[TIF]) regardless of whether the RTI is enabled. To enable generation of an RTI when the RTI Timer Interrupt Flag is set, write 1 to TCTRL*n*[TIE]. To clear the RTI Timer Interrupt Flag, write 1 to it.

The RTI is typically used for periodic wake-up from Standby mode, but it can also be used to generate a general-purpose interrupt.

68.4 External signals

PIT has no external inputs.

68.5 Initialization

To initialize PIT:

1. For each timer *n* you plan to use:
    a. Specify the timer period (LDVAL*n*[TSV]).

       To calculate the load value for a timer:

       Start value = (timer period / clock period) - 1

    b. If you want the timer to generate an interrupt each time it expires, enable the timer interrupt (write 1 to TCTRL*n*[TIE]).
    c. If you want to use this timer as the divisor of a pair of chained timers, chain this timer to the next lower-numbered timer (write 1 to TCTRL*n*[CHN]). You cannot chain timer 0. If you want to use the lifetimer, chain timer 1 to timer 0.
    d. Enable the timer (write 1 to TCTRL*n*[TEN]).
2. If you want to use the RTI feature:
    a. Specify the RTI timer period (RTI_LDVAL*n*[TSV]).
    b. Enable the RTI timer interrupt (write 1 to RTI_TCTRL*n*[TIE]).
    c. Enable the RTI timer (write 1 to RTI_TCTRL*n*[TEN]).
3. Activate, or enable, PIT (write 0 to MCR[MDIS]).

   All enabled timers load their start values and begin counting down.

For example configurations, see Application information.

68.6 Application information

68.6.1 Example configuration for general timers

In this example configuration:

- The PIT clock frequency is 50 MHz, which equates to a clock cycle of 20 ns.
- The RTI clock frequency is 10 MHz, which equates to a clock cycle of 100 ns.
- The RTI timer needs to generate a wake-up interrupt every 500 ms.
- Timer 1 needs to generate an interrupt every 5.12 ms.
- Timer 3 needs to generate a trigger event every 30 ms.

To configure PIT for this example:

1. Disable PIT (write 1 to MCR[MDIS]).
2. Use the formula in Initialization to calculate the load value for the RTI timer:

   (500 ms/100 ns) - 1 = 4,999,999 cycles, or 004C4B3Fh.
3. Specify the RTI timer load value (write 004C4B3Fh to RTI_LDVAL[TSV]).
4. Enable the RTI (write 1 to RTI_TCTRL[TIE])
5. Enable the RTI timer (write 1 to RTI_TCTRL*n*[TEN]).
6. Use the formula in Initialization to calculate the load value for timer 1:

   (5.12 ms/20 ns) - 1 = 255,999 cycles, or 0003E7FFh.
7. Disable timer 1 (write 0 to TCTRL1[TEN]).
8. Specify the timer 1 value (write 0003E7FFh to LDVAL1[TSV]).
9. Enable the timer 1 interrupt (write 1 to TCTRL1[TIE])
10. Enable timer 1 (write 1 to TCTRL1[TEN]).
11. Calculate the load value for timer 3:

    (30 ms/20 ns) - 1 = 1,499,999 cycles, or 0016E35Fh.
12. Disable timer 3 (write 0 to TCTRL3[TEN]).
13. Specify the timer 3 value (write 0016E35Fh to LDVAL3[TSV]).
14. Enable timer 3 (write 1 to TCTRL3[TEN]).
15. Activate, or enable, PIT (write 0 to MCR[MDIS]).

NOTE

You can configure timers without first disabling PIT or the timer itself. However, if PIT is running, be aware that when a timer begins using a new start value depends on whether the timer is running or disabled when you specify the new start value. For more information, see Timer operation.

The following example code provides the described configuration:

```
// Disable PIT (see note above)
PIT_MCR = 0x02;
```

```
// RTI
PIT_RTI_LDVAL = 0x004C4B3F; // Configure RTI for 5000000 cycles
```

```
PIT_RTI_TCTRL = PIT_TIE;    // Enable RTI
PIT_RTI_TCTRL |= PIT_TEN;   // Start RTI timer
```

```
// Timer 1
PIT_LDVAL1 = 0x0003E7FF;    // Configure timer 1 for 256000 cycles
PIT_TCTRL1 = TIE;           // Enable Timer 1 interrupts
PIT_TCTRL1 |= TEN;          // Start Timer 1
```

```
// Timer 3
PIT_LDVAL3 = 0x0016E35F;    // Configure timer 3 for 1500000 cycles
PIT_TCTRL3 = TEN;           // Enable Timer 3
```

```
 // Enable PIT
 PIT_MCR = 0x00;
```

68.6.2 Example configuration using chained timers

In this example configuration:

- The PIT clock frequency is 100 MHz, which equates to a clock cycle of 10 ns.
- Timers 1 and 2 are available.
- An interrupt is needed every minute. Using the formula in Initialization, a 1 minute timer period requires:

  (60 s/100 ns) - 1 = 5,999,999,999 cycles, or 165A0BBFFh

  This value requires more than 32 bits.

To configure PIT for this example:

1. Disable PIT (write 1 to MCR[MDIS]).
2. Disable timer 1 (write 0 to TCTRL1[TEN]).
3. Disable timer 2 (write 0 to TCTRL2[TEN]).
4. Chain timer 2 to timer 1 (write 1 to TCTRL2[CHN]).
5. For the timer 2 start value, select a value that divides evenly into the desired period. For this example, we are using 10. For the timer 1 value, divide the full timer period by the selected divisor: 6,000,000,000/10 = 600,000,000.
6. Specify the timer 2 value as the selected divisor (write 00000009h to LDVAL2[TSV]).
7. Specify the timer 1 value as the quotient minus 1, or 599,999,999 (write 0x23C345FFh to LDVAL1[TSV]).
8. Enable the timer 2 interrupt (write 1 to TCTRL2[TIE])
9. Enable timer 2 (write 1 to TCTRL2[TEN]).
10. Enable timer 1 (write 1 to TCTRL1[TEN]).
11. Activate, or enable, PIT (write 0 to MCR[MDIS]).

The following example code matches the described configuration:

```
// Disable PIT
PIT_MCR = 0x02;
```

```
// Timer 2
PIT_LDVAL2 = 0x00000009; // Configure Timer 2 to the divisor (10)
PIT_TCTRL2 = TIE;        // Enable Timer 2 interrupt
PIT_TCTRL2 |= CHN;       // Chain Timer 2 to Timer 1
PIT_TCTRL2 |= TEN;       // Enable Timer 2
```

```
// Timer 1
PIT_LDVAL1 = 0x23C345FF; // Configure Timer 1 to 600,000,000 cycles
PIT_TCTRL1 = TEN;        // Enable Timer 1
```

```
 // Enable PIT
 PIT_MCR = 0x00;
```

68.6.3 Example configuration using the lifetimer

To configure the lifetimer:

1. Disable PIT (write 1 to MCR[MDIS]).
2. Disable timer 0 (write 0 to TCTRL0[TEN]).
3. Disable timer 1 (write 0 to TCTRL1[TEN]).
4. Chain timer 1 to timer 0 (write 1 to TCTRL1[CHN]).
5. Specify the maximum start value for timer 1 (write FFFFFFFFh to LDVAL1[TSV]).
6. Specify the maximum start value for timer 0 (write FFFFFFFFh to LDVAL0[TSV]).
7. Enable timer 1 (write 1 to TCTRL1[TEN]).
8. Enable timer 0 (write 1 to TCTRL0[TEN]).
9. Activate, or enable, PIT (write 0 to MCR[MDIS]).

To access the lifetimer, first read LTMR64H and then LTMR64L. For more information, see Lifetimer.

The following example code matches the described setup:

```
// Disable PIT
PIT_MCR = 0x02;
```

```
// Timer 1
PIT_LDVAL1 = 0xFFFFFFFF; // setup timer 1 for maximum counting period
PIT_TCTRL1 = 0x0; // disable timer 1 interrupts
PIT_TCTRL1 |= CHN; // chain timer 1 to timer 0
PIT_TCTRL1 |= TEN; // start timer 1
```

```
// Timer 0
```

```
PIT_LDVAL0 = 0xFFFFFFFF; // setup timer 0 for maximum counting period
PIT_TCTRL0 = TEN; // start timer 0
```

```
// Read lifetimer
current_uptime = PIT_LTMR64H<<32;
current_uptime = current_uptime + PIT_LTMR64L;
```

```
 // Enable PIT
 PIT_MCR = 0x00;
```

68.7 PIT register descriptions

This section provides a detailed description of all PIT registers.

Program RTI registers only when the RTI clock is running.

68.7.1 PIT memory map

PIT_0 base address: 400B_0000h

PIT_1 base address: 400B_4000h

PIT_2 base address: 402F_C000h

PIT_3 base address: 4030_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	PIT Module Control (MCR)	32	RW	See section
E0h	PIT Upper Lifetimer (LTMR64H)	32	R	0000_0000h
E4h	PIT Lower Lifetimer (LTMR64L)	32	R	0000_0000h
ECh	RTI Timer Load Value Sync Status (RTI_LDVAL_STAT)	32	RW	0000_0000h
F0h	RTI Timer Load Value (RTI_LDVAL)	32	RW	0000_0000h
F4h	Current RTI Timer Value (RTI_CVAL)	32	R	0000_0000h
F8h	RTI Timer Control (RTI_TCTRL)	32	RW	0000_0000h
FCh	RTI Timer Interrupt Flag (RTI_TFLG)	32	RW	0000_0000h
100h	Timer Load Value (LDVAL0)	32	RW	0000_0000h
104h	Current Timer Value (CVAL0)	32	R	0000_0000h
108h	Timer Control (TCTRL0)	32	RW	0000_0000h
10Ch	Timer Flag (TFLG0)	32	RW	0000_0000h
110h	Timer Load Value (LDVAL1)	32	RW	0000_0000h
114h	Current Timer Value (CVAL1)	32	R	0000_0000h
118h	Timer Control (TCTRL1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
11Ch	Timer Flag (TFLG1)	32	RW	0000_0000h
120h	Timer Load Value (LDVAL2)	32	RW	0000_0000h
124h	Current Timer Value (CVAL2)	32	R	0000_0000h
128h	Timer Control (TCTRL2)	32	RW	0000_0000h
12Ch	Timer Flag (TFLG2)	32	RW	0000_0000h
130h	Timer Load Value (LDVAL3)	32	RW	0000_0000h
134h	Current Timer Value (CVAL3)	32	R	0000_0000h
138h	Timer Control (TCTRL3)	32	RW	0000_0000h
13Ch	Timer Flag (TFLG3)	32	RW	0000_0000h

68.7.2 PIT Module Control (MCR)

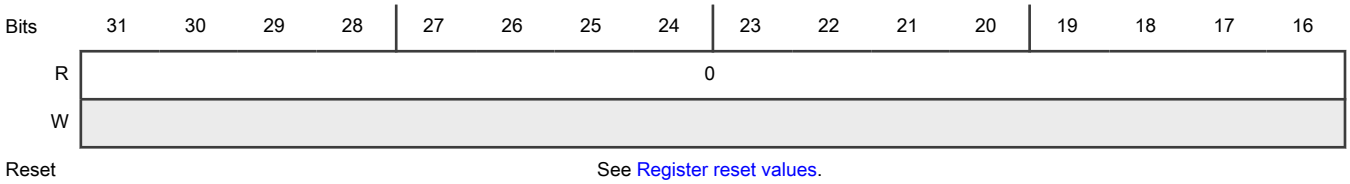
Offset

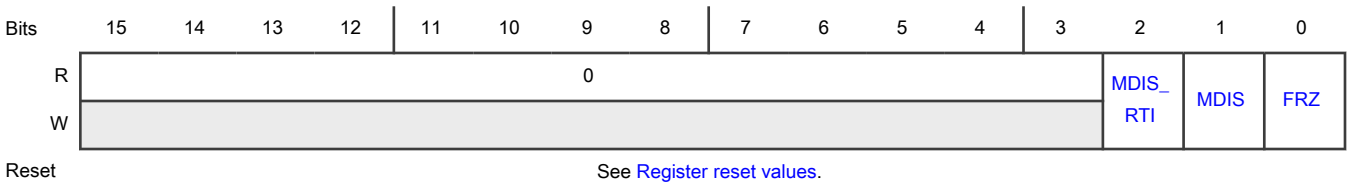
Register	Offset
MCR	0h

Function

Enables the PIT timer clocks and specifies the behavior of the timers when PIT enters Debug mode.

Diagram





Register reset values

Register	Reset value
MCR	PIT_0: 0000_0006h PIT_1–PIT_3: 0000_0002h

Fields

Field	Function
31-3 —	Reserved
2 MDIS_RTI	Module Disable for RTI Disables the RTI timer. You must write 1 to this field before setting up an RTI. **NOTE** This field is not supported in every instance. The following table includes only supported registers.

Instance	Field supported in	Field not supported in
PIT_0	MCR	—
PIT_1	—	MCR
PIT_2	—	MCR
PIT_3	—	MCR

	0b - Enables 1b - Disables
1 MDIS	Module Disable for PIT Disables the standard timers. This field does not affect the RTI timer. You must disable the clock (write 1 to MDIS) before configuring PIT. After configuration, you must enable the clock (write 0 to MDIS). 0b - Enables 1b - Disables
0 FRZ	Freeze Stops the timers when the device enters Debug mode. 0b - Timers run in Debug mode 1b - Timers stop in Debug mode

68.7.3 PIT Upper Lifetimer (LTMR64H)

Offset

Register	Offset
LTMR64H	E0h

Function

Combined with LTMR64L, provides a 64-bit lifetimer constructed from the values in two chained PIT timers.

For more information, see Lifetimer.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
PIT_0	LTMR64H	—
PIT_1	—	LTMR64H
PIT_2	—	LTMR64H
PIT_3	—	LTMR64H

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LTH															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LTH															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 LTH	Lifetimer Value Indicates the timer value of timer 1. This value is the upper 32 bits of the 64-bit lifetimer value. When you read this register at t1, the value of timer 0 at t1 is latched into LTMR64L.

68.7.4 PIT Lower Lifetimer (LTMR64L)

Offset

Register	Offset
LTMR64L	E4h

Function

Combined with LTMR64H, provides a 64-bit lifetimer.

For more information, see Lifetimer.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
PIT_0	LTMR64L	—
PIT_1	—	LTMR64L
PIT_2	—	LTMR64L
PIT_3	—	LTMR64L

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LTL															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LTL															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 LTL	Lifetimer Value Indicates the timer value of timer 0 at the moment LTMR64H was last read. This value is the lower 32 bits of the 64-bit lifetimer value. This field updates only when LTMR64H is read.

68.7.5 RTI Timer Load Value Sync Status (RTI_LDVAL_STAT)

Offset

Register	Offset
RTI_LDVAL_STAT	ECh

Function

Indicates whether the RTI timer start value has been loaded (RTI_LDVAL). This status is necessary because it takes several cycles for the RTI start value to synchronize into the RTI clock domain.

NOTE

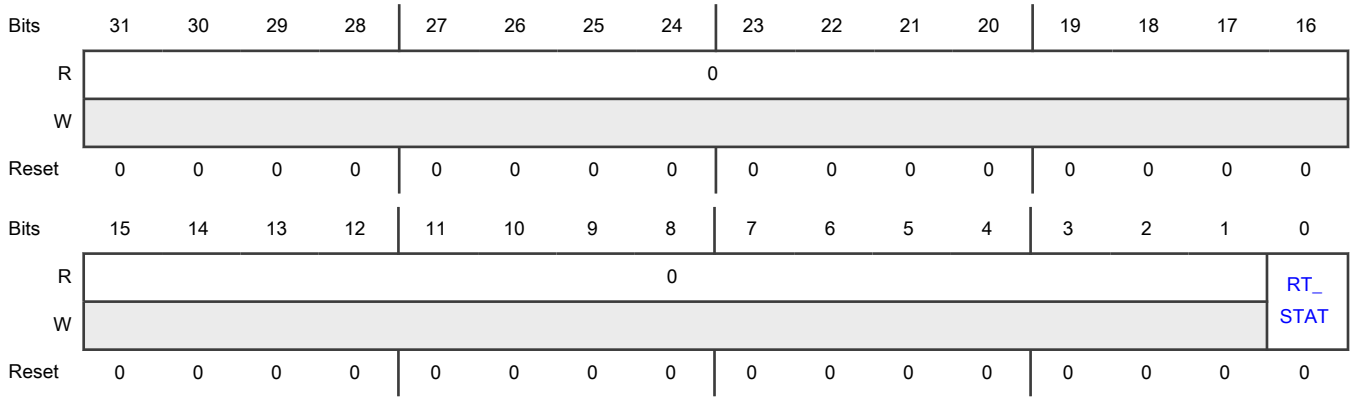
In the case of dynamic loading (Figure 385) of the RTI timer, this status may be unreliable .

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
PIT_0	RTI_LDVAL_STAT	—
PIT_1	—	RTI_LDVAL_STAT
PIT_2	—	RTI_LDVAL_STAT
PIT_3	—	RTI_LDVAL_STAT

Diagram



Fields

Field	Function
31-1 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 RT_STAT	Sync Status Indicates whether the RTI start value is loaded. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not loaded 1b - Loaded When writing 0b - Clears status 1b - Clears status

68.7.6 RTI Timer Load Value (RTI_LDVAL)

Offset

Register	Offset
RTI_LDVAL	F0h

Function

Specifies the length of the RTI timeout period in clock cycles.

It takes several cycles for this value to synchronize into the RTI clock domain.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
PIT_0	RTI_LDVAL	—
PIT_1	—	RTI_LDVAL
PIT_2	—	RTI_LDVAL
PIT_3	—	RTI_LDVAL

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	TSV															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	TSV															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TSV	Timer Start Value Specifies the starting RTI timer value. The timer counts down until it reaches 0, then sets the interrupt flag and reloads this value. When you write a new value to this register, the timer does not restart with the new value until the current timing period expires. To terminate the current period and start a new period with the new value, you must disable the timer (write 0 to RTI_TCTRL*n*[TEN]) and then enable it again (write 1 to RTI_TCTRL*n*[TEN]). **NOTE** Because it may take several cycles to clear the RTI interrupt, setting the value to something less than 32 cycles may result in a lost interrupt.

68.7.7 Current RTI Timer Value (RTI_CVAL)

Offset

Register	Offset
RTI_CVAL	F4h

Function

Indicates the current RTI timer value. Because this timer might be driven from a different clock domain from the other timers, the value may be several cycles old.

NOTE

Each module instance supports a different number of registers.

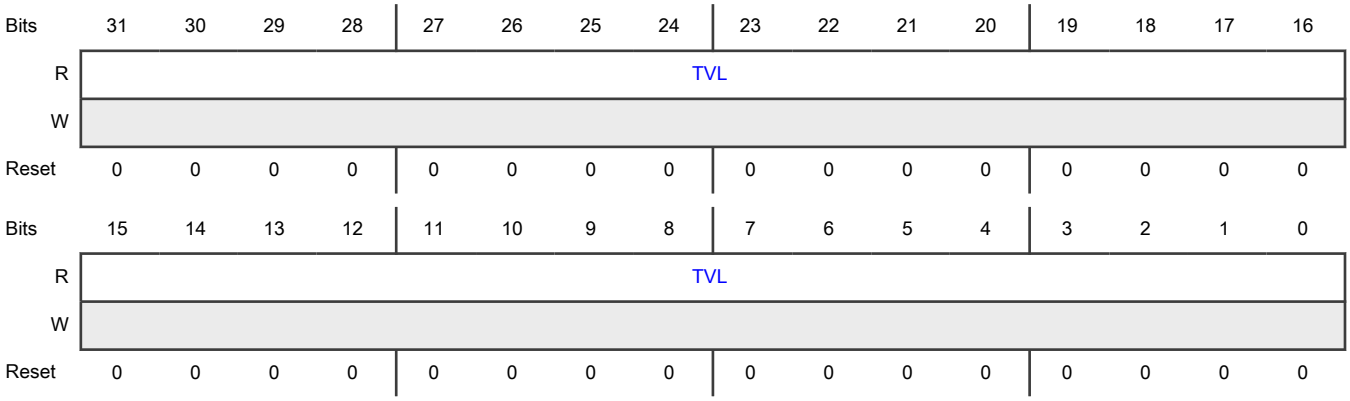
Instance	Register supported	Register not supported
PIT_0	RTI_CVAL	—
PIT_1	—	RTI_CVAL

Table continues on the next page...

Table continued from the previous page...

Instance	Register supported	Register not supported
PIT_2	—	RTI_CVAL
PIT_3	—	RTI_CVAL

Diagram



Fields

Field	Function
31-0 TVL	Current Timer Value Indicates the current RTI timer value. **NOTE** • If the timer is disabled (RTI_TCTRL*n*[TEN] = 0), this value has no meaning. • If PIT is frozen in Debug mode (MCR[FRZ] = 1), timer values are frozen when the device is in Debug mode.

68.7.8 RTI Timer Control (RTI_TCTRL)

Offset

Register	Offset
RTI_TCTRL	F8h

Function

Controls RTI timer behavior.

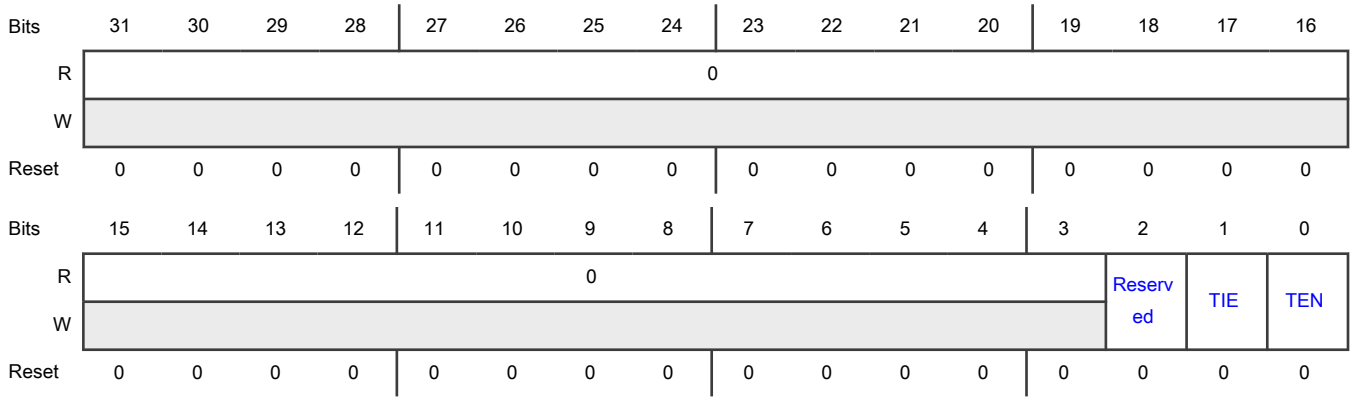
The RTI may take several RTI clock cycles to enable or update. Therefore, you must wait for at least three RTI clock cycles after RTI configuration before relying on the RTI timer.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
PIT_0	RTI_TCTRL	—
PIT_1	—	RTI_TCTRL
PIT_2	—	RTI_TCTRL
PIT_3	—	RTI_TCTRL

Diagram



Fields

Field	Function
31-3 —	Reserved
2 —	Reserved
1 TIE	Timer Interrupt Enable Enables the RTI timer interrupt. If the interrupt is enabled, when an interrupt is pending or the Timer Interrupt Flag is set (RTI_TFLG*n*[TIF] = 1), PIT generates an interrupt. To avoid an interrupt when you first enable the interrupt, clear the Timer Interrupt Flag (write 1 to RTI_TFLG*n*[TIF]) before you enable the interrupt. 0b - Disables 1b - Enables
0 TEN	Timer Enable Bit Enables the RTI timer.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disables 1b - Enables. The RTI timer begins counting down.

68.7.9 RTI Timer Interrupt Flag (RTI_TFLG)

Offset

Register	Offset
RTI_TFLG	FCh

Function

Indicates the RTI timer has expired.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
PIT_0	RTI_TFLG	—
PIT_1	—	RTI_TFLG
PIT_2	—	RTI_TFLG
PIT_3	—	RTI_TFLG

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															TIF
W																W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-1 —	Reserved
0 TIF	Timer Interrupt Flag Indicates the RTI timer period has expired (CVAL*n*[TVL] = 0). If interrupts are enabled (RTI_TCTRL*n*[TIE] = 1), TIF triggers an interrupt request. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Timer still counting down 1b - Timer has expired When writing 0b - No effect 1b - Clears the flag

68.7.10 Timer Load Value (LDVAL0 - LDVAL3)

Offset

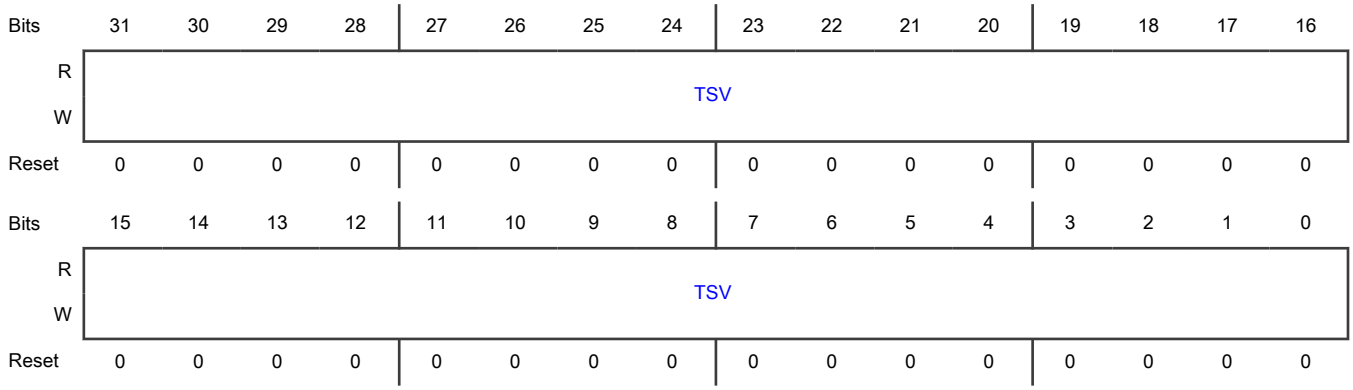
Register	Offset
LDVAL0	100h
LDVAL1	110h
LDVAL2	120h
LDVAL3	130h

Function

Specifies the length of the timeout period in clock cycles.

The value change is visible immediately. The synchronization mechanism allows 0 wait states in this case.

Diagram



Fields

Field	Function
31-0 TSV	Timer Start Value Specifies the starting timer value. The timer counts down until it reaches 0, then sets the interrupt flag and reloads this value. When you write a new value to this register, the timer does not restart with the new value until the current timing period expires. To terminate the current period and start a new period with the new value, you must disable the timer (write 0 to TCTRL*n*[TEN]) and then enable it again (write 1 to TCTRL*n*[TEN]). **NOTE** Because the CPU requires several cycles to service an interrupt, there is a practical limit to the lowest start value for a timer. Setting the value too low may result in a lost interrupt.

68.7.11 Current Timer Value (CVAL0 - CVAL3)

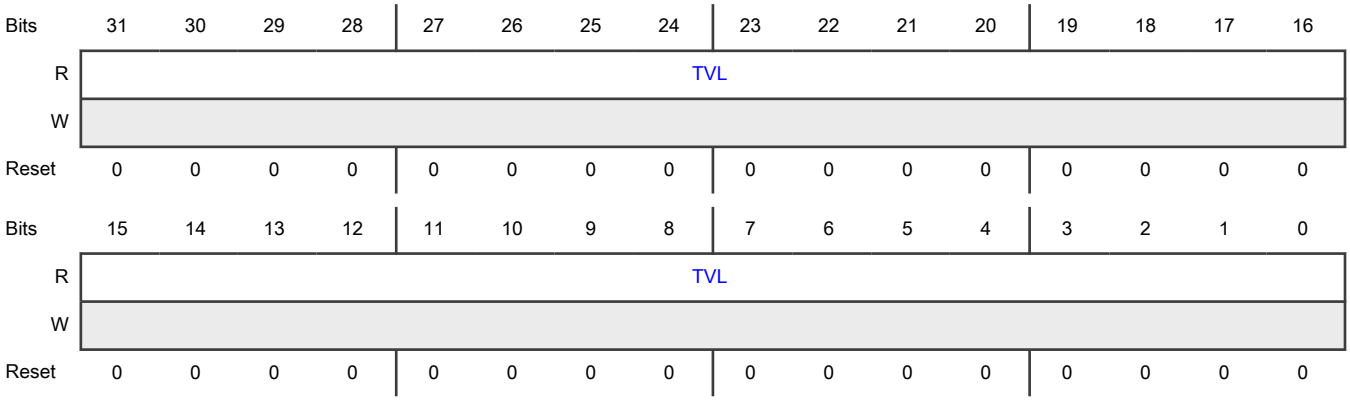
Offset

Register	Offset
CVAL0	104h
CVAL1	114h
CVAL2	124h
CVAL3	134h

Function

Indicates the current timer value.

Diagram



Fields

Field	Function
31-0 TVL	Timer Value Indicates the current timer value. **NOTE** • If the timer is disabled (TCTRL*n*[TEN] = 0), this value has no meaning. • If PIT is frozen in Debug mode (MCR[FRZ] = 1), timer values are frozen when the device is in Debug mode.

68.7.12 Timer Control (TCTRL0 - TCTRL3)

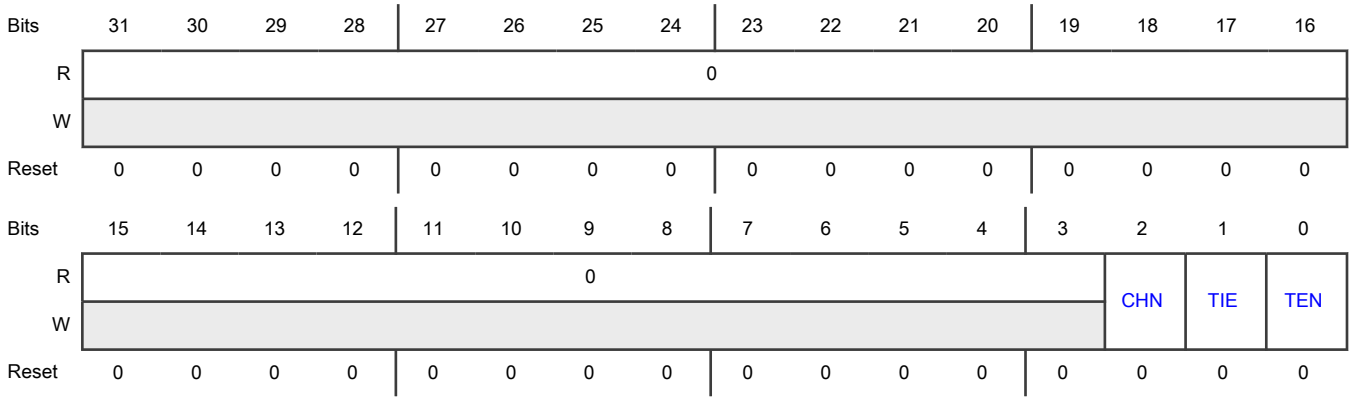
Offset

Register	Offset
TCTRL0	108h
TCTRL1	118h
TCTRL2	128h
TCTRL3	138h

Function

Controls timer behavior.

Diagram



Fields

Field	Function
31-3 —	Reserved
2 CHN	Chain Mode Chains this timer (timer *n*, where *n* is > 0) to the next lower numbered timer (timer *n* - 1) to create a 64-bit lifetimer. When chained, timer *n* decrements when timer *n* - 1 expires. See LTMR64H and LTMR64L. You cannot chain timer 0 or the RTI timer. 0b - Unchains 1b - Chains
1 TIE	Timer Interrupt Enable Enables the timer interrupt. If the interrupt is enabled, when an interrupt is pending or the Timer Interrupt Flag is set (TFLG*n*[TIF] = 1), PIT generates an interrupt. To avoid an interrupt when you first enable the interrupt, clear the Timer Interrupt Flag (write 1 to TFLG*n*[TIF]) before you enable the interrupt. 0b - Disables 1b - Enables
0 TEN	Timer Enable Enables the timer. 0b - Disables 1b - Enables. The timer begins counting down.

68.7.13 Timer Flag (TFLG0 - TFLG3)

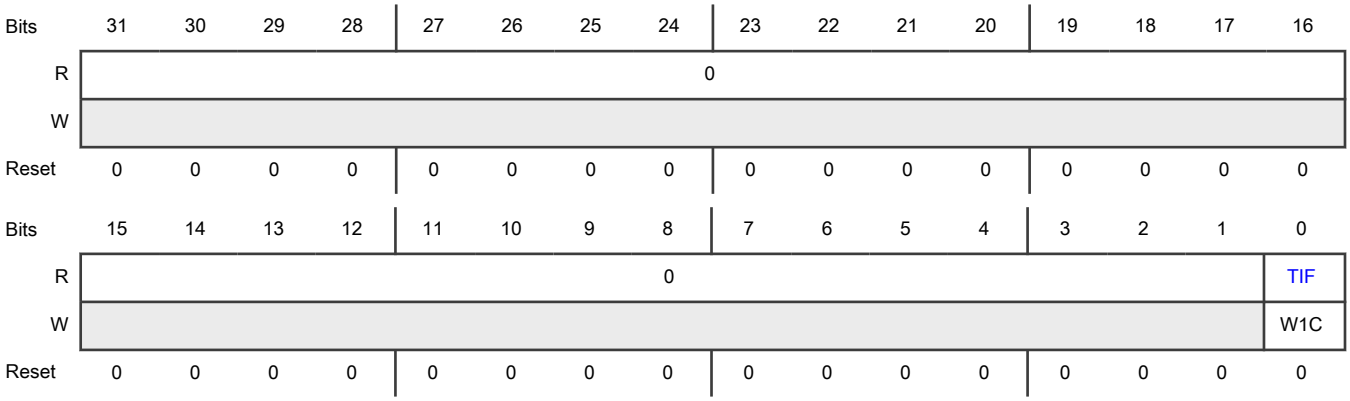
Offset

Register	Offset
TFLG0	10Ch
TFLG1	11Ch
TFLG2	12Ch
TFLG3	13Ch

Function

Indicates the PIT timer has expired.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 TIF	Timer Interrupt Flag Indicates the timer period has expired (CVAL*n*[TVL] = 0). If interrupts are enabled (TCTRL*n*[TIE] = 1), TIF triggers an interrupt request. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Timer has not expired 1b - Timer expired When writing

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No effect 1b - Clears flag

68.8 Glossary

RTI Real-time interrupt

Chapter 69
Real Time Clock (RTC)

69.1 Chip-specific RTC information

69.1.1 RTC instances and configuration

The chip contains one instance of RTC (Real Time Clock) timer and API (Autonomous Periodic Interrupt) timer, where both can perform 32-bit comparisons.

The RTC is present in always ON domain, hence available in RUN mode as well as in STANDBY mode. Both RTC and API timers can generate interrupts as well as wakeup from low power modes.

RTC supports a resolution of 1µs with high speed internal oscillator clock (1/48MHz (FIRC) * 48 (RTC counter) = 1µs)

This chip supports seamless RTC operation across functional reset with the clock sources SIRC and SXOSC.

NOTE

SXOSC is not present in K311, thus RTC will run from SIRC clock.

API has the capability to change the Timer compare value (APIVAL) independently without stopping the timer.

API can also be used in conjunction with the Comparator module. In STANDBY mode configuration, the API is configured to generate a START / NEXT type signal to inform the Comparator module that a 'compare' must be carried out.

69.2 Overview

The Real-Time Clock (RTC) is a free-running counter used for time keeping applications. The RTC can be configured to generate an interrupt at a pre-defined interval independent of the mode of operation (run mode or low power mode). If in a low power mode, the RTC interval is reached, the RTC first generates a wakeup and then asserts the interrupt request. The RTC also supports an API function used to generate a periodic wakeup request to exit a low-power mode or an interrupt request.

69.2.1 Block diagram

The following figure shows clock gating for RTC clocks.

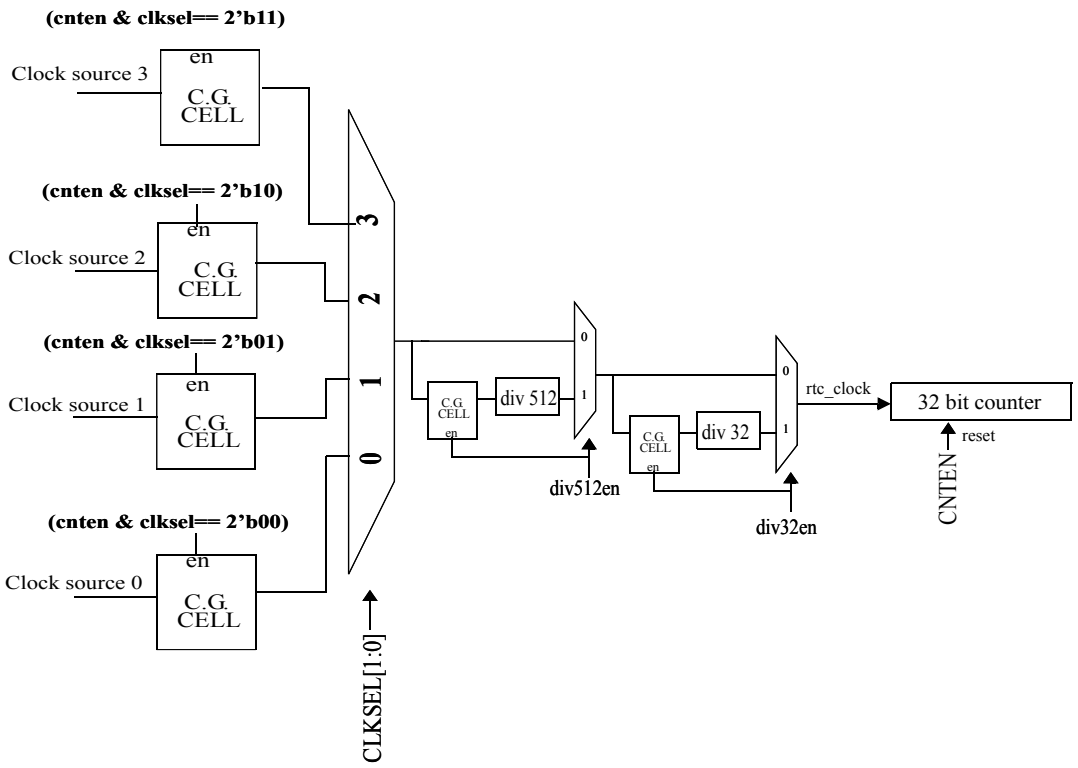


Figure 386. Clock gating for RTC clocks

The following figure shows the block diagram of RTC.

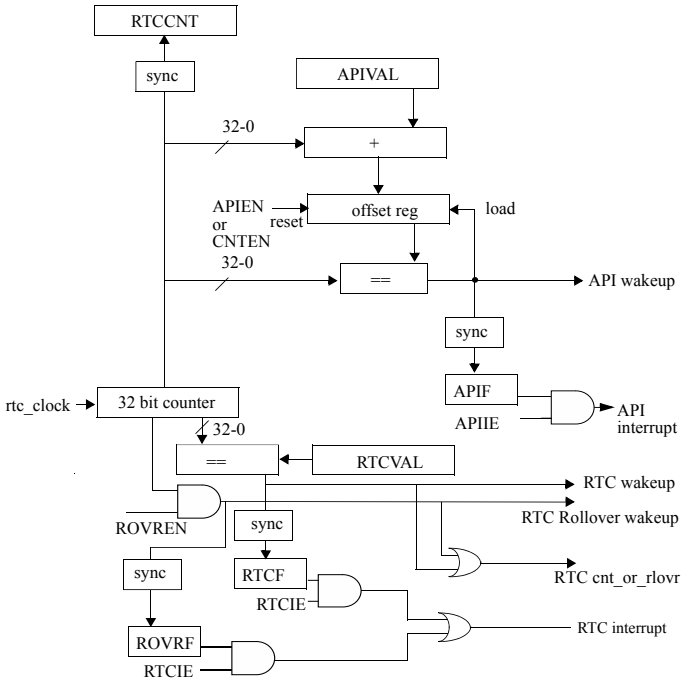


Figure 387. Block diagram

69.2.2 Features

RTC features include:

- 32-bit counter
- Selectable counter clock sources (IRCs and OSCs)
  - Clock source 0
  - Clock source 1
  - Clock source 2
  - Clock source 3

NOTE

See the chip-specific RTC information and clocking chapter for mapping of clock sources.

- Optional 512 prescaler and optional 32 prescaler to run the 32 bit counter.
- RTC interrupt with interrupt enable.
- Counter runs in all modes of operation.
- RTC counter is reset when the counter is disabled by software and by reset to RTC block.
- Autonomous periodic interrupt support includes:
  - 32-bit compare value to support range of wakeup intervals/interrupts
  - API logic has a separate enable to support changing compare value while RTC is running
  - API interrupt with interrupt enable
  - Operates in all modes
  - API compare value can be modified while RTC is running
- Optional interrupt for RTC match, API match, and RTC rollover.

69.3 Functional description

69.3.1 RTC

The RTC consists of a 32-bit free running counter enabled with the CNTEN bit (CNTEN, when negated, asynchronously resets the counter and synchronously enables the counter when enabled). After disabling CNTEN, RTCVAL, APIVAL, needs to be written again for desired functionality. The value of the counter may be read via the RTC Count (RTCCNT) register. Note that because of clock synchronization, the RTC Count (RTCCNT) value may represent a previous counter value. The difference between the counter and the read value depends on the ratio of counter clock and bus clock. Maximum possible difference between the two is 6 count values.

The clock source to the counter is selected with the CLKSEL field, which gives four options for clocking the RTC/API. The four clock sources are assumed to be on these:Clock source 0, Clock source 1, Clock source 2, and Clock source 3. The output of the clock mux can be optionally divided by a combination of 512 and 32 to give various count periods for different clock sources. Note that the CNTEN bit should be disabled when the RTC/API clock source is switched.

When the RTC Count (RTCCNT) counter value for counter bits 31-0 match the 32-bit value in the RTCVAL field, then the RTCF interrupt flag bit is set (after proper clock synchronization). If the RTCIE interrupt enable bit is set, then the RTC interrupt request is generated. RTC Compare value register (RTCVAL) register can be written only when INV_RTC bit is clear. Initially INV_RTC=0, and hence RTC Compare value register (RTCVAL) can be written once and hence INV_RTC gets set. This bit can now be cleared only by enabling the RTC counter. After the counter is enabled, RTC Compare value register (RTCVAL) can be written anytime, until RTC is disabled again. RTC Compare value register (RTCVAL) is first synchronized to the RTC clock domain, therefore, if RTC Compare value register (RTCVAL) is updated at the point where a counter match is due to happen in the next 2-3 RTC clocks because of previous RTC Compare value register (RTCVAL), the RTCF flag is set. However, if the RTC Compare value register

(RTCVAL) is updated at the point where no counter match is due as per the previous RTC Compare value register (RTCVAL), the RTCF flag is set when the counter matches the new RTC Compare value register (RTCVAL). If there is a match when in the low-power mode, then the RTC first generates a wakeup request to force a wakeup to run mode,and then the RTCF flag is set.

If RTC Compare value register (RTCVAL) is updated just after counter match, new RTC Compare value register (RTCVAL) value should not be within next 6 RTC counter values.

A rollover wakeup and/or interrupt can be generated when the RTC transitions from a count of 0xFFFF_FFFF to 0x0000_0000. The rollover flag is enabled by setting the ROVREN bit. If RTCIE is enabled, an interrupt request is generated for an RTC counter rollover. If system is in low power mode, RTC counter rollover with this bit causes a wakeup from the low-power mode.

Both rollover wakeup and RTC wakeup gets asynchronously de-asserted with disabling of CNTEN.

All the flags and counter values are synchronized with the Bus clock. It is assumed that the Bus clock and RTC clock selected through CLKSEL follows the below relation:

Bus clock >= (1.5 * RTC clock)/(div_factor)

- if both DIV32EN and DIV512EN bits are disabled, div_factor = 1
- if DIV32EN=1 and DIV512EN=0, div_factor = 32
- if DIV32EN=0 and DIV512EN=1, div_factor = 512
- if both DIV32EN and DIV512EN bits are enabled, div_factor = 512*32 = 16384

In case, RTC wakeup's are used as a wakeup source, bus clock (system clock or register interface clock of RTC) should be disabled (to save power in standby mode) after enabling the required wakeup and ensuring sufficient time gap (3-6 bus clock or RTC clock (rtc_clock) cycles, whichever is slower) between bus clock disabling and wakeup event. To ensure writing the register to enable the wakeup, the bus clock must be active, hence it should not be disabled before enabling wakeup. Correct operation is not guaranteed if the specification is not followed.

69.3.2 API functional description

Setting APIEN bit enables the autonomous interrupt function. The 32-bit APIVAL field selects the time interval for triggering an interrupt and/or wakeup event. Because the RTC is a free-running counter, the APIVAL is added to the current count to calculate an offset. When the counter reaches the offset count, an interrupt and/or wakeup request is generated. Then the offset value is recalculated and again re-triggers a new request when the new value is reached. API function is enabled only when CNTEN and APIEN bits are asserted and APIVAL is non-zero. Also APIVAL can be updated anytime. After APIVAL is updated, the first API interrupt is generated according to the previous value. From the second interrupt onwards, the API interrupt is generated with the new APIVAL. When a compare is reached, the APIF interrupt flag bit is set (after proper clock synchronization). If the APIIE interrupt enable bit is set, then the API interrupt request is generated. If there is a match while being in the low-power mode, then the API first generates a wakeup request to force a wakeup into normal operation, and then the APIF flag is set.

When the CNTEN is de-asserted, the API function is reset , though wakeup API is not asynchronously de-asserted with CNTEN. If APIEN is disabled when counter matches the offset APIF can be missed.

69.3.3 Modes of operation

69.3.3.1 Functional mode

There are two functional modes of operation for RTC, normal operation and low-power mode. In normal operation, all RTC registers can be read or written . The RTC/API and associated interrupts are optionally enabled. In low-power mode, the bus interface is disabled. The RTC/API is enabled (if enabled prior to entry into low-power mode).

69.3.3.2 Debug mode

On entering into the debug mode, the RTC counter freezes on the last valid count if the FRZEN is set. On exit from debug mode, counter continues from the frozen value.

69.4 Initialization

Recommended programming flow as follow

1. Program RTCVAL register with the value greater than 4 if RTCF related functionality is required.
2. Program APIVAL register with the value greater than 4 if API functionality is required.
3. Program all fields (as required) of RTC Control (RTCC) register with CNTEN=1. After setting CNTEN field, counter starts running and RTC/API functionality will be active as per the configurations.
4. If RTCVAL or APIVAL needs to be updated during run, check corresponding INV_RTC or INV_API bit is cleared before writing and do meet all the restrictions of the corresponding register.
5. If CLKSEL or DIV32EN or DIV512EN needs to be updated, first update RTC Control (RTCC) register with CNTEN=0 (when all INV_RTC or INV_API are 0) keeping other field values same. Wait for minimum 3 RTC clock cycles so that CNTEN will be synchronized to RTC clock domain. Then write the RTC Control (RTCC) register with new configuration required with CNTEN=1.

69.5 RTC register descriptions

The RTC registers are listed in this section.

NOTE

Address offset - 0x18h should not be accessed by application as corresponding feature/s are not available. Therefore, transfer error will not be generated at this offset.

NOTE

XFR error will be generated when RTCSUPV is accessed in user mode, any other register is accessed in user mode when SUPV bit is set, write attempt is made for RTCCNT register and any register accessed out of address range.

69.5.1 RTC memory map

RTC base address: 4028_8000h

Offset	Register	Width (In bits)	Access	Reset value
0h	RTC Supervisor control register (RTCSUPV)	32	RW	8000_0000h
4h	RTC Control (RTCC)	32	RW	0000_0000h
8h	RTC Status register (RTCS)	32	RW	0000_0000h
Ch	RTC Count (RTCCNT)	32	R	0000_0000h
10h	API Compare value register (APIVAL)	32	RW	0000_0000h
14h	RTC Compare value register (RTCVAL)	32	RW	0000_0000h

69.5.2 RTC Supervisor control register (RTCSUPV)

Offset

Register	Offset
RTCSUPV	0h

Function

The RTCSUPV register contains the SUPV bit that determines whether other registers are accessible in supervisor mode or user mode.

NOTE

You can access this register only in Supervisor mode, and you must write a value only to the SUPV field of the register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SUPV	0														
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SUPV	RTC Supervisor Bit 0b - All registers are accessible in both user as well as supervisor mode 1b - All other registers are accessible in the supervisor mode only
30-0 —	Reserved

69.5.3 RTC Control (RTCC)

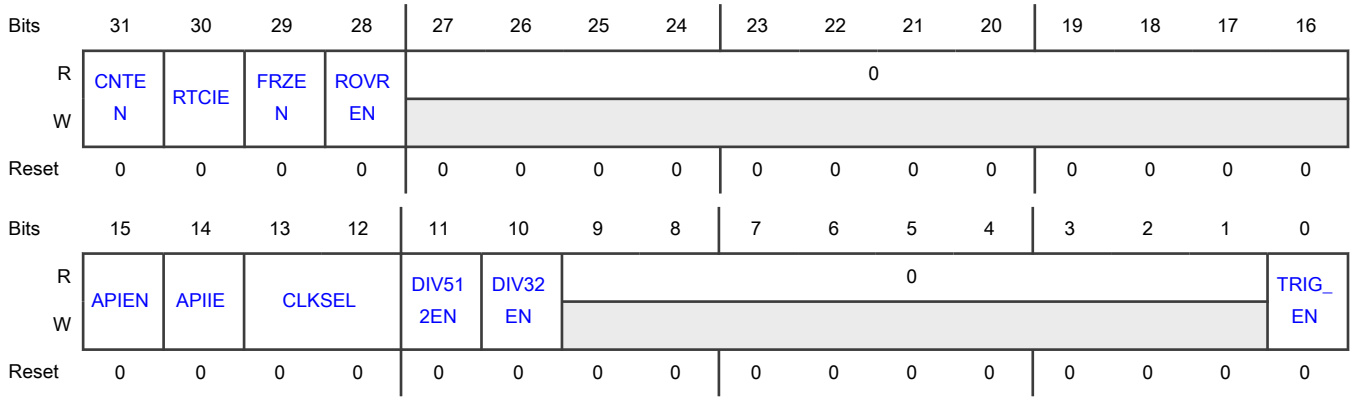
Offset

Register	Offset
RTCC	4h

Function

RTC Control register

Diagram



Fields

Field	Function
31 CNTEN	Counter Enable The CNTEN bit enables the RTC counter. Setting CNTEN bit to 0b has the effect of asynchronously resetting (synchronous reset negation) all the RTC and API logic. This allows RTC configuration and clock source selection to be updated without causing synchronization issues. **NOTE** CNTEN should be disabled when INV_RTC, INV_API are cleared. 0b - Counter disabled 1b - Counter enabled
30 RTCIE	RTC Interrupt Enable The RTCIE bit enables interrupts requests to the system if RTCF is asserted. 0b - RTC interrupts disabled 1b - RTC interrupts enabled
29 FRZEN	Freeze Enable Bit The counter freezes on entering the debug mode on the last valid count value if the FRZEN bit is set. After passing of the debug mode counter starts from the frozen value. This bit should not be changed when debug mode is enabled. 0b - Counter does not freeze in debug mode 1b - Counter freezes in debug mode
28 ROVREN	Counter Roll Over wakeup/Interrupt Enable The ROVREN bit enables wakeup and interrupt requests when the RTC has rolled over from 0xFFFF_FFFF to 0x0000_0000. The RTCIE bit must also be set in order to generate an interrupt from a counter rollover. 0b - RTC rollover wakeup/interrupt disabled 1b - RTC rollover wakeup/interrupt enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
27-16 —	Reserved
15 APIEN	Autonomous Periodic Interrupt Enable The APIEN bit enables the autonomous periodic interrupt function. Setting this bit to 0b, asynchronously disables API wakeup output of RTC as well. 0b - API disabled 1b - API enabled
14 APIIE	API Interrupt Enable The APIIE bit enables interrupts requests to the system if APIF is asserted. 0b - API interrupts disabled 1b - API interrupts enabled
13-12 CLKSEL	Clock select The CLKSEL[1:0] bits select the clock source for the RTC. CLKSEL may only be updated when CNTEN is 0. The user should ensure that oscillator is enabled before selecting it as a clock source for RTC. **NOTE** See the RTC-clocking and resolution information in the clocking chapter elsewhere in this document. 00b - Clock source 0 01b - Clock source 1 10b - Clock source 2 11b - Clock source 3
11 DIV512EN	Divide by 512 enable The DIV512EN bit enables the 512 clock divider. DIV512EN may only be updated when CNTEN is 0. 0b - Divide by 512 is disabled 1b - Divide by 512 is enabled
10 DIV32EN	Divide by 32 enable The DIV32EN bit enables the 32 clock divider. DIV32EN may only be updated when CNTEN is 0. 0b - Divide by 32 is disabled 1b - Divide by 32 is enabled
9-1 —	Reserved
0	Trigger enable for Analog Comparator

Table continues on the next page...

Table continued from the previous page...

Field	Function
TRIG_EN	This bit (TRIG_EN) when set will de-assert the wakeup_api on the next RTC clock (required for CMP when bus clock is disabled). **NOTE** If API wakeup is asserted with TRIG_EN set, and bus clock enabled, setting of RTCS[APIF] may be missed.

69.5.4 RTC Status register (RTCS)

Offset

Register	Offset
RTCS	8h

Function

NOTE

W1C has priority over setting of the RTCF, APIF and ROVRF bits, in case both clearing and setting occurs at the same time.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		RTCF		0									INV_RTC	INV_API	Reserved
W			W1C													
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		APIF	0		ROVRF	0									
W			W1C			W1C										
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29	RTC Interrupt Flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
RTCF	The RTCF bit indicates that the RTC counter has reached the counter value matching RTC Compare value register (RTCVAL). RTCF is cleared by writing a 1 to RTCF. Writing a 0 to RTCF has no effect. 0b - RTC counter is not equal to RTCVAL 1b - RTC counter matches RTCVAL
28-19 —	Reserved
18 INV_RTC	Invalid RTC write This bit returns value 1 after a value is written to the RTCVAL register and the synchronization process is in progress. During this synchronization period, any attempt to write to the RTCVAL register again is ignored. Synchronization will complete only when CNTEN is set.
17 INV_API	Invalid APIVAL write This bit returns value 1 after a value is written to the APIVAL register and the synchronization process is in progress. During this synchronization period, any attempt to write to the APIVAL register again is ignored. Synchronization will complete only when CNTEN is set.
16 —	Reserved
15-14 —	Reserved
13 APIF	API Interrupt Flag The APIF bit indicates that the RTC counter has reached the counter value matching API offset value. APIF is cleared by writing a 1 to APIF. Writing a 0 to APIF has no effect. 0b - Counter is not equal to API offset value 1b - Counter matches the API offset value
12-11 —	Reserved
10 ROVRF	Counter Roll Over Interrupt Flag The ROVRF bit indicates that the RTC has rolled over from 0xFFFF_FFFF to 0x0000_0000. ROVRF is cleared by writing a 1 to ROVRF. 0b - RTC has not rolled over 1b - RTC has rolled over
9-0 —	Reserved

69.5.5 RTC Count (RTCCNT)

Offset

Register	Offset
RTCCNT	Ch

Function

RTC Counter register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RTCCNT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RTCCNT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 RTCCNT	RTC Counter Value Because of clock synchronization, the RTCCNT value may represent a previous counter value.

69.5.6 API Compare value register (APIVAL)

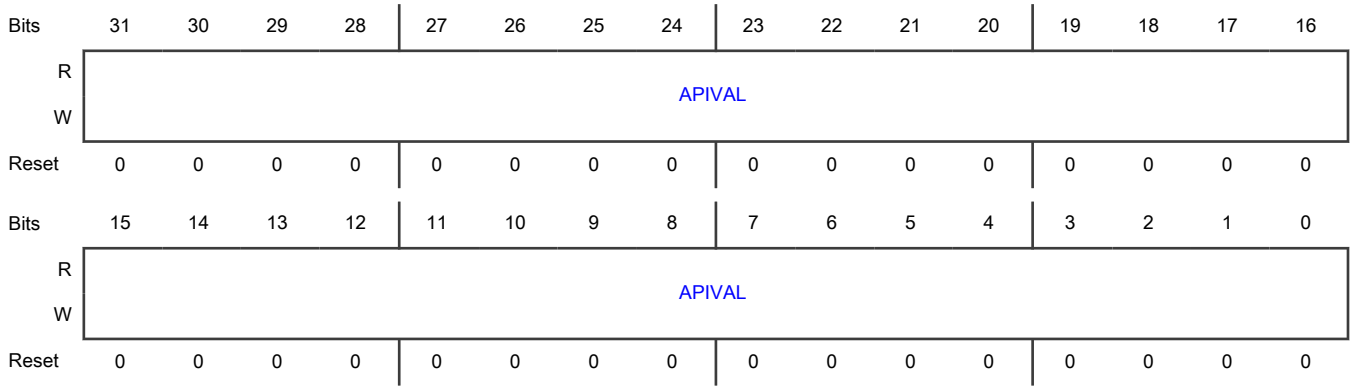
Offset

Register	Offset
APIVAL	10h

Function

The APIVAL offset bits are compared to the RTC counter bits and if a match occurs, an interrupt/wakeup request is asserted.

Diagram



Fields

Field	Function
31-0 APIVAL	API Compare Value APIVAL bits are added to the current count to calculate an offset. The APIVAL offset bits are compared to the RTC counter bits and if a match occurs, an interrupt/wakeup request is asserted. **NOTE** API functionality is active only when APIVAL is non zero. The first API interrupt takes two more cycles because of synchronization of APIVAL to RTC clock. After that, interrupts are periodic in nature and it takes APIVAL+1 cycles. The Minimum supported value of APIVAL is 4. This is because of synchronization.

69.5.7 RTC Compare value register (RTCVAL)

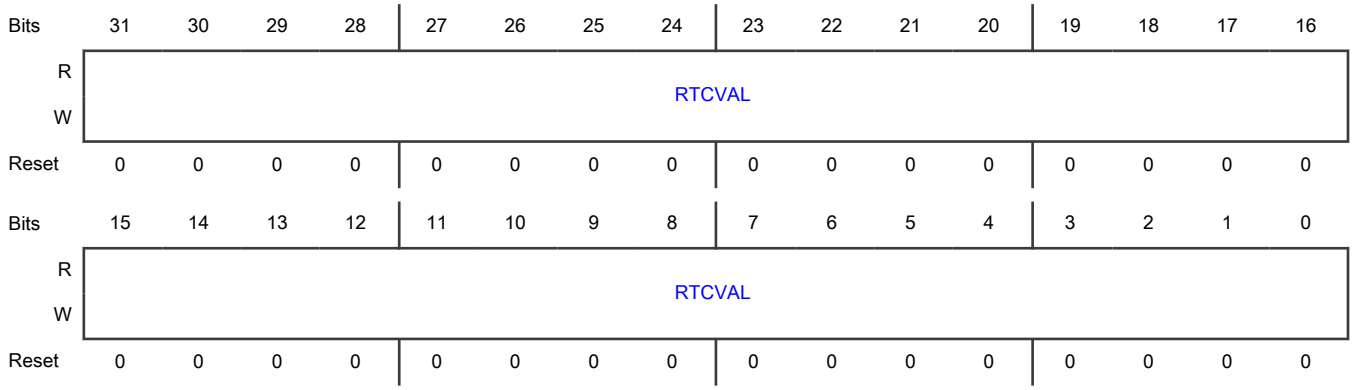
Offset

Register	Offset
RTCVAL	14h

Function

The RTCVAL bits are compared to the RTC counter bits and if a match occurs, RTCF is set. The minimum value of RTCVAL should be 4.

Diagram



Fields

Field	Function
31-0 RTCVAL	RTC Compare Value The RTCVAL bits are compared to the RTC counter bits and if a match occurs, RTCF is set.

69.6 Glossary

API	Autonomous Periodic Interrupt
CMP	Comparator
IRCs	Internal RC Oscillator
OSCs	Oscillator

Chapter 70
Low Power Serial Peripheral Interface (LPSPI)

70.1 Chip-specific LPSPI information

70.1.1 LPSPI instances and configuration

Table 437. LPSPI instances

Instance	S32K314/S32K324/S32K344/S32K358/S32K356/ S32K348/S32K338/S32K328/S32K388/S32K389	S32K310/S32K311/S32K312/S32K322/ S32K341/S32K342
LPSPI0	Yes	Yes
LPSPI1	Yes	Yes
LPSPI2	Yes	Yes
LPSPI3	Yes	Yes
LPSPI4	Yes	No
LPSPI5	Yes	No

Table 438. LPSPI instances configuration

Instances	TX FIFO Size	RX FIFO Size	Chip Selects
LPSPI0	4x32 bit	4x32 bit	8
LPSPI1	4x32 bit	4x32 bit	6
LPSPI2	4x32 bit	4x32 bit	4
LPSPI3	4x32 bit	4x32 bit	4
LPSPI4	4x32 bit	4x32 bit	4
LPSPI5	4x32 bit	4x32 bit	4

- For supported data rates see table 'Peripheral data rates' table in Clocking chapter.
- Low leakage and Wait modes are not supported in this device.

The number of chip selects that each LPSPI instance supports varies. Because of that, in the Configuration Register 1 (CFGR1), the Peripheral Chip Select (PCS) field and the Peripheral Chip Select Polarity (PCSPOL) field also vary. See the next 2 tables.

Table 439. LPSPI instances mapped against Peripheral Chip Select (PCS) field support

PCS supported in	PCS not supported in
LPSPI0_TCR[26-24]	—
LPSPI1_TCR[26-24]	—
LPSPI2_TCR[25–24]	LPSPI2_TCR[26]
LPSPI3_TCR[25–24]	LPSPI3_TCR[26]
LPSPI4_TCR[25–24]	LPSPI4_TCR[26]
LPSPI5_TCR[25–24]	LPSPI5_TCR[26]

Table 440. LPSPI instances mapped against Peripheral Chip Select Polarity (PCSPOL) field support

PCSPOL supported in	PCSPOL not supported in
LPSPI0_CFGR1[15-8]	—
LPSPI1_CFGR1[13–8]	LPSPI1_CFGR1[15–14]
LPSPI2_CFGR1[11–8]	LPSPI2_CFGR1[15–12]
LPSPI3_CFGR1[11–8]	LPSPI3_CFGR1[15–12]
LPSPI4_CFGR1[11–8]	LPSPI4_CFGR1[15–12]
LPSPI5_CFGR1[11–8]	LPSPI5_CFGR1[15–12]

70.1.2 LPSPI HREQ considerations for S32K314, S32K324 and S32K344

It is recommended that the HREQ pin (when PCS[1] is used as HREQ) should get de-asserted before the completion of frame transfer. In case if the HREQ state is still asserted and LPSPI returns to idle state after frame transfer completion, the HREQ state is internally latched and the next data is written to the transmit FIFO without waiting for the next HREQ assertion.

This limitation is present only while using PCS[1] as HREQ and HREQ remains asserted throughout frame transfer. In case of using trigger input as HREQ, there is no issue.

70.2 Overview

LPSPI provides an efficient interface (either as a controller or peripheral) to an SPI bus, which is a synchronous serial communication interface used in embedded systems. It is typically used to perform short distance communications between microcontrollers and peripheral devices, on printed circuit boards. Typical applications include interfacing with secure digital cards and LCD displays.

NOTE

The terminology in this chapter has been updated to align with NXP's inclusive language standards, as shown in the table below.

Table 441. Updated terms

Updated term	Deprecated term
Controller	Master
Peripheral	Slave

70.2.1 Block diagram

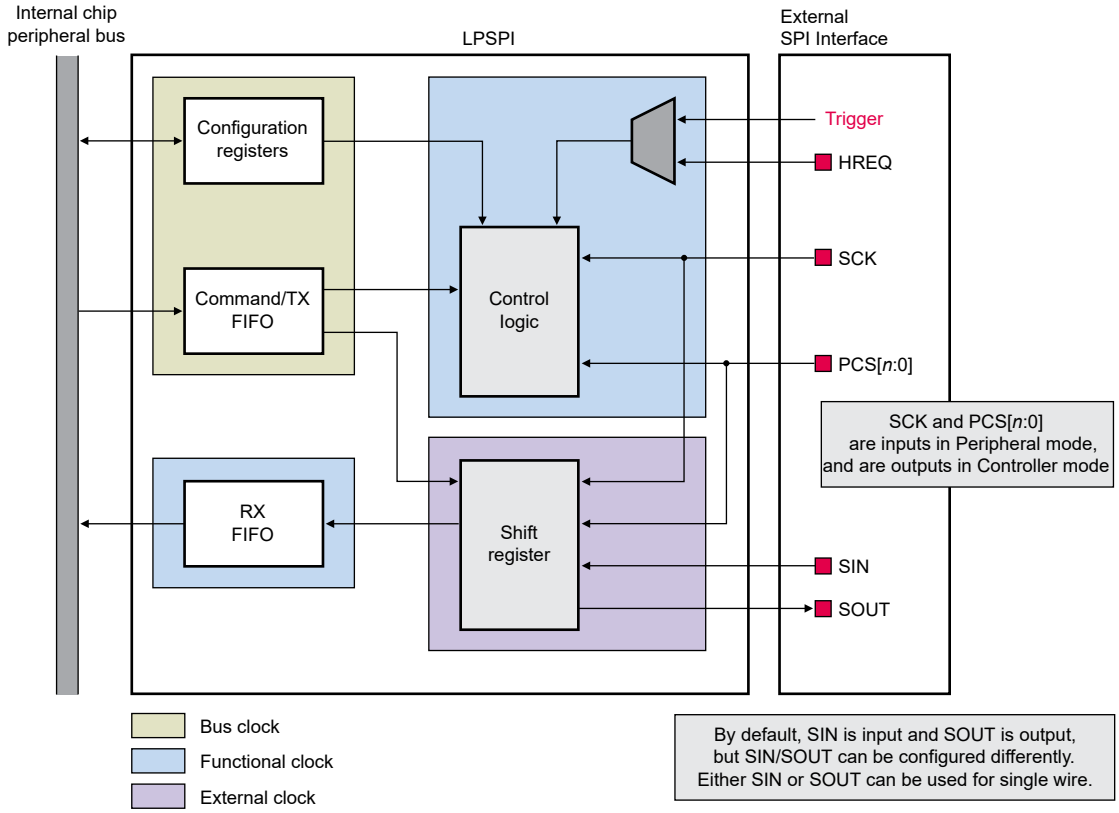


Figure 388. Block diagram

70.2.2 Features

- Minimal CPU overhead, with DMA transmit and receive requests supporting FIFO register accesses
- Support available for 32-bit word size
- Configurable clock polarity and phase
- Support available for 8 peripheral chip selects in Controller mode
- Support available for Peripheral mode
- 4-word transmit and command FIFO
- 4-word receive FIFO
- Flexible timing parameters in Controller mode, including SCK frequency and duty cycle, and delays between PCS and SCK edges
- Continuous transfer option to keep PCS asserted across multiple frames
- Full-duplex transfers that support 1-bit transmit and receive on each clock edge
- Half-duplex transfers that support:
  - 1-bit transmit or receive on each clock edge
  - 2-bit transmit or receive on each clock edge
  - 4-bit transmit or receive on each clock edge
  - 8-bit transmit or receive on each clock edge

- Option to use host request to control the start of an SPI bus transfer

70.3 Functional description

70.3.1 Controller mode

70.3.1.1 Transmit and command FIFO commands

The transmit and command FIFO is a combined FIFO that includes both transmit data words and command words. You store:

- Transmit data words in the transmit and command FIFO, by writing to Transmit Data (TDR).
- Command words in the transmit and command FIFO, by writing to Transmit Command (TCR).

When a command word is at the top of the transmit and command FIFO, the actions that can occur depend on whether LPSPI is busy or between frames (see TCR[CONT] and TCR[CONTC]). See Table 442 for conditions and possible corresponding actions when a command word is at the top of the transmit and command FIFO.

Table 442. Possible actions when a command word is at the top of the transmit and command FIFO

Condition	Action
LPSPI is enabled and idle.	The command word is pulled from the FIFO, and this command word controls all subsequent transfers.
LPSPI is busy and TCR[CONTC] is 0.	The SPI frame completes at the end of the existing word, ignoring TCR[FRAMESZ]. The command word is then pulled from the FIFO and that command word controls all subsequent transfers (or until the next update to the command word). Note that a command word with TCR[CONTC] = 0 always terminates the existing transfer regardless of the previous TCR[CONT] value.
LPSPI is busy; the existing TCR[CONT] value is 1 and the new TCR[CONTC] value is 1.	The command word must be updated at the frame boundary. The command word is pulled from the FIFO during the last SCK pulse of the existing frame (based on the value of FRAMESZ), and the frame continues using the new command value for the rest of the frame (or until the next update to the command word). When TCR[CONTC] = 1, only the lower 24 bits of the command word are updated. If the command word is updated at a word boundary, then the transfer halts (stops) after that word. TCR[CONTC] is ignored when not at a frame boundary, so the frame ends prematurely.

TCR[CONT] = 1 keeps PCS asserted at end of frame, allowing the transfer to continue.

TCR[CONTC] = 1 specifies that this command word must not terminate the existing frame, and the transfer can continue using the new command word.

TCR[CONTC] = 1 is restricted in the sense that the new command must load on a frame boundary, and the only way for a transfer to continue from a frame boundary is when the previous command has TCR[CONT] = 1.

You can read the current state of the existing command word from Transmit Command (TCR). It requires at least three LPSPI functional clock cycles for Transmit Command (TCR) to update after you write to it (assuming an empty FIFO), and LPSPI must be enabled (CR[MEN] = 1).

Writing to Transmit Command (TCR) does not initiate an SPI bus transfer, unless TCR[TXMSK] = 1. When TCR[TXMSK] = 1, a new command word is not loaded until the end of the existing frame (based on the value of TCR[FRAMESZ]); at the end of the transfer, TCR[TXMSK] transitions to 0.

In Controller mode, the LPSPI command word in Transmit Command (TCR) controls SPI attributes based on the selections in register fields. See Table 443 for TCR fields and associated functionality related to data transfer.

Table 443. Command word in Controller mode

Transmit Command (TCR)		Description	Can this field be modified during a data transfer?
Field	Name		
CPOL	Clock polarity	Specifies the polarity of the SCK pin. Any change of CPOL value causes a transition on the SCK pin.	N
CPHA	Clock phase	Specifies the clock phase of the transfer.	N
PRESCALE	Prescaler value	Specifies a prescaler used to divide the LPSPI functional clock, to generate the timing parameters of the SPI bus transfer. Changing PRESCALE in conjunction with PCS enables LPSPI to connect to different peripheral devices at different frequencies.	N
PCS	Peripheral chip select	Specifies which PCS pin asserts for the transfer; the polarity of PCS is static and specified by CFGR1[PCSPOL]. If CFGR1[PCSCFG] = 1, do not select PCS[3:2].	N
LSBF	LSB first	Specifies whether LSB (bit 0) or MSB (bit 31 for a 32-bit word) is transmitted or received first.	Y
BYSW	Byte swap	Enables byte swap on each 32-bit word when transmitting and receiving data. Byte swapping can be useful when interfacing with devices that organize data as big-endian.	Y
CONT	Continuous transfer	Configures LPSPI for a continuous transfer that keeps PCS asserted between frames (as specified by FRAMESZ). You must write a new command word to cause PCS to negate. Also, this field supports changing the command word at frame size boundaries.	Y
CONTC	Continuing command	Indicates that this is a new command word for the existing continuous transfer. When CONTC = 1, the command word must only be written to the transmit and command FIFO on a frame boundary.	Y
RXMSK	Receive data mask	Masks the receive data and does not store the masked receive data in the receive FIFO or perform receive data matching. This option is useful for half-duplex transfers or to specify which fields are compared during receive data matching.	Y
TXMSK	Transmit data mask	Masks the transmit data; masked transmit data is not pulled from the transmit FIFO, and the output data pin is 3-stated (unless otherwise configured by CFGR1[OUTCFG]). This option is useful for half-duplex transfers.	Y
WIDTH	Transfer width	Specifies the number of bits shifted on each SCK pulse: • 1-bit transfers support traditional SPI bus transfers in either half-duplex or full-duplex data formats. • 2-bit and 4-bit half-duplex transfers are useful for interfacing with QuadSPI memory devices, and either TXMSK or RXMSK must also be 1. • 8-bit half-duplex transfers are useful for interfacing with OctelSPI memory devices, and either TXMSK or RXMSK must also be 1.	Y

Table continues on the next page...

Table 443. Command word in Controller mode (continued)

Transmit Command (TCR)		Description	Can this field be modified during a data transfer?
Field	Name		
FRAMESZ	Frame size	Configures the frame size in number of bits equal to (FRAMESZ + 1): • The minimum frame size is 8 bits, or 16 bits for an 8-bit transfer. • If the frame size is larger than 32 bits, then the frame is divided into multiple words of 32 bits; each word is loaded from the transmit FIFO and stored in the receive FIFO separately. • If the size of the frame is not divisible by 32, then the last load of the transmit FIFO and store of the receive FIFO contains the remaining bits. For example, a 72-bit transfer consists of three words: the first and second words are 32 bits, and the third word is 8 bits.	Y

70.3.1.1.1 SPI bus transfers

LPSPI initiates an SPI bus transfer when all these conditions are true:

- Data is written to the transmit FIFO.
- The HREQ pin is asserted (or the HREQ function is disabled).
- LPSPI is enabled.

To perform the SPI bus transfer, LPSPI uses the attributes configured in Transmit Command (TCR) and the timing parameters defined in Clock Configuration (CCR).

The SPI bus transfer ends after the number of bits indicated by the value of FRAMESZ have been transferred (provided CONT = 0), or at the end of a word when a new transmit command word is at the top of the transmit and command FIFO. When LPSPI is disabled, the SPI bus transfers end after the transmit FIFO is empty and LPSPI is idle.

The HREQ input is only checked when PCS is negated.

70.3.1.1.2 Circular FIFO

The transmit and command FIFO supports a circular FIFO feature. This feature enables the LPSPI controller to (periodically) repeat a short data transfer that fits within the transmit and command FIFO, without requiring additional FIFO accesses. When the circular FIFO is enabled (CFGR0[CIRFIFO] = 1), the current state of the FIFO read pointer is saved and the status flags are not updated. After the FIFO is empty and LPSPI is idle, the FIFO read pointer is restored with the saved version, so the contents of the transmit and command FIFO are not permanently pulled from the FIFO when Circular FIFO mode is enabled.

70.3.1.2 Receive FIFO and data match

The receive FIFO stores received data during SPI bus transfers. When TCR[RXMSK] = 1, the received data is discarded instead of being stored in the receive FIFO:

- Received data is written to the receive FIFO when the last bit of the word is sampled.
- If the transmit FIFO is empty during a multiple-word or continuous transfer, then the receive data is written to the receive FIFO before the transfer stalls (assuming CFGR1[NOSTALL] = 0) while waiting for new transmit data or for a command word to be written.

LPSPI provides a receive data match function that can match received data against one of the two words in DMR0 and DMR1, or against a masked data word. You can also configure the received data match function to compare only the first one or two received data words since the start of the frame:

- Received data that is already discarded because of TCR[RXMSK] cannot cause the data match flag to set, and delays the receive data match on the first received data word, until all discarded data is received.
- You can configure the receive data match function to discard all received data until a data match is detected, using CFGR0[RDMO].
- After a receive data match, to allow all subsequent data to be received, write 0 to CFGR0[RDMO], and then write 0 to SR[DMF].

70.3.1.3 Timing parameters

The timing parameters that are used for all SPI bus transfers are relative to the LPSPI functional clock divided by the selection specified in TCR[PRESCALE]. Although you can only change Clock Configuration (CCR) when LPSPI is disabled (CR[MEN] = 0), to support interfacing with different peripheral devices at different frequencies, you can change the TCR[PRESCALE] selection between SPI bus transfers by using Transmit Command (TCR).

NOTE

The minimum value shown in Table 444 is the minimum counter value, but the values of Clock Configuration (CCR) must also satisfy the data sheet specs based on the LPSPI functional clock frequency and prescaler value.

Table 444. Timing parameters

Clock Configuration (CCR) Clock Configuration 1 (CCR1)		Description	Minimum value	Maximum value
Field	Name			
SCKSET	SCK setup phase	Configures the SCK setup phase to (SCKSET + 1) cycles. The setup phase is the SCK high period when either CPHA = 0 and CPOL = 1, or CPHA = 1 and CPOL = 0. Otherwise, it is the SCK low period. The SCK period is defined as (SCKSET + SCKHLD + 2) and the duty cycle is the difference between SCKSET and SCKHLD.	0 (1 cycle)	255 (256 cycles)
SCKHLD	SCK hold phase	Configures the SCK hold phase to (SCKHLD + 1) cycles. The hold phase is the SCK low period when either CPHA = 0 and CPOL = 1, or CPHA = 1 and CPOL = 0. Otherwise, it is the SCK high period. The SCK period is defined as (SCKSET + SCKHLD + 2) and the duty cycle is the difference between SCKSET and SCKHLD.	0 (1 cycle)	255 (256 cycles)
PCSPCS	PCS-to-PCS delay	Configures the minimum delay between PCS negation and the next PCS assertion to (PCSPCS + PCSPCS + 2) cycles. When the command word is updated between transfers, there is a minimum of (PCSPCS + 1) cycles between the command word update and any change on PCS pins.	0 (2 cycles)	255 (512 cycles)
SCKSCK	SCK-to-SCK delay	Configures the delay during a continuous transfer between the last SCK edge of a frame and the first SCK edge of the continuing frame to (SCKSCK + 1) cycles. This is useful when the external peripheral	0 (1 cycle)	255 (256 cycles)

Table continues on the next page...

Table 444. Timing parameters (continued)

Clock Configuration (CCR) Clock Configuration 1 (CCR1)		Description	Minimum value	Maximum value
Field	Name			
		requires a large delay between different words of an SPI bus transfer.		
PCSSCK	PCS-to-SCK delay	Configures the minimum delay between PCS assertion and the first SCK edge to (PCSSCK + 1) cycles.	0 (1 cycle)	255 (256 cycles)
SCKPCS	SCK-to-PCS delay	Configures the minimum delay between the last SCK edge and the PCS negation to (SCKPCS + 1) cycles.	0 (1 cycle)	255 (256 cycles)

Figure 389 shows the timing settings controlled by:

- TCR[CPHA]
- TCR[CPOL]
- CCR[SCKPCS]
- CCR[PCSSCK]
- CCR1[SCKSET]
- CCR1[SCKHLD]

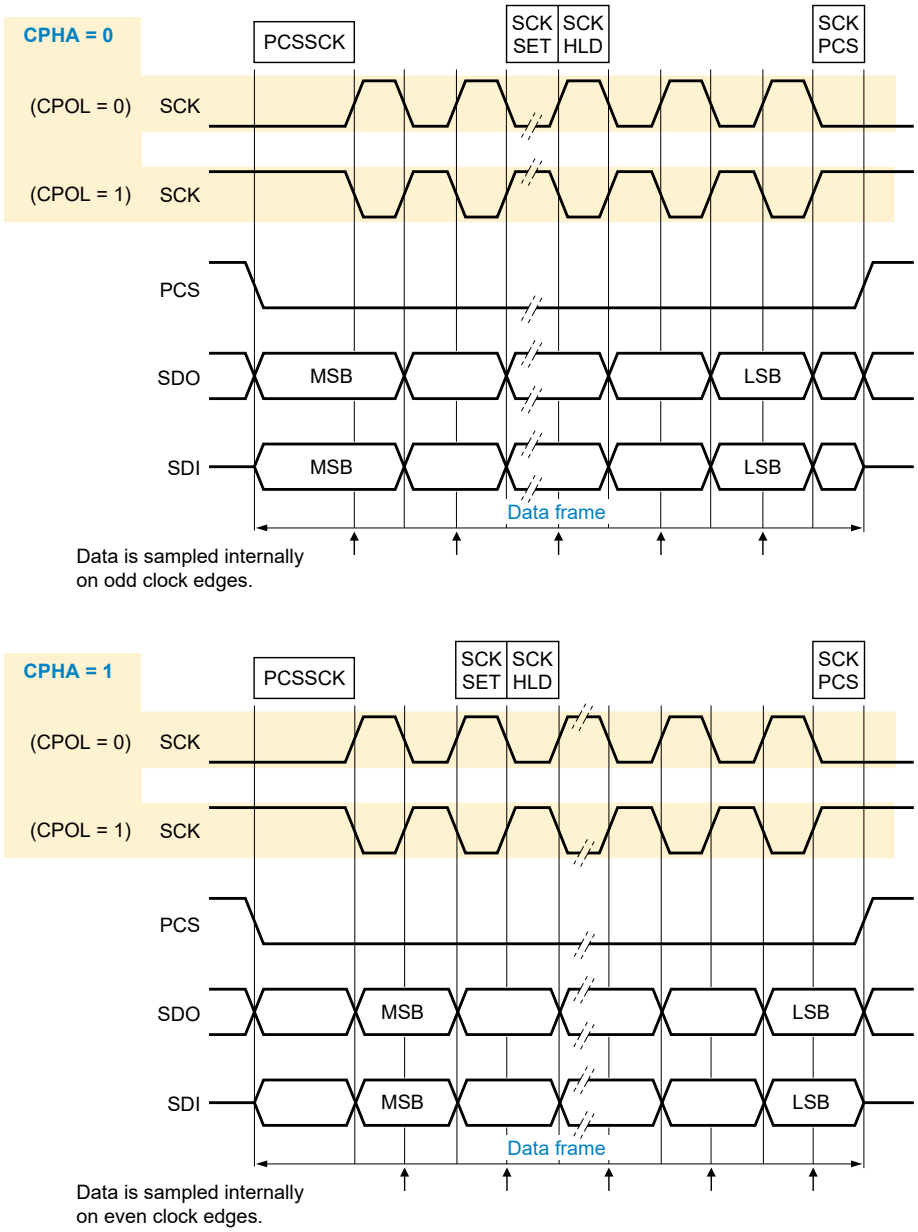


Figure 389. Clock phase (TCR[CPHA]) timing diagram example

To configure for a baud rate of 10 MHz with 50/50 duty cycle and with a functional clock frequency of 100 MHz, use the following settings:

- CCR1[SCKSET] = 0x4 (5 cycles)
- CCR1[SCKHLD] = 0x4 (5 cycles)
- CCR1[PCSPCS] = 0x8 (10 cycles)
- CCR1[SCKSCK] = 0x4 (5 cycles)
- CRR[PCSSCK] = 0x4 (5 cycles)
- CRR[SCKPCS] = 0x4 (5 cycles)
- TCR[PRESCALE] = 0x0 (divide by 1)

70.3.1.4 Pin configuration

Following are the pin configuration settings for half-duplex transfers:

- To swap directions or to support half-duplex transfers on the same pin, you can configure the SIN and SOUT pins using CFGR1[PINCFG].
- To specify whether an output data pin (SOUT, for example) 3-states when PCS is negated, or if the output data pin retains the last value, use CFGR1[OUTCFG].
- When configuring half-duplex transfers, you must configure the output data pins to 3-state when PCS is negated (CFGR1[OUTCFG] = 1).
- When performing half-duplex 2-bit transfers, you can write any value to CFGR1[PCSCFG].
- When performing half-duplex 4-bit transfers, you must write 1h to CFGR1[PCSCFG].

70.3.1.5 Clock loopback

Configure the LPSPI controller to use one of the following clocks to sample the input data:

- The SCK output clock
- A delayed version of the SCK output clock

The delayed version of the SCK is chosen by the SCK pin output delay, plus the SCK pin input delay, and is selected by writing 1 to CFGR1[SAMPLE]. Enabling the loopback version of the SCK pin can improve the setup time of the input data from the peripheral.

See the chip data sheet for the specific input setup time in Controller Loopback mode.

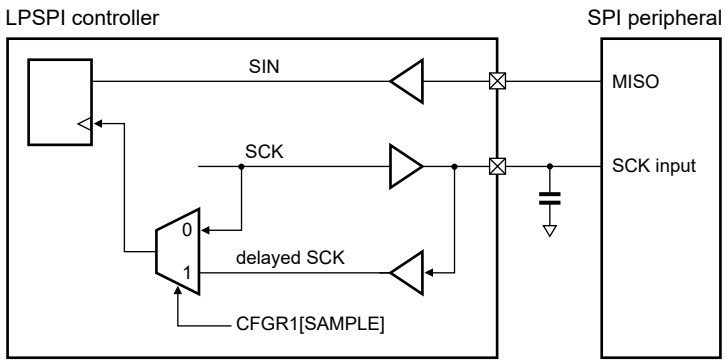


Figure 390. Clock loopback

70.3.2 Peripheral mode

LPSPI Peripheral mode:

- Uses the same shift register and logic that Controller mode uses.
- Does not use Clock Configuration (CCR).
- Requires Transmit Command (TCR) to remain static (unchanged) during SPI bus transfers.

70.3.2.1 Transmit and command FIFO commands

You must initialize Transmit Command (TCR) before enabling LPSPI in Peripheral mode, although this register is not updated until after LPSPI is enabled. After LPSPI is enabled, you must make changes to this register only when LPSPI is idle. In Peripheral mode, the LPSPI command word in this register controls SPI attributes. Before the PCS input asserts, the transmit FIFO must be filled with transmit data, or the transmit error flag sets.

Table 445. Command word in Peripheral mode

Transmit Command (TCR)		Description
Field	Name	
CPOL	Clock polarity	Specifies the polarity of the external SCK input.
CPHA	Clock phase	Specifies the clock phase of transfer.
PRESCALE	Prescaler value	Specifies the LPSPI functional clock prescaler.
PCS	Peripheral chip select	Specifies which PCS is used. The polarity of PCS is static and configured by CFGR1[PCSPOL]. If CFGR1[PCSCFG] is not equal to zero, then do not select the PCS[3:2] pins.
LSBF	LSB first	Specifies whether LSB (bit 0) or MSB (bit 31 for a 32-bit word) is transmitted or received first.
BYSW	Byte swap	Enables byte swap on each 32-bit word when transmitting and receiving data. Byte swapping can be useful when interfacing with devices that organize data as big-endian.
CONT	Continuous transfer	When continuous transfer is selected in Peripheral mode, after the number of bits indicated by FRAMESZ are transferred, LPSPI passes through and transmits the received data until the next PCS negation. Whatever is shifted in on the receive data is shifted out as transmit data considering that there is a 32-bit shift register.
CONTC	Continuing command	When the continuing command is enabled in Peripheral mode, after the number of bits indicated by FRAMESZ are transferred, RXMSK is considered equal to 1 and TXMSK is considered equal to 0 until the next PCS negation. CONTC can be used to change the direction of a transfer after the number of bits indicated by FRAMESZ.
RXMSK	Receive data mask	Masks the receive data; LPSPI does not store masked receive data in the receive FIFO or perform receive data matching. This option is useful for half-duplex transfers or to specify which fields are compared during receive data matching.
TXMSK	Transmit data mask	Masks the transmit data so that the masked transmit data is not pulled from transmit FIFO, and the output data pin is 3-stated (unless otherwise specified in CFGR1[OUTCFG]). This option is useful for half-duplex transfers.
WIDTH	Transfer width	Specifies the number of bits shifted on each SCK pulse: • 1-bit transfers support traditional SPI bus transfers in either half-duplex or full-duplex data formats. • 2-bit and 4-bit half-duplex transfers are useful for interfacing with QuadSPI memory devices, and at least either TCR[TXMSK] or TCR[RXMSK] must be 1. • 8-bit half-duplex transfers are useful for interfacing with OctelSPI memory devices, and at least either TCR[TXMSK] or TCR[RXMSK] must also be 1.
FRAMESZ	Frame size	Specifies the frame size in number of bits equal to (FRAMESZ + 1): • The minimum frame size is 8 bits, or 16 bits for an 8-bit parallel transfer. • If the frame size is larger than 32 bits, then the frame is divided into multiple words of 32 bits; each word is loaded from the transmit FIFO and stored in the receive FIFO separately. • If the size of the frame is not divisible by 32, then the last load of the transmit FIFO and store of the receive FIFO contain the remainder bits. For example, a 72-bit transfer consists of three words: the first and second words are 32 bits, and the third word is 8 bits.

70.3.2.2 Receive FIFO and data match

The receive FIFO stores receive data during SPI bus transfers. When TCR[RXMSK] = 1, the received data is discarded instead of storing the received data in the receive FIFO.

Receive data supports a receive data match function that can match received data against one of the two words in DMR0 and DMR1 or against a masked data word. You can also configure the data match function to compare only the first one or two received data words since the start of the frame.

- Received data that is already discarded because TCR[RXMSK] = 1 cannot cause the data match to set, and delays the match on the first received data word, until all discarded data is received.
- By using CFGR0[RDMO], you can also configure the receiver match function to discard all received data until a data match is detected.
- After a receive data match, to allow all subsequent data to be received, first write 0 to CFGR0[RDMO], then clear SR[DMF].

70.3.2.3 Partial received word

When the PCS pin deasserts and the receive shift register shifts in a partial word, you can configure the receive shift register to either discard the partial word or to store it in the receive FIFO. You must specify this using CFGR1[PARTIAL].

A partial word is defined as less than TCR[FRAMESZ] bits (when TCR[FRAMESZ] is equal or less than 32 bits, or it is the last word in a multi-word frame) or less than 32 bits (when TCR[FRAMESZ] is greater than 32 bits and not the last word in a multi-word frame).

A single-bit frame is not supported. A partial received word of 1 bit is supported, but a partial received frame of 1 bit is not supported.

70.3.2.4 Clocked interface

LPSPI supports interfacing with external controllers that provide only clock and data pins (PCS is not required). This interface requires:

- Writing 1 to TCR[CPHA] (data is changed on the leading edge of SCK and captured on the following edge).
- Configuring the PCS input to be always asserted (CFGR1[PCSPOL*n*] = 1). For example, to configure PCS[0] to be always asserted, write 1 to PCSPOL[0], and do not configure PCS[0] in the pin muxing. The chip-level drives PCS to a certain value (ideally 1); you could use CFGR1[PCSPOL*n*] to invert that value.
- Writing 1 to CFGR1[AUTOPCS] to enable automatic PCS generation. When CFGR1[AUTOPCS] = 1, a minimum of four LPSPI functional clock cycles (divided by the selection specified in TCR[PRESCALE]) is required between the last SCK edge of one word and the first SCK edge of the next word.

70.3.3 Debug mode

Table 446. Debug mode

Chip mode	LPSPI operation
Debug (the core is in Debug or Halted mode)	Can continue operating in Debug mode, if CR[DBGEN] = 1

70.3.4 Clocking

Table 447. LPSPI clocks

Type of clock	Description
Functional	• Asynchronous to the bus clock. • If the LPSPI functional clock remains enabled in low-power modes, then LPSPI can perform SPI bus transfers and low-power wakeups in both Controller and Peripheral modes. • LPSPI divides the functional clock by a prescaler; the resulting frequency must be at least two times faster than the SPI external clock frequency (SCK).
External	• The LPSPI shift register is clocked directly by the SCK clock. • How the SCK clock is generated or supplied depends on the mode (Controller or Peripheral): — In Controller mode, the SCK clock is generated internally. — In Peripheral mode, the SCK clock is supplied externally.
Bus	The bus clock is only used for bus accesses to the LPSPI control and configuration registers. The bus clock frequency must be high enough to support the data bandwidth requirements of the LPSPI registers, including the FIFOs.

See the chip-specific LPSPI information for more.

70.3.5 Reset

Table 448. LPSPI resets

Type of reset	Description
Chip	Resets the LPSPI logic and registers to their default states.
Software	• Resets the LPSPI logic and registers to their default states, except for the Control register. • The LPSPI software reset is controlled using CR[RST].
FIFO	• Resets the transmit and command FIFO and the receive FIFO. • CR[RTF] and CR[RRF] are write-only. • After being reset, FIFO is empty.

70.3.6 Interrupts and DMA requests

The following table lists sources (status flags) that can generate LPSPI interrupts and LPSPI transmit and receive DMA requests.

Table 449. Interrupts and DMA requests

Status (SR)		Description	Can generate		
Status flag	Name		Interrupt?	DMA request?	Low-power wake-up?
TDF	Transmit data flag	Indicates that data can be written to transmit FIFO, as configured by the transmit FIFO watermark, FCR[TXWATER].	Y	TX	Y

Table continues on the next page...

Table 449. Interrupts and DMA requests (continued)

Status (SR)		Description	Can generate		
Status flag	Name		Interrupt?	DMA request?	Low-power wake-up?
RDF	Receive data flag	Indicates that data can be read from the receive FIFO, as configured by the receive FIFO watermark, FCR[RXWATER].	Y	RX	Y
WCF	Word complete flag	Indicates that the word is complete and the last bit of the word has been sampled.	Y	N	Y
FCF	Frame complete flag	Indicates that the frame is complete and PCS is deasserted.	Y	N	Y
TCF	Transfer complete flag	Indicates that transfer is complete, PCS is deasserted, and the transmit and command FIFO is empty.	Y	N	Y
TEF	Transmit error flag	Indicates a transmit and command FIFO underrun. In Controller mode, when CFGR1[NOSTALL] = 0 (transfers stall when transmit FIFO is empty), TEF cannot be set.	Y	N	Y
REF	Receive error flag	Indicates a receive FIFO overflow. In Controller mode, when CFGR1[NOSTALL] = 0 (transfers stall when receive FIFO is full), REF cannot be set.	Y	N	Y
DMF	Data match flag	Indicates that the received data matches the configured data match value.	Y	N	Y
MBF	Module busy flag	Indicates that LPSPI is busy performing an SPI bus transfer.	N	N	N

70.3.6.1 DMA support registers

To support efficient DMA transfers to the transmit and control FIFO, an alias register supports 32-bit write access to the Transmit Command (TCR) and an alias region supports incrementing 32-bit write accesses to the Transmit Data (TDR).

- Transmit Command Burst (TCBR) is a 32-bit alias register for writing to TCR.
- Transmit Data Burst (TDBR0 - TDBR127) is a 512-byte alias region that supports writing to the TDR.

The burst alias locations are contiguous. A DMA transfer can start by writing to the TCBR register to initialize the transfer and then increment into the TDBR region without changing the DMA transfer size. The alias registers can also be used by the DMA to perform burst transfers when accessing the transmit FIFO.

The transmit FIFO prevents writes that overflow the FIFO, but this prevention does not signal an error. Do not perform writes to the TDBR unless there is sufficient room in the transmit FIFO.

Receive Data Burst (RDBR0 - RDBR127) is a 512-byte alias region that supports reading the Receive Data. This can be used by the DMA to perform burst transfers when accessing the receive FIFO.

70.3.7 Peripheral triggers

The connection of the LPSPI peripheral triggers with other peripherals depends on the device that is used.

Table 450. Peripheral triggers

Type of trigger	Description	Additional information
Frame output	The frame output trigger: • Asserts at the end of each frame (when PCS deasserts). • Remains asserted for one cycle of the LPSPI functional clock divided by the configuration defined in TCR[PRESCALE].	LPSPI generates two output triggers that can be connected to other peripherals on the chip.
Word output	The word output trigger: • Asserts at the end of each received word. • Remains asserted for one cycle of the LPSPI functional clock divided by the configuration defined in TCR[PRESCALE].	
Input	To control the start of an LPSPI bus transfer, the LPSPI input trigger can be selected instead of the HREQ input: • The LPSPI input trigger is synchronized, and must assert for at least two cycles of the LPSPI functional clock divided by the configuration defined in TCR[PRESCALE] so that the input trigger can be detected. • When LPSPI is busy, the HREQ input (and therefore the LPSPI input trigger) is ignored. • When LPSPI is busy, both the HREQ and LPSPI input triggers are ignored. They are used to start a new transfer when LPSPI is idle.	

70.4 External signals

Table 451. External signals

Signal	Name	Description	I/O
SCK	Serial clock	• Input in Peripheral mode • Output in Controller mode	I/O
PCS[0]	Peripheral chip select	• Input in Peripheral mode • Output in Controller mode	I/O
PCS[1]/HREQ	Peripheral chip select or host request	Host request pin is selected when CFGR0[HREN] = 1 and CFGR0[HRSEL] = 0: • Input in either Peripheral mode or when used as controller host request • Output in either Controller mode or when used as peripheral host request	I/O
PCS[2]/DATA[2]	Peripheral chip select or data pin 2 during parallel data transfers	When CFGR1[PCSCFG] = 0: • Input in Peripheral mode • Output in Controller mode When CFGR1[PCSCFG] = 1:	I/O

Table continues on the next page...

Table 451. External signals (continued)

Signal	Name	Description	I/O
		• Input in half-duplex parallel data receive transfers • Output in half-duplex parallel data transmit transfers	
PCS[3]/DATA[3]	Peripheral chip select or data pin 3 during parallel data transfers	When CFGR1[PCSCFG] = 0: • Input in Peripheral mode • Output in Controller mode When CFGR1[PCSCFG] = 1: • Input in half-duplex parallel data receive transfers • Output in half-duplex parallel data transmit transfers	I/O
PCS[4]/DATA[4]	Peripheral chip select or data pin 4 during parallel data transfers	When CFGR1[PCSCFG] = 0: • Input in Peripheral mode • Output in Controller mode When CFGR1[PCSCFG] = 11b: • Input in half-duplex parallel data receive transfers • Output in half-duplex parallel data transmit transfers	I/O
PCS[5]/DATA[5]	Peripheral chip select or data pin 5 during parallel data transfers	When CFGR1[PCSCFG] = 0: • Input in Peripheral mode • Output in Controller mode When CFGR1[PCSCFG] = 11b: • Input in half-duplex parallel data receive transfers • Output in half-duplex parallel data transmit transfers	I/O
PCS[6]/DATA[6]	Peripheral chip select or data pin 6 during parallel data transfers	When CFGR1[PCSCFG] = 0: • Input in Peripheral mode • Output in Controller mode When CFGR1[PCSCFG] = 11b: • Input in half-duplex parallel data receive transfers • Output in half-duplex parallel data transmit transfers	I/O

Table continues on the next page...

Table 451. External signals (continued)

Signal	Name	Description	I/O
PCS[7]/DATA[7]	Peripheral chip select or data pin 7 during parallel data transfers	When CFGR1[PCSCFG] = 0: • Input in Peripheral mode • Output in Controller mode When CFGR1[PCSCFG] = 11b: • Input in half-duplex parallel data receive transfers • Output in half-duplex parallel data transmit transfers	I/O
SOUT/DATA[0]	Serial data output	Can be configured as serial data input signal (used as data pin 0 in half-duplex parallel data transfers)	I/O
SIN/DATA[1]	Serial data input	Can be configured as serial data output signal (used as data pin 1 in half-duplex parallel data transfers)	I/O

70.5 Initialization

This module does not require initialization.

70.6 Memory map and registers

NOTE

- Writing to a read-only register or reading a write-only register can cause bus errors.
- LPSPI does not check values programmed in registers for validity, so you must take care to write valid values only.

70.6.1 LPSPI register descriptions

70.6.1.1 LPSPI memory map

LPSPI_0 base address: 4035_8000h

LPSPI_1 base address: 4035_C000h

LPSPI_2 base address: 4036_0000h

LPSPI_3 base address: 4036_4000h

LPSPI_4 base address: 404B_C000h

LPSPI_5 base address: 404C_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID (VERID)	32	R	0200_0004h
4h	Parameter (PARAM)	32	R	See section

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
10h	Control (CR)	32	RW	0000_0000h
14h	Status (SR)	32	RW	0000_0001h
18h	Interrupt Enable (IER)	32	RW	0000_0000h
1Ch	DMA Enable (DER)	32	RW	0000_0000h
20h	Configuration 0 (CFGR0)	32	RW	0000_0000h
24h	Configuration 1 (CFGR1)	32	RW	0000_0000h
30h	Data Match 0 (DMR0)	32	RW	0000_0000h
34h	Data Match 1 (DMR1)	32	RW	0000_0000h
40h	Clock Configuration (CCR)	32	RW	0000_0000h
44h	Clock Configuration 1 (CCR1)	32	RW	0000_0000h
58h	FIFO Control (FCR)	32	RW	0000_0000h
5Ch	FIFO Status (FSR)	32	R	0000_0000h
60h	Transmit Command (TCR)	32	RW	0000_001Fh
64h	Transmit Data (TDR)	32	W	0000_0000h
70h	Receive Status (RSR)	32	R	0000_0002h
74h	Receive Data (RDR)	32	R	0000_0000h
78h	Receive Data Read Only (RDROR)	32	R	0000_0000h
3FCh	Transmit Command Burst (TCBR)	32	W	0000_0000h
400h - 5FCh	Transmit Data Burst (TDBR0 - TDBR127)	32	W	0000_0000h
600h - 7FCh	Receive Data Burst (RDBR0 - RDBR127)	32	R	0000_0000h

70.6.1.2 Version ID (VERID)

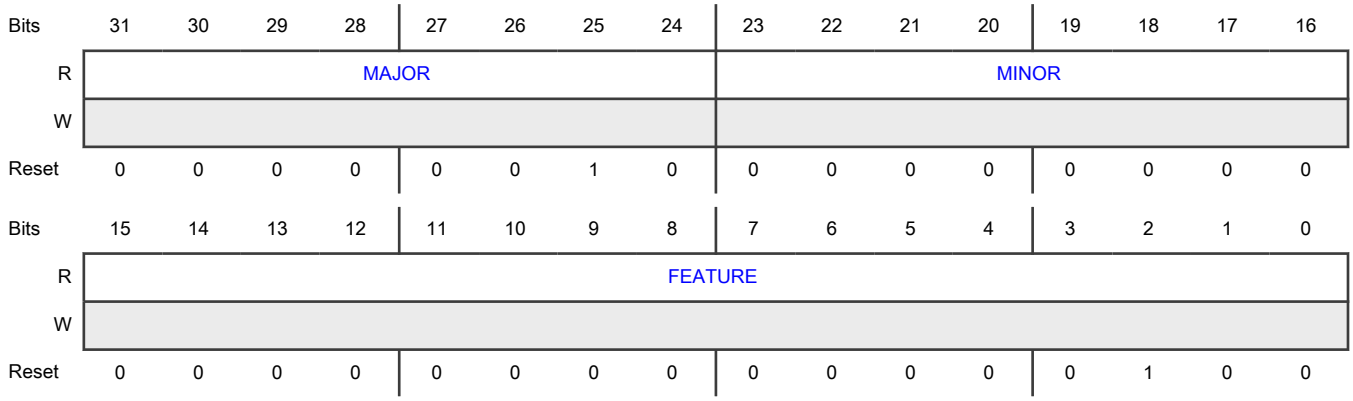
Offset

Register	Offset
VERID	0h

Function

Contains version numbers for the module design and feature set.

Diagram



Fields

Field	Function
31-24 MAJOR	Major Version Number Indicates the major version number of the module specification.
23-16 MINOR	Minor Version Number Indicates the minor version number of the module specification.
15-0 FEATURE	Module Identification Number Indicates the feature set number 0000_0000_0000_0100b - Standard feature set supporting a 32-bit shift register. All other values are reserved.

70.6.1.3 Parameter (PARAM)

Offset

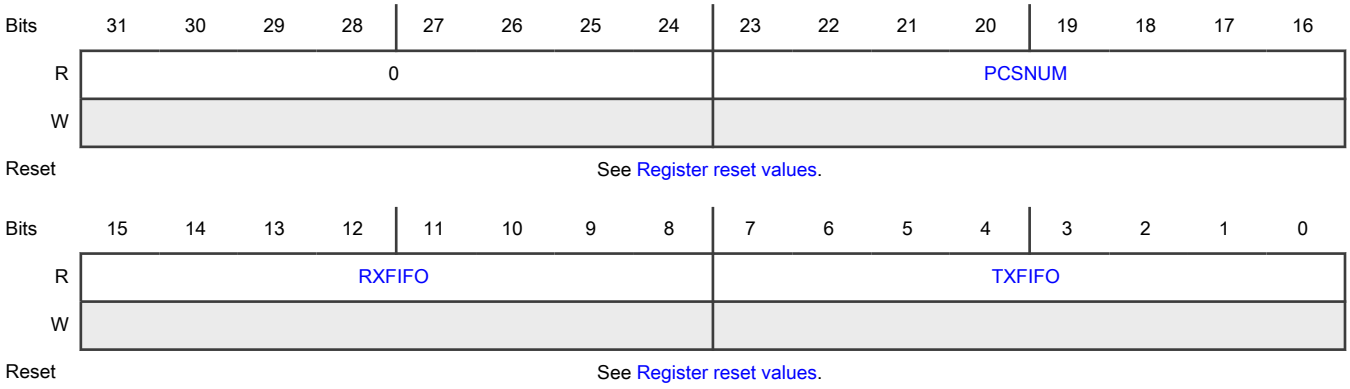
Register	Offset
PARAM	4h

Function

Contains:

- Number of PCS pins.
- Receive FIFO size.
- Transmit FIFO size.

Diagram



Register reset values

Register	Reset value
PARAM	LPSPI_0: 0008_0202h LPSPI_1: 0006_0202h LPSPI_2–LPSPI_5: 0004_0202h

Fields

Field	Function
31-24 —	Reserved
23-16 PCSNUM	PCS Number Indicates the number of PCS pins supported.
15-8 RXFIFO	Receive FIFO Size Indicates the maximum number of words in the receive FIFO. The maximum number of words is 2^{RXFIFO}.
7-0 TXFIFO	Transmit FIFO Size Indicates the maximum number of words in the transmit FIFO. The maximum number of words is 2^{TXFIFO}.

70.6.1.4 Control (CR)

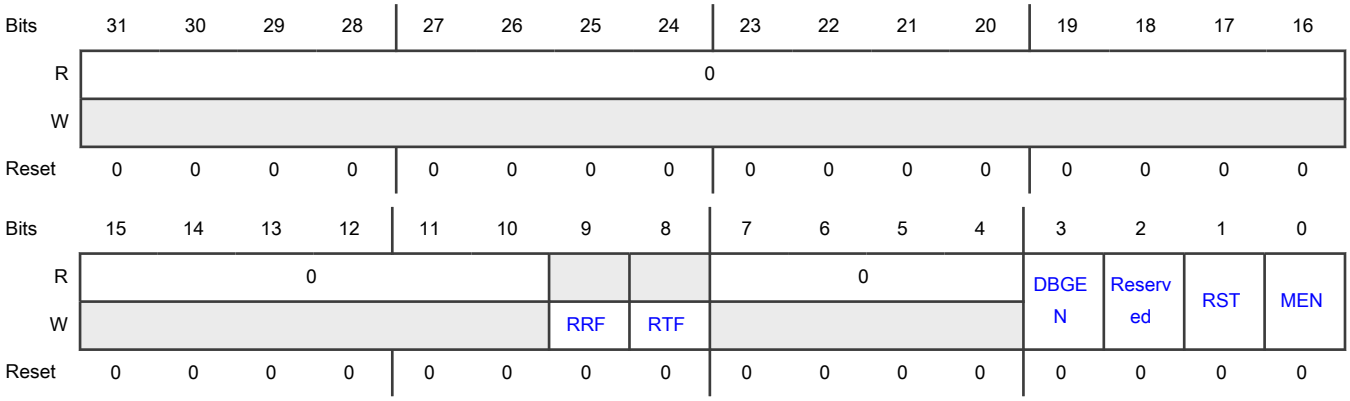
Offset

Register	Offset
CR	10h

Function

Contains fields that control the module operation.

Diagram



Fields

Field	Function
31-10 —	Reserved
9 RRF	Reset Receive FIFO Deletes all entries in the receive FIFO. This field always reads 0. 0b - No effect 1b - Reset
8 RTF	Reset Transmit FIFO Deletes all entries in the transmit FIFO. This field always reads 0. 0b - No effect 1b - Reset
7-4 —	Reserved
3 DBGEN	Debug Enable Enables LPSPI when the CPU is in Debug mode. If this field is 0, LPSPI is disabled when the CPU is halted; the PCS pin is deasserted after the transmit FIFO is empty regardless of the state of Transmit Command (TCR). You must update this field only when LPSPI is disabled (MEN = 0). 0b - Disable 1b - Enable
2 —	Reserved
1	Software Reset

Table continues on the next page...

Table continued from the previous page...

Field	Function
RST	Resets all internal logic and registers, except Control (CR). The reset takes effect immediately and remains asserted until you write 0 to it. There is no minimum delay required before clearing the software reset by writing 0. 0b - Not reset 1b - Reset
0 MEN	Module Enable Enables the module. After writing 0, MEN remains set until LPSPI has completed the current transfer and is idle. 0b - Disable 1b - Enable

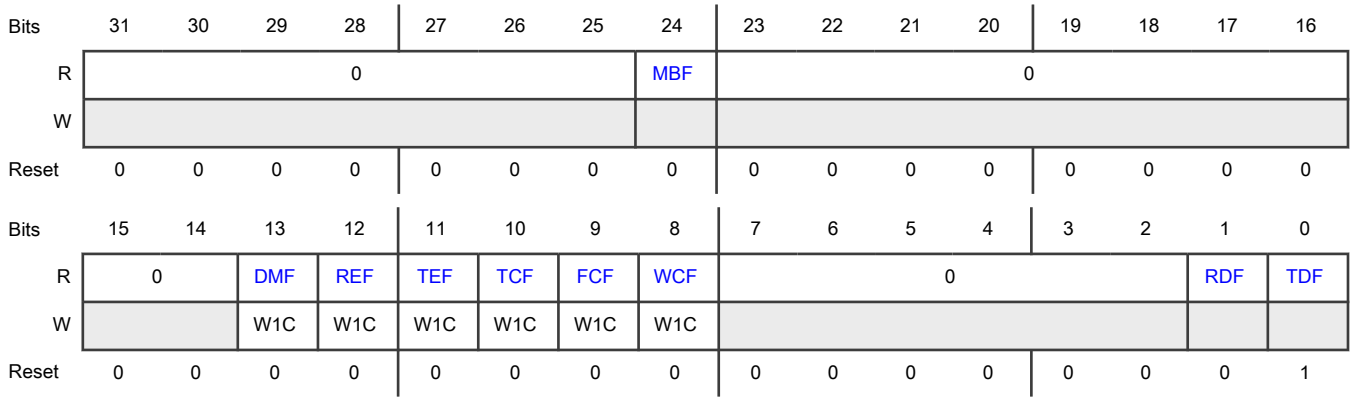
70.6.1.5 Status (SR)

Offset

Register	Offset
SR	14h

Function
Contains data flow status.

Diagram



Fields

Field	Function
31-25	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
24 MBF	Module Busy Flag Indicates, in Controller mode, whether there is data to transmit and LPSPI is able to transmit (for example, the HREQ pin is asserted). The HREQ pin deasserts after the PCS pin deasserts and the LPSPI controller has waited for half the time specified in CCR[DBT] with no new data to transmit. Peripheral mode sets this flag when LPSPI is enabled and PCS is asserted. 0b - LPSPI is idle 1b - LPSPI is busy
23-14 —	Reserved
13 DMF	Data Match Flag Indicates whether the received data matches DMR0[MATCH0] and/or DMR1[MATCH1] (as configured by CFGR1[MATCFG]). **NOTE** This field behaves differently for register reads and writes. When reading 0b - No match 1b - Match When writing 0b - No effect 1b - Clear the flag
12 REF	Receive Error Flag Indicates a receive FIFO overflow error. When this flag is set: 1. End the transfer. 2. Empty the receive FIFO. 3. Clear this flag. 4. Restart the transfer from the beginning. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No overflow 1b - Overflow When writing

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No effect 1b - Clear the flag
11 TEF	Transmit Error Flag Indicates a transmit FIFO underrun error. When this flag is set: 1. End the transfer. 2. Clear this flag. 3. Restart the transfer from the beginning. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No underrun 1b - Underrun When writing 0b - No effect 1b - Clear the flag
10 TCF	Transfer Complete Flag Indicates, in Controller mode, whether all transfers are complete and LPSPI has returned to the Idle state and the transmit FIFO is empty. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not complete 1b - Complete When writing 0b - No effect 1b - Clear the flag
9 FCF	Frame Complete Flag Indicates whether a frame transfer is complete after PCS deasserts. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not complete

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Complete When writing 0b - No effect 1b - Clear the flag
8 WCF	Word Complete Flag Indicates whether the last bit of a received word is sampled. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not complete 1b - Complete When writing 0b - No effect 1b - Clear the flag
7-2 —	Reserved
1 RDF	Receive Data Flag Indicates whether the number of words in the receive FIFO is greater than the value in FCR[RXWATER]. 0b - Receive data not ready 1b - Receive data ready
0 TDF	Transmit Data Flag Indicates whether the number of words in the transmit FIFO is equal to or less than the value in FCR[TXWATER]. 0b - Transmit data not requested 1b - Transmit data requested

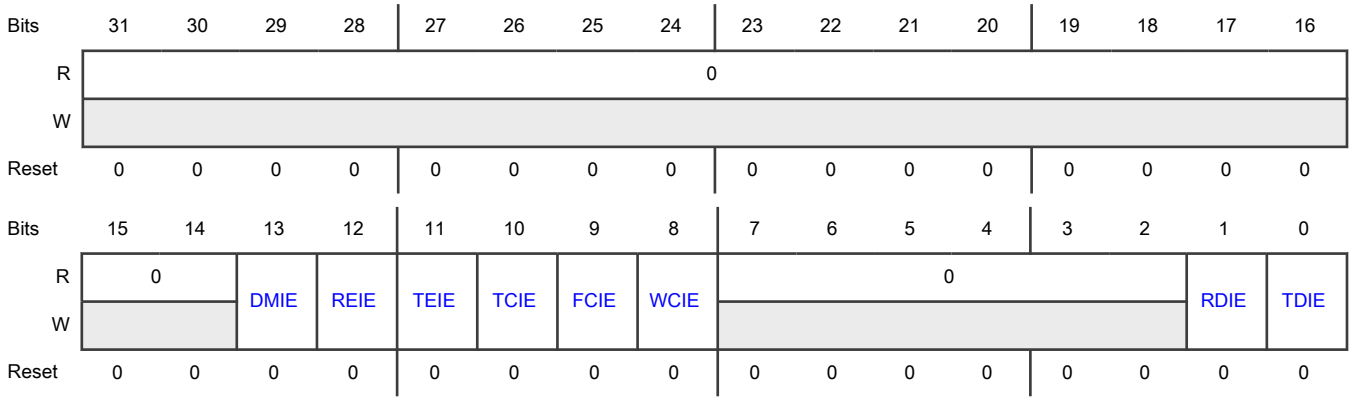
70.6.1.6 Interrupt Enable (IER)

Offset

Register	Offset
IER	18h

Function
Enables interrupts based on data flow and errors.

Diagram



Fields

Field	Function
31-14 —	Reserved
13 DMIE	Data Match Interrupt Enable Enables the data match interrupt. 0b - Disable 1b - Enable
12 REIE	Receive Error Interrupt Enable Enables the receive error interrupt. 0b - Disable 1b - Enable
11 TEIE	Transmit Error Interrupt Enable Enables the transmit error interrupt. 0b - Disable 1b - Enable
10 TCIE	Transfer Complete Interrupt Enable Enables the transfer complete interrupt. 0b - Disable 1b - Enable
9 FCIE	Frame Complete Interrupt Enable Enables the frame complete interrupt. 0b - Disable 1b - Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
8 WCIE	Word Complete Interrupt Enable Enables the word complete interrupt. 0b - Disable 1b - Enable
7-2 —	Reserved
1 RDIE	Receive Data Interrupt Enable Enables the receive data interrupt. 0b - Disable 1b - Enable
0 TDIE	Transmit Data Interrupt Enable Enables the transmit data interrupt. 0b - Disable 1b - Enable

70.6.1.7 DMA Enable (DER)

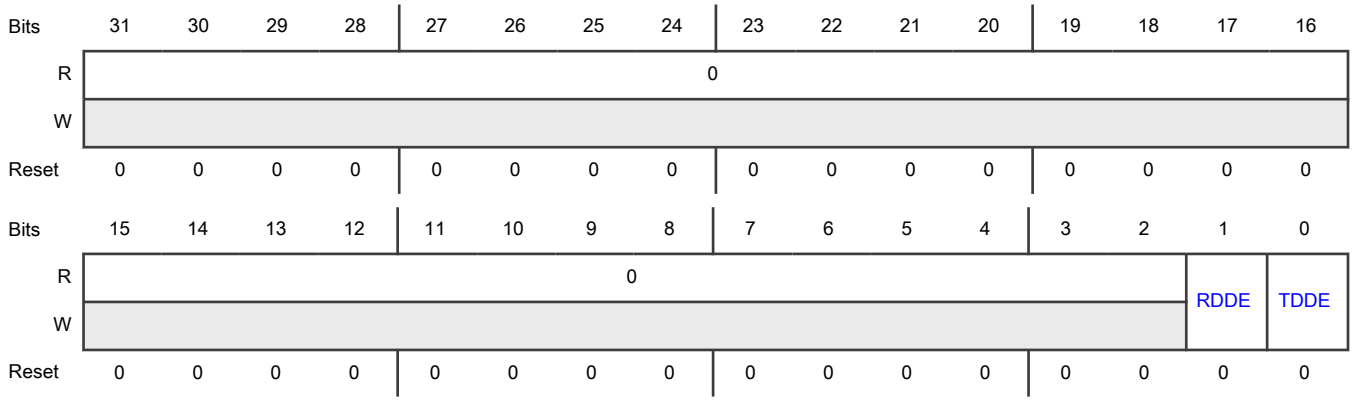
Offset

Register	Offset
DER	1Ch

Function

Enables the DMA data flow.

Diagram



Fields

Field	Function
31-2 —	Reserved
1 RDDE	Receive Data DMA Enable Enables the receive data DMA. 0b - Disable 1b - Enable
0 TDDE	Transmit Data DMA Enable Enables the transmit data DMA. 0b - Disable 1b - Enable

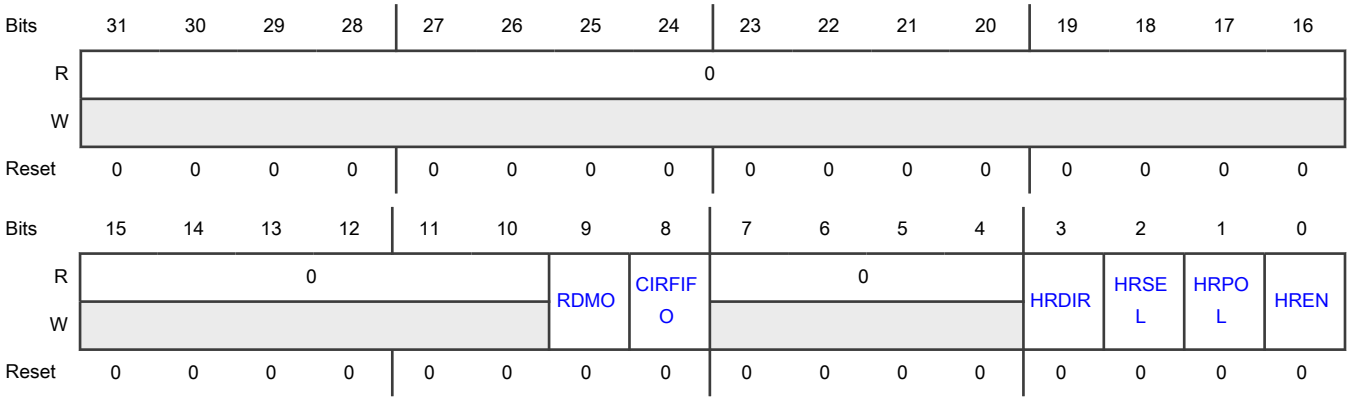
70.6.1.8 Configuration 0 (CFGR0)

Offset

Register	Offset
CFGR0	20h

Function
Includes fields to configure LPSPI.

Diagram



Fields

Field	Function
31-10 —	Reserved
9 RDMO	Receive Data Match Only Enables receive data match. When enabled, all received data that does not cause SR[DMF] to assert is discarded: • Write 1 to this field when LPSPI is idle and SR[DMF] = 0. • After SR[DMF] = 1, this field is ignored. • To ensure that no receive data is lost when disabling RDMO, write 0 to this field before clearing SR[DMF]. See CFGR1[MATCFG] for the received data matching options. When disabled, all received data is stored in the receive FIFO. 0b - Disable 1b - Enable
8 CIRFIFO	Circular FIFO Enable Enables circular FIFO. When enabled, the transmit FIFO read pointer is saved to a temporary register. The transmit FIFO is emptied as in normal operation, but when LPSPI is idle and the transmit FIFO is empty, the read pointer value is restored from the temporary register. This restoring of the read pointer causes the contents of the transmit FIFO to be cycled through repeatedly. **NOTE** The read pointer is restored for as long as this field is 1. Writing additional words to the FIFO when this field is 1 adds them to the end of the FIFO, up to the size of the transmit FIFO. 0b - Disable 1b - Enable
7-4 —	Reserved
3 HRDIR	Host Request Direction Specifies the direction of the HREQ pin. You must configure the HREQ pin only as an output when LPSPI is in Peripheral mode. The HREQ pin direction must be an input for Controller mode. 0b - Input 1b - Output
2 HRSEL	Host Request Select Specifies the source of the host request input. When the host request function is enabled with the HREQ pin, the PCS[1] function is disabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - HREQ pin 1b - Input trigger
1 HRPOL	Host Request Polarity Specifies the polarity of the HREQ pin or input trigger. 0b - Active high 1b - Active low
0 HREN	Host Request Enable Enables LPSPI, in Controller mode, to start a new SPI bus transfer only if the host request input is asserted. When LPSPI is busy, the host request input is ignored. In Peripheral mode, causes the HREQ output pin to assert when data is available to be transmitted. 0b - Disable 1b - Enable

70.6.1.9 Configuration 1 (CFGR1)

Offset

Register	Offset
CFGR1	24h

Function

Includes fields to configure LPSPI. You must write to this register only when LPSPI is disabled.

In addition to pin and output configurations, this register contains match configuration details; the following table shows match conditions specified in MATCFG.

Table 452. Match conditions for CFGR1[MATCFG]

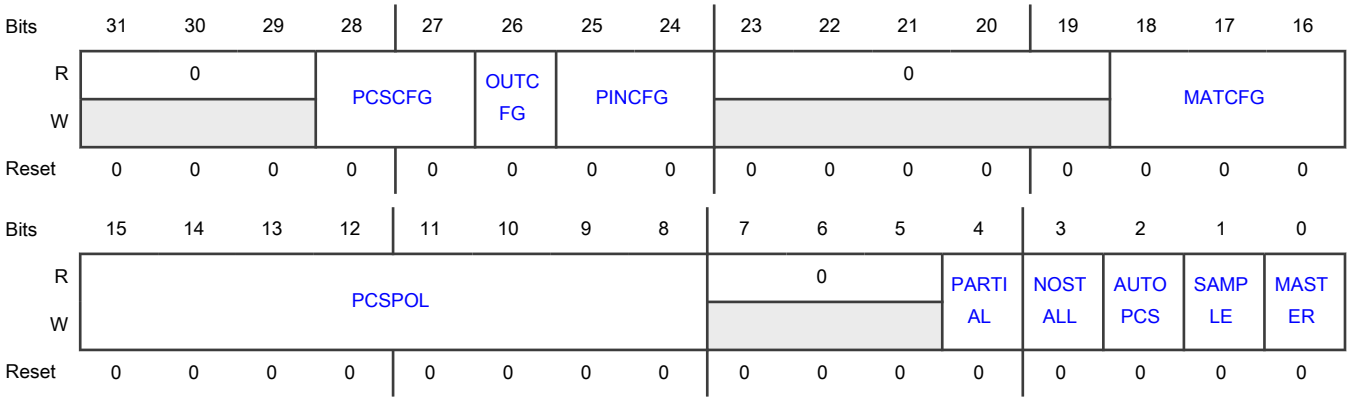
Condition	Description
Match first data word with compare word	Match if first data word equals MATCH0 logically ORed with MATCH1 first_data_word == (MATCH0 || MATCH1)
Match any data word with compare word	Match if any data word equals MATCH0 logically ORed with MATCH1 any_data_word == (MATCH0 || MATCH1)
Sequential match, first data word	Match if first data word equals MATCH0, and second data word equals MATCH1 (first_data_word == MATCH0) && (second_data_word == MATCH1)
Sequential match, any data word	Match if any data word equals MATCH0, and the next data word equals MATCH1

Table continues on the next page...

Table 452. Match conditions for CFGR1[MATCFG] (continued)

Condition	Description
	(any_data_word == MATCH0) && (next_data_word == MATCH1)
Match first data word (masked) with compare word (masked)	Match if first data word logically ANDed with MATCH1 equals MATCH0 logically ANDed with MATCH1 (first_data_word && MATCH1) == (MATCH0 && MATCH1)
Match any data word (masked) with compare word (masked)	Match if any data word logically ANDed with MATCH1 equals MATCH0 logically ANDed with MATCH1 (any_data_word && MATCH1) == (MATCH0 && MATCH1)

Diagram



Fields

Field	Function
31-29 —	Reserved
28-27 PCSCFG	Peripheral Chip Select Configuration Specifies PCS pin configuration. When performing parallel transfers, you must configure this field to enable the desired transfer. **NOTE** This field is not supported in every instance. The following table includes only supported registers.

Instance	Field supported in	Field not supported in
LPSPI_0	CFGR1	—
LPSPI_1	CFGR1[27]	CFGR1[28]
LPSPI_2	CFGR1[27]	CFGR1[28]

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>LPSPI_3</td><td>CFGR1[27]</td><td>CFGR1[28]</td></tr>
<tr><td>LPSPI_4</td><td>CFGR1[27]</td><td>CFGR1[28]</td></tr>
<tr><td>LPSPI_5</td><td>CFGR1[27]</td><td>CFGR1[28]</td></tr>
</table>
NOTE
The descriptions of the field settings vary by module instance.
<table>
<tr><th>Instance</th><th>Field value and description</th></tr>
<tr><td>LPSPI_0</td><td>00b - PCS[7:2] configured for chip select function
01b - PCS[3:2] configured for half-duplex 4-bit transfers (PCS[3:2] = DATA[3:2])
11b - PCS[7:2] configured for half-duplex 4-bit and 8-bit transfers (PCS[7:2] = DATA[7:2])</td></tr>
<tr><td>LPSPI_1</td><td>0b - PCS[5:2] configured for chip select function
1b - PCS[3:2] configured for half-duplex 4-bit transfers (PCS[3:2] = DATA[3:2])</td></tr>
<tr><td>LPSPI_2
LPSPI_3
LPSPI_4
LPSPI_5</td><td>0b - PCS[3:2] configured for chip select function
1b - PCS[3:2] configured for half-duplex 4-bit transfers (PCS[3:2] = DATA[3:2])</td></tr>
</table>
</td></tr>
<tr><td>26
OUTCFG</td><td>Output Configuration
Specifies whether the output data is 3-stated between accesses (when PCS is deasserted). When performing half-duplex transfers, this field must be 1.
0b - Retain last value
1b - 3-stated</td></tr>
<tr><td>25-24
PINCFG</td><td>Pin Configuration
Specifies the pins used for input and output data during serial transfers.
NOTE
When performing parallel transfers, this field must be 00b or 10b.
00b - SIN is used for input data; SOUT is used for output data
01b - SIN is used for both input and output data; only half-duplex serial transfers are supported</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10b - SOUT is used for both input and output data; only half-duplex serial transfers are supported 11b - SOUT is used for input data; SIN is used for output data
23-19 —	Reserved
18-16 MATCFG	Match Configuration Specifies the condition that causes SR[DMF] to assert. See the match conditions listed in Table 1 for more information. **NOTE** When writing to this field, either the old value or new value must be in the disabled state (0). You cannot transition from a nonzero value to another nonzero value. 000b - Match is disabled 001b - Reserved 010b - Match first data word with compare word 011b - Match any data word with compare word 100b - Sequential match, first data word 101b - Sequential match, any data word 110b - Match first data word (masked) with compare word (masked) 111b - Match any data word (masked) with compare word (masked)
15-8 PCSPOL	Peripheral Chip Select Polarity Specifies the polarity of each PCS pin. Bit *n* in this field (the least-significant bit is bit 0) corresponds to PCS[*n*]. **NOTE** This field is not supported in every instance. The following table includes only supported registers.

Instance	Field supported in	Field not supported in
LPSPI_0	CFGR1	—
LPSPI_1	CFGR1[13–8]	CFGR1[15–14]
LPSPI_2	CFGR1[11–8]	CFGR1[15–12]
LPSPI_3	CFGR1[11–8]	CFGR1[15–12]
LPSPI_4	CFGR1[11–8]	CFGR1[15–12]

Table continues on the next page...

Table continued from the previous page...

Field	Function
	<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>LPSPI_5</td><td>CFGR1[11–8]</td><td>CFGR1[15–12]</td></tr></table> **NOTE** The descriptions of the field settings vary by module instance. <table><tr><th>Instance</th><th>Field value and description</th></tr><tr><td>LPSPI_0</td><td>0000_0000b - Active low 0000_0001b - Active high</td></tr><tr><td>LPSPI_1</td><td>00_0000b - Active low 00_0001b - Active high</td></tr><tr><td>LPSPI_2 LPSPI_3 LPSPI_4 LPSPI_5</td><td>0000b - Active low 0001b - Active high</td></tr></table>
7-5 —	Reserved
4 PARTIAL	Partial Enable Specifies whether LPSPI, when in Peripheral mode, stores a partial received word in the receive FIFO, or discards it, when PCS deasserts. See Partial received word for more information. 0b - Discard 1b - Store
3 NOSTALL	No Stall Disables a normal operating feature that causes LPSPI, when in Controller mode, to stall transfers when the transmit FIFO is empty or when the receive FIFO is full. This feature prevents transmit FIFO underruns and receive FIFO overruns. Writing 1 to this field disables this functionality. 0b - Disable 1b - Enable
2 AUTOPCS	Automatic PCS Enables automatic PCS generation. For correct operation in Peripheral mode, LPSPI requires the PCS signal to deassert between frames. Writing 1 to this field generates an internal PCS signal at the end of each transfer word when TCR[CPHA] = 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When this field is 1, SCK must remain idle for at least four LPSPI functional clock cycles, divided by the prescaler (see TCR[PRESCALE]) selected between each word to ensure correct operation. This field is ignored in Controller mode. 0b - Disable 1b - Enable
1 SAMPLE	Sample Point Specifies the SCK clock edge on which LPSPI, when in Controller mode, samples input data. Writing 1 to this field causes LPSPI to sample input data on a delayed loopback SCK clock edge, which improves the setup time when sampling data (see Clock loopback). In this configuration, the input data setup time in Controller mode is equal to the input data setup time in Peripheral mode. In Peripheral mode, this field is ignored. **NOTE** When SAMPLE = 1, both the input and output buffers must be enabled for the SCK pin. Buffers are configured at the chip level. 0b - SCK edge 1b - Delayed SCK edge
0 MASTER	Controller Mode Specifies the LPSPI operating mode, Controller or Peripheral. This field directly controls the direction of the SCK and PCS pins. 0b - Peripheral mode 1b - Controller mode

70.6.1.10 Data Match 0 (DMR0)

Offset

Register	Offset
DMR0	30h

Function

Specifies the match data to be used when data matching is enabled. See CFGR1[MATCFG] for the received data matching options.

NOTE

Do not change the value in this register when CFGR1[MATCFG] > 0.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MATCH0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MATCH0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 MATCH0	Match 0 Value Specifies the MATCH0 value to be compared against received data.

70.6.1.11 Data Match 1 (DMR1)

Offset

Register	Offset
DMR1	34h

Function

Specifies the match data to be used when data matching is enabled. See CFGR1[MATCFG] for the received data matching options.

NOTE

Do not change the value in this register while CFGR1[MATCFG] > 0.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MATCH1															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MATCH1															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 MATCH1	Match 1 Value Specifies the MATCH1 value to be compared against received data.

70.6.1.12 Clock Configuration (CCR)

Offset

Register	Offset
CCR	40h

Function

Contains clock configuration fields that are used only in Controller mode; you can only change them when LPSPI is disabled (CR[MEN] = 0).

Warning

Writing a 32-bit value to this register overwrites Clock Configuration 1 (CCR1); DBT and SCKDIV always read 0.

To avoid overwriting CCR1, do one of the following:

- Write to all four fields in Clock Configuration (CCR) simultaneously and only once in a 32-bit data.
- Modify the values of CCR[SCKPCS] and/or CCR[PCSSCK]; write only these two upper bytes in a 16-bit data or one of them in an 8-bit data.
- Modify CCR1[PCSPCS] and CCR1[SCKSCK] only or CCR1[SCKSET] and CCR1[SCKHLD] only, write respectively to CCR[DBT] or CCR[SCKDIV] in 8-bit data.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SCKPCS								PCSSCK							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								0							
W	DBT								SCKDIV							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24	SCK-to-PCS Delay

Table continues on the next page...

Table continued from the previous page...

Field	Function
SCKPCS	Configures SCK-to-PCS delay. In Controller mode, this field helps you configure the delay from the last SCK edge to PCS negation: • The delay is equal to (SCKPCS + 1) cycles of the LPSPI functional clock divided by the selected prescaler (see TCR[PRESCALE]). • The minimum delay is one cycle. See Figure 389 for more information.
23-16 PCSSCK	PCS-to-SCK Delay Configures PCS-to-SCK delay. In Controller mode, this field helps you configure the delay from PCS assertion to the first SCK edge: • The delay is equal to (PCSSCK + 1) cycles of the LPSPI functional clock divided by the selected prescaler (see TCR[PRESCALE]). • The minimum delay is one cycle. See Figure 389 for more information.
15-8 DBT	Delay Between Transfers Configures the delay between transfers. Writing to this field updates the contents of CCR1[PCSPCS] and CCR1[SCKSCK].
7-0 SCKDIV	SCK Divider Updates the contents of CCR1[SCKSET] and CCR1[SCKHLD]. Baud rate = function clock ÷ (2^PRESCALE × (SCKSET + SCKHLD + 2))

70.6.1.13 Clock Configuration 1 (CCR1)

Offset

Register	Offset
CCR1	44h

Function

Contains clock configuration fields, which are used only in Controller mode. You can change them only when LPSPI is disabled (CR[MEN] = 0).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SCKSCK								PCSPCS							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SCKHLD								SCKSET							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 SCKSCK	SCK Inter-Frame Delay Configures SCK inter-frame delay in Controller mode: • This field helps you configure the delay from the last SCK pulse of a frame and the first SCK pulse of the following frame, in a continuous transfer. • The delay is equal to (SCKSCK + 1) cycles of the LPSPI functional clock divided by the selected prescaler (see TCR[PRESCALE]). • The minimum delay is one cycle. **NOTE** For backward compatibility, writing to CCR[DBT] updates this field with the value written.
23-16 PCSPCS	PCS to PCS Delay Configures PCS to PCS delay in Controller mode: • This field helps you configure the delay from the PCS negation to the next PCS assertion. • The delay is equal to (PCSPCS + PCSPCS + 2) cycles of the LPSPI functional clock divided by the selected prescaler (see TCR[PRESCALE]). • The minimum delay is two cycles. • Half of the delay (PCSPCS + 1) occurs before PCS assertion and the other half of the delay (PCSPCS + 1) occurs after PCS negation. If the command word is updated between two transfers, then the command word is updated halfway between the PCS negation of the last transfer and PCS assertion of the next transfer. • The command word specifies which PCS signal is used, the polarity and phase of the SCK signal, and the selected prescaler. **NOTE** For backward compatibility, writing to CCR[DBT] updates this field with (DBT÷2) rounded up.
15-8	SCK Hold

Table continues on the next page...

Table continued from the previous page...

Field	Function
SCKHLD	Configures the hold phase of the SCK pin in Controller mode: • The hold phase is the delay between the SCK edge that samples the receive data and the SCK edge that drives the transmit data. • It is the SCK low period when CPHA = 0 and CPOL = 1, or CPHA = 1 and CPOL = 0. It is the SCK high period when CPHA = 0, CPOL = 0 and CPHA = 1, CPOL = 1. • The SCK hold phase delay is equal to (SCKHLD + 1) cycles of the LPSPI functional clock divided by the selected prescaler (see TCR[PRESCALE]). • The minimum delay is one cycle. • The SCK period is equal to (SCKSET + SCKHLD + 2) cycles of the LPSPI functional clock divided by the selected prescaler (see TCR[PRESCALE]). • The SCK duty cycle is based on the difference between SCKSET and SCKHLD. You must configure both these fields to the same value for a 50/50 duty cycle. See Figure 389 for more information. **NOTE** For backward compatibility, writing to CCR[SCKDIV] updates this field with (SCKDIV ÷ 2) rounded down.
7-0 SCKSET	SCK Setup Configures the setup phase of the SCK pin in Controller mode: • The setup phase is the delay between the SCK edge that drives the transmit data and the SCK edge that samples the receive data. • It is the SCK high period when CPHA = 0 and CPOL = 1, or CPHA = 1 and CPOL = 0. It is the SCK low period when CPHA = 0 and CPOL = 0, or CPHA = 1 and CPOL = 1. • The SCK setup phase delay is equal to (SCKSET + 1) cycles of the LPSPI functional clock divided by the selected prescaler (see TCR[PRESCALE]). • The minimum delay is one cycle. • The SCK period is equal to (SCKSET + SCKHLD + 2) cycles of the LPSPI functional clock divided by the selected prescaler (see TCR[PRESCALE]). • The SCK duty cycle is based on the difference between SCKSET and SCKHLD. You must configure both these fields to the same value for a 50/50 duty cycle. See Figure 389 for more information. **NOTE** For backward compatibility, writing to CCR[SCKDIV] updates this field with (SCKDIV ÷ 2) rounded up.

70.6.1.14 FIFO Control (FCR)

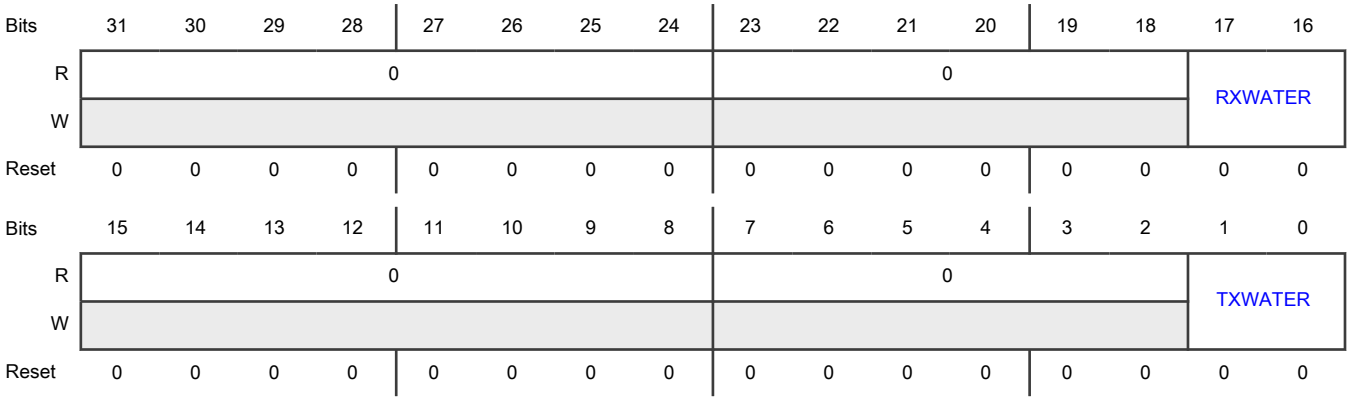
Offset

Register	Offset
FCR	58h

Function

Contains the receive FIFO and transmit FIFO watermark values.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-18 —	Reserved
17-16 RXWATER	Receive FIFO Watermark Causes LPSPI to set SR[RDF] when the number of words in the receive FIFO is greater than RXWATER. Writing a value equal to or greater than the FIFO size truncates the written value.
15-8 —	Reserved
7-2 —	Reserved
1-0 TXWATER	Transmit FIFO Watermark Causes LPSPI to set SR[TDF] when the number of words in the transmit FIFO is equal to or less than TXWATER. Writing a value equal to or greater than the FIFO size truncates the written value.

70.6.1.15 FIFO Status (FSR)

Offset

Register	Offset
FSR	5Ch

Function

Contains fields that indicate the number of words currently stored in the receive and transmit FIFOs.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								0				RXCOUNT			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								0				TXCOUNT			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-19 —	Reserved
18-16 RXCOUNT	Receive FIFO Count Indicates the number of words currently stored in the receive FIFO.
15-8 —	Reserved
7-3 —	Reserved
2-0 TXCOUNT	Transmit FIFO Count Indicates the number of words currently stored in the transmit FIFO.

70.6.1.16 Transmit Command (TCR)

Offset

Register	Offset
TCR	60h

Function

Pushes the data into the transmit FIFO, in the same order as written.

When you write to either this register or to Transmit Data (TDR), each write pushes data into the transmit FIFO. You must write to this register only using 32-bit writes, which are tagged and cause the command register to update; after that the entry reaches the top of the FIFO and LPSPI is enabled. This allows changes to the command word and the transmit data itself to be interleaved. That is, writes to the two registers can be interleaved (write command word, then data word, then command word, and so on). Changing the command word causes all subsequent SPI bus transfers to be performed using the new command word:

- In Controller mode, writing a new command word does not initiate a new transfer, unless TXMSK is 1. Transfers are initiated by transmit data in the transmit FIFO, or by a new command word (with TXMSK = 1). Hardware writes 0 to TXMSK when PCS deasserts.
- In Controller mode, if the command word is changed before an existing frame has completed, then the existing frame terminates and the command word updates. The command word can be changed during a continuous transfer, if CONTC of the new command word is 1 and the command word is written on a frame size boundary.
- In Peripheral mode, the command word must be changed only when LPSPI is idle and there is no SPI bus transfer.

Avoid resetting the transmit FIFO after writing to this register; wait for the command register to update from the FIFO first.

Avoid register reading problems: Reading this register returns the current state of the register. Reading this register at the same time that it is loaded from the transmit FIFO can return an incorrect register value. It is recommended to:

- Read this register when the transmit FIFO is empty.
- Read this register more than once and then compare the returned values.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CPOL	CPHA	PRESCALE			PCS			LSBF	BYSW	CONT	CONTC	RXMSK	TXMSK	WIDTH	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved				FRAMESZ											
W	0															
Reset	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1

Fields

Field	Function
31 CPOL	Clock Polarity Specifies the value of SCK when it is idle. You can update this field only when PCS is deasserted. See Figure 389 for more information. 0b - Inactive low 1b - Inactive high
30 CPHA	Clock Phase Indicates whether data is captured or changed on the leading edge of SCK and captured or changed on the following edge of SCK. You can update this field only when PCS is deasserted. See Figure 389 for more information. 0b - Captured 1b - Changed
29-27 PRESCALE	Prescaler Value Specifies the division of the LPSPI functional clock. For all SPI bus transfers, this value is applied to Clock Configuration (CCR). You can update this field only when PCS is deasserted. 000b - Divide by 1 001b - Divide by 2 010b - Divide by 4 011b - Divide by 8 100b - Divide by 16 101b - Divide by 32 110b - Divide by 64 111b - Divide by 128

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>26-24
PCS</td><td>Peripheral Chip Select
Configures the peripheral chip select used for the transfer. This field is updated only when PCS is deasserted.
NOTE
This entire field is not fully supported in every LPSPI module instance. See the chip-specific LPSPI information.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>LPSPI_0</td><td>TCR</td><td>—</td></tr>
<tr><td>LPSPI_1</td><td>TCR</td><td>—</td></tr>
<tr><td>LPSPI_2</td><td>TCR[25–24]</td><td>TCR[26]</td></tr>
<tr><td>LPSPI_3</td><td>TCR[25–24]</td><td>TCR[26]</td></tr>
<tr><td>LPSPI_4</td><td>TCR[25–24]</td><td>TCR[26]</td></tr>
<tr><td>LPSPI_5</td><td>TCR[25–24]</td><td>TCR[26]</td></tr>
</table>
NOTE
The descriptions of the field settings vary by module instance.
<table>
<tr><th>Instance</th><th>Field value and description</th></tr>
<tr><td>LPSPI_0</td><td>000b - Transfer using PCS[0]
001b - Transfer using PCS[1]
010b - Transfer using PCS[2]
011b - Transfer using PCS[3]
100b - Transfer using PCS[4]
101b - Transfer using PCS[5]
110b - Transfer using PCS[6]
111b - Transfer using PCS[7]</td></tr>
<tr><td>LPSPI_1</td><td>000b - Transfer using PCS[0]</td></tr>
</table>
</td></tr>
</table>

Table continued from the previous page...

Field	Function
	<table><tr><th>Instance</th><th>Field value and description</th></tr><tr><td></td><td>001b - Transfer using PCS[1] 010b - Transfer using PCS[2] 011b - Transfer using PCS[3] 100b - Transfer using PCS[4] 101b - Transfer using PCS[5]</td></tr><tr><td>LPSPI_2 LPSPI_3 LPSPI_4 LPSPI_5</td><td>00b - Transfer using PCS[0] 01b - Transfer using PCS[1] 10b - Transfer using PCS[2] 11b - Transfer using PCS[3]</td></tr></table>
23 LSBF	LSB First Indicates whether data is transferred with MSB first or LSB first. 0b - MSB first 1b - LSB first
22 BYSW	Byte Swap Swaps the contents of [31:24] with [7:0] and [23:16] with [15:8] for each transmit data word read from the FIFO and for each received data word stored to the FIFO (or compared with match registers). 0b - Disable byte swap 1b - Enable byte swap
21 CONT	Continuous Transfer Enables continuous transfer: • In Controller mode, this field keeps PCS asserted at the end of the frame size until a command word is received that starts a new frame. • In Peripheral mode, when this field is enabled, LPSPI only transmits the first FRAMESZ bits, after which LPSPI transmits received data (assuming a 32-bit shift register) until the next PCS negation. 0b - Disable 1b - Enable
20 CONTC	Continuing Command Enables the command word to be changed within a continuous transfer in Controller mode: • The initial command word must enable continuous transfer (CONT = 1). • The continuing command must have CONTC = 1.

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
The continuing command word must be loaded on a frame size boundary.
For example, if the continuous transfer has a frame size of 64 bits, then a continuing command word must be loaded on a 64-bit boundary.
In Peripheral mode, this field modifies the internal RXMSK and TXMSK configuration after the first FRAMESZ bits and until PCS negation:
Receive data is discarded after the first FRAMESZ bits. If CONT is also 1, this does not block the transmission of received data.
Transmit data is not masked after the first FRAMESZ bits. This allows the first FRAMESZ bits to be received and a response transmitted.
0b - Command word for start of new transfer
1b - Command word for continuing transfer
</td></tr>
<tr><td>19
RXMSK</td><td>Receive Data Mask
Masks receive data (receive data is not stored in the receive FIFO).
0b - Normal transfer
1b - Mask receive data</td></tr>
<tr><td>18
TXMSK</td><td>Transmit Data Mask
Masks transmit data (no data is loaded from the transmit FIFO and the output pin is 3-stated). In Controller mode, TXMSK initiates a new transfer that cannot be aborted by another command word. TXMSK automatically transitions to 0 at the end of the transfer.
0b - Normal transfer
1b - Mask transmit data</td></tr>
<tr><td>17-16
WIDTH</td><td>Transfer Width
Configures serial (1-bit) or parallel transfers. For half-duplex parallel transfers, either RXMSK or TXMSK must be 1.
NOTE
The descriptions of the field settings vary by module instance.
<table>
<tr><th>Instance</th><th>Field value and description</th></tr>
<tr><td>LPSPI_0</td><td>00b - 1-bit transfer
01b - 2-bit transfer
10b - 4-bit transfer
11b - 8-bit transfer</td></tr>
<tr><td>LPSPI_1
LPSPI_2</td><td>00b - 1-bit transfer
01b - 2-bit transfer</td></tr>
</table>
</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	<table><tr><th>Instance</th><th>Field value and description</th></tr><tr><td>LPSPI_3 LPSPI_4 LPSPI_5</td><td>10b - 4-bit transfer 11b - Reserved</td></tr></table>
15-12 —	Reserved
11-0 FRAMESZ	Frame Size Configures the frame size in number of bits equal to (FRAMESZ + 1): • The minimum frame size is 8 bits, or 16 bits for an 8-bit transfer. • If the frame size is larger than 32 bits, then the frame is divided into multiple words of 32 bits; each word is loaded from the transmit FIFO and stored in the receive FIFO separately. • If the size of the frame is not divisible by 32, then the last load of the transmit FIFO and store of the receive FIFO contains the remainder bits. For example, a 72-bit transfer consists of three words: the first and second words are 32 bits, and the third word is 8 bits.

70.6.1.17 Transmit Data (TDR)

Offset

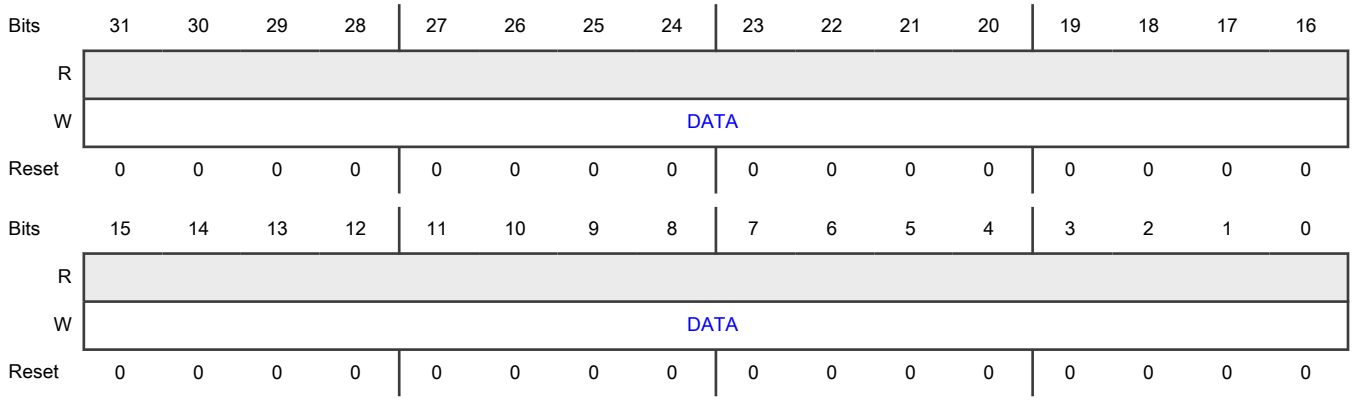
Register	Offset
TDR	64h

Function

Pushes the data into the transmit FIFO, in the same order that the data is written. You can write to this register using 32-, 16-, or 8-bit writes.

When you write to this register or to Transmit Command (TCR), each write pushes data into the FIFO with zero pushed in unwritten bytes.

Diagram



Fields

Field	Function
31-0 DATA	Transmit Data Indicates transmit data. Both 8-bit and 16-bit writes of transmit data zero-extend the data written and push the data into the transmit FIFO. To zero-extend 8-bit and 16-bit writes (to 32 bits) means that the higher order (most significant) empty parts of the 8-bit and 16-bit writes are filled with zeroes.

70.6.1.18 Receive Status (RSR)

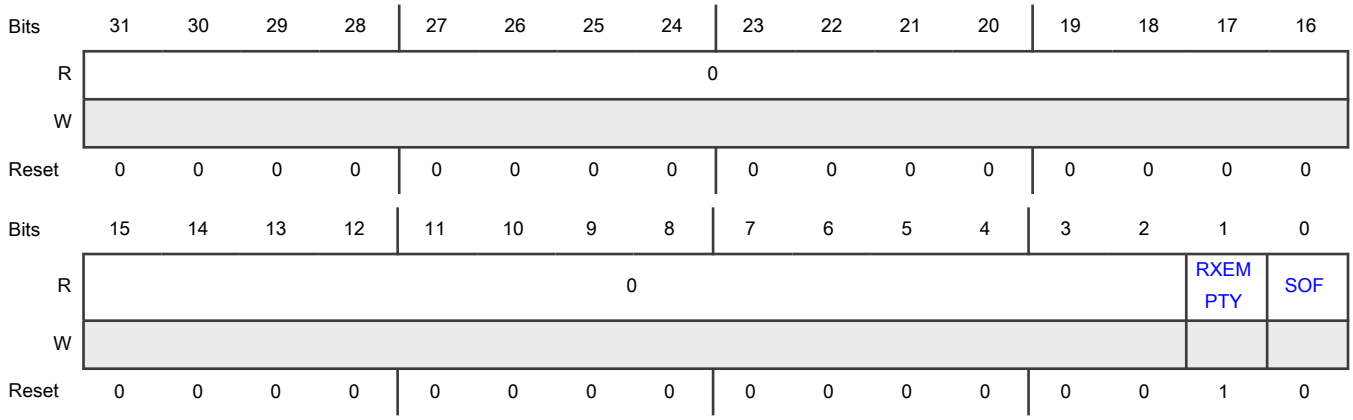
Offset

Register	Offset
RSR	70h

Function

Contains data flow status fields for receive FIFO.

Diagram



Fields

Field	Function
31-2 —	Reserved
1 RXEMPTY	RX FIFO Empty Indicates whether the receive FIFO is empty. 0b - Not empty 1b - Empty
0 SOF	Start of Frame Indicates whether this is the first data word received after PCS assertion. 0b - Subsequent data word or RX FIFO is empty (RXEMPTY=1). 1b - First data word

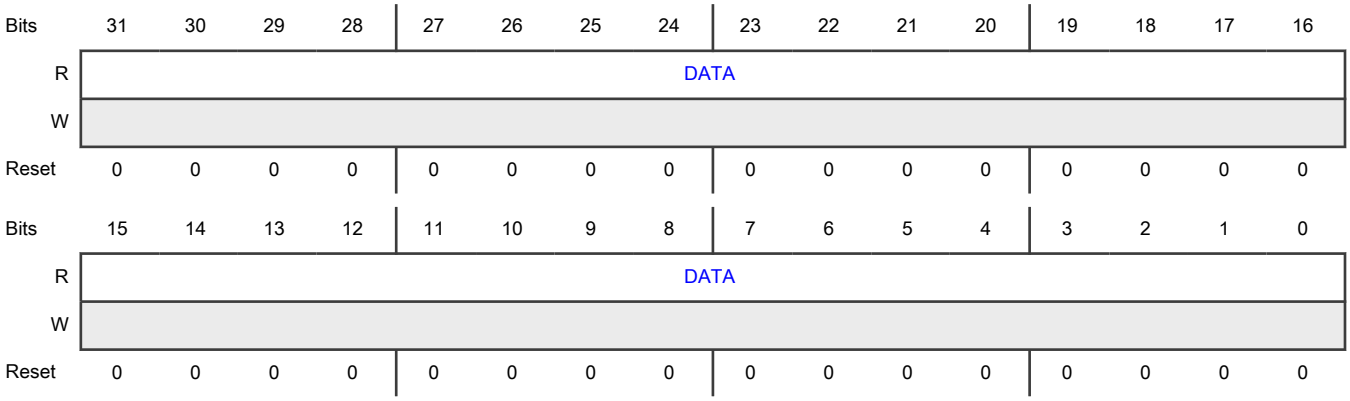
70.6.1.19 Receive Data (RDR)

Offset

Register	Offset
RDR	74h

Function
Pulls the first entry from the receive FIFO.

Diagram



Fields

Field	Function
31-0 DATA	Receive Data

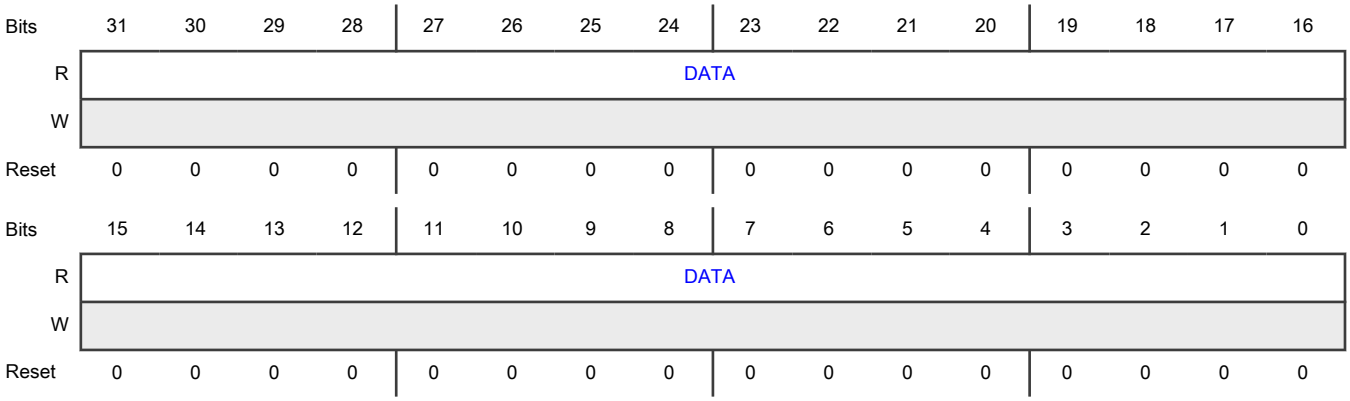
70.6.1.20 Receive Data Read Only (RDROR)

Offset

Register	Offset
RDROR	78h

Function
Returns the first entry in the receive FIFO but does not remove the data from the FIFO.

Diagram



Fields

Field	Function
31-0 DATA	Receive Data

70.6.1.21 Transmit Command Burst (TCBR)

Offset

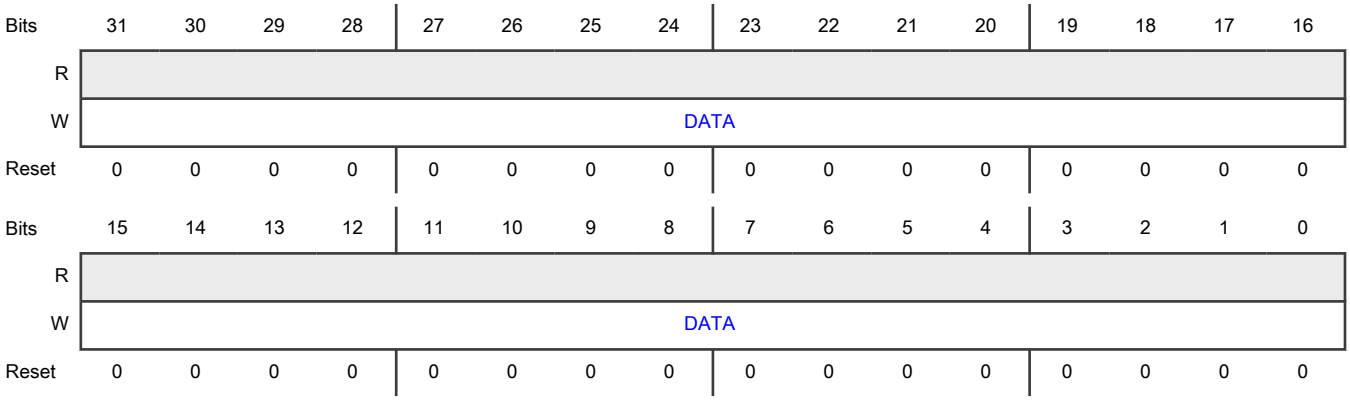
Register	Offset
TCBR	3FCh

Function

Supports burst transfers of command data to the transmit FIFO for use with the DMA controller.

See DMA support registers.

Diagram



Fields

Field	Function
31-0 DATA	Command Data Writes data to Transmit Command (TCR).

70.6.1.22 Transmit Data Burst (TDBR0 - TDBR127)

Offset

For n = 0 to 127:

Register	Offset
TDBRn	400h + (n × 4h)

Function

Supports burst transfers of data to the transmit FIFO for use with the DMA controller. The size of this register is 512 bytes.

See DMA support registers.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	DATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	DATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 DATA	Data Writes data to Transmit Data (TDR).

70.6.1.23 Receive Data Burst (RDBR0 - RDBR127)

Offset

For n = 0 to 127:

Register	Offset
RDBRn	600h + (n × 4h)

Function

Supports burst transfers of data from the receive FIFO. The size of this register is 512 bytes.

See DMA support registers.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DATA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DATA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 DATA	Data Reads data from Receive Data (RDR).

70.7 Glossary

PCS	Peripheral chip select
SCK	Serial clock
SDI	Peripheral data in
SDO	Peripheral data out
SS	Peripheral select

Chapter 71
Low Power Inter-Integrated Circuit (LPI2C)

71.1 Chip-specific LPI2C information

71.1.1 LPI2C instances and configuration

This chip has two instances of LPI2C: LPI2C_0 and LPI2C_1.

Table 453. LPI2C configuration

Instances	TX FIFO Size	RX FIFO Size	SMBus[1]	Slave Mode Enable
LPI2C_0	4x11 bit	4x8 bit	Yes	Yes
LPI2C_1	4x11 bit	4x8 bit	Yes	Yes

1. System Management Bus

- LPI2C target mode operation supports up to high speed mode = 3.4MHz (3.4 Mbps data rate; effective data rate reduces according to I2C protocol).
- LPI2C controller mode operation supports up to fast Mode = 400KHz (400 kbps data rate; effective data rate reduces according to I2C protocol).
- The LPI2C module includes SMBus (System Management Bus) support and DMA support.
- LPI2C RX FIFO is 8-bit and TX FIFO is 11-bit (8bit data + 3bit command)
- LPI2C supports fast data communication as per I2C standard v3.0.

71.2 Overview

LPI2C supports an efficient interface to an I2C bus as a controller and target:

- Implements logic support for Standard, Fast, Fast+, HS-mode (target only) and Ultra-Fast modes of operation
- Uses little CPU overhead, with DMA offloading of FIFO register accesses

LPI2C also complies with the System Management Bus (SMBus) Specification, version 3. The SMBus is a single-ended simple two-wire bus, which is typically used for low-bandwidth communications.

The Inter-Integrated Circuit (I^2C) serial bus is multi-controller, multi-target, packet-switched, and single-ended, and is often used to attach microcontroller ICs to lower-speed peripheral ICs.

NOTE

Terminology in this chapter has been updated to align with I^2C-bus specification, Rev. 7.0, as shown in Table 454.

Table 454. Updated terms

Updated term	Deprecated term
Controller	Master
Target	Slave

71.2.1 Block diagram

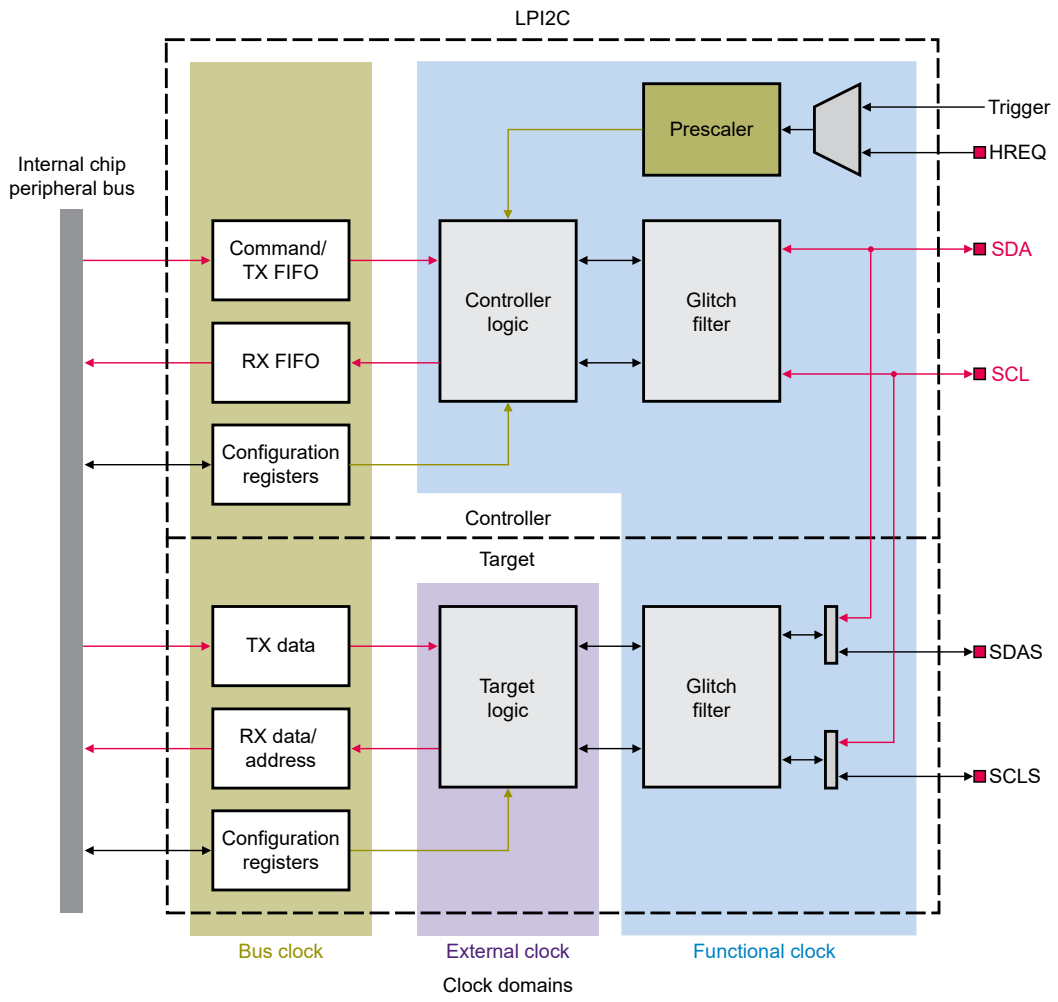


Figure 391. Block diagram

71.2.2 Features

LPI2C supports:

- Standard, Fast, Fast+ and Ultra Fast modes
- HS mode in target mode
- Multicontroller, including synchronization and arbitration, means that any number of controller nodes can be present. Also, controller and target roles can be changed between messages (after a Stop signal is sent).
- Clock stretching. Used on the SCL line, as an I2C flow control mechanism.
- Arbitration for when the system has more than one controller. When used on the SDA line, ensures that there is only one I2C transmitter at a time.
- General call, seven-bit addressing, and ten-bit addressing
- Software reset, Start byte, and device ID (also require software support)

The LPI2C controller supports:

- Command and transmit FIFO of 4 words (8-bit transmit data + 3-bit command)

- Receive FIFO of 4 words (8-bit receive data).
- Command FIFO waiting for an I2C idle bus before initiating a transfer
- Initiation of repeated Start and Stop conditions and one or more controller-receiver transfers by command FIFO
- Stop condition generation from command FIFO, or automatic generation of Stop condition when the transmit FIFO is empty
- Host request input to control the start time of an I2C bus transfer
- Interrupt generation on data match and unwanted data rejection, via flexible receive data match
- Flags and optional interrupt signals at repeated Start condition, Stop condition, loss of arbitration, unexpected NACK, and command word errors
- Configurable bus idle timeout and pin-stuck-low timeout

The LPI2C target supports:

- Separate I2C target registers to minimize software overhead because of controller or target switching
- 7-bit or 10-bit addressing, address range, SMBus alert, and general call address.
- Transmit data register that supports interrupt or DMA requests
- Receive data register that supports interrupt or DMA requests
- Software-controllable ACK or NACK, with optional clock stretching on ACK or NACK field
- Configurable clock stretching, to avoid transmit-FIFO-underrun and receive-FIFO-overrun errors
- Flags and optional interrupt at end of packet, Stop condition, or bit error detection

71.3 Functional description

71.3.1 Controller mode

The LPI2C controller logic operates independently from the target logic to perform all controller-mode transfers on the I2C bus.

71.3.1.1 Transmit and Command FIFO commands

The transmit FIFO stores command data to initiate various I2C operations. The following operations can be initiated through commands in the transmit FIFO:

- Start or repeated Start condition with address byte, expecting ACK or NACK.
- Transmit data. This operation is the default for zero-extended-byte writes to the transmit FIFO.
- Receive 1-256 bytes of data. You can configure this operation to discard received data and not to store it in the receive FIFO.
- Stop condition. You can configure this operation to send a Stop condition when the transmit FIFO is empty.

Multiple transmit and receive commands can be inserted between the Start and Stop conditions. To comply with the I2C specification, transmit and receive commands must not be interleaved. The receive data command and the receive data and discard commands can be interleaved. This interleaving ensures that only the desired received data is stored in the receive FIFO (or compared with the data match logic).

The LPI2C controller automatically transmits a NACK on the last byte of a receive data command. It transmits the NACK unless the next command in the FIFO is also a receive data command. If the transmit FIFO is empty when a receive data command completes, a NACK is also automatically transmitted.

The LPI2C controller supports 10-bit addressing via a (repeated) Start condition, followed by a transmit data byte containing the second address byte, followed by any number of data bytes with the controller transmit data.

A Start or repeated Start condition expecting a NACK (for example, HS mode controller code) must be followed by a Stop or (repeated) Start condition.

71.3.1.2 Controller operations

When LPI2C is enabled, it monitors the I2C bus to detect when the I2C is idle (MSR[BBF]). If either SCL or SDA are low, the I2C bus is no longer considered idle. The bus becomes idle if a Stop condition is detected or if a bus idle timeout is detected (as configured by MCFGR2[BUSIDLE]). After the bus is idle, if the transmit FIFO is not empty and the host request is asserted or disabled, the LPI2C controller initiates a transfer on the bus. This transfer involves the following steps:

1. Wait the bus idle time equal to (MCCR0[CLKLO] + 1) multiplied by the prescaler (MCFGR1[PRESCALE]).
2. Transmit a Start condition and address byte using the timing configuration in Controller Clock Configuration 0 (MCCR0). If an HS mode transfer is configured, the timing configuration from Controller Clock Configuration 1 (MCCR1) is used instead.
3. Perform controller transmit or controller receive transfers, as configured by the transmit FIFO.
4. Transmit NACK on the last byte of a controller receive transfer. This action is performed unless the next command in the transmit FIFO is also a receive data command and the transmit FIFO is not empty.
5. Transmit a repeated Start or Stop condition as configured by the transmit FIFO or MCFGR1[AUTOSTOP]. A repeated Start can change which timing configuration register is used.

When the LPI2C controller is disabled, LPI2C continues emptying the transmit FIFO until a Stop condition is transmitted. (The controller could be disabled due to MCR[MEN] being 0, or automatically due to mode entry.) However, LPI2C no longer stalls the I2C bus by waiting for the transmit or receive FIFO. After the transmit FIFO is empty, LPI2C generates a Stop condition automatically.

The LPI2C controller can stall the I2C bus under certain conditions. This stalling results in SCL pulled low continuously on the first bit of a byte, until these conditions change:

- The LPI2C controller is enabled and busy, the transmit FIFO is empty, and MCFGR1[AUTOSTOP] is 0. The LPI2C controller continues to stall the bus until the transmit FIFO is loaded with more data.
- The LPI2C controller is enabled and receiving data, receive data is not being discarded (due to command or receive data match), and the receive FIFO is full. The LPI2C controller continues to stall the I2C bus until the receive FIFO is emptied.

71.3.1.3 Receive FIFO and data matching

The receive FIFO stores receive data during controller-receiver transfers. You can configure the LPI2C controller to discard received data instead of storing it in the receive FIFO. This option is configured via the command word in the transmit FIFO.

Received data supports a receive data match function that can match received data against one of two bytes, or against a masked data byte. You can configure the data match function to compare only the first one or two data words received since the last (repeated) Start condition. Received data that is already discarded due to the command word cannot cause a data match. It delays the match on the first data word received until after the discarded data is received.

You can configure the receiver match function to discard all received data until a data match is detected, using MCFGR0[RDMO]. Following a data match, write 0 to MCFGR0[RDMO] before writing 0 to MSR[DMF] to allow all subsequent data to be received.

71.3.1.4 Timing parameters

The LPI2C controller can configure the following timing parameters. Parameters are configured separately for HS mode (Controller Clock Configuration 1 (MCCR1)) and other modes (Controller Clock Configuration 0 (MCCR0)). This separation allows the HS mode controller code to be sent using regular timing parameters. Then it allows a switch to HS mode timing (following a repeated Start) until the next STOP condition.

Configure the LPI2C controller timing parameters, measured in LPI2C functional clock cycles, as shown in Table 455. You must configure these parameters to meet the I2C timing specification for the required mode.

Table 455. Timing parameters

I2C specification timing parameter	I2C specification timing symbol	LPI2C timing parameter (in LPI2C functional clock cycles)
SCL clock period	tSCL	(CLKHI + CLKLO + 2 + SCL_LATENCY) × (2 ^ PRESCALE)
Hold time (repeated) Start condition	tHD:STA	(SETHOLD +1) × (2 ^ PRESCALE)
Low period of the SCL clock	tLOW	(CLKLO + 1) × (2 ^ PRESCALE)
High period of the SCL clock	tHIGH	(CLKHI + 1 + SCL_LATENCY) × (2 ^ PRESCALE)
Setup time for a repeated Start condition or Stop condition	tSU:STA, tSU:STO	(SETHOLD + 1 + SCL_LATENCY) × (2 ^ PRESCALE)
Data hold time	tHD:DAT	(DATAVD + 1) × (2 ^ PRESCALE)
Data setup time	tSU:DAT	(SDA_LATENCY + 1) × (2 ^ PRESCALE)
Bus free time between a Stop and Start condition	tBUF	(CLKLO + 1 + SDA_LATENCY) × (2 ^ PRESCALE)
Data valid time, data valid acknowledge time	tVD:DAT, tVD:ACK	(DATAVD + 1) × (2 ^ PRESCALE)

Table 456 defines the latency parameters. These parameters assume that the risetime is less than one LPI2C functional clock cycle. The risetime depends on a number of factors, including the I/O propagation delay, the I2C bus loading, and the external pullup resistor sizing. A larger risetime increases the number of cycles that the signal takes to propagate through the synchronizer (and glitch filter), which increases the latency.

Table 456. Synchronization latency

Timing parameter	Timing definition
SCL_LATENCY	ROUNDDOWN ((2 + FILTSCL + SCL_RISETIME) ÷ (2 ^ PRESCALE))
SDA_LATENCY	ROUNDDOWN ((2 + FILTSDA + SDA_RISETIME) ÷ (2 ^ PRESCALE))

The following timing restrictions must be enforced to avoid unexpected Start or Stop conditions on the I2C bus. These restrictions also avoid unexpected Start or Stop conditions detected by the LPI2C controller. The timing restrictions can be summarized as "SDA cannot change when SCL is high outside a transmitted (repeated) Start or Stop condition."

Table 457. LPI2C timing parameter restrictions

Timing parameter	Minimum	Maximum	Comment
CLKLO	03h	—	CLKLO x (2 ^ PRESCALE) > SCL_LATENCY
CLKHI	01h	—	Configure CLKHI to meet the duty cycle requirements in the I2C specification
SETHOLD	02h	—	SETHOLD × (2 ^ PRESCALE) > SDA_LATENCY
DATAVD	01h	CLKLO – SDA_LATENCY – 1	Configure DATAVD to meet the data hold requirement in the I2C specification

Table continues on the next page...

Table 457. LPI2C timing parameter restrictions (continued)

Timing parameter	Minimum	Maximum	Comment
FILTSCL	00h	[CLKLO × (2 ^ PRESCALE)] – 3	FILTSCL and FILTSDA are the only parameters not multiplied by (2 ^ PRESCALE)
FILTSDA	FILTSCL	[CLKLO × (2 ^ PRESCALE)] – 3	Configuring FILTSDA greater than FILTSCL can delay the SDA input to compensate for board level skew
BUSIDLE	(CLKLO + SETHOLD + 2) × 2	—	Must also be greater than (CLKHI + 1)

See the UM10204, *I2C-bus specification and user manual*.

See Application information for example LPI2C timing configurations.

71.3.1.5 Error conditions

The LPI2C controller monitors errors while it is active. The following conditions generate an error flag and block a new Start condition from being sent, until the flag is cleared by software:

- A Start or Stop condition is detected and is not generated by the LPI2C controller (MSR[ALF] becomes 1).
- Transmitting data on SDA and different values are received (MSR[ALF] becomes 1).
- NACK is detected when transmitting data, and MCFGR1[IGNACK] is 0 (MSR[NDF] becomes 1).
- NACK is detected and is expecting ACK for the address byte, and MCFGR1[IGNACK] is 0 (MSR[NDF] becomes 1).
- ACK is detected and is expecting NACK for the address byte, and MCFGR1[IGNACK] is 0 (MSR[NDF] becomes 1).
- Transmit FIFO is requesting to transmit or receive data without a Start condition (MSR[FEF] becomes 1).
- SCL (or SDA if MCFGR1[TIMECFG] is 1) is low for (MCFGR2[TIMELOW] × 256) prescaler cycles without a pin transition (MSR[PLTF] becomes 1).

You must respond to MSR[PLTF] to terminate the existing command. You can terminate the command cleanly by writing 0 to MCR[MEN], or you can terminate it abruptly by writing 1 to MCR[RST].

You can use MCFGR2[BUSIDLE] to force the I2C bus to be considered idle when SCL and SDA remain high for (BUSIDLE + 1) prescaler cycles. The bus is considered idle when the LPI2C controller is first enabled. When BUSIDLE is configured greater than zero, then SCL or SDA must be high for (BUSIDLE + 1) prescaler cycles before the I2C bus is considered idle.

71.3.1.6 Pin configuration

Configuration	Description
Open-drain support	The LPI2C controller defaults to open-drain configuration of the SDA and SCL pins. Support for true open drain depends on the specific device, and requires the pins where LPI2C pins are muxed to support true open drain.
HS mode support	Support for HS mode depends on the specific device. This mode requires the SCL pin to support the current source pullup required in the I2C specification.
Ultra-Fast mode support	The LPI2C controller supports the output-only push-pull function required for I2C Ultra-Fast mode using the SDA and SCL pins. Support for Ultra-Fast mode also requires MCFGR1[IGNACK] to be 1.

Table continues on the next page...

Table continued from the previous page...

Configuration	Description
Push-pull two-wire support	A push-pull two-wire configuration is available to the LPI2C controller. If LPI2C is the only controller and all I2C pins on the bus are at the same voltage, this configuration may support a partial HS mode. A partial HS mode, not a full HS mode, because this configuration actively drives high rather than enabling a current service pull-up. This configuration sets the SCL pin as push-pull for every clock except the first clock pulse, to allow HS-mode-compatible targets to perform clock stretching. In this mode, the SDA pin is tristated for controller-receive data bits and controller-transmit ACK/NACK bits, and is configured as push-pull at other times. To avoid the risk of contention when SDA is push-pull, configure the pin for open-drain operation, as part of the device-specific configuration.
Push-pull four-wire support	The push-pull four-wire configuration separates the SCL input data and output data into separate pins. It also separates the SDA input data and output data into separate pins. The SCL/SDA pins are used for input data; the SCLS/SDAS pins are used for output data, with configurable polarity. This configuration simplifies external connections when connecting the LPI2C to the I2C bus through external level shifters or discrete components. When using this four-wire configuration, the LPI2C controller logic and LPI2C target logic cannot connect to separate I2C buses.

71.3.2 Target mode

To perform all target mode transfers on the I2C bus, the LPI2C target logic operates independently from the LPI2C controller logic.

71.3.2.1 Address matching

You can configure the LPI2C target:

- To match one of two addresses, using either 7-bit or 10-bit addressing modes for each address.
- To match a range of addresses in either 7-bit or 10-bit addressing modes.
- To match the general call address and generate appropriate flags.
- To match the SMBus alert address and generate appropriate flags.
- To detect the HS mode controller code address, and to disable the digital filters and output valid delay time until the next Stop condition is detected.

After a valid address is matched, the LPI2C target automatically performs target-transmit or target-receive transfers until:

- A NACK is detected (unless SCFGR1[IGNACK] becomes 1).
- A bit error is detected (the LPI2C target is driving SDA, but a different value is sampled).
- A (repeated) Start or Stop condition is detected.

71.3.2.2 Transmit and receive data

Target Transmit Data (STDR) and Target Receive Data (SRDR) are double-buffered and only update during a target-transmit and target-receive transfer, respectively.

You can configure the target address that was received to be read from SRDR (for example, when using DMA to transfer data) or from Target Address Status (SASR).

You can configure STDR to request data only after a target-transmit transfer is detected. You can also configure it to request new data whenever STDR is empty.

Write to STDR only when SSR[TDF] is set.

Read SRDR only when SSR[RDF] is set, or when SSR[AVF] is set and SCFGR1[RXCFG] = 1.

Read SASR only when SSR[AVF] is set.

71.3.2.3 Clock stretching

The LPI2C target supports many configurable options for clock stretching. You can configure these conditions to perform clock stretching:

- SSR[AVF] is set during the ninth clock pulse of the address byte.
- SSR[TDF] is set during the ninth clock pulse of a target-transmit transfer.
- SSR[RDF] is set during the ninth clock pulse of a target-receive transfer.
- SSR[TAF] is set during the eighth clock pulse of an address byte or a target-receive transfer. In HS mode, this option is disabled.
- Clock stretching can be extended for a number of cycles equal to the value of SCFGR2[CLKHOLD] cycles. This stretching allows additional setup time to sample the SDA pin externally. In HS mode, this option is disabled.

When clock stretching is enabled, clock stretching extends for one peripheral bus clock cycle after SDA updates, unless extended by the SCFGR2[CLKHOLD] configuration.

71.3.2.4 Timing parameters

The LPI2C target can configure the following timing parameters:

- SDA data valid time from SCL negation to SDA update
- SCL hold time when clock stretching is enabled to increase setup time when sampling SDA externally
- SCL glitch filter time
- SDA glitch filter time

These parameters are disabled when SCR[FILTEN] is 0, when SCR[FILTDZ] is 1 in Doze mode, and when LPI2C target detects HS mode. When disabled, the LPI2C target is clocked directly from the I2C bus. In this case, the target may not satisfy all timing requirements of the I2C specification (such as SDA minimum hold time in Standard/Fast mode).

The LPI2C target places the following restrictions on the timing parameters:

- You must configure SCFGR2[FILTSDA] to be greater than or equal to SCFGR2[FILTSCL] (unless compensating for board level skew between SDA and SCL).
- You must configure SCFGR2[DATAVD] to be less than the minimum SCL low period.

71.3.2.5 Error conditions

The LPI2C target can flag the following error conditions:

- SSR[BEF] is set when the LPI2C target is driving SDA but it samples a different value than what is expected.
- SSR[FEF] is set due to a transmit data underrun or a receive data overrun. To eliminate the possibility of underrun and overrun, enable clock stretching.
- SSR[FEF] is also set due to an address overrun, but only when SCFGR1[RXCFG] is 1. To eliminate the possibility of overrun, enable clock stretching.

The LPI2C target does not implement a timeout due to SCL or SDA being stuck low. If this detection is required, use the LPI2C controller logic so you can reset the LPI2C target when this condition is detected.

71.3.3 Low-power modes

LPI2C remains functional during low-power modes, if MCR[DOZEN] = 0 and LPI2C uses an external or internal clock source that remains enabled. LPI2C can generate an interrupt or DMA request to cause a wake-up from low-power modes.

You can configure LPI2C to be disabled in low-power modes when MCR[DOZEN] = 1. In this case, LPI2C waits for the current transfer to complete any pending operation.

NOTE

See the chip-specific information for low-power modes available on your chip.

71.3.4 Debug mode

Table 458. Debug mode

Mode	LPI2C operation
Debug	If MCR[DBGEN] = 1, can continue operating in Debug mode.

71.3.5 Peripheral triggers

The connection of the LPI2C peripheral triggers to other peripherals depends upon the specific device being used.

Table 459. LPI2C triggers

Trigger	Description
Controller output trigger	Generates an output trigger that can be connected to other peripherals on the device. The controller output trigger asserts on either a repeated Start or a Stop condition. The trigger remains asserted for one cycle of the LPI2C functional clock divided by MCFGR1[PRESCALE].
Target output trigger	Generates an output trigger that can be connected to other peripherals on the device. The target output trigger asserts on either a repeated Start or a Stop condition that occurs after a target address match. The target output trigger remains asserted until the next target SCL pin negation.
Input trigger	To control the start of a LPI2C bus transfer, the LPI2C input trigger can be selected instead of the HREQ input. The input trigger is synchronized. To be detected, the input trigger must assert for at least two cycles of the LPI2C functional clock divided by the value of MCFGR1[PRESCALE]. When LPI2C is busy, the HREQ input (and therefore the input trigger) is ignored.

71.3.6 Clocking

Table 460. LPI2C clocks

Clock	Description
LPI2C functional clock	The LPI2C functional clock is asynchronous to the bus clock. It can remain enabled in low-power modes to support I2C bus transfers by the LPI2C controller. The functional clock is also used by the LPI2C target to support digital filter and data hold time configurations. The LPI2C controller divides the functional clock by a prescaler (MCFGR1[PRESCALE]) and the resulting frequency must be at least eight times faster than the I2C bus bandwidth.
External clock	The LPI2C target logic is clocked directly from the external pins. These pins are SCL and SDA, or SCLS and SDAS if the controller and target are implemented on separate pins). This clocking allows the LPI2C target to remain operational, even when the LPI2C functional clock is disabled. **NOTE** If the LPI2C functional clock is disabled, the LPI2C target digital filter must be disabled. This condition can affect compliance with some timing parameters of the I2C specification, such as data hold time.
Bus clock	The bus clock is only used for bus accesses to the control and configuration registers. The bus clock frequency must be sufficient to support the data bandwidth requirements of the LPI2C controller and target registers.

For chip-specific clocking information, see the Clocking chapter.

71.3.7 Reset

Table 461. LPI2C resets

Reset	Description
Chip reset	The logic and registers for the LPI2C controller and target are reset to their default states after a chip reset.
Software reset	The LPI2C controller implements a software reset field in its control register. MCR[RST] resets all controller logic and registers to their default states, except for Controller Control (MCR) itself. The LPI2C target implements a software reset field in its control register. SCR[RST] resets all target logic and registers to their default states, except for Target Control (SCR) itself.
FIFO reset	The LPI2C controller implements write-only control fields that reset the transmit FIFO (MCR[RTF]) and receive FIFO (MCR[RRF]). After a FIFO is reset, that FIFO is empty. The LPI2C target implements write-only control fields that reset the transmit data register (SCR[RTF]) and receive data register (SCR[RRF]). After a data register is reset, that data register is empty.

71.3.8 Interrupts and DMA requests

Depending on the configuration, interrupts and DMA requests can be combined:

- LPI2C controller and target interrupts
- LPI2C controller and target transmit DMA requests
- LPI2C controller and target receive DMA requests

71.3.8.1 Controller mode

Table 462 lists the Controller mode sources that can generate LPI2C controller interrupts and LPI2C controller transmit and receive DMA requests.

Table 462. Controller interrupts and DMA requests

Status flag	Description	Can generate		
		Interrupt?	DMA request?	Low-power wake-up?
Transmit Data Flag (MSR[TDF])	Data can be written to transmit FIFO, as configured by MFCR[TXWATER].	Y	TX	Y
Receive Data Flag (MSR[RDF])	Data can be read from the receive FIFO, as configured by MFCR[RXWATER].	Y	RX	Y
End Packet Flag (MSR[EPF])	Controller has transmitted a repeated Start or Stop condition.	Y	N	Y
Stop Detect Flag (MSR[SDF])	Controller has transmitted a Stop condition .	Y	N	Y
NACK Detect Flag (MSR[NDF])	During an address byte, the controller expects an ACK but detects a NACK. During an address byte, the controller expects a NACK but detects an ACK.	Y	N	Y

Table continues on the next page...

Table 462. Controller interrupts and DMA requests (continued)

Status flag	Description	Can generate		
		Interrupt?	DMA request?	Low-power wake-up?
	During a controller-transmitter data byte, the controller detects a NACK.			
Arbitration Lost Flag (MSR[ALF])	The controller lost arbitration due to a Start or Stop condition detected at the wrong time, or the controller was transmitting data but received data different from the data that was transmitted.	Y	N	Y
FIFO Error Flag (MSR[FEF])	The controller expects a Start condition in the command FIFO, but the next entry in the command FIFO is not a Start condition.	Y	N	Y
Pin Low Timeout Flag (MSR[PLTF])	Pin low timeout is enabled and SCL (or SDA, if configured) is low for longer than the configured timeout.	Y	N	Y
Data Match Flag (MSR[DMF])	The received data matches the configured data match, but the received data is not discarded due to a command FIFO entry.	Y	N	Y
Controller Busy Flag (MSR[MBF])	LPI2C controller is busy transmitting or receiving data.	N	N	N
Bus Busy Flag (MSR[BBF])	LPI2C controller is enabled and activity is detected on the I2C bus, but no Stop condition is detected and no bus idle timeout (if enabled) occurred.	N	N	N

71.3.8.2 Target mode

Table 463 lists the target mode sources that can generate LPI2C target interrupts and LPI2C target transmit and receive DMA requests.

Table 463. Target interrupts and DMA requests

Status flag	Description	Can generate		
		Interrupt?	DMA request?	Low-power wake-up?
Transmit Data Flag (SSR[TDF])	Data can be written to Target Transmit Data (STDR).	Y	TX	Y
Receive Data Flag (SSR[RDF])	Data can be read from Target Receive Data (SRDR).	Y	RX	Y
Address Valid Flag (SSR[AVF])	Address can be read from Target Address Status (SASR).	Y	RX	Y
Transmit ACK Flag (SSR[TAF])	ACK or NACK can be written to Target Transmit ACK (STAR).	Y	N	Y

Table continues on the next page...

Table 463. Target interrupts and DMA requests (continued)

Status flag	Description	Can generate		
		Interrupt?	DMA request?	Low-power wake-up?
Repeated Start Flag (SSR[RSF])	Target has detected an address match followed by a repeated Start condition.	Y	N	Y
Stop Detect Flag (SSR[SDF])	Target has detected an address match followed by a Stop condition.	Y	N	Y
Bit Error Flag (SSR[BEF])	Target was transmitting data, but received data is different from what was transmitted.	Y	N	Y
FIFO Error Flag (SSR[FEF])	This flag is set by: • Transmit data underrun • Receive data overrun • Address status overrun when SCFGR1[RXCFG] = 1 This flag can only be set when clock stretching is disabled.	Y	N	Y
Address Match 0 Flag (SSR[AM0F])	Target detected an address match SAMR[ADDR0].	Y	N	N
Address Match 1 Flag (SSR[AM1F])	Target detected an address match with SAMR[ADDR1] or using an address range.	Y	N	N
General Call Flag (SSR[GCF])	Target detected an address match with the general call address.	Y	N	N
SMBus Alert Response Flag (SSR[SARF])	Target detected an address match with the SMBus alert address.	Y	N	N
Target Busy Flag (SSR[SBF])	LPI2C target is busy receiving an address byte or is transmitting or receiving data.	N	N	N
Bus Busy Flag (SSR[BBF])	LPI2C target is enabled and a Start condition is detected on I2C bus, but no Stop condition detected.	N	N	N

71.4 External signals

Table 464. External signals

Signal	Name	Two-wire scheme	Four-wire scheme	Direction
SCL	LPI2C clock line	SCL	In Four-Wire mode, this pin is the SCL input pin.	Input or output
SDA	LPI2C data line	SDA	In Four-Wire mode, this pin is the SDA input pin.	Input or output

Table continues on the next page...

Table 464. External signals (continued)

Signal	Name	Two-wire scheme	Four-wire scheme	Direction
SCLS	Secondary I2C clock line	Not used	In Four-Wire mode, this pin is the SCLS output pin. If LPI2C controller/target are configured to use separate pins, then this pin is the LPI2C target SCL pin.	Input or output
SDAS	Secondary I2C data line	Not used	In Four-Wire mode, this pin is the SDAS output pin. If LPI2C controller/target are configured to use separate pins, then this pin is the LPI2C target SDA pin.	Input or output
HREQ	Host request	If host request is asserted and the I2C bus is idle, then it initiates an LPI2C controller transfer. HREQ is an additional pin separate from the two-wire or four-wire scheme.		Input

Figure 392 shows the two-signal connection.

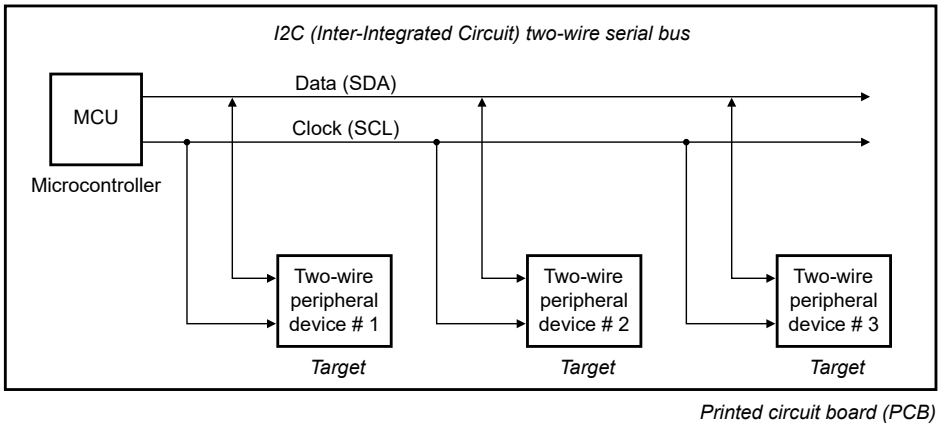


Figure 392. I²C two-wire serial bus

Figure 393 shows a possible four-signal connection.

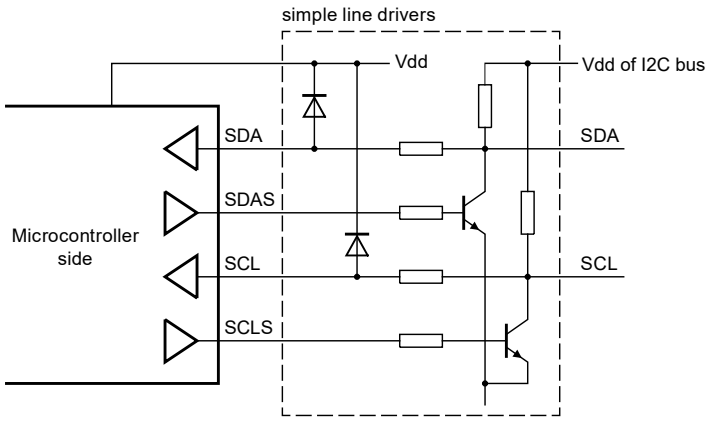


Figure 393. I²C four-wire serial bus

71.5 Initialization

To initialize the LPI2C controller:

1. Configure Controller Configuration 0 (MCFGR0)–Controller Configuration 3 (MCFGR3) as required by the application.
2. Configure Controller Clock Configuration 0 (MCCR0) and Controller Clock Configuration 1 (MCCR1) to satisfy the timing requirements of the I2C mode supported by the application.
3. Enable controller interrupts and DMA requests as required by the application.
4. Enable the LPI2C controller by writing 1 to MCR[MEN].

To initialize the LPI2C target:

1. Configure Target Address Match (SAMR) with the I2C address of the target location on the I2C bus.
2. Configure Target Configuration 1 (SCFGR1) as required by the application.
3. Configure Target Configuration 2 (SCFGR2) to satisfy the timing requirements of the I2C mode supported by the application.
4. Enable target interrupts and DMA requests as required by the application.
5. Enable the LPI2C target by writing 1 to SCR[SEN].

71.6 Application information

Configure the I2C timing parameters to meet the requirements of the I2C specification. This configuration depends on the supported mode and LPI2C functional clock frequency. When switching between two modes using different clock configuration registers (for example, Fast mode and HS mode), MCFGR1[PRESCALE] must remain constant between the modes.

Table 465. Example timing configurations

I2C mode	Clock frequency	Baud rate	PRESCALE	FILTSCL / FILTSDA	SETHOLD	CLKLO	CLKHI	DATAVD
Standard	8 MHz	100 kbit/s	0h	0h/0h	24h	28h	24h	02h
Standard	48 MHz	100 kbit/s	2h	1h/1h	37h	3Fh	37h	03h
Standard	60 MHz	100 kbit/s	2h	1h/1h	45h	50h	44h	04h
Fast	8 MHz	400 kbit/s	0h	0h/0h	04h	0Bh	05h	02h
Fast+	8 MHz	1 Mbit/s	0h	0h/0h	02h	03h	01h	01h
Fast	48 MHz	400 kbit/s	0h	1h/1h	1Dh	3Eh	35h	0Fh
Fast	48 MHz	400 kbit/s	2h	1h/1h	07h	11h	0Bh	03h
Fast+	48 MHz	1 Mbit/s	2h	1h/1h	03h	06h	04h	04h
HS	48 MHz	3.2 Mbit/s	0h	0h/0h	07h	08h	03h	01h
Fast	60 MHz	400 kbit/s	1h	2h/2h	11h	28h	1Fh	08h
Fast+	60 MHz	1 Mbit/s	1h	2h/2h	07h	0Fh	0Bh	01h
HS	60 MHz	3.33 Mbit/s	1h	0h/0h	04h	04h	02h	01h

71.7 Memory map and registers

71.7.1 LPI2C register descriptions

Writing to a read-only register or reading from a write-only register can cause bus errors. This module does not check whether programmed values in the registers are correct; you must ensure that valid programmed values are written to the registers.

71.7.1.1 LPI2C memory map

LPI2C_0 base address: 4035_0000h

LPI2C_1 base address: 4035_4000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID (VERID)	32	R	0102_0003h
4h	Parameter (PARAM)	32	R	0000_0202h
10h	Controller Control (MCR)	32	RW	0000_0000h
14h	Controller Status (MSR)	32	RW	0000_0001h
18h	Controller Interrupt Enable (MIER)	32	RW	0000_0000h
1Ch	Controller DMA Enable (MDER)	32	RW	0000_0000h
20h	Controller Configuration 0 (MCFGR0)	32	RW	0000_0000h
24h	Controller Configuration 1 (MCFGR1)	32	RW	0000_0000h
28h	Controller Configuration 2 (MCFGR2)	32	RW	0000_0000h
2Ch	Controller Configuration 3 (MCFGR3)	32	RW	0000_0000h
40h	Controller Data Match (MDMR)	32	RW	0000_0000h
48h	Controller Clock Configuration 0 (MCCR0)	32	RW	0000_0000h
50h	Controller Clock Configuration 1 (MCCR1)	32	RW	0000_0000h
58h	Controller FIFO Control (MFCR)	32	RW	0000_0000h
5Ch	Controller FIFO Status (MFSR)	32	R	0000_0000h
60h	Controller Transmit Data (MTDR)	32	W	0000_0000h
70h	Controller Receive Data (MRDR)	32	R	0000_4000h
110h	Target Control (SCR)	32	RW	0000_0000h
114h	Target Status (SSR)	32	RW	0000_0000h
118h	Target Interrupt Enable (SIER)	32	RW	0000_0000h
11Ch	Target DMA Enable (SDER)	32	RW	0000_0000h
124h	Target Configuration 1 (SCFGR1)	32	RW	0000_0000h
128h	Target Configuration 2 (SCFGR2)	32	RW	0000_0000h
140h	Target Address Match (SAMR)	32	RW	0000_0000h
150h	Target Address Status (SASR)	32	R	0000_4000h
154h	Target Transmit ACK (STAR)	32	RW	0000_0000h
160h	Target Transmit Data (STDR)	32	W	0000_0000h
170h	Target Receive Data (SRDR)	32	R	0000_4000h

71.7.1.2 Version ID (VERID)

Offset

Register	Offset
VERID	0h

Function

Contains version numbers for the module design and feature set.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MAJOR								MINOR							
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FEATURE															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Fields

Field	Function
31-24 MAJOR	Major Version Number Returns the major version number for the module design specification.
23-16 MINOR	Minor Version Number Returns the minor version number for the module design specification.
15-0 FEATURE	Feature Specification Number Returns the feature set number. 0000_0000_0000_0010b - Controller only, with standard feature set 0000_0000_0000_0011b - Controller and target, with standard feature set

71.7.1.3 Parameter (PARAM)

Offset

Register	Offset
PARAM	4h

Function

Contains parameter values implemented in the module.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				MRXFIFO				0				MTXFIFO			
W																
Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0

Fields

Field	Function
31-16 —	Reserved
15-12 —	Reserved
11-8 MRXFIFO	Controller Receive FIFO Size Configures the number of words in the controller receive FIFO to $2^{MRXFIFO}$.
7-4 —	Reserved
3-0 MTXFIFO	Controller Transmit FIFO Size Configures the number of words in the controller transmit FIFO to $2^{MTXFIFO}$.

71.7.1.4 Controller Control (MCR)

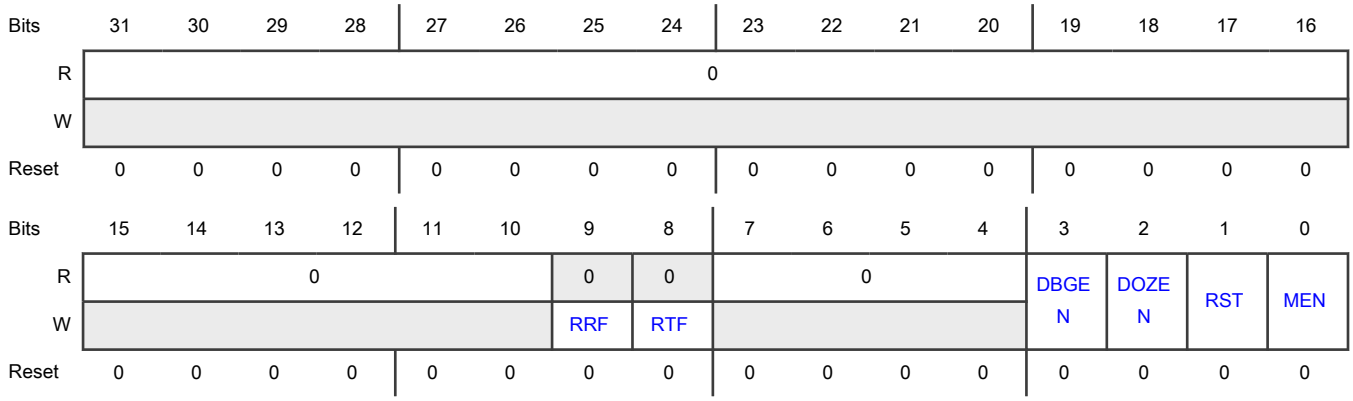
Offset

Register	Offset
MCR	10h

Function

Contains resets, debug enable, and other controller control settings.

Diagram



Fields

Field	Function
31-10 —	Reserved
9 RRF	Reset Receive FIFO Resets the receive FIFO in Controller Receive Data (MRDR). 0b - No effect 1b - Reset receive FIFO
8 RTF	Reset Transmit FIFO Resets the transmit FIFO in Controller Transmit Data (MTDR). 0b - No effect 1b - Reset transmit FIFO
7-4 —	Reserved
3 DBGEN	Debug Enable Enables the controller in Debug mode. 0b - Disable 1b - Enable
2 DOZEN	Doze Mode Enable Enables the controller in Doze mode. 0b - Enable 1b - Disable
1	Software Reset

Table continues on the next page...

Table continued from the previous page...

Field	Function
RST	Resets all internal controller logic and registers except Controller Control (MCR). This field remains 1 (enabled) until you write 0 to it. The reset takes effect immediately and remains asserted until negated by software. There is no minimum delay required before clearing the software reset. 0b - No effect 1b - Reset
0 MEN	Controller Enable Enables the controller logic. 0b - Disable 1b - Enable

71.7.1.5 Controller Status (MSR)

Offset

Register	Offset
MSR	14h

Function

Contains status flags for transmit and receive data, for start and stop conditions, and for bus and controller busy or idle status.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0						BBF	MBF	0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	DMF	PLTF	FEF	ALF	NDF	SDF	EPF	0						RDF	TDF
W		W1C	W1C	W1C	W1C	W1C	W1C	W1C								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Fields

Field	Function
31-26 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
25 BBF	Bus Busy Flag Specifies whether the I2C bus is busy. 0b - Idle 1b - Busy
24 MBF	Controller Busy Flag Specifies whether the I2C controller is busy. 0b - Idle 1b - Busy
23-16 —	Reserved
15 —	Reserved
14 DMF	Data Match Flag Indicates whether the received data matches MDMR[MATCH0] or MDMR[MATCH1] (as configured by MCFGR1[MATCFG]). Received data discarded due to MTDR[CMD] does not cause this flag to set. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Matching data not received 1b - Matching data received When writing 0b - No effect 1b - Clear the flag
13 PLTF	Pin Low Timeout Flag Indicates whether pin low timeout has occurred. Sets when the SCL or SDA input is low for more than the number of PINLOW cycles defined by MCFGR3[PINLOW], even when the LPI2C controller is idle. You must resolve the pin low condition via software. PLTF cannot be cleared as long as the pin low timeout continues. Before LPI2C can initiate a Start condition, you must clear this flag. See MCFGR1[TIMECFG] for the SCL and/or SDA timeout settings. **NOTE** This field behaves differently for register reads and writes. When reading

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Pin low timeout did not occur 1b - Pin low timeout occurred When writing 0b - No effect 1b - Clear the flag
12 FEF	FIFO Error Flag Detects the LPI2C controller's attempt to send or receive data without first generating a (repeated) Start condition. This error can occur when the transmit FIFO underflows when MCFGR1[AUTOSTOP] = 1. When this flag is set, the LPI2C controller sends a Stop condition (if busy). The controller does not initiate a new Start condition until the flag is cleared. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No FIFO error 1b - FIFO error When writing 0b - No effect 1b - Clear the flag
11 ALF	Arbitration Lost Flag Indicates whether arbitration is lost. Either of these conditions sets this flag: • The LPI2C controller transmits a logic 1 and detects a logic 0 on the I2C bus. • The LPI2C controller detects a Start or Stop condition when the LPI2C controller is transmitting data. When ALF is set, the LPI2C controller releases the I2C bus (goes idle), and the LPI2C controller does not initiate a new Start condition until the ALF is cleared. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Controller did not lose arbitration 1b - Controller lost arbitration When writing 0b - No effect 1b - Clear the flag
10	NACK Detect Flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
NDF	Indicates whether an unexpected NACK has been detected. This flag is set when the LPI2C controller detects a NACK it was not expecting when transmitting an address or data. When set, the controller does not initiate a new Start condition until this flag is cleared. If a NACK is expected for a given address (as configured by the command word), this flag is set if a NACK is not generated. When this flag is set, the LPI2C controller automatically transmits a Stop condition if MCFGR1[AUTOSTOP] = 1, or if the transmit FIFO is not empty. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No unexpected NACK detected 1b - Unexpected NACK detected When writing 0b - No effect 1b - Clear the flag
9 SDF	Stop Detect Flag Indicates whether the LPI2C controller has generated a Stop condition. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No Stop condition generated 1b - Stop condition generated When writing 0b - No effect 1b - Clear the flag
8 EPF	End Packet Flag Indicates whether the LPI2C controller has generated a repeated Start condition or a Stop condition. When the controller first generates a Start condition, this flag is not set. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No Stop or repeated Start generated 1b - Stop or repeated Start generated When writing

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No effect 1b - Clear the flag
7-2 —	Reserved
1 RDF	Receive Data Flag Indicates whether the receive data is ready. This flag is set when the number of words in the receive FIFO is greater than MFCR[RXWATER]. 0b - Receive data not ready 1b - Receive data ready
0 TDF	Transmit Data Flag Indicates whether transmit data is requested. This flag is set when the number of words in the transmit FIFO is equal or less than MFCR[TXWATER]. 0b - Transmit data not requested 1b - Transmit data requested

71.7.1.6 Controller Interrupt Enable (MIER)

Offset

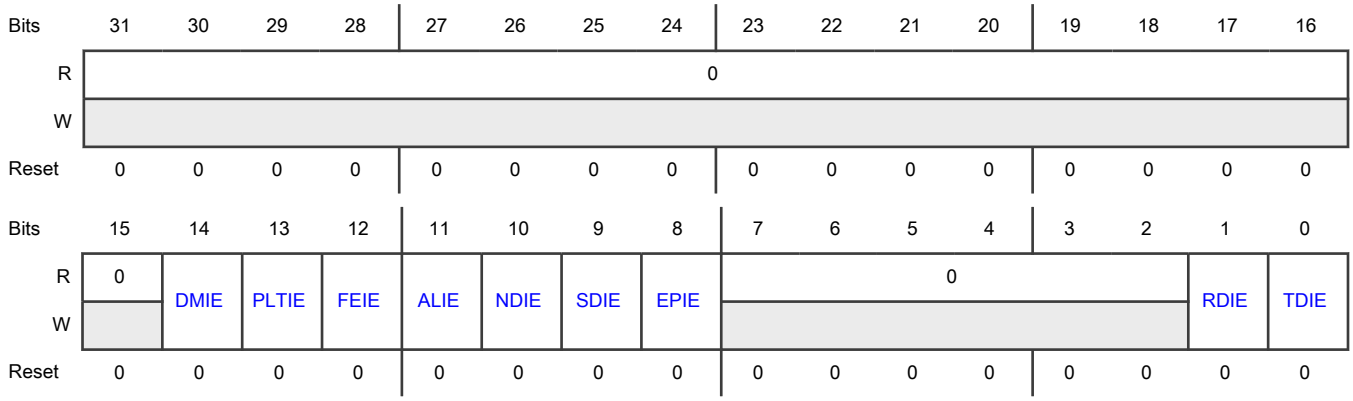
Register	Offset
MIER	18h

Function

Contains enables for:

- Transmit and receive data interrupts
- Start, Stop, and NACK detection interrupts
- DMA interrupts

Diagram



Fields

Field	Function
31-16 —	Reserved
15 —	Reserved
14 DMIE	Data Match Interrupt Enable Enables interrupt for data match. 0b - Disable 1b - Enable
13 PLTIE	Pin Low Timeout Interrupt Enable Enables interrupt for pin-low timeout. 0b - Disable 1b - Enable
12 FEIE	FIFO Error Interrupt Enable Enables interrupt for FIFO error. 0b - Disable 1b - Enable
11 ALIE	Arbitration Lost Interrupt Enable Enables interrupt for arbitration lost. 0b - Disable 1b - Enable
10	NACK Detect Interrupt Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
NDIE	Enables interrupt for NACK detection. 0b - Disable 1b - Enable
9 SDIE	Stop Detect Interrupt Enable Enables interrupt for Stop detection. 0b - Disable 1b - Enable
8 EPIE	End Packet Interrupt Enable Enables interrupt for end packet. 0b - Disable 1b - Enable
7-2 —	Reserved
1 RDIE	Receive Data Interrupt Enable Enables interrupt for receive data. 0b - Disable 1b - Enable
0 TDIE	Transmit Data Interrupt Enable Enables interrupt for transmit data. 0b - Disable 1b - Enable

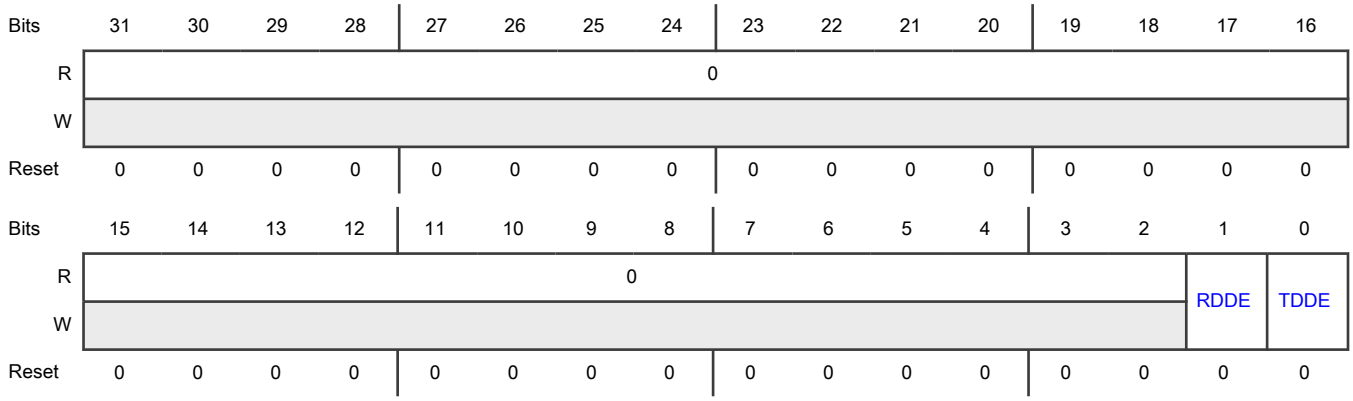
71.7.1.7 Controller DMA Enable (MDER)

Offset

Register	Offset
MDER	1Ch

Function
Contains DMA transmit, request, and receive enables.

Diagram



Fields

Field	Function
31-2 —	Reserved
1 RDDE	Receive Data DMA Enable Enables DMA receive data. 0b - Disable 1b - Enable
0 TDDE	Transmit Data DMA Enable Enables DMA transmit data. 0b - Disable 1b - Enable

71.7.1.8 Controller Configuration 0 (MCFGR0)

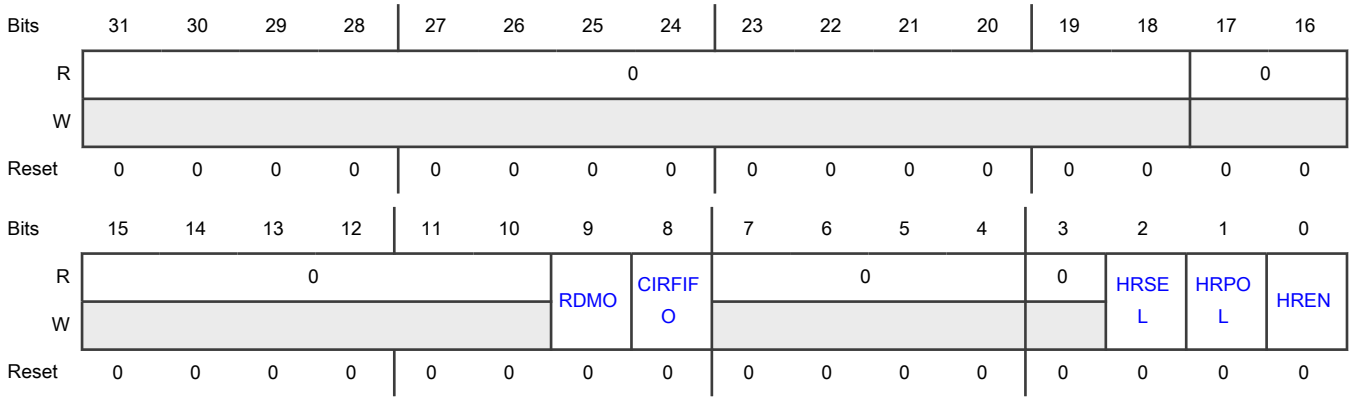
Offset

Register	Offset
MCFGR0	20h

Function

Contains host settings and other receive and transfer settings.

Diagram



Fields

Field	Function
31-18 —	Reserved
17-16 —	Reserved
15-10 —	Reserved
9 RDMO	Receive Data Match Only Determines whether all received data that does not set MSR[DMF] is discarded. After MSR[DMF] is set, the RDMO configuration is ignored. When disabling RDMO, write 0 to this field before writing 0 to MSR[DMF] to ensure that no receive data is lost. 0b - Received data is stored in the receive FIFO 1b - Received data is discarded unless MSR[DMF] is set
8 CIRFIFO	Circular FIFO Enable Enables the transmit FIFO read pointer to be saved to a temporary register. The transmit FIFO empties as normal. After the LPI2C controller is idle and the transmit FIFO is empty, the read pointer value is restored from the temporary register. This setting causes the contents of the transmit FIFO to be cycled through repeatedly. If MCFGR1[AUTOSTOP] is 1, then a Stop condition is sent whenever the transmit FIFO is empty and the read pointer is restored. 0b - Disable 1b - Enable
7-4 —	Reserved
3	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
2 HRSEL	Host Request Select Selects the source of the host request input. When host request is enabled, this field must not change. 0b - Host request input is pin HREQ 1b - Host request input is input trigger
1 HRPOL	Host Request Polarity Configures the polarity of the host request input. When host request is enabled, this field must not change. HRPOL sets the polarity for both the HREQ pin and the input trigger. • When HRPOL=0, the polarity is configured for active low, so host request is asserted if the HREQ pin or input trigger are logic 0. • When HRPOL=1, the polarity is configured for active high, so host request is asserted if the HREQ pin or input trigger are logic 1. 0b - Active low 1b - Active high
0 HREN	Host Request Enable Enables host request. When enabled, the LPI2C controller only initiates a Start condition if the host request input is asserted and the bus is idle. A repeated Start condition is not affected by the host request. 0b - Disable 1b - Enable

71.7.1.9 Controller Configuration 1 (MCFGR1)

Offset

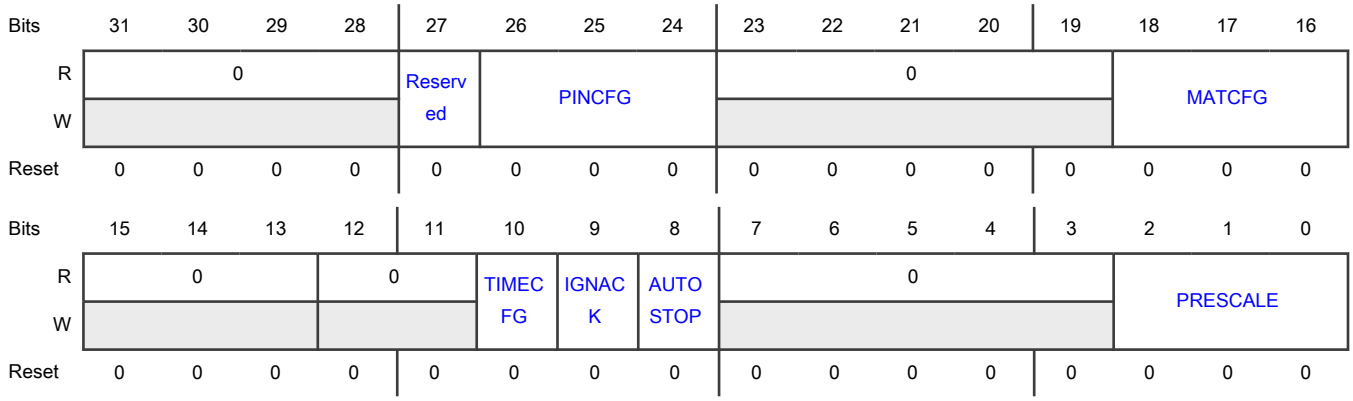
Register	Offset
MCFGR1	24h

Function

Contains controls for pin configuration, clock prescaler, and various other control settings.

Write to this register only when the I2C controller is disabled.

Diagram



Fields

Field	Function
31-28 —	Reserved
27 —	Reserved
26-24 PINCFG	Pin Configuration Configures the pin mode for LPI2C. 000b - Two-pin open drain mode. SCL/SDA pins: Bidirectional open drain for controller and target. SCLS/SDAS pins: Not used. 001b - Two-pin output only mode (Ultra-Fast mode). SCL/SDA pins: Output-only (Ultra-Fast mode) open drain for controller and target. SCLS/SDAS pins: Not used. 010b - Two-pin push-pull mode. SCL/SDA pins: Bidirectional push-pull for controller and target. SCLS/SDAS pins: Not used. 011b - Four-pin push-pull mode. SCL/SDA pins: Input only for controller and target. SCLS/SDAS pins: Output-only push-pull for controller and target. 100b - Two-pin open-drain mode with separate LPI2C target. SCL/SDA pins: Bidirectional open drain for controller. SCLS/SDAS pins: Bidirectional open drain for target. 101b - Two-pin output only mode (Ultra-Fast mode) with separate LPI2C target. SCL/SDA pins: Output-only (Ultra-Fast mode) open drain for controller. SCLS/SDAS pins: Output-only open drain for target. 110b - Two-pin push-pull mode with separate LPI2C target. SCL/SDA pins: Bidirectional push-pull for controller. SCLS/SDAS pins: Bidirectional push-pull for target. 111b - Four-pin push-pull mode (inverted outputs). SCL/SDA pins: Input only for controller and target. SCLS/SDAS pins: Inverted output-only push-pull for controller and target.
23-19 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
18-16 MATCFG	Match Configuration Configures the condition that sets MSR[DMF]. See Controller Data Match (MDMR). 000b - Match is disabled 001b - Reserved 010b - Match is enabled: first data word equals MDMR[MATCH0] OR MDMR[MATCH1] 011b - Match is enabled: any data word equals MDMR[MATCH0] OR MDMR[MATCH1] 100b - Match is enabled: (first data word equals MDMR[MATCH0]) AND (second data word equals MDMR[MATCH1) 101b - Match is enabled: (any data word equals MDMR[MATCH0]) AND (next data word equals MDMR[MATCH1) 110b - Match is enabled: (first data word AND MDMR[MATCH1]) equals (MDMR[MATCH0] AND MDMR[MATCH1]) 111b - Match is enabled: (any data word AND MDMR[MATCH1]) equals (MDMR[MATCH0] AND MDMR[MATCH1])
15-13 —	Reserved
12-11 —	Reserved
10 TIMECFG	Timeout Configuration Configures which signals must be low for longer than the configured timeout to set MSR[PLTF]. When this field is 0, MSR[PLTF] is set when SCL is low for longer than the configured timeout. 0b - SCL 1b - SCL or SDA
9 IGNACK	Ignore NACK Determines whether the LPI2C controller ignores a received NACK and treats it as an ACK. This field must be 1 in Ultra-Fast mode. 0b - No effect 1b - Treat a received NACK as an ACK
8 AUTOSTOP	Automatic Stop Generation Determines whether a Stop condition is generated when the LPI2C controller is busy and the transmit FIFO is empty. A Stop condition can also be generated using a transmit FIFO command. When this field is 1, a Stop condition is automatically generated when the transmit FIFO is empty and the LPI2C controller is busy. 0b - No effect

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Stop automatically generated
7-3 —	Reserved
2-0 PRESCALE	Prescaler Configures the clock prescaler used for all LPI2C controller logic except the digital glitch filters. 000b - Divide by 1 001b - Divide by 2 010b - Divide by 4 011b - Divide by 8 100b - Divide by 16 101b - Divide by 32 110b - Divide by 64 111b - Divide by 128

71.7.1.10 Controller Configuration 2 (MCFGR2)

Offset

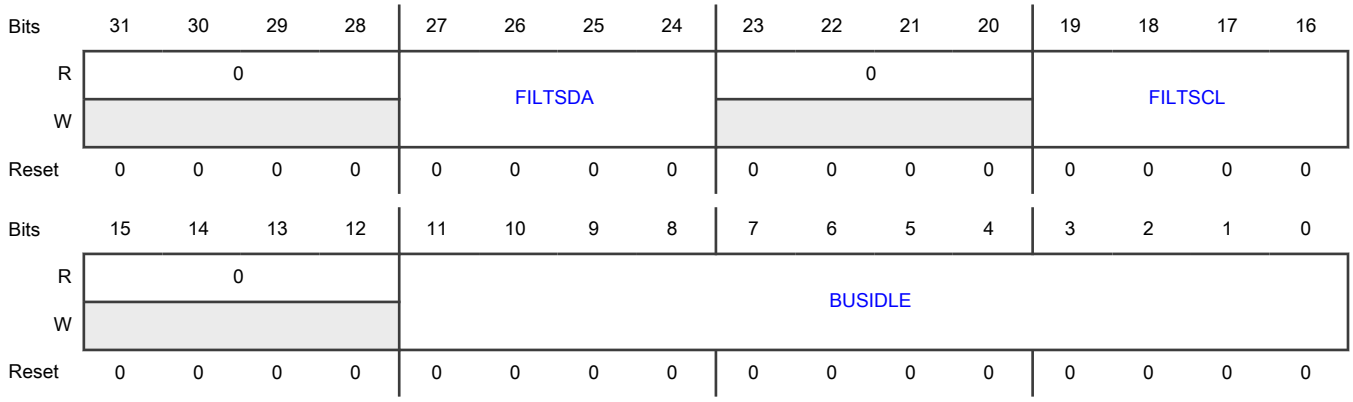
Register	Offset
MCFGR2	28h

Function

Contains the configuration for the bus idle timeout and glitch filters for SDA and SCL.

Write to this register only when the I2C controller is disabled.

Diagram



Fields

Field	Function
31-28 —	Reserved
27-24 FILTSDA	Glitch Filter SDA Configures the I2C controller digital glitch filters for the SDA input. The latency through the glitch filter is equal to the number of cycles defined by this field. The value of this field must be less than the minimum SCL low or high period. Glitches equal to or less than the number of cycles defined by this field are filtered out and ignored. Writing 0 to this field disables the glitch filter. MCFGR1[PRESCALE] does not affect the glitch filter cycle count. It is automatically bypassed in HS mode.
23-20 —	Reserved
19-16 FILTSCL	Glitch Filter SCL Configures the I2C controller digital glitch filters for SCL input. The latency through the glitch filter is equal to the number of cycles defined by this field. The value of this field must be less than the minimum SCL low or high period. Glitches equal to or less than the number of cycles defined by this field are filtered out and ignored. These cycles are based on the functional clock. Writing 0 to this field disables the glitch filter. MCFGR1[PRESCALE] does not affect the glitch filter cycle count. It is automatically bypassed in HS mode.
15-12 —	Reserved
11-0 BUSIDLE	Bus Idle Timeout Configures the bus idle timeout period, in clock cycles. If both SCL and SDA are high for longer than the number of cycles defined by this field, the I2C bus is assumed to be idle and the controller can generate a Start condition. Writing 0 to this field disables the bus idle timeout.

71.7.1.11 Controller Configuration 3 (MCFGR3)

Offset

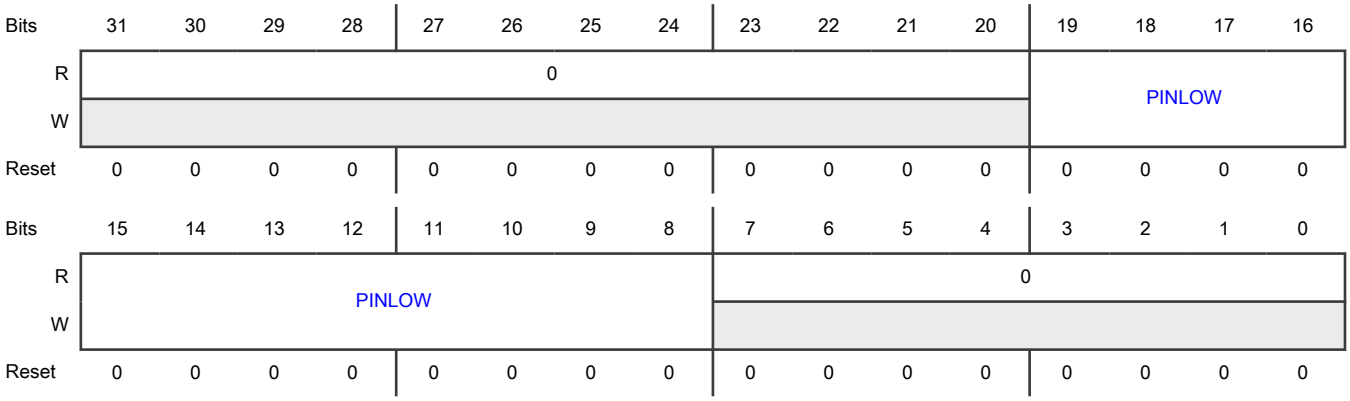
Register	Offset
MCFGR3	2Ch

Function

Configures the threshold value for the pin low timeout flag.

Write to this register only when the I2C controller is disabled.

Diagram



Fields

Field	Function
31-20 —	Reserved
19-8 PINLOW	Pin Low Timeout Configures the threshold value, in clock cycles, that sets MSR[PLTF]. If SCL or SDA (selected by MCFGR1[TIMECFG]) is low for longer than (PINLOW × 256) cycles, MSR[PLTF] is set. When this field is 0, the pin low timeout feature is disabled.
7-0 —	Reserved

71.7.1.12 Controller Data Match (MDMR)

Offset

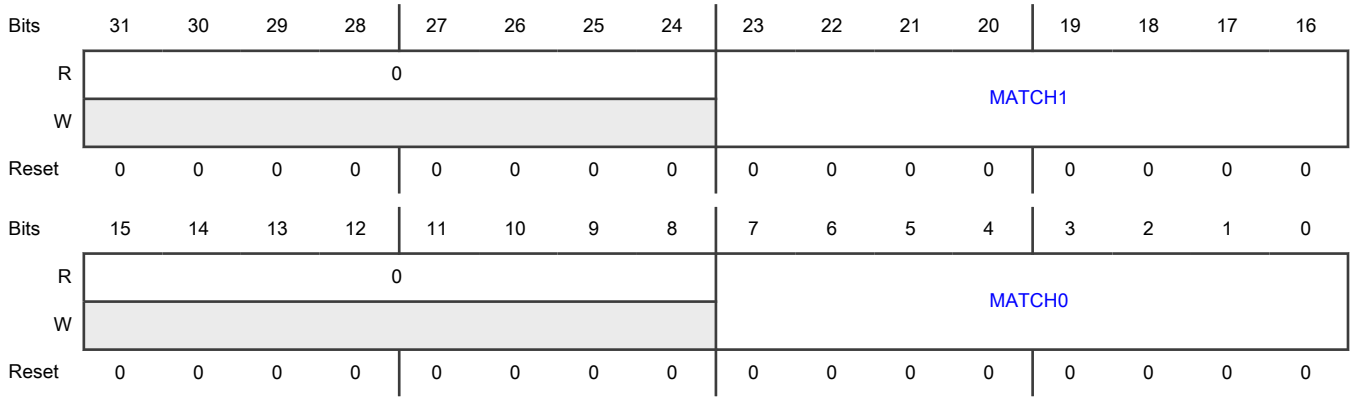
Register	Offset
MDMR	40h

Function

Contains data match values.

Write to this register only when the I2C controller is disabled or idle.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-16 MATCH1	Match 1 Value Specifies match 1 value that is compared to the received data when receive data match is enabled.
15-8 —	Reserved
7-0 MATCH0	Match 0 Value Specifies match 0 value that is compared to the received data when receive data match is enabled.

71.7.1.13 Controller Clock Configuration 0 (MCCR0)

Offset

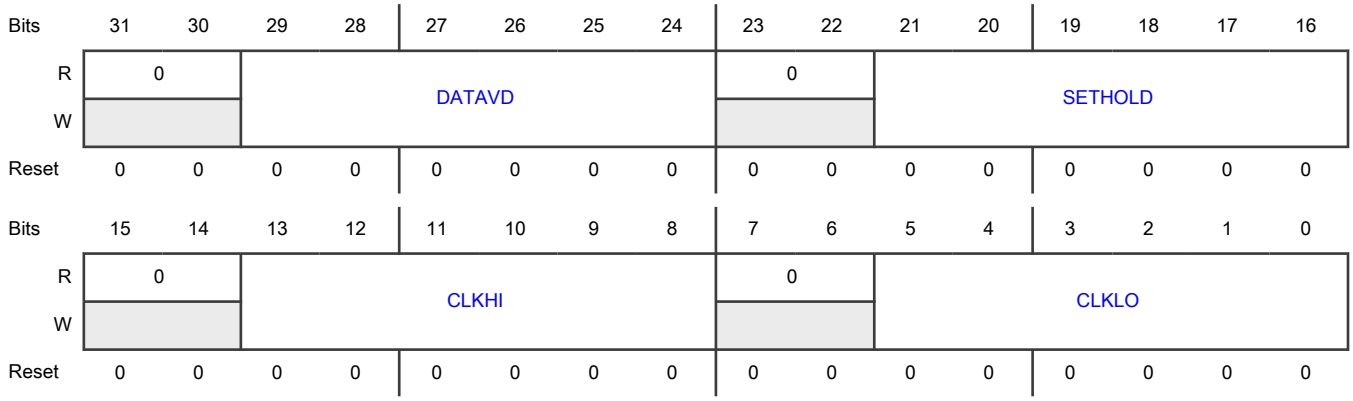
Register	Offset
MCCR0	48h

Function

Configures various clock controls.

You cannot make changes to this register when the I2C controller is enabled and is used for standard, fast, fast-mode plus, and ultra-fast transfers.

Diagram



Fields

Field	Function
31-30 —	Reserved
29-24 DATAVD	Data Valid Delay Specifies the minimum number of cycles (minus one) used as the data hold time for SDA. This value must be less than the minimum SCL low period.
23-22 —	Reserved
21-16 SETHOLD	Setup Hold Delay Specifies the minimum number of cycles (minus one) used by the controller for these conditions: • Hold time for a Start • Setup and hold time for a repeated Start • Setup time for a Stop The setup time is extended by the time it takes to detect a rising edge on the external SCL pin. Ignoring any additional board delay due to external loading, this time is equal to (2 + FILTSCL) ÷ 2^PRESCALE cycles.
15-14 —	Reserved
13-8 CLKHI	Clock High Period Specifies the minimum number of cycles (minus one) that the controller drives the SCL clock high. The SCL high time is extended by the time needed to detect a rising edge on the external SCL pin. Ignoring any additional board delay due to external loading, this time is equal to (2 + FILTSCL) ÷ 2^PRESCALE cycles.
7-6 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
5-0 CLKLO	Clock Low Period Specifies the minimum number of cycles (minus one) that the controller drives the SCL clock low. This value is also used for the minimum bus free time between a Stop and a Start condition. This period is extended by the time needed to detect a rising edge on the external SCL pin. Ignoring any additional board delay due to external loading, this time is equal to (2 + FILTSCL) ÷ 2^PRESCALE cycles.

71.7.1.14 Controller Clock Configuration 1 (MCCR1)

Offset

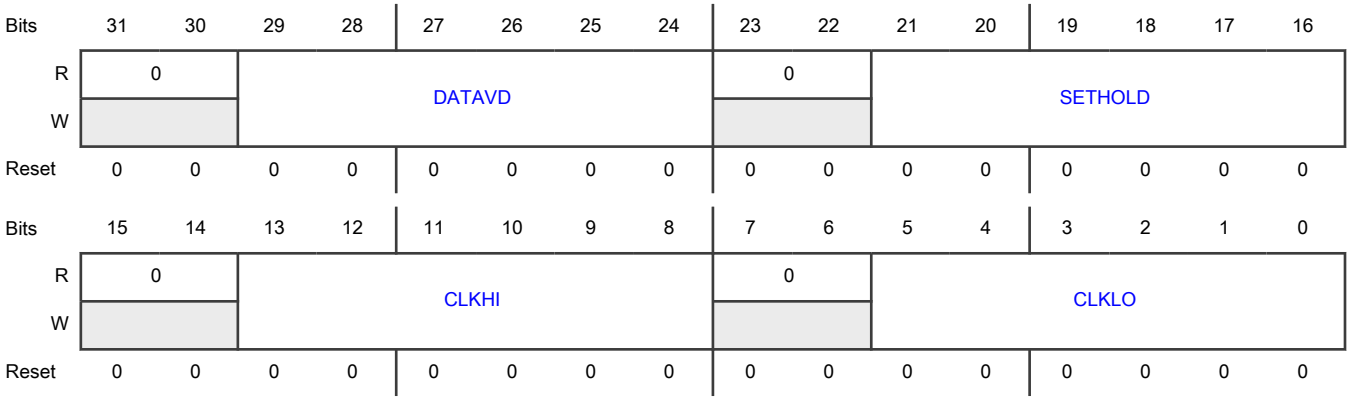
Register	Offset
MCCR1	50h

Function

Configures various clock controls.

You cannot makes changes to this register when the I2C controller is enabled and is used for HS mode transfers. The separate clock configuration for HS mode allows arbitration to take place in Fast mode (with timing configured by Controller Clock Configuration 0 (MCCR0)), before switching to HS mode (with timing configured by MCCR1).

Diagram



Fields

Field	Function
31-30 —	Reserved
29-24 DATAVD	Data Valid Delay

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Specifies the minimum number of cycles (minus one) used as the data hold time for SDA. This value must be less than the minimum SCL low period.
23-22 —	Reserved
21-16 SETHOLD	Setup Hold Delay Specifies the minimum number of cycles (minus one) used by the controller for these conditions: • Hold time for a Start condition • Setup and hold time for a repeated Start condition • Setup time for a Stop condition The setup time is extended by the time needed to detect a rising edge on the external SCL pin. Ignoring any additional board delay due to external loading, this time is equal to (2 + FILTSCL) ÷ 2^PRESCALE cycles.
15-14 —	Reserved
13-8 CLKHI	Clock High Period Specifies the minimum number of cycles (minus one) that the controller drives the SCL clock high. The SCL high time is extended by the time needed to detect a rising edge on the external SCL pin. Ignoring any additional board delay due to external loading, this time is equal to (2 + FILTSCL) ÷ 2^PRESCALE cycles.
7-6 —	Reserved
5-0 CLKLO	Clock Low Period Specifies the minimum number of cycles (minus one) that the controller drives the SCL clock low. This value is also used for the minimum bus free time between a Stop and a Start condition. This period is extended by the time needed to detect a rising edge on the external SCL pin. Ignoring any additional board delay due to external loading, this time is equal to (2 + FILTSCL) ÷ 2^PRESCALE cycles.

71.7.1.15 Controller FIFO Control (MFCR)

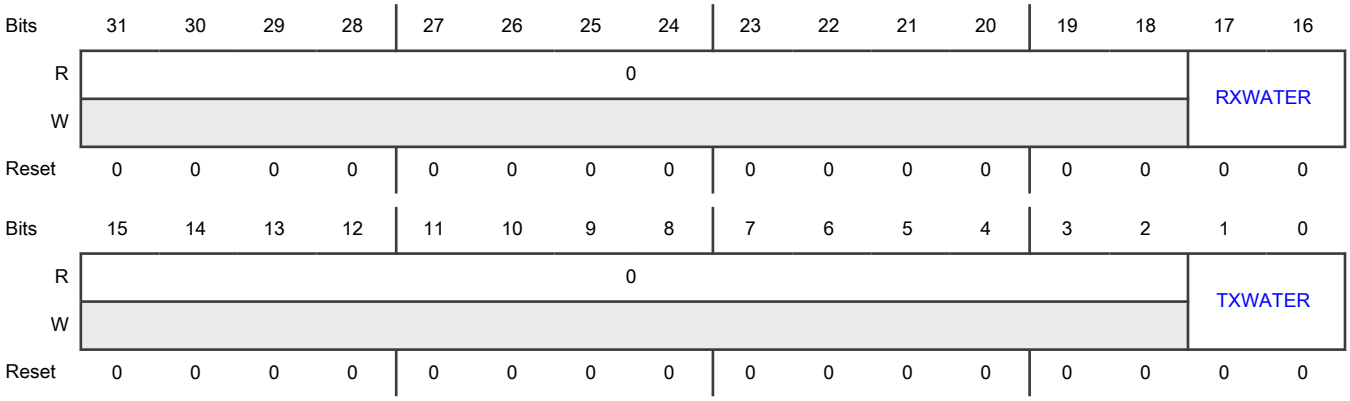
Offset

Register	Offset
MFCR	58h

Function

Controls the receive and transmit FIFO watermark values.

Diagram



Fields

Field	Function
31-18 —	Reserved
17-16 RXWATER	Receive FIFO Watermark Determines the watermark for setting SSR[RDF]. That flag is set when the number of words in the receive FIFO is greater than the value of this field. Writing a value equal to or greater than the FIFO size truncates the value.
15-2 —	Reserved
1-0 TXWATER	Transmit FIFO Watermark Determines the watermark for setting SSR[TDF]. That flag is set when the number of words in the transmit FIFO is equal or less than the value of this field. Writing a value equal to or greater than the FIFO size truncates the value.

71.7.1.16 Controller FIFO Status (MFSR)

Offset

Register	Offset
MFSR	5Ch

Function

Specifies the number of words in the transmit and receive FIFOs.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0													RXCOUNT		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0													TXCOUNT		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-19 —	Reserved
18-16 RXCOUNT	Receive FIFO Count Specifies the number of words in the receive FIFO.
15-3 —	Reserved
2-0 TXCOUNT	Transmit FIFO Count Specifies the number of words in the transmit FIFO.

71.7.1.17 Controller Transmit Data (MTDR)

Offset

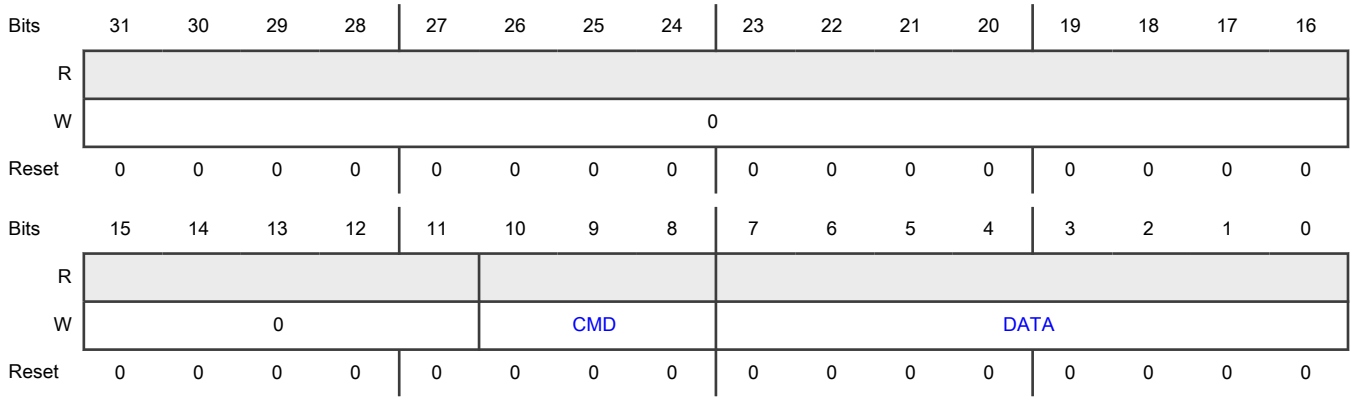
Register	Offset
MTDR	60h

Function

Configures transmit data:

- An 8-bit write to MTDR[CMD] is ignored and does not increment the FIFO write pointer.
- An 8-bit write to MTDR[DATA] zero-extends the value of MTDR[CMD] and increments the FIFO write pointer.
- A 16-bit or 32-bit write operation writes to both MTDR[CMD] and MTDR[DATA] and increments the FIFO write pointer.

Diagram



Fields

Field	Function
31-11 —	Reserved
10-8 CMD	Command Data Selects command transmitted by controller. 000b - Transmit the value in DATA[7:0] 001b - Receive (DATA[7:0] + 1) bytes. DATA[7:0] is used as a byte counter. Receive that many bytes and check each for a data match (if configured) before storing the received data in the receive FIFO. 010b - Generate Stop condition on I2C bus 011b - Receive and discard (DATA[7:0] + 1) bytes. DATA[7:0] is used as a byte counter. Receive that many bytes but do not check for a data match or store those bytes in the receive FIFO. 100b - Generate (repeated) Start on the I2C bus and transmit the address in DATA[7:0] 101b - Generate (repeated) Start on the I2C bus and transmit the address in DATA[7:0] (this transfer expects a NACK to be returned) 110b - Generate (repeated) Start on the I2C bus and transmit the address in DATA[7:0] using HS mode 111b - Generate (repeated) Start on the I2C bus and transmit the address in DATA[7:0] using HS mode (this transfer expects a NACK to be returned)
7-0 DATA	Transmit Data Contains data used by the commands listed in MTDR[CMD]. Performing an 8-bit write to this field zero-extends the value of MTDR[CMD].

71.7.1.18 Controller Receive Data (MRDR)

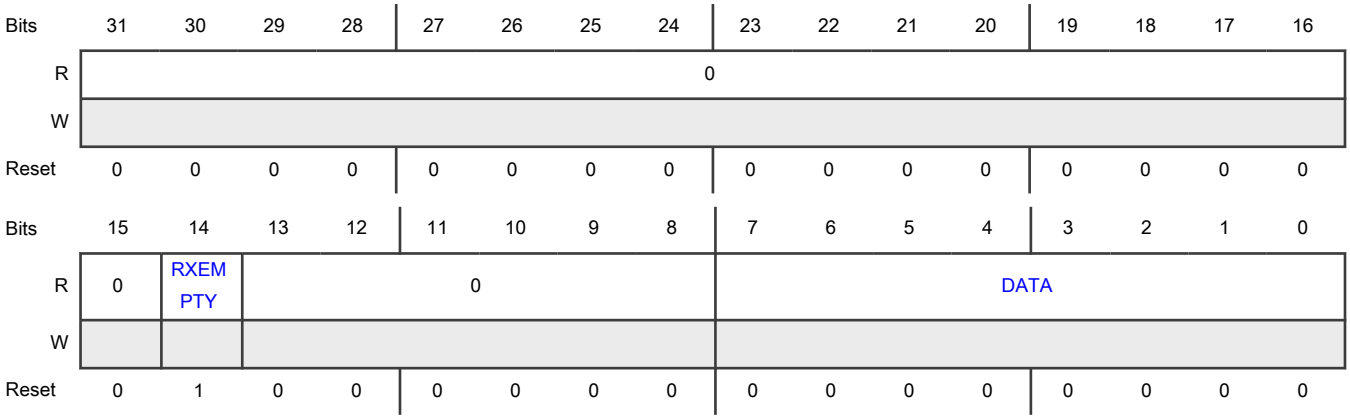
Offset

Register	Offset
MRDR	70h

Function

Contains the status of the receive FIFO and the data received by the I2C controller that has not been discarded.

Diagram



Fields

Field	Function
31-15 —	Reserved
14 RXEMPTY	Receive Empty Indicates whether the controller receive data FIFO is empty. 0b - Not empty 1b - Empty
13-8 —	Reserved
7-0 DATA	Receive Data Contains data received by the I2C controller that has not been discarded. Received data can be discarded due to the command in MTDR[CMD], or the controller can be configured to discard nonmatching data.

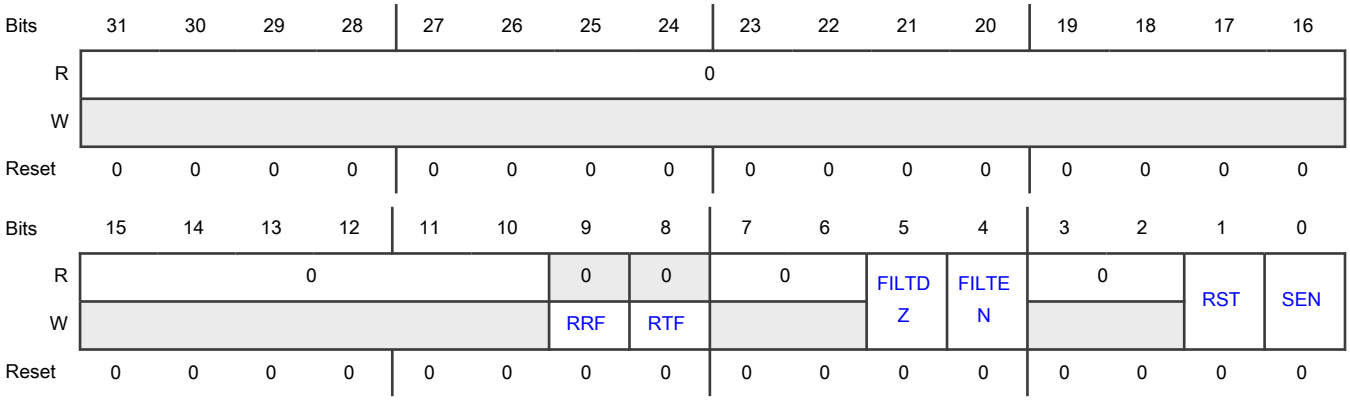
71.7.1.19 Target Control (SCR)

Offset

Register	Offset
SCR	110h

Function
Contains resets and other target control settings.

Diagram



Fields

Field	Function
31-10 —	Reserved
9 RRF	Reset Receive FIFO Empties the receive FIFO in Target Receive Data (SRDR). 0b - No effect 1b - SRDR is now empty
8 RTF	Reset Transmit FIFO Empties the transmit FIFO in Target Transmit Data (STDR). 0b - No effect 1b - STDR is now empty
7-6 —	Reserved
5	Filter Doze Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
FILTDZ	Enables filter in Doze mode. Update this field only when the I2C target is disabled. 0b - Enable 1b - Disable
4 FILTEN	Filter Enable Enables digital filter and output delay counter for target mode. Update this field only when the I2C target is disabled. 0b - Disable 1b - Enable
3-2 —	Reserved
1 RST	Software Reset Resets target mode logic. The reset takes effect immediately. The value of this field remains 1 until you write 0 to it. There is no minimum delay required before clearing the software reset. 0b - Not reset 1b - Reset
0 SEN	Target Enable Enables I2C Target mode. 0b - Disable 1b - Enable

71.7.1.20 Target Status (SSR)

Offset

Register	Offset
SSR	114h

Function

Contains status flags for transmit and receive data, for error conditions, and for bus and target busy or idle status.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0						BBF	SBF	0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SARF	GCF	AM1F	AM0F	FEF	BEF	SDF	RSF	0				TAF	AVF	RDF	TDF
W					W1C	W1C	W1C	W1C								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-26 —	Reserved
25 BBF	Bus Busy Flag Indicates whether an I2C bus is idle or busy. 0b - Idle 1b - Busy
24 SBF	Target Busy Flag Indicates whether an I2C target is idle or busy. 0b - Idle 1b - Busy
23-16 —	Reserved
15 SARF	SMBus Alert Response Flag Indicates whether an SMBus alert response has been detected. You can clear this flag by reading Target Address Status (SASR). This flag cannot generate an asynchronous wakeup. 0b - Disabled or not detected 1b - Enabled and detected
14 GCF	General Call Flag Indicates whether a target has detected the general call address. You can clear this flag by reading Target Address Status (SASR). This flag cannot generate an asynchronous wakeup.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - General call address disabled or not detected 1b - General call address detected
13 AM1F	Address Match 1 Flag Indicates whether the received address matches the value in ADDR1, or it falls within the ADDR0 to ADDR1 range as configured by SCFGR1[ADDRCFG]. This flag is cleared by reading Target Address Status (SASR). This flag cannot generate an asynchronous wakeup. 0b - Matching address not received 1b - Matching address received
12 AM0F	Address Match 0 Flag Indicates whether the received address matches the ADDR0 field, as configured by SCFGR1[ADDRCFG]. This flag is cleared by reading Target Address Status (SASR). This flag cannot generate an asynchronous wakeup. 0b - ADDR0 matching address not received 1b - ADDR0 matching address received
11 FEF	FIFO Error Flag Indicates whether there is a FIFO error. This flag can only be set when clock stretching is disabled. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No FIFO error 1b - FIFO error When writing 0b - No effect 1b - Clear the flag
10 BEF	Bit Error Flag Indicates whether the LPI2C target has transmitted a logic 1 and detects a logic 0 on the I2C bus. The target ignores the rest of the transfer until the next (repeated) Start condition. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No bit error occurred

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Bit error occurred When writing 0b - No effect 1b - Clear the flag
9 SDF	Stop Detect Flag Indicates whether the LPI2C target detects a Stop condition, and if the LPI2C target matched the last address byte. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No Stop detected 1b - Stop detected When writing 0b - No effect 1b - Clear the flag
8 RSF	Repeated Start Flag Indicates whether the LPI2C target detects a repeated Start condition and if the LPI2C target matched the last address byte. This flag is not set when the target first detects a Start condition. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No repeated Start detected 1b - Repeated Start detected When writing 0b - No effect 1b - Clear the flag
7-4 —	Reserved
3 TAF	Transmit ACK Flag Indicates whether a transmit ACK or NACK is required. You can clear this flag by writing to Target Transmit ACK (STAR). 0b - Not required 1b - Required

Table continues on the next page...

Table continued from the previous page...

Field	Function
2 AVF	Address Valid Flag Indicates whether the contents of Target Address Status (SASR) are valid. You can clear this flag by reading SASR. When SCFGR1[RXCFG] = 1, this flag is also cleared by reading Target Receive Data (SRDR). 0b - Not valid 1b - Valid
1 RDF	Receive Data Flag Indicates whether receive data is ready. You can clear this flag by reading Target Receive Data (SRDR). When SCFGR1[RXCFG] = 1, this flag is not cleared when reading Target Receive Data (SRDR) if SSR[AVF] = 1. 0b - Not ready 1b - Ready
0 TDF	Transmit Data Flag Indicates whether transmit data has been requested. This flag is cleared by writing to Target Transmit Data (STDR). When SCFGR1[TXCFG] = 0, if a NACK, repeated Start, or Stop condition is detected, this flag is also cleared. 0b - Transmit data not requested 1b - Transmit data is requested

71.7.1.21 Target Interrupt Enable (SIER)

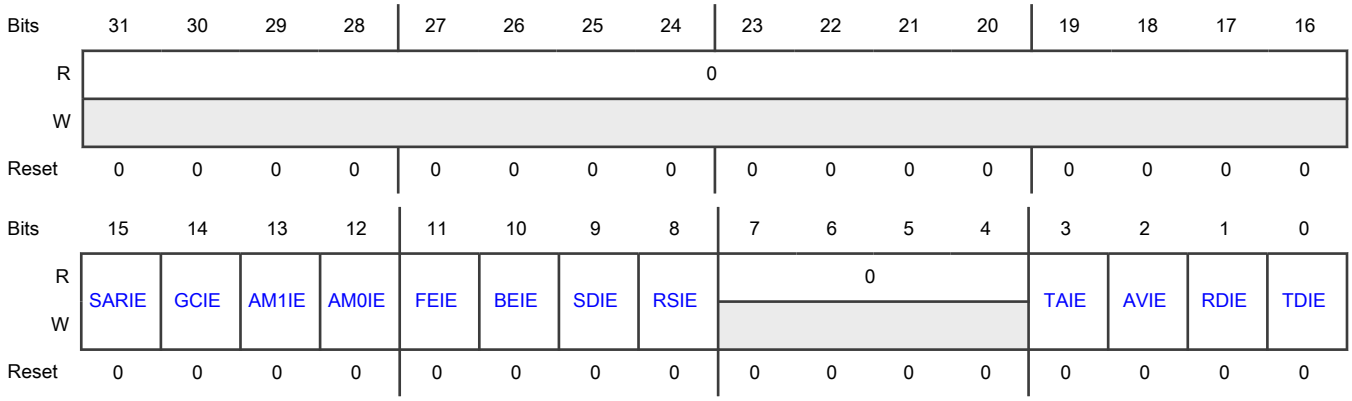
Offset

Register	Offset
SIER	118h

Function

Contains transmit and receive data interrupt enables, start and stop detect interrupt enables, and other target interrupt enables.

Diagram



Fields

Field	Function
31-16 —	Reserved
15 SARIE	SMBus Alert Response Interrupt Enable Enables interrupt for SMBus alert response. 0b - Disable 1b - Enable
14 GCIE	General Call Interrupt Enable Enables interrupt for general call. 0b - Disabled 1b - Enabled
13 AM1IE	Address Match 1 Interrupt Enable Enables interrupt for address match 1. 0b - Disable 1b - Enable
12 AM0IE	Address Match 0 Interrupt Enable Enables interrupt for address match 0. 0b - Disable 1b - Enable
11 FEIE	FIFO Error Interrupt Enable Enables interrupt for FIFO error. 0b - Disable 1b - Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
10 BEIE	Bit Error Interrupt Enable Enables interrupt for bit error. 0b - Disable 1b - Enable
9 SDIE	Stop Detect Interrupt Enable Enables interrupt for Stop detection. 0b - Disable 1b - Enable
8 RSIE	Repeated Start Interrupt Enable Enables interrupt for repeated start. 0b - Disable 1b - Enable
7-4 —	Reserved
3 TAIE	Transmit ACK Interrupt Enable Enables interrupt for transmit ACK. 0b - Disable 1b - Enable
2 AVIE	Address Valid Interrupt Enable Enables interrupt for valid address. 0b - Disable 1b - Enable
1 RDIE	Receive Data Interrupt Enable Enables interrupt for receive data. 0b - Disable 1b - Enable
0 TDIE	Transmit Data Interrupt Enable Enables interrupt for transmit data. 0b - Disable 1b - Enable

71.7.1.22 Target DMA Enable (SDER)

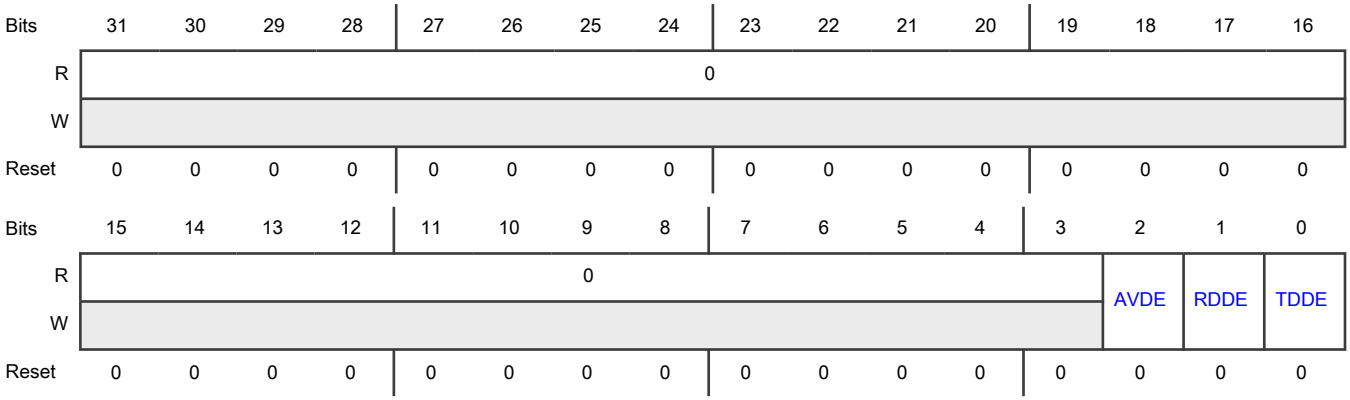
Offset

Register	Offset
SDER	11Ch

Function

Contains the transmit, request, and receive enables for DMA.

Diagram



Fields

Field	Function
31-3 —	Reserved
2 AVDE	Address Valid DMA Enable Enables address valid DMA request. The address valid DMA request is shared with the receive data DMA request. If both are enabled, write 1 to SCFGR1[RXCFG] to allow the DMA to read the address from Target Receive Data (SRDR). 0b - Disable 1b - Enable
1 RDDE	Receive Data DMA Enable Enables receive data for DMA. 0b - Disable DMA request 1b - Enable DMA request
0 TDDE	Transmit Data DMA Enable Enables transmit data for DMA.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disable 1b - Enable

71.7.1.23 Target Configuration 1 (SCFGR1)

Offset

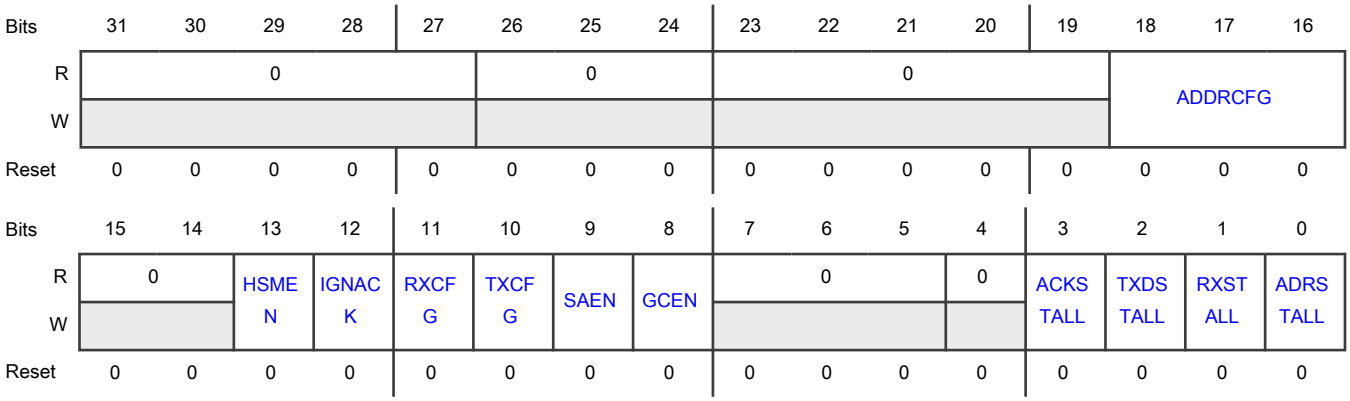
Register	Offset
SCFGR1	124h

Function

Configures various aspects of the target.

Write to this register only when the I2C target is disabled.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-24 —	Reserved
23-19 —	Reserved
18-16	Address Configuration

Table continues on the next page...

Table continued from the previous page...

Field	Function
ADDRCFG	Configures the condition that causes an address to match. 000b - Address match 0 (7-bit) 001b - Address match 0 (10-bit) 010b - Address match 0 (7-bit) or address match 1 (7-bit) 011b - Address match 0 (10-bit) or address match 1 (10-bit) 100b - Address match 0 (7-bit) or address match 1 (10-bit) 101b - Address match 0 (10-bit) or address match 1 (7-bit) 110b - From address match 0 (7-bit) to address match 1 (7-bit) 111b - From address match 0 (10-bit) to address match 1 (10-bit)
15-14 —	Reserved
13 HSMEN	HS Mode Enable Enables detection of the HS mode controller code of target address 0000_1XX, but does not cause an address match on this code. When this field is 1 and any HS mode controller code is detected, SCR[FILTEN] and SCFGR1[ACKSTALL] are ignored until the next Stop condition is detected. 0b - Disable 1b - Enable
12 IGNACK	Ignore NACK Determines whether the target ends transfer when a NACK condition is detected. When this field is 1, the LPI2C target continues transfers after a NACK is detected. This field is required to be 1 in Ultra-Fast mode. 0b - End transfer on NACK 1b - Do not end transfer on NACK
11 RXCFG	Receive Data Configuration Configures which data is returned and which flags are cleared when reading Target Receive Data (SRDR). When this field is 0, reading SRDR returns received data and clears SSR[RDF]. When this field is 1, reading SRDR: • Returns the value of Target Address Status (SASR) and clears SSR[AVF] when SSR[AVF] is set. • Returns received data and clears SSR[RDF] when SSR[AVF] is not set. 0b - Return received data, clear SSR[RDF] 1b - Return SASR and clear SSR[AVF] when SSR[AVF] is set, return received data and clear SSR[RDF] when SSR[AFV] is not set
10 TXCFG	Transmit Flag Configuration Determines which conditions set SSR[TDF].

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field always becomes 1 before a NACK is detected at the end of a target-transmit transfer. This change can cause an extra word to be written to the transmit data FIFO. When this field is 0, Target Transmit Data (STDR) is automatically emptied when a target-transmit transfer is detected. SSR[TDF] is set when a target-transmit transfer is detected, and SSR[TDF] is cleared at the end of the target-transmit transfer. When this field is 1, SSR[TDF] is set when STDR is empty, and SSR[TDF] is cleared when STDR is full. This setting allows STDR to be filled before a target-transmit transfer is detected. However, it can cause STDR to be written before a NACK is detected on the last byte of a target-transmit transfer. 0b - SSR[TDF] is set only during a target-transmit transfer when STDR is empty 1b - SSR[TDF] is set whenever STDR is empty
9 SAEN	SMBus Alert Enable Enables a match on an SMBus alert. 0b - Disable 1b - Enable
8 GCEN	General Call Enable Enables a general call address. 0b - Disable 1b - Enable
7-5 —	Reserved
4 —	Reserved
3 ACKSTALL	ACK SCL Stall Enables SCL clock stretching during target-transmit address bytes and target-receiver address and data bytes, so you can write to Target Transmit ACK (STAR) before the ACK or NACK is transmitted. Clock stretching occurs when transmitting the ninth bit, and is therefore not compatible with HS mode. If this field is 1: • You do not need to write 1 to SCFGR1[RXSTALL] or SCFGR1[ADRSTALL]. • When there is an address match on the first byte of a 10-bit address, SSR[AVF] is set, allowing you to read the received address before writing to Target Transmit ACK (STAR). 0b - Disable 1b - Enable
2 TXDSTALL	Transmit Data SCL Stall

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Enables SCL clock stretching when SSR[TDF] = 1 during a target-transmit transfer. Clock stretching occurs following the ninth bit, and is therefore compatible with HS mode. 0b - Disable 1b - Enable
1 RXSTALL	RX SCL Stall Enables SCL clock stretching when SSR[RDF] = 1 during a target-receive transfer. Clock stretching occurs following the ninth bit, and is therefore compatible with HS mode. 0b - Disable 1b - Enable
0 ADRSTALL	Address SCL Stall Enables SCL clock stretching when SSR[AVF] = 1. Clock stretching only occurs following the ninth bit, and is therefore compatible with HS mode. 0b - Disable 1b - Enable

71.7.1.24 Target Configuration 2 (SCFGR2)

Offset

Register	Offset
SCFGR2	128h

Function

Configures data valid delay, clock hold time, and glitch filters for SDA and SCL.

Write to this register only when the I2C target is disabled.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				FILTSDA				0				FILTSCL			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		DATAVD						0				CLKHOLD			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved
27-24 FILTSDA	Glitch Filter SDA Configures the I2C target digital glitch filters for SDA input. Writing 0 to this field disables the glitch filter. Glitches equal to or less than the number of cycles defined by this field are filtered out and ignored. The latency through the glitch filter is equal to the number of cycles defined by this field + 3. The latency must be configured to be less than the minimum SCL low or high period. MCFGR1[PRESCALE] does not affect the glitch filter cycle count, and the glitch filter cycle count is disabled in HS mode.
23-20 —	Reserved
19-16 FILTSCL	Glitch Filter SCL Configures the I2C target digital glitch filters for SCL input. Writing 0 to this field disables the glitch filter. Glitches equal to or less than the number of cycles defined by this field are filtered out and ignored. The latency through the glitch filter is equal to the number of cycles defined by this field + 3. The latency must be configured to be less than the minimum SCL low or high period. MCFGR1[PRESCALE] does not affect the glitch filter cycle count, and the glitch filter cycle count is disabled in HS mode.
15-14 —	Reserved
13-8 DATAVD	Data Valid Delay Configures the SDA data valid delay time for the I2C target, which is equal to FILTSCL + DATAVD + 3 cycles. The data valid delay must be configured to be less than the minimum SCL low period. MCFGR1[PRESCALE] does not affect the I2C target data valid delay time, and the I2C target data valid delay time is disabled in HS mode.
7-4 —	Reserved
3-0 CLKHOLD	Clock Hold Time Configures the minimum clock hold time for the I2C target, when clock stretching is enabled. The minimum hold time is equal to the number of cycles defined by this field + 3.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	MCFGR1[PRESCALE] does not affect the I2C target clock hold time, and the I2C target clock hold time is disabled in HS mode.

71.7.1.25 Target Address Match (SAMR)

Offset

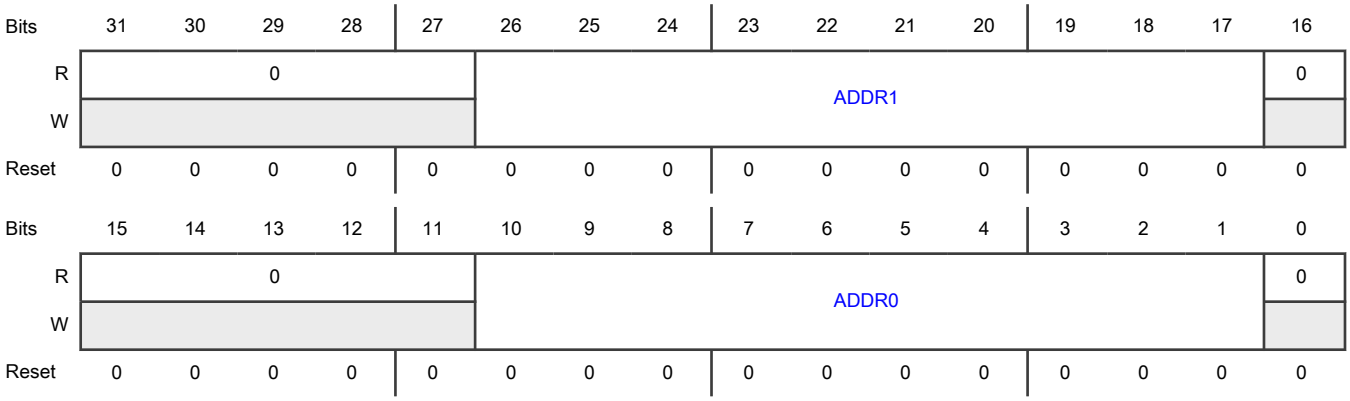
Register	Offset
SAMR	140h

Function

Contains address values for received target match comparison.

Write to this register only when the I2C target is disabled.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-17 ADDR1	Address 1 Value Contains the value of address 1, which is compared to the received address to detect the target address. In 10-bit mode, the first address byte is compared to {11110, ADDR1[26:25]} and the second address byte is compared to ADDR1[24:17]. In 7-bit mode, the address is compared to ADDR1[23:17].

Table continues on the next page...

Table continued from the previous page...

Field	Function
16-11 —	Reserved
10-1 ADDR0	Address 0 Value Contains the value of address 0, which is compared to the received address to detect the target address. In 10-bit mode, the first address byte is compared to {11110, ADDR0[10:9]} and the second address byte is compared to ADDR0[8:1]. In 7-bit mode, the address is compared to ADDR0[7:1].
0 —	Reserved

71.7.1.26 Target Address Status (SASR)

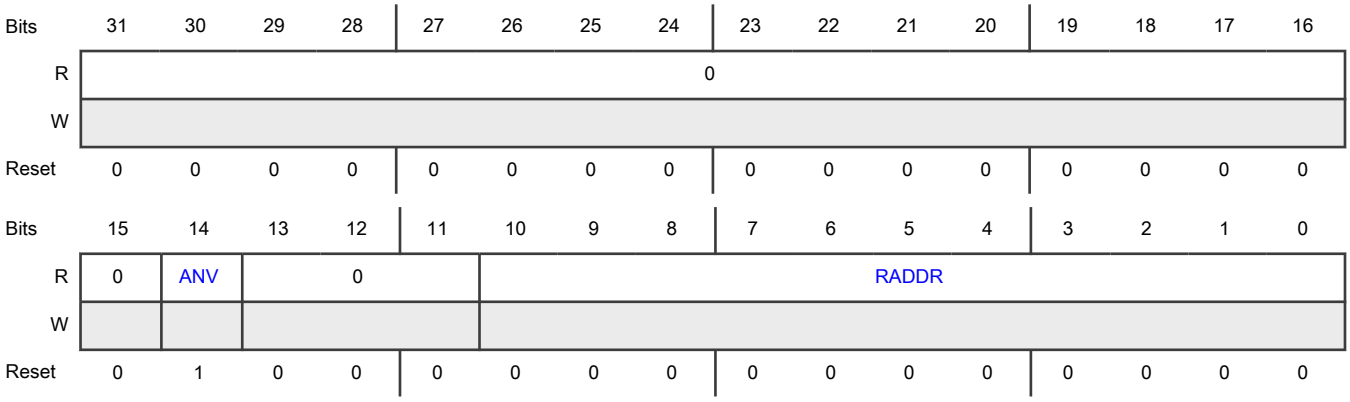
Offset

Register	Offset
SASR	150h

Function

Contains the received address and its validity.

Diagram



Fields

Field	Function
31-15	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
14 ANV	Address Not Valid Indicates whether SASR[RADDR] is valid. 0b - Valid 1b - Not valid
13-11 —	Reserved
10-0 RADDR	Received Address Contains the received address. Updates whenever SSR[AM0F] or SSR[AM1F] is set. Reading Target Address Status (SASR) clears SSR[AM0F] and SSR[AM1F]. In 7-bit mode, the address byte is stored in RADDR[7:0]. In 10-bit mode, the first address byte is {11110, RADDR[10:9], RADDR[0]} and the second address byte is RADDR[8:1]. The Read-or-Write bit is therefore always stored in RADDR[0]. When SCFGR1[ACKSTALL] = 1, if the first address byte matches in 10-bit mode, the first address byte is stored in RADDR[7:0] so you can read this field before writing the Transmit ACK. If the second address byte matches, this field is then updated with the full 10-bit address.

71.7.1.27 Target Transmit ACK (STAR)

Offset

Register	Offset
STAR	154h

Function

Configures choice of ACK or NACK on each received word.

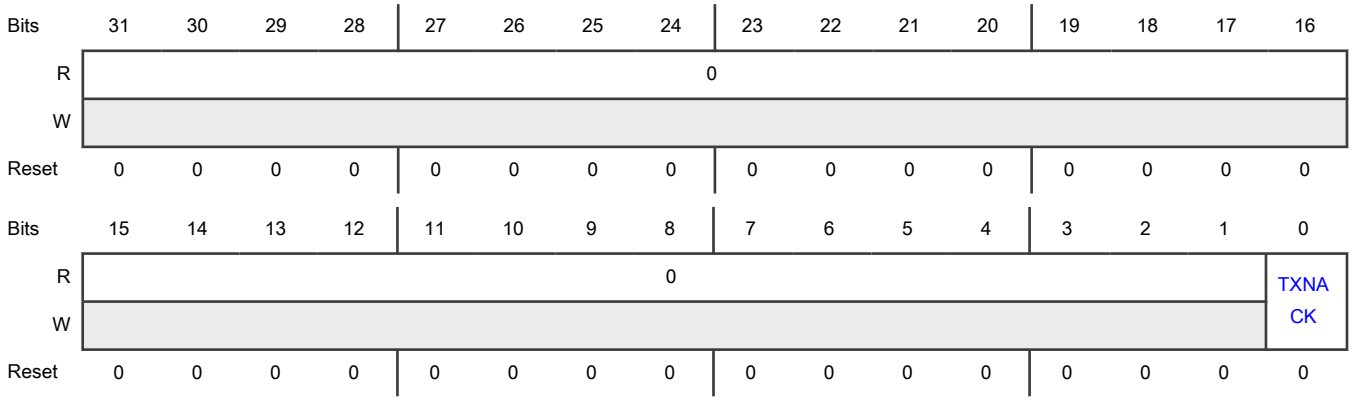
You can write to this register only when SCFGR1[ACKSTALL] = 1.

SCFGR1[ACKSTALL] enables clock stretching during the ACK-or-NACK bit slot. During this time, you can write to this register.

The logic ensures that the clock stretching continues for at least one bus clock cycle after this register is updated.

This clock stretching time can be extended via SCFGR2[CLKHOLD].

Diagram



Fields

Field	Function
31-1 —	Reserved
0 TXNACK	Transmit NACK Selects whether transmit ACK (logic 0) or NACK (logic 1) is returned on the bus by the I2C target after receiving each word. • When SCFGR1[ACKSTALL] = 1, a transmit NACK signal must be written once for each matching address byte and each received word. SCFGR1[ACKSTALL] must be 1, because that setting stalls the data transfer until software reads the received word (and determines whether to respond with an ACK or NACK). • To configure the default (ACK or NACK), you can write to this field when LPI2C target is disabled or idle. 0b - Transmit ACK 1b - Transmit NACK

71.7.1.28 Target Transmit Data (STDR)

Offset

Register	Offset
STDR	160h

Function

Contains the I2C target data to transmit.

Clock stretching (enabled or disabled) affects when the transmit data is transferred. SCFGR1[TXDSTALL] enables clock stretching during the first data bit of a target-transmit transfer.

If clock stretching is enabled (SCFGR1[TXDSTALL] = 1), the transmit data transfer is stalled until this register is updated. Clock stretching is extended by at least 1 bus clock cycle after this register is updated. Clock stretching can be delayed further by using SCFGR2[CLKHOLD].

If clock stretching is disabled (SCFGR1[TXDSTALL] = 0), the transmit data must be written before the start of the target-transmit transfer, otherwise SSR[FEF] is set.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	0								DATA							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-8 —	Reserved
7-0 DATA	Transmit Data Contains the I2C target data to transmit. Writing data to this register stores I2C target transmit data in this register.

71.7.1.29 Target Receive Data (SRDR)

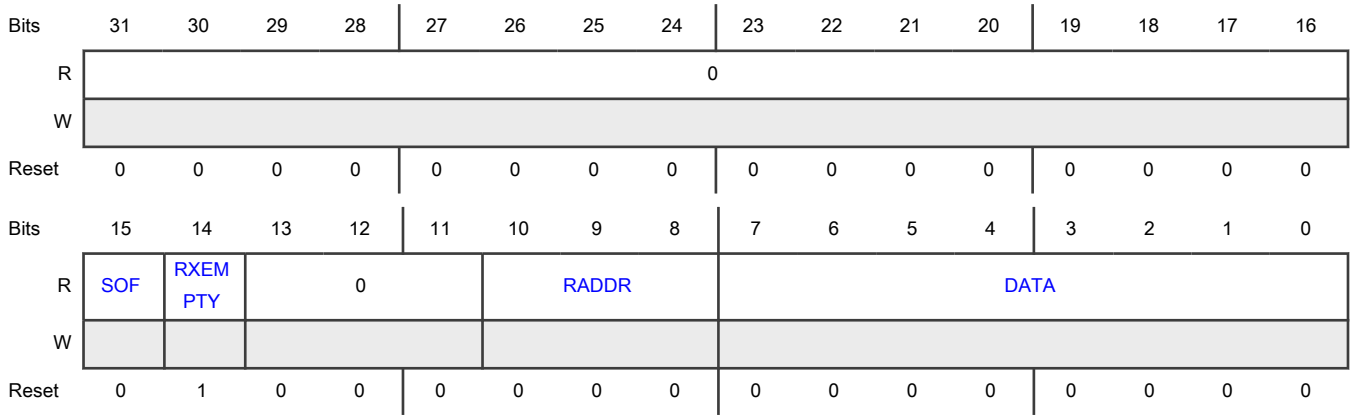
Offset

Register	Offset
SRDR	170h

Function

Contains status of target receive data transfer.

Diagram



Fields

Field	Function
31-16 —	Reserved
15 SOF	Start of Frame Indicates whether this data word is the first data word since a (repeated) Start or Stop condition. 0b - Not first 1b - First
14 RXEMPTY	Receive Empty Indicates whether this register is empty. 0b - Not empty 1b - Empty
13-11 —	Reserved
10-8 RADDR	Received Address Contains the address received by the IC2 target. When both SCFGR1[RXCFG] and SSR[AVF] are 1, bits [10:8] of SASR[RADDR] are returned. Otherwise, this field returns zero.
7-0 DATA	Received Data Contains the data received by the I2C target. When both SCFGR1[RXCFG] and SSR[AVF] are 1, bits [7:0] of SASR[RADDR] are returned.

71.8 Glossary

HREQ	Host request
SCL	Serial clock line

SCLS	Secondary serial clock line
SDA	Serial data line
SDAS	Secondary serial data line

Chapter 72
Flexible I/O (FlexIO)

72.1 Chip-specific FlexIO information

72.1.1 FlexIO instances and configuration

The device contains one instance of FlexIO.

FlexIO is a highly configurable module providing a wide range of functionality including:

- Emulation of a variety of serial communication protocols like UART, I2C, SPI, SAI. It can work either as master or slave.
- Flexible 16-bit timers with support for a variety of trigger, reset, enable and disable conditions.

Table 466. FlexIO instances

Chip	Instance	No. of pins	No. of timers	No. of shift registers	No. of bits in counter
S32K312, S32K322, S32K342, S32K341, S32K314, S32K324, S32K344, S32K358, S32K356, S32K348, S32K338, S32K328, S32K388 and S32K389	FlexIO	32	8	8	16
S32K311, S32K310	FlexIO	16	8	8	16

Table 467. Data rate limitation

FlexIO operation	Data rate limitation
Master Tx	FlexIO_clk/4
Slave Tx	FlexIO_clk/10
Master Rx	FlexIO_clk/8
Slave Rx	FlexIO_clk/6

NOTE

FlexIO supports 32 bit accesses only.

NOTE

When I2S is emulated on FlexIO, you could observe a skew between bit-clock (BCLK) and frame sync pulse amounting to 1/2 of BCLK.

72.2 Overview

FLEXIO is a highly configurable module that provides:

- Emulation of various serial or parallel communication protocols.
- Flexible 16-bit timers with support for various trigger, reset, enable, and disable conditions.
- Programmable logic blocks that allow the implementation of digital logic functions on-chip and configurable interaction of internal and external modules.
- Programmable state machine for offloading basic system control functions from the CPU.

72.2.1 Block diagram

The following diagram provides a high-level overview of the FLEXIO timer and shifter configuration.

FLEXIO uses shifters, timers, and external triggers to shift data into or out of FLEXIO. As shown in the block diagram, timers control the timing of this data shift. You can configure the timers to use generic timer functions, external triggers, or various other conditions to determine the control.

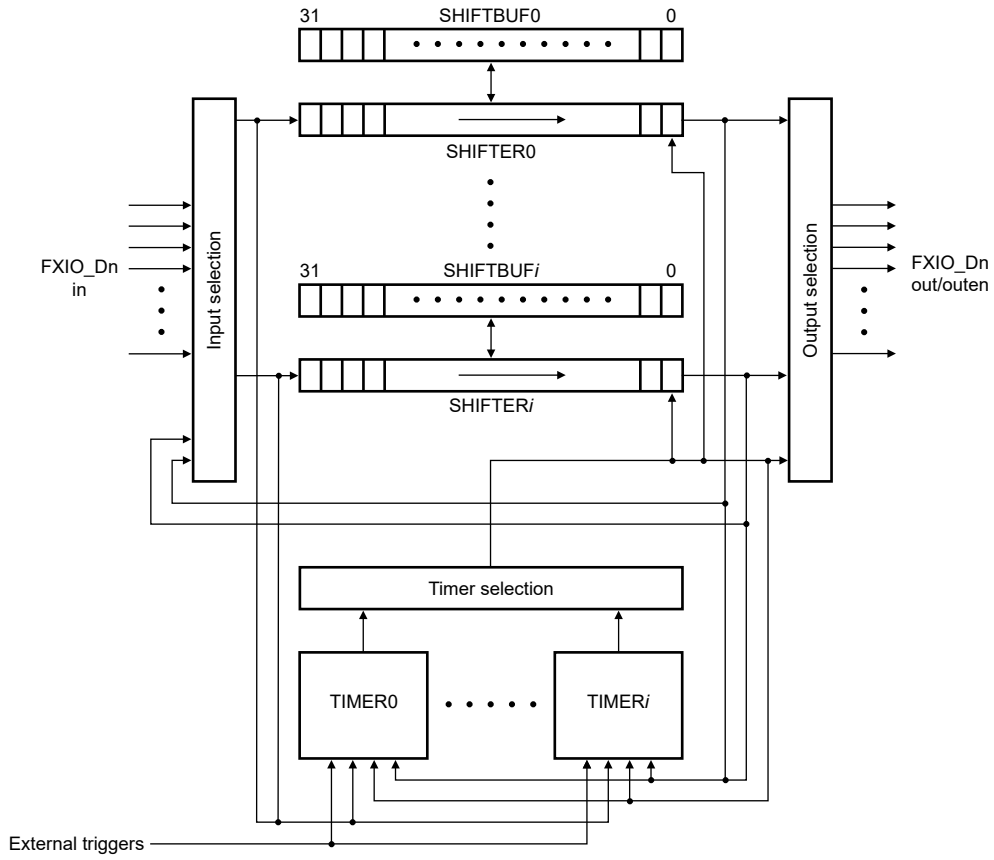


Figure 394. Block diagram

72.2.2 Features

- Array of 32-bit shift registers with transmit, receive, data match, logic, and state modes:
  - Double-buffered shifter operation for continuous data transfer
  - Shifter concatenation to support large transfer sizes
  - Automatic start and stop bit generation

- 1, 2, 4, 8, 16, or 32 multi-bit shift widths for parallel interface support
- Interrupt, DMA, or polled transmit and receive operation

- Highly flexible 16-bit timers with support for various internal or external triggers, reset, enable, and disable conditions:
  - Programmable baud rates independent of bus clock frequency
  - Programmable logic mode for integrating external digital logic functions on-chip, or combining pin, shifter, or timer functions to generate complex outputs
  - Programmable state machine for offloading basic system control functions from CPU, with support for up to eight states, eight outputs, and three selectable inputs per state
- Integrated general-purpose I/O registers and pin rising or falling edge interrupts to simplify software support
- Support for a wide range of protocols, including but not limited to:
  - UART
  - I2C
  - SPI
  - I2S
  - Camera IF
  - Motorola 68K or Intel 8080 bus
  - PWM or waveform generation
  - Input-capture (pulse-edge interval measurement), such as SENT

72.3 Functional description

72.3.1 Shifter operation

Shifters are responsible for buffering and shifting data into or out of FLEXIO. The timer assigned to the shifter controls the timing of shift, load, and store events via SHIFTCTL*n*[TIMSEL]. Shifters are designed to support either DMA, interrupt, or polled operations. The following figure provides a detailed view of the shifter microarchitecture.

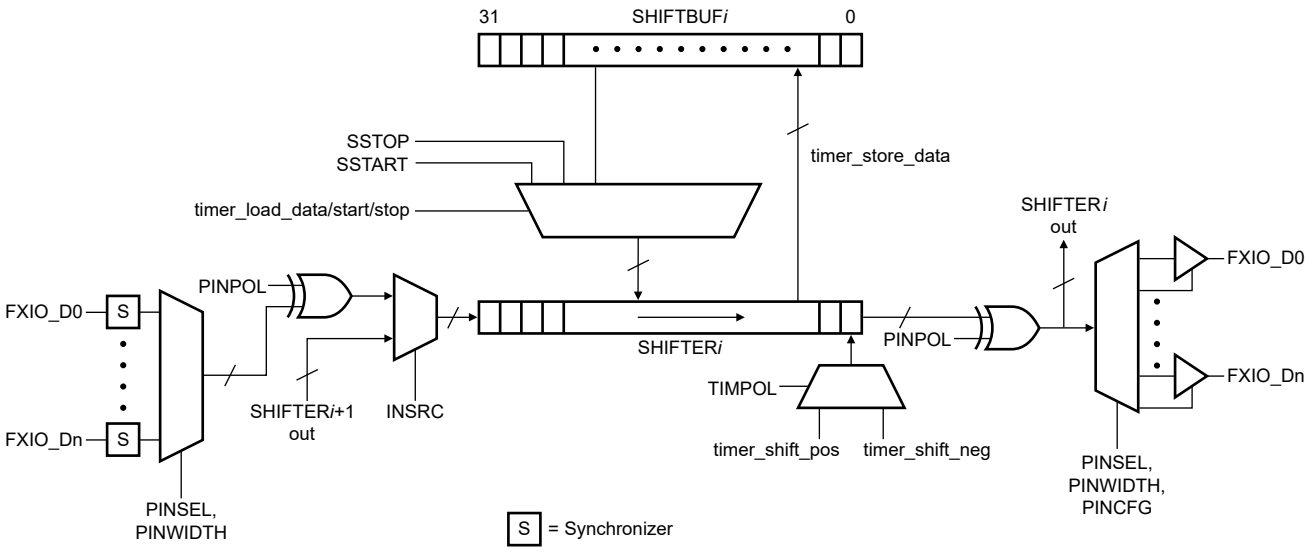


Figure 395. Shifter microarchitecture

72.3.1.1 Transmit mode

In Transmit mode (SHIFTCTL*n*[SMOD] = 010b), the shifter loads data from Shifter Buffer (SHIFTBUF0 - SHIFTBUF7) and shifts data out when the assigned timer signals a load event. An optional start and stop bit can be automatically loaded before or after SHIFTBUF register data by configuring either SHIFTCFG[SSTART] and TIMCFG[TSTART], or SHIFTCFG[SSTOP] and TIMCFG[TSTOP] in the shifter and timer.

NOTE

If a stop bit is enabled, the shifter immediately loads a stop bit when it is initially configured for Transmit mode.

The shifter status flag (SHIFTSTAT[SSF]) and any enabled interrupts or DMA requests are set when data has either been loaded from the SHIFTBUF register into the shifter or when the shifter is initially configured for Transmit mode. To clear the flag, write 1 or write new data to SHIFTBUF. In Transmit mode, write any value to the SHIFTBUF register to clear the corresponding shifter status flag, which is cleared regardless of what is writing to the register (DMA or interrupt), or the state of the DMA or interrupt enables. See the functional description of SHIFTSTAT[SSF] for information on how the flag is set and cleared for each mode.

The shifter error flag (SHIFTERR[SEF]) and any enabled interrupts are set when an attempt to load data from an empty SHIFTBUF register occurs (buffer underrun). Clear the flag by writing 1.

72.3.1.2 Receive mode

When the assigned timer signals a store event in Receive mode (SHIFTCTL*n*[SMOD] = 001b), the shifter shifts and stores data in Shifter Buffer (SHIFTBUF0 - SHIFTBUF7). You can check for a start and stop bit before or after the shifter data is sampled by configuring either SHIFTCFG[SSTART] and TIMCFG[TSTART], or SHIFTCFG[SSTOP] and TIMCFG[TSTOP] in the shifter and timer.

The shifter status flag (SHIFTSTAT[SSF]) and any enabled interrupts or DMA requests are set when data is stored in the SHIFTBUF register from the shifter. To clear the flag, write 1 to or read the data from SHIFTBUF. Any read of the SHIFTBUF register clears the corresponding shifter status flag when the shifter is in Receive mode. The flag is cleared regardless of what is reading the register (DMA or interrupt) or the state of the DMA or interrupt enables. See the functional description of SHIFTSTAT[SSF] for information on how the flag is set or cleared for each mode.

The shifter error flag (SHIFTERR[SEF]) and any enabled interrupts are set either when an attempt to store data into a full SHIFTBUF register occurs (buffer overrun) or when a mismatch occurs on a start or stop bit check. Write 1 to clear the flag.

72.3.1.3 Match Store mode

In Match Store mode (SHIFTCTL*n*[SMOD] = 100b), the shifter shifts data in, checks for a match result, and stores matched data in Shifter Buffer (SHIFTBUF0 - SHIFTBUF7) when the assigned timer signals a store event. By configuring either SHIFTCFG[SSTART], TIMCFG[TSTART], and SHIFTCFG[SSTOP], or TIMCFG[TSTOP] in the shifter and timer, you can check for a start and stop bit before or after the shifter data is sampled. You can compare up to 16 bits of data using SHIFTBUF[31:16] to configure the data to be matched and SHIFTBUF[15:0] to mask the match result.

The shifter status flag (SHIFTSTAT[SSF]) and any enabled interrupts or DMA requests are set when a match occurs and the matched data is stored in the SHIFTBUF register from the shifter. To clear the flag, read the matched data from the SHIFTBUF register or write 1 to the flag. Any read of the SHIFTBUF register clears the corresponding shifter status flag when the shifter is configured in Match Store mode. The flag is cleared regardless of what is reading the register (DMA or interrupt) or the state of the DMA or interrupt enables. See the functional description for SHIFTSTAT[SSF] to know how the flag is set or cleared for each mode.

The shifter error flag (SHIFTERR[SEF]) and any enabled interrupts are set when an attempt to store matched data into a full SHIFTBUF register occurs (buffer overrun), or when a mismatch occurs on a start or stop bit check. Write 1 to clear the flag.

72.3.1.4 Match Continuous mode

In Match Continuous mode (SHIFTCTL*n*[SMOD] = 101b), the shifter shifts data in and continuously checks for a match result whenever a shift event is signaled by the assigned timer. You can compare up to 16 bits of data using SHIFTBUF[31:16] to configure the data to be matched and SHIFTBUF[15:0] to mask the match result.

The shifter status flag (SHIFTSTAT[SSF]) and any enabled interrupts or DMA requests are set when a match occurs. The flag clears automatically as soon as no match exists between the shifter data and Shifter Buffer (SHIFTBUF0 - SHIFTBUF7).

You cannot clear the flag by reading the SHIFTBUF register.

The shifter error flag (SHIFTERR[SEF]) and any enabled interrupts are set when a match occurs. To clear the flag, write 1 or perform a read from the SHIFTBUF register.

72.3.1.5 State mode

State mode enables you to implement any state machine with up to eight states, eight outputs, and three selectable inputs per state. This feature allows basic control functions to be offloaded from the CPU.

In State mode (SHIFTCTL*n*[SMOD] = 110b), when the shifter is selected by the current state pointer (SHIFTSTATE[STATE]), use the SHIFTBUF register to drive the output and compute the next state values. The following figure provides a detailed view of the shifter microarchitecture when configured for State mode.

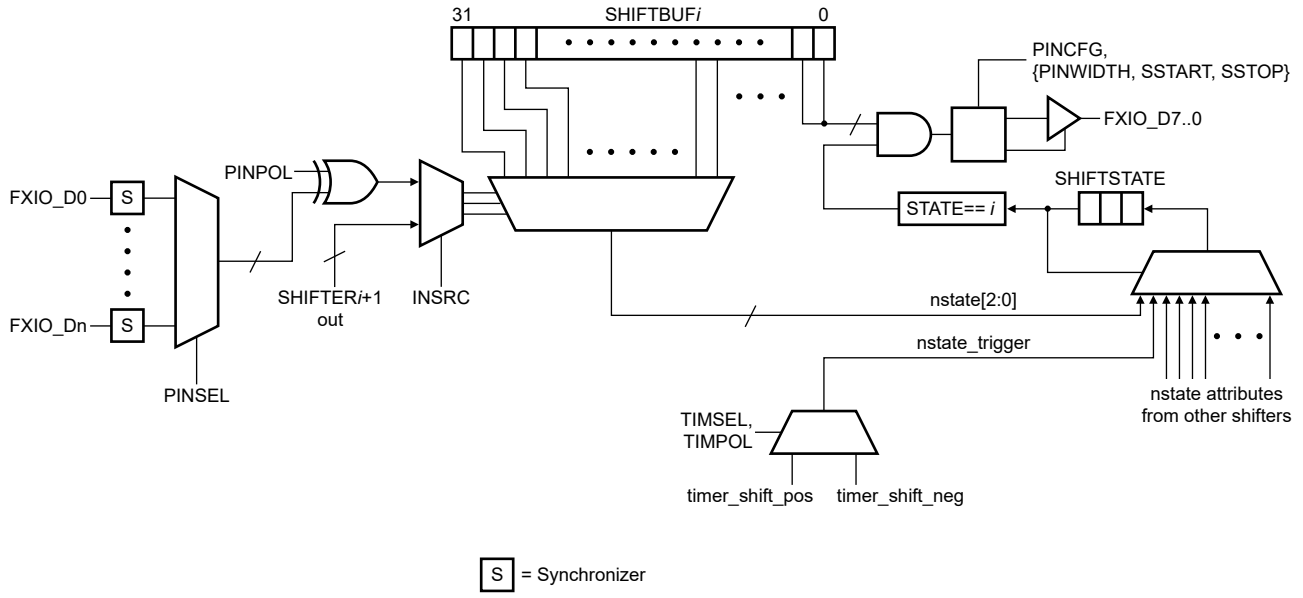


Figure 396. State microarchitecture

When the current state pointer selects a specific shifter (shifter *n*), output pins FXIO_D[7:0] are driven by SHIFTBUF*n*[31:24]; the configuration is defined by SHIFTCTL*n*[PINCFG]. Write 1 to Shifter Configuration (SHIFTCFG0 - SHIFTCFG7) {PWIDTH[3:0],SSTOP[1:0],SSTART[1:0]} to disable the output drive on pins FXIO_D[7:0] for state machine applications that require less than eight output pins.

Use the three input pins selected by SHIFTCTL*n*[PINSEL] and SHIFTBUF*n*[23:0] to compute the next state value.

NOTE

Each state can use a different set of three input pins.

The following table shows how the next state value is computed when the current state pointer is pointing to shifter *n*.

Table 468. Next state computation for SHIFTSTATE[STATE] = *n*

FXIO_D[PINSEL + 2]	FXIO_D[PINSEL + 1]	FXIO_D[PINSEL]	Next state value
0	0	0	SHIFTBUF*n*[2:0]
0	0	1	SHIFTBUF*n*[5:3]

Table continues on the next page...

Table 468. Next state computation for SHIFTSTATE[STATE] = *n* (continued)

FXIO_D[PINSEL + 2]	FXIO_D[PINSEL + 1]	FXIO_D[PINSEL]	Next state value
0	1	0	SHIFTBUF*n*[8:6]
0	1	1	SHIFTBUF*n*[11:9]
...	...	...	...
1	1	1	SHIFTBUF*n*[23:21]

NOTE

You can configure other shifters and timers to drive the input pins of a given state, allowing you to create complex combinations of shifters and timers as needed. For example, the output of a shifter configured for Logic mode can be used to drive a state machine input.

The next state transition is triggered using the timer output selected by SHIFTCTL*n*[TIMSEL], with polarity controlled by SHIFTCTL*n*[TIMPOL].

NOTE

Each state can use a different timer to trigger each next state transition, allowing various internal or external trigger sources and clocking configurations to be used. See Timer section for more information.

The current state pointer defaults to shifter 0 at reset; however, you can write to select a different shifter for the initial state. If the current state pointer selects a shifter that is not configured for State mode, then outputs are not driven and the next state transition is never triggered.

The shifter status flag (SHIFTSTAT[SSF]) and any enabled interrupts or DMA requests are set when the shifter is selected by the current state pointer. The flag is cleared when the current state pointer is updated to a different shifter.

72.3.1.6 Logic mode

Logic mode enables you to implement a small amount of programmable digital logic within a FLEXIO shifter.

In Logic mode (SHIFTCTL*n*[SMOD] = 111b), use Shifter Buffer (SHIFTBUF0 - SHIFTBUF7) to implement a 5-input, 32-bit programmable logic lookup table. The following figure provides a detailed view of shifter microarchitecture when configured for Logic mode.

Use the SHIFTBUF register to configure the lookup table for the four pin inputs. You can also use SHIFTER*n* to configure a feedback or delayed pin source as the fifth input to the lookup table.

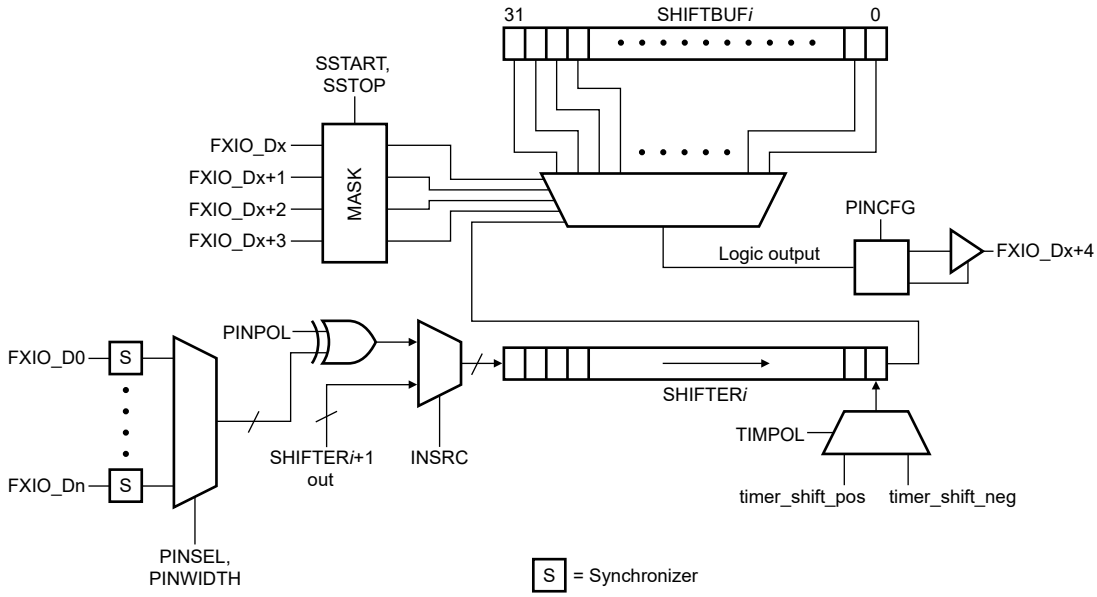


Figure 397. Logic microarchitecture

The lookup table is driven using four pin inputs (maskable using SHIFTCFG[SSTOP] and SHIFTCFG[SSTART]), plus one input from the internal shifter. It can be configured to drive an output pin using SHIFTCTL[PINCFG]. Pin inputs and outputs are fixed for each logic lookup table and are not selectable. The following table lists the logic output value selected by the lookup table for shifter *n*.

Table 469. Logic lookup table for shifter *n*

SHIFTER*n*[0]	FXIO_D[*x* + 3][1]	FXIO_D[*x* + 2]	FXIO_D[*x* + 1]	FXIO_D[*x*]	Logic output to FXIO_D[*x* + 4]
0	0	0	0	0	SHIFTBUF*n*[0]
0	0	0	0	1	SHIFTBUF*n*[1]
0	0	0	1	0	SHIFTBUF*n*[2]
0	0	0	1	1	SHIFTBUF*n*[3]
...	...	...	...	...	...
1	1	1	1	1	SHIFTBUF*n*[31]

1. for shifters *n* = 0...3, *x* = *n*

   for shifter *n* = 4...7, *x* = *n* + 4

To minimize output glitches, use SHIFTCFG*n*[SSTOP] and SHIFTCFG*n*[SSTART] to mask unused input pins. When these fields are 1, {SSTOP[1:0] and SSTART[1:0]} mask FXIO_D[*x* + 3]...FXIO_D[*x*] inputs respectively so that any transitions on these pins do not cause the logic output to glitch.

NOTE

You can configure other shifters and timers to drive the input pins of a given lookup table, allowing you to concatenate lookup tables or create complex combinations of shifters and timers as needed.

SHIFTCFG[PWIDTH] controls the number of delay stages introduced by the internal shifter input (SHIFTER*n*[0]). For example, when configured for a 1-bit shift (SHIFTCFG[PWIDTH] = 0), the internal shifter introduces a 32-shift clock delay before passing its

input (selected by SHIFTCTL[PINSEL]) to the lookup table. When configured for a 32-bit shift (SHIFTCFG[PWIDTH] = 16...31), the internal shifter introduces a 1-shift clock delay to its input.

The shifter status flag (SHIFTSTAT[SSF]) and any enabled interrupts or DMA requests are set whenever the output pin allocated to the logic lookup table has a value of 1 after being synchronized with the FLEXIO clock. The flag clears when the output pin has a value of 0. This also allows SHIFTSTAT[SSF] to be used as a trigger to a timer if needed.

The shifter error flag (SHIFTERR[SEF]) and any enabled interrupts are set when the output pin allocated to the logic lookup table is asserted. Clear the flag by writing 1 to it.

The Logic mode input pins, including pins driven by other shifters and timers, are synchronized with the FLEXIO functional clock before they are input to the programmable logic lookup table.

72.3.2 Timer operation

The FLEXIO 16-bit timers control the loading, shifting, and storing of the shift registers. The counters load the contents of the compare register and decrement down to zero on the FLEXIO clock. The counters can perform generic timer functions such as generating a clock, select output, or a PWM waveform. You can configure these timers to perform any of the following functions:

- Enable in response to a trigger, pin, or shifter condition.
- Decrement always or only on a trigger or pin edge.
- Reset in response to a trigger or pin condition.
- Disable on a trigger or pin condition or on a timer compare.

Timers can optionally include a start condition and a stop condition.

Although each timer operates independently, you can configure a timer to enable or disable at the same time as the previous timer (for example, timer 1 can enable or disable at the same time as timer 0) and a timer output can be used to trigger any other timer. The trigger used by each timer is configured independently as a timer output, shifter status flag, pin input, or an external trigger input. The trigger configuration is separate from pin configuration; you can perform it to configure input, output data, or output enable. See the chip-specific FLEXIO information for information on external trigger connections.

You must configure Timer Configuration (TIMCFG0 - TIMCFG7) before writing 1 to TIMCTL*n*[TIMOD].

72.3.2.1 Timer 8-bit Baud Counter mode

In 8-bit Baud Counter mode, the 16-bit counter is divided into two 8-bit counters. The lower 8 bits are used to configure the baud rate of the shift clock and the upper 8 bits are used to configure the number of shift clock edges in the transfer. When the lower 8 bits decrement to zero, the timer output is toggled and the lower 8 bits reload from the compare register. The upper 8 bits decrement when the lower 8 bits become zero and decrement.

NOTE

A timer reset event in 8-bit Baud Counter mode only resets the lower 8-bit counter. The upper 8-bit counter is not affected and can decrement if the timer reset is configured to update the state of the timer output, which toggles as a result of the timer reset event.

A timer compare event occurs when the upper 8 bits equal zero and decrement. The timer status flag is set on a timer compare event.

72.3.2.2 Timer 8-bit High PWM mode

In 8-bit High PWM mode, the 16-bit counter is divided into two 8-bit counters. The lower 8 bits are used to configure the timer output high period and the upper 8 bits are used to configure the timer output low period. The lower 8 bits decrement when the output is high. When the lower 8 bits become zero and decrement, the timer output is cleared and the lower 8 bits are reloaded from the compare register. The upper 8 bits decrement when the output is low. When the upper 8 bits become zero and decrement, the timer output is set and the upper 8 bits are reloaded from the compare register.

A timer compare event occurs when the upper 8 bits become zero and decrement. The timer status flag is set on a timer compare event.

72.3.2.3 Timer 16-bit Counter mode

In 16-bit Counter mode, you can use the 16-bit counter to configure either the baud rate of the shift clock (for example, TIMDEC[1:0] ≠ 10 or 11) or the number of shift clock edges in the transfer (for example, TIMDEC[1:0] = 10 or 11). When the 16-bit counter equals zero and decrements, the timer output toggles and the counter reloads from the compare register.

A timer compare event occurs when the 16-bit counter equals zero and decrements. The timer status flag is set on a timer compare event.

72.3.2.4 Timer 16-bit Counter Disable mode

In 16-bit Counter Disable mode, the 16-bit counter can be used to configure either the baud rate of the shift clock (for example, TIMDEC[1:0] ≠ 10 or 11) or the number of shift clock edges in the transfer (for example, TIMDEC[1:0] = 10 or 11). When the 16-bit counter equals zero and decrements, the timer output toggles and the counter reloads from the compare register.

A timer compare event occurs when the 16-bit counter equals zero and decrements. The timer status flag is set on a timer disable event.

72.3.2.5 Timer 8-bit Word Counter mode

In 8-bit Word Counter mode, the 16-bit counter is divided into two 8-bit counters. The lower 8 bits are used to configure the number of shift clock edges in each word and the upper 8 bits are used to configure the number of words in the transfer. When the lower 8 bits decrement to zero, the timer output is toggled and the lower 8 bits reload from the compare register. The upper 8 bits only decrement when the lower 8 bits become zero and decrement.

A timer compare event occurs when the lower 8 bits become zero and decrement. The timer status flag is set when the upper 8 bits become zero and decrement.

72.3.2.6 Timer 8-bit Low PWM mode

In 8-bit Low PWM mode, the 16-bit counter is divided into two 8-bit counters. The lower 8 bits are used to configure the timer output low period and the upper 8 bits are used to configure the timer output high period. The lower 8 bits decrement when the output is low. When the lower 8 bits become zero and decrement, the timer output is set and the lower 8 bits are reloaded from the compare register. The upper 8 bits decrement when the output is high. When the upper 8 bits become zero and decrement, the timer output is cleared and the upper 8 bits are reloaded from the compare register.

A timer compare event occurs when the upper 8 bits become zero and decrement. The timer status flag is set on a timer compare event.

72.3.2.7 Timer enable and start functions

The following events occur when you configure TIMCTL*n*[TIMOD] for the desired mode and the condition configured by the timer enable (TIMCFG*n*[TIMENA]) is detected. When TIMCTL*n*[ONETIM] is 1, the timer status flag must be clear to generate a timer enable event; otherwise, the timer enable event is blocked. You can use this to enforce software intervention after each timer iteration:

- The timer counter loads the current value of the compare register and starts decrementing, as configured by TIMCFG*n*[TIMDEC].
- The timer output may update to its initial state depending on the configuration of TIMCFG*n*[TIMOUT]. Shifters that are controlled by this timer do not see this as a rising edge on the timer shift clock.
- Transmit shifters controlled by this timer either output their start bit value or load the shift register from the shift buffer and output the first bit, as configured by SHIFTCFG*n*[SSTART].

If the timer start bit is enabled, the timer counter reloads with the compare register on the first rising edge of the shift clock after the timer starts decrementing. If there is no falling edge on the shift clock before the first rising edge (for example, when TIMCFG*n*[TIMOUT] = 1), a shifter that is configured to shift on the falling edge and load on the first shift does not load correctly.

72.3.2.8 Timer decrement and reset functions

The timer generates the timer output and timer shift clock depending on the fields, TIMCTL*n*[TIMOD] and TIMCFG*n*[TIMDEC]. The shifter clock is either equal to the timer output (when TIMCFG*n*[TIMDEC] ≠ 10 or 11) or equal to the decrement clock (when TIMCFG*n*[TIMDEC] = 10 or 11). If you configure TIMCFG*n*[TIMDEC] to decrement from a pin or trigger, the timer decrements on both rising and falling edges.

If a timer is configured to decrement on the FLEXIO functional clock divided by 16 or 256 (when TIMCFG*n*[TIMDEC] = 100 or 101), then a common prescaler that is shared by all timers is used to generate the two divide ratios. This prescaler is reset when all timers are either idle or configured not to use the prescaler (TIMCFG*n*[TIMDEC] ≠ 100 or 101).

If you configure the timer to reset as determined by TIMCFG*n*[TIMRST], then the timer counter loads the current value of the compare register again. You can configure the timer output and timer shift clock to update on timer reset, as configured by TIMCFG*n*[TIMOUT]. If the time output toggles as a result of the timer reset, this can result in a timer shift clock edge. In 8-bit Baud Counter mode, this also decrements the upper 8 bits of the counter.

In general, when the timer counter decrements to zero, a timer compare event is triggered. The timer compare event causes:

- The timer counter to load the contents of the timer compare register.
- The timer output to toggle.
- Any configured transmit shift registers to load.
- Any configured receive shift registers to store.

Depending on the timer mode, the timer status flag may also be set.

72.3.2.9 Timer disable and stop functions

When the timer is configured to add a stop bit on each compare, the following additional events occur:

- Transmit shifters controlled by this timer output their stop bit value (if configured by SHIFTCFG*n*[SSTOP]).
- Receive shifters controlled by this timer store the contents of the shift register in their shift buffer, as configured by SHIFTCFG*n*[SSTOP].
- The timer counter reloads the current value of the compare register on the first rising edge of the shifter clock after the compare.

If you configure the timer to insert a stop bit on each compare, you must configure the transmit shifters to load on the first shift.

When the condition configured by timer disable (TIMCFG*n*[TIMDIS]) is detected, the following events occur:

- Timer counter reloads the current value of the compare register and starts decrementing as configured by TIMCFG*n*[TIMDEC].
- Timer output clears. Shifters that are controlled by this timer do not see this as a falling edge on the timer shift clock, but can generate a shift event if the timer shift clock otherwise generates one.
- Transmit shifters controlled by this timer output their stop bit value (if configured by SHIFTCFG*n*[SSTOP]).
- Receive shifters controlled by this timer store the contents of the shift register in their shift buffer, as configured by SHIFTCFG*n*[SSTOP].

If the timer stop bit is enabled, the timer counter continues decrementing until the next rising edge of the shift clock is detected, at which point it finishes decrementing. Although the timer output is forced low during the stop bit, the timer shift clock can toggle during the stop bit. The timer output does not generate shift events during the stop bit.

A timer enable condition can be detected in the same cycle as a timer disable condition (if timer stop bit is disabled), or on the first rising edge of the shift clock after the disable condition (if stop bit is enabled). When TIMCTL*n*[ONETIM] is 1, the timer status flag must be clear before the next timer enable condition is detected. When the timer is in the stop state condition, receive shift registers with stop bit enabled store the contents of the shift register into the shift buffer and verify the state of the input data on the configured shift edge. If there is no configured edge between the timer disable and the next rising edge of the shift clock, then the final store and verify do not occur.

72.3.3 Pin operation

The pin configuration for each timer and shifter can be set to use any FLEXIO pin with either polarity. You can configure each timer and shifter as an input, output data, output enable, or bidirectional output. A pin configured for output enable can be used as an open drain (with inverted polarity because the output enable assertion causes logic zero to be output on the pin) or to control the enable on the bidirectional output. You can configure any timer or shifter to control the output enable for a pin where the bidirectional output data is driven by another timer or shifter.

When more than one shifter or timer is configured to use the same pin for output data, then the output data to the pin is combined using an OR function. This also applies if more than one timer or shifter is configured to use the same pin for output enable.

72.3.3.1 Parallel interface

You can configure shifters to use multiple FLEXIO pins simultaneously by using SHIFTCFG*n*[PWIDTH], which is used to configure the following settings of a shifter:

1. Number of bits shifted per shift clock.
2. Number of pins driven by the shifter per shift clock (only on shifters supporting parallel transmit—that is, SHIFTER0 and SHIFTER4.)
3. Number of pins sampled by the shifter per shift clock (only on shifters supporting parallel receive—that is, SHIFTER3 and SHIFTER7.)

When configured for parallel shift, either 4, 8, 16, or 32 bits can be shifted on every shift clock. If an adjacent shifter is selected as the input source (SHIFTCFG*n*[INSRC] = 1), the least significant 4, 8, 16, or 32 bits from the adjacent shifter are sampled on each shift clock.

For shifters supporting parallel receive (SHIFTER3, SHIFTER7), you can configure the shifter to sample multiple pins (INSRC = 0), with PWIDTH and PINSEL selecting the pins as FXIO_D[PINSEL+PWIDTH]:FXIO_D[PINSEL].

NOTE

If PWIDTH is less than the number of bits being shifted on each shift clock, then the most significant bits are masked with 0. For example, if PINSEL = 7 and PWIDTH = 6, then SHIFTER[31:24] samples {0,0,FXIO_D[12:7]} on each shift clock.

For shifters supporting parallel transmit (SHIFTER0, SHIFTER4), you can configure the shifter to drive multiple pins using SHIFTCTL*n*[PINCFG], with PWIDTH and PINSEL selecting the pins as follows: FXIO_D[PINSEL+PWIDTH]:FXIO_D[PINSEL].

NOTE

If PWIDTH is less than the number of bits being shifted on each shift clock, then the most significant pins are not driven. For example, if PINSEL = 7 and PWIDTH = 6, then SHIFTER[5:0] drives only FXIO_D[12:7] on each shift clock.

72.3.3.2 Pin synchronization

When you configure a pin as an input (this includes a timer trigger configured as a pin input), the input signal is first synchronized with the FLEXIO clock before a timer or shifter could use the signal. This introduces a small latency of 0.5–1.5 FLEXIO clock cycles when using an external pin input to generate an output or control a shifter. This sets the maximum setup time at 1.5 FLEXIO clock cycles.

If an input is used by more than one timer or shifter, then the synchronization occurs once to ensure any edge is seen on the same cycle by all timers and shifters using that input.

NOTE

FLEXIO pins are also connected internally. Configuring a FLEXIO shifter or timer to output data on an unused pin makes an internal connection that allows other shifters and timers to use this pin as an input. This allows a shifter output to trigger a timer or a timer output to be shifted into a shifter. This path is also synchronized with the FLEXIO clock and therefore incurs a one-cycle latency.

When using a pin input as a timer trigger, timer clock, or shifter data input, the following synchronization delays occur:

- 0.5–1.5 FLEXIO clock cycles for an external pin
- One FLEXIO clock cycle for an internally driven pin

See Application information for timing considerations such as output valid time and input setup time for specific applications (SPI controller, SPI target, I2C controller, I2S controller, and I2S target).

72.3.3.3 Pin override

You can change the state of any FLEXIO pin at any time. Pin Output Enable (PINOUTE) configures any pin as an output and drives that pin with the value in Pin Output Data (PINOUTD).

Alias registers for PINOUTE and data registers also exist. Writing a logic 1 to an alias register updates the corresponding register fields in both PINOUTE and PINOUTD as follows:

- Pin Output Disable (PINOUTDIS) clears Pin Output Enable (PINOUTE) and Pin Output Data (PINOUTD).
- Pin Output Clear (PINOUTCLR) sets Pin Output Enable (PINOUTE) and clears Pin Output Data (PINOUTD).
- Pin Output Set (PINOUTSET) sets Pin Output Enable (PINOUTE) and Pin Output Data (PINOUTD).
- Pin Output Toggle (PINOUTTOG) sets Pin Output Enable (PINOUTE) and toggles Pin Output Data (PINOUTD).

72.3.3.4 Pin interrupt

You can read the state of any FLEXIO pin at any time and also configure any pin to set a status flag when either a rising or falling edge is detected on that pin. Additionally, you can configure the pin status flag to generate an interrupt.

72.3.4 Low-power modes

FLEXIO remains functional during low-power modes, if the FLEXIO functional clock remains enabled.

72.3.5 Debug mode

FLEXIO remains functional in Debug mode, provided the value of CTRL[DBGE] is 1.

72.3.6 Clocking

Table 470. FLEXIO clocks

Clock	Description
Functional clock	Is asynchronous to the bus clock and can remain enabled in low-power modes. You must enable the FLEXIO functional clock before accessing any of the FLEXIO registers. Provided the FLEXIO functional clock is at least equal to the bus clock, you can configure CTRL[FASTACC] to support fast register accesses.
Bus clock	Is used only for bus accesses to the control and configuration registers.

72.3.7 Reset

Table 471. FLEXIO reset types

Reset	Description
Chip reset	Resets the FLEXIO logic and registers to their default states on chip reset.
Software reset	Resets, using CTRL[SWRST], all logic and registers to their default states, except for the Control register.

72.3.8 Interrupts and DMA requests

The following table shows the status flags that generate the FLEXIO interrupt and DMA requests.

Table 472. FLEXIO interrupts and DMA requests

Flag	Description	Interrupt	DMA request	Low-power wake-up
SHIFTSTAT[SSF]	Shifter status flag	Y	Y	Y
SHIFTERR[SEF]	Shifter error flag	Y	N	Y
TIMSTAT[TSF]	Timer status flag	Y	Y	Y
PINSTAT[PSF]	Pin status flag	Y	N	Y
TRGSTAT[ETSF]	External trigger status flag	Y	N	Y

72.3.9 Peripheral triggers

The connection between FLEXIO peripheral triggers and other peripherals is device-specific.

72.3.9.1 Output triggers

Each FLEXIO timer generates an output trigger equal to the timer output. The output trigger is not affected by the timer pin polarity configuration.

72.3.9.2 Input trigger

FLEXIO supports multiple external trigger inputs that can be used to trigger one or more FLEXIO timers. If a rising edge is detected on an external trigger when FLEXIO is enabled, then the external trigger status flag is set. The external triggers are synchronized to the FLEXIO functional clock and must assert for at least two cycles of the FLEXIO functional clock to be sampled correctly.

72.4 External signals

Table 473. External signals

Signal	Description	Direction
FXIO_D*n* (*n* = 0...31)	Bidirectional FLEXIO shifter and timer pin	Input or output

72.5 Initialization

Perform the following procedure to initialize FLEXIO registers:

1. Enable FLEXIO by writing 1 to CTRL[FLEXEN].
2. Configure shift registers for the given application. It is recommended to write to Shifter Configuration (SHIFTCFG0 - SHIFTCFG7) before writing to the corresponding register, Shifter Control (SHIFTCTL0 - SHIFTCTL7).
3. Configure timer registers for the given application. It is recommended to write to Timer Compare (TIMCMP0 - TIMCMP7) and Timer Configuration (TIMCFG0 - TIMCFG7) before writing to the corresponding register, Timer Control (TIMCTL0 - TIMCTL7).
4. Enable interrupts and/or DMA requests, as appropriate, for the given application.
5. Write transmit data to initiate a transfer (depending on the given application).

72.6 Application information

This section provides examples for a variety of FLEXIO module applications. See FLEXIO register descriptions for more information.

72.6.1 UART transmit

UART transmit can be supported using one timer, one shifter, and one pin (two pins, if supporting CTS). The start and stop bit insertion is handled automatically, and multiple transfers are supported using the DMA controller. The timer status flag is used to indicate when the stop bit of each word is transmitted.

Break and idle characters require software intervention. Before transmitting a break or idle character, you must modify SHIFTCFG*n*[SSTART] and SHIFTCFG*n*[SSTOP] to transmit the required state, and the data to transmit must equal FFh or 00h. Supporting a second stop bit requires the stop bit to be inserted into the data stream using software (and increasing the number of bits to transmit). When performing byte writes to Shifter Buffer (SHIFTBUF0 - SHIFTBUF7) (or Shifter Buffer Bit Swapped (SHIFTBUFBIS0 - SHIFTBUFBIS7) for transmitting MSB first), the rest of the register remains unaltered. This allows an address mark bit or additional stop bit to remain undisturbed.

NOTE

FLEXIO does not support automatic insertion of parity bits.

Table 474. UART transmit configuration

Register	Value	Configuration
SHIFTCFG*n*	0000_0032h	Configure start bit of 0 and stop bit of 1.
SHIFTCTL*n*	0003_0002h	Configure transmit using timer 0 on the positive edge of clock with output data on pin 0. You can configure the PINPOL field to invert output data, or support open-drain by writing 1h to the PINPOL and PINCFG fields.
TIMCMP*n*	0000_0F01h	Configure 8-bit transfer with baud rate of divide by 4 of the FLEXIO clock. Set TIMCMP[15:8] as (number of bits × 2) - 1, and set TIMCMP[7:0] as (baud rate divider ÷ 2) - 1.
TIMCFG*n*	0000_2222h	Configure start bit, stop bit, enable on trigger asserted and disable on compare. You can support CTS by configuring the TIMENA field as 3h.
TIMCTL*n*	01C0_0001h	Configure the dual 8-bit counter using the shifter 0 status flag as an inverted internal trigger source. To support CTS, configure the PINSEL (for pin 1) and PINPOL fields as 1h.
SHIFTBUF*n*	Data to transmit	Transmit data can be written to SHIFTBUF[7:0] to initiate an 8-bit transfer. Use the shifter status flag to indicate when data can be written using an interrupt or a DMA request. Write to SHIFTBUFBBS[7:0] instead to support MSB first transfer.

The following table shows an alternative configuration that supports slower baud rates. This configuration requires two timers.

Table 475. UART transmit configuration for slow baud rate

Register	Value	Configuration
SHIFTCFG*n*	0000_0032h	Configure start bit of 0 and stop bit of 1.
SHIFTCTL*n*	0003_0002h	Configure transmit using timer 0 on the positive edge of clock with output data on pin 0. Invert output data by writing 1 to the PINPOL field. Support open-drain by configuring the PINPOL and PINCFG fields as 1h.
TIMCMP*n*	0000_000Fh	Configure for 8-bit transfer, and configure TIMCMP[15:0] as (number of bits × 2) - 1.
TIMCFG*n*	0030_2622h	Configure start bit, stop bit, enable on trigger rising edge, decrement on trigger and disable on compare.
TIMCTL*n*	0740_0003h	Configure the 16-bit counter using the timer 1 output as an internal trigger source.
TIMCMP(*n* + 1)	0000_0001h	Configure baud rate of divide by 4 of the FLEXIO clock, and configure TIMCMP[15:0] as (baud rate divider ÷ 2) - 1.
TIMCFG(*n* + 1)	0000_1200h	Configure enable on trigger asserted and disable on timer 0 disable. You can configure the TIMEN field as 3h to support CTS.
TIMCTL(*n* + 1)	01C0_0003h	Configure the 16-bit counter using the shifter 0 status flag as an inverted internal trigger source. You can support CTS by configuring the PINSEL (for pin 1) and PINPOL fields as 1h.
SHIFTBUF*n*	Data to transmit	Transmit data can be written to SHIFTBUF[7:0] to initiate an 8-bit transfer. Use the shifter status flag to indicate when data can be written using an interrupt or a DMA request. Write to SHIFTBUFBBS[7:0] instead to support MSB first transfer.

72.6.2 UART receive

UART receive can be supported using one timer, one shifter, and one pin (two timers and two pins, if supporting RTS). The start and stop bit verification is handled automatically and multiple transfers are supported using the DMA controller. The timer status flag is used to indicate when the stop bit of each word is received.

FLEXIO does not support triple voting of the received data, which is sampled only once in the middle of each bit. You can use a timer to implement a glitch filter on the incoming data and a different timer to detect an idle line of programmable length. Break characters cause the error flag to set, and the shifter buffer register returns 00h.

NOTE

FLEXIO does not support automatic verification of parity bits.

Table 476. UART receiver configuration

Register	Value	Configuration
SHIFTCFG*n*	0000_0032h	Configure start bit of 0 and stop bit of 1.
SHIFTCTL*n*	0080_0001h	Configure receive using timer 0 on the negative edge of clock with input data on pin 0. You can invert input data by writing 1 to the PINPOL field.
TIMCMP*n*	0000_0F01h	Configure 8-bit transfer with baud rate of divide by 4 of the FLEXIO clock. Set TIMCMP[15:8] as (number of bits × 2) - 1, and set TIMCMP[7:0] as (baud rate divider ÷ 2) - 1.
TIMCFG*n*	0204_2422h	Configure start bit, stop bit, enable on pin positive edge and disable on compare. Enable resynchronization to received data with TIMOUT = 2h and TIMRST = 4h.
TIMCTL*n*	0000_0081h	Configure the dual 8-bit counter using the inverted pin 0 input.
SHIFTBUF*n*	Data to receive	You can read received data from SHIFTBUFBYS[7:0]. Use the shifter status flag to indicate when data can be read using interrupt or DMA request. Read from SHIFTBUFBIS[7:0] instead to support MSB first transfer.

The UART receiver with RTS configuration uses a second timer to generate the RTS output. RTS asserts when the start bit is detected and negates when the data is read from the shifter buffer register. If no start bit is detected when the RTS is asserted, the received data is ignored.

Table 477. UART receiver with RTS configuration

Register	Value	Configuration
SHIFTCFG*n*	0000_0032h	Configure start bit of 0 and stop bit of 1.
SHIFTCTL*n*	0080_0001h	Configure receive using timer 0 on the negative edge of clock with input data on pin 0. Invert input data by writing 1 to the PINPOL field.
TIMCMP*n*	0000_0F01h	Configure 8-bit transfer with baud rate of divide by 4 of the FLEXIO clock. Set TIMCMP[15:8] as (number of bits × 2) - 1, and set TIMCMP[7:0] as (baud rate divider ÷ 2) - 1.
TIMCFG*n*	0204_2522h	Configure start bit, stop bit, enable on pin positive edge with trigger

Table continues on the next page...

Table 477. UART receiver with RTS configuration (continued)

Register	Value	Configuration
		asserted and disable on compare. Enable resynchronization to received data with TIMOUT = 2h and TIMRST = 4h.
TIMCTL*n*	02C0_0081h	Configure dual 8-bit counter using the inverted pin 0 input. Trigger is internal using the inverted pin 1 input.
TIMCMP(*n* + 1)	0000_FFFFh	Never compare.
TIMCFG(*n* + 1)	0030_6100h	Enable on timer *n* enable and disable on the trigger falling edge. Decrement on trigger to ensure no compare.
TIMCTL(*n* + 1)	0143_0003h	Configure 16-bit counter and output on pin 1. Trigger is internal using the shifter 0 flag.
SHIFTBUF*n*	Data to receive	You can read received data using SHIFTBUFBYS[7:0]. Use the shifter status flag to indicate when data can be read using interrupt or DMA request. Read from SHIFTBUFBIS[7:0] instead to support MSB first transfer.

72.6.3 SPI controller

SPI Controller mode can be supported using two timers, two shifters, and four pins. Using the DMA controller, either CPHA = 0 or CPHA = 1 and transfers can be supported. For CPHA = 1, the chip select can remain asserted for multiple transfers and the timer status flag can be used to indicate the end of the transfer.

The stop bit is used to guarantee a minimum of one clock cycle between the target chip select negating and before the next transfer. To initiate each transfer, either the core or DMA writes to the transmit buffer.

NOTE

Because of synchronization delays, the setup time for the serial input data is 1.5 FLEXIO clock cycles. This means the maximum baud rate is divide by 4 of the FLEXIO clock frequency.

Table 478. SPI controller (CPHA = 0) configuration

Register	Value	Configuration
SHIFTCFG*n*	0000_0000h	Start and stop bit disabled.
SHIFTCTL*n*	0083_0002h	Configure transmit using timer 0 on the negative edge of clock with output data on pin 0.
SHIFTCFG(*n* + 1)	0000_0000h	Start and stop bit disabled.
SHIFTCTL(*n* + 1)	0000_0101h	Configure receive using timer 0 on the positive edge of clock with input data on pin 1.

Table continues on the next page...

Table 478. SPI controller (CPHA = 0) configuration (continued)

Register	Value	Configuration
TIMCMP*n*	0000_3F01h	Configure 32-bit transfer with baud rate of divide by 4 of the FLEXIO clock. Set TIMCMP[15:8] as (number of bits × 2) - 1, and set TIMCMP[7:0] as (baud rate divider ÷ 2) - 1.
TIMCFG*n*	0100_2222h	Configure start bit, stop bit, enable on trigger high and disable on compare; initial clock state is logic 0.
TIMCTL*n*	01C3_0201h	Configure dual 8-bit counter using the pin 2 output (shift clock), with shifter 0 flag as the inverted trigger. Write 1 to the PINPOL field to invert the output shift clock.
TIMCMP(*n* + 1)	0000_FFFFh	Never compare.
TIMCFG(*n* + 1)	0000_1100h	Enable when timer 0 is enabled and disable when timer 0 is disabled.
TIMCTL(*n* + 1)	0003_0383h	Configure 16-bit counter (never compare) using the inverted pin 3 output as target select.
SHIFTBUF*n*	Data to transmit	You can write transmit data to Shifter Buffer (SHIFTBUF0 - SHIFTBUF7). Use the shifter status flag to indicate when data can be written using interrupt or DMA request. Write to Shifter Buffer Bit Swapped (SHIFTBUFBIS0 - SHIFTBUFBIS7) instead to support MSB first transfer.
SHIFTBUF(*n* + 1)	Data to receive	Received data can be read from Shifter Buffer (SHIFTBUF0 - SHIFTBUF7). Use the shifter status flag to indicate when data can be read using interrupt or DMA request. Read from Shifter Buffer Bit Swapped (SHIFTBUFBIS0 - SHIFTBUFBIS7) instead to support MSB first transfer.

Table 479. SPI controller (CPHA = 1) configuration

Register	Value	Configuration
SHIFTCFG*n*	0000_0021h	Start bit loads data on first shift.
SHIFTCTL*n*	0003_0002h	Configure transmit using timer 0 on the positive edge of clock with output data on pin 0.
SHIFTCFG(*n* + 1)	0000_0000h	Start and stop bit disabled.

Table continues on the next page...

Table 479. SPI controller (CPHA = 1) configuration (continued)

Register	Value	Configuration
SHIFTCTL(*n* + 1)	0080_0101h	Configure receive using timer 0 on the negative edge of clock with input data on pin 1.
TIMCMP*n*	0000_3F01h	Configure 32-bit transfer with baud rate of divide by 4 of the FLEXIO clock. Set TIMCMP[15:8] as (number of bits x 2) - 1, and set TIMCMP[7:0] as (baud rate divider ÷ 2) - 1.
TIMCFG*n*	0100_2222h	Configure start bit, stop bit, enable on trigger high and disable on compare; initial clock state is logic 0.
TIMCTL*n*	01C3_0201h	Configure dual 8-bit counter using pin 2 output (shift clock), with the shifter 0 flag as the inverted trigger. Write 1 to the PINPOL field to invert the output shift clock, and set the TIMDIS field as 3 to keep target select asserted for as long as there is data in the transmit buffer.
TIMCMP(*n* + 1)	0000_FFFFh	Never compare.
TIMCFG(*n* + 1)	0000_1100h	Enable when timer 0 is enabled and disable when timer 0 is disabled.
TIMCTL(*n* + 1)	0003_0383h	Configure 16-bit counter (never compare) using inverted pin 3 output (as target select).
SHIFTBUF*n*	Data to transmit	Transmit data can be written to SHIFTBUF. Use the shifter status flag to indicate when data can be written using interrupt or DMA request. Write to Shifter Buffer Bit Swapped (SHIFTBUFBIS0 - SHIFTBUFBIS7) instead to support MSB first transfer.
SHIFTBUF(*n* + 1)	Data to receive	Received data can be read from Shifter Buffer (SHIFTBUF0 - SHIFTBUF7). Use the shifter status flag to indicate when data can be read using interrupt or DMA request. Read from Shifter Buffer Bit Swapped (SHIFTBUFBIS0 - SHIFTBUFBIS7) instead to support MSB first transfer.

72.6.4 SPI target

SPI Target mode can be supported using one timer, two shifters, and four pins. Either CPHA = 0 or CPHA = 1 can be supported and transfers can be supported using the DMA controller. For CPHA = 1, the select can remain asserted for multiple transfers and the timer status flag can be used to indicate the end of the transfer.

You must write the transmit data to the transmit buffer register before the external target select asserts; otherwise, the shifter error flag is set.

NOTE

Because of synchronization delays, the output valid time for the serial output data is 2.5 FLEXIO clock cycles. This means the maximum baud rate is divide by 6 of the FLEXIO clock frequency.

Table 480. SPI target (CPHA = 0) configuration

Register	Value	Configuration
SHIFTCFG*n*	0000_0000h	Start and stop bit disabled.
SHIFTCTL*n*	0083_0002h	Configure transmit using timer 0 on the falling edge of shift clock with output data on pin 0.
SHIFTCFG(*n* + 1)	0000_0000h	Start and stop bit disabled.
SHIFTCTL(*n* + 1)	0000_0101h	Configure receive using timer 0 on the rising edge of shift clock with input data on pin 1.
TIMCMP*n*	0000_003Fh	Configure 32-bit transfer. Set TIMCMP[15:0] as (number of bits × 2) - 1.
TIMCFG*n*	0120_0600h	Configure enable on trigger rising edge. Initial clock state is logic 0 and decrements on pin input.
TIMCTL*n*	06C0_0203h	Configure 16-bit counter using pin 2 input (shift clock), with pin 3 input (target select) as the inverted trigger.
SHIFTBUF*n*	Data to transmit	Transmit data can be written to Shifter Buffer (SHIFTBUF0 - SHIFTBUF7). Use the shifter status flag to indicate when data can be written using interrupt or DMA request. Write to Shifter Buffer Bit Swapped (SHIFTBUFBIS0 - SHIFTBUFBIS7) instead to support MSB first transfer.
SHIFTBUF(*n* + 1)	Data to receive	Received data can be read from Shifter Buffer (SHIFTBUF0 - SHIFTBUF7). Use the shifter status flag to indicate when data can be read using interrupt or DMA request. Read from Shifter Buffer Bit Swapped (SHIFTBUFBIS0 - SHIFTBUFBIS7) instead to support MSB first transfer.

Table 481. SPI target (CPHA = 1) configuration

Register	Value	Configuration
SHIFTCFG*n*	0000_0001h	Shifter configured to load on first shift and stop bit disabled.
SHIFTCTL*n*	0003_0002h	Configure transmit using timer 0 on rising edge of shift clock with output data on pin 0.
SHIFTCFG(*n* + 1)	0000_0000h	Start and stop bit disabled.
SHIFTCTL(*n* + 1)	0080_0101h	Configure receive using timer 0 on falling edge of shift clock with input data on pin 1.
TIMCMP*n*	0000_003Fh	Configure 32-bit transfer. Set TIMCMP[15:0] as (number of bits × 2) - 1).
TIMCFG*n*	0120_6602h	Configure start bit, enable on trigger rising edge, disable on trigger falling edge. Initial clock state is logic 0 and decrements on pin input.
TIMCTL*n*	06C0_0203h	Configure 16-bit counter using pin 2 input (shift clock), with pin 3 input (target select) as the inverted trigger.
SHIFTBUF*n*	Data to transmit	Transmit data can be written to Shifter Buffer (SHIFTBUF0 - SHIFTBUF7). Use the shifter status flag to indicate when data can be written using interrupt or DMA request. Write to Shifter Buffer Bit Swapped (SHIFTBUFBIS0 - SHIFTBUFBIS7) instead to support MSB first transfer.
SHIFTBUF(*n* + 1)	Data to receive	Received data can be read from Shifter Buffer (SHIFTBUF0 - SHIFTBUF7). Use the shifter status flag to indicate when data can be read using interrupt or DMA request. Read from Shifter Buffer Bit Swapped (SHIFTBUFBIS0 - SHIFTBUFBIS7) instead to support MSB first transfer.

72.6.5 I2C controller

I2C Controller mode can be supported using two timers, two shifters, and two pins. One timer is used to generate the SCL output and another one is used to control the shifters. The two shifters that are used to transmit and receive for every word, when receiving the transmitter, must transmit FFh to 3-state the output. FLEXIO inserts a stop bit after every word to generate and verify the ACK or NACK. FLEXIO waits for the first write to the transmit data buffer before enabling SCL generation. Data transfers can be supported using the DMA controller and the shifter error flag sets on transmit underrun or receive overflow.

The first timer generates the bit clock for the entire packet (start to repeated start or stop condition), so you must program the compare register with the total number of clock edges in the packet (minus one). The timer supports clock stretching using the reset counter when pin is equal to output. However, this increases both the clock high and clock low periods by at least one FLEXIO

clock cycle each. The second timer uses the SCL input pin to control the transmit and receive shift registers. This enforces an SDA data hold time by an extra two FLEXIO clock cycles.

Both the transmit and receive shifters must be serviced for each word in the transfer. The transmit shifter must transmit FFh when receiving, and the receive shifter returns the data present on the SDA pin. The transmit shifter loads one additional word on the last falling edge of the SCL pin. When generating a stop condition or a repeated start condition, this word must be 00h and FFh, respectively. During the last word of a controller-receiver transfer, you must set the transmit SHIFTCFG*n*[SSTOP] field to generate a NACK.

The receive shift register asserts an error interrupt if a NACK is detected, but you are responsible for generating the stop or repeated start condition. If a NACK is detected during controller-transmit, the interrupt routine must immediately write 00h (when generating a stop condition) or FFh (when generating a repeated start condition) to the transmit shifter register. You must wait for the next rising edge on SCL before disabling both timers. The transmit shifter must be disabled after the setup delay for a repeated start or stop condition.

NOTE

Because of synchronization delays, the data valid time for the transmit output is two FLEXIO clock cycles. This means the maximum baud rate is divide by 6 of the FLEXIO clock frequency.

To guarantee SDA hold time, the I2C controller data valid is delayed by two cycles because the clock output is passed through a synchronizer before clocking the transmit or receive shifter. Because the SCL output is synchronous with FLEXIO clock, the synchronization delay is one cycle, and then an additional cycle is involved to generate the output.

Table 482. I2C controller configuration

Register	Value	Configuration
SHIFTCFG*n*	0000_0032h	Start bit enabled (logic 0) and stop bit enabled (logic 1).
SHIFTCTL*n*	0101_0082h	Configure transmit using timer 1 on the rising edge of clock with inverted output enable (open-drain output) on pin 0.
SHIFTCFG(*n* + 1)	0000_0020h	Start bit disabled and stop bit enabled (logic 0) for ACK or NACK detection.
SHIFTCTL(*n* + 1)	0180_0001h	Configure receive using timer 1 on the falling edge of clock with input data on pin 0.
TIMCMP*n*	0000_2501h	Configure 2 word transfer with baud rate of divide by 4 of the FLEXIO clock. Set TIMCMP[15:8] as (number of words × 18) + 1, and set TIMCMP[7:0] as (baud rate divider ÷ 2) - 1.
TIMCFG*n*	0102_2222h	Configure start bit, stop bit, enable on trigger high, disable on compare, reset if output equals pin. Initial clock state is logic 0 and is not affected by reset.
TIMCTL*n*	01C1_0101h	Configure dual 8-bit counter using pin 1 output enable (SCL open-drain), with the shifter 0 flag as the inverted trigger.
TIMCMP(*n* + 1)	0000_000Fh	Configure 8-bit transfer. Set TIMCMP[15:0] as (number of bits x 2) - 1.

Table continues on the next page...

Table 482. I2C controller configuration (continued)

Register	Value	Configuration
TIMCFG(*n* + 1)	0020_1112h	Enable when timer 0 is enabled; disable when timer 0 is disabled. Enable start bit and stop bit at the end of each word and decrement on pin input.
TIMCTL(*n* + 1)	01C0_0183h	Configure 16-bit counter using inverted pin 1 input (SCL).
SHIFTBUF*n*	Data to transmit	Transmit data can be written to SHIFTBUFBBS[7:0]. Use the shifter status flag to indicate when data can be written using interrupt or DMA request.
SHIFTBUF(*n* + 1)	Data to receive	Received data can be read from SHIFTBUFBIS[7:0]. Use the shifter status flag to indicate when data can be read using interrupt or DMA request.

72.6.6 I2S controller

I2S Controller mode can be supported using two timers, two shifters, and four pins. One timer is used to generate the bit clock and control the shifters and another timer is used to generate the frame sync. FLEXIO waits for the first write to the transmit data buffer before enabling bit clock and frame sync generation. Data transfers are supported using the DMA controller and the shifter error flag sets on transmit underrun or receive overflow.

The bit clock frequency is an even integer divide of the FLEXIO clock frequency. The initial frame sync assertion occurs at the same time as the first bit clock edge. The timer uses the start bit to ensure that the frame sync is generated one clock cycle before the first output data.

NOTE

Because of synchronization delays, the setup time for the receiver input is 1.5 FLEXIO clock cycles. This means that the maximum baud rate is divide by 4 of the FLEXIO clock frequency.

Table 483. I2S controller configuration

Register	Value	Configuration
SHIFTCFG*n*	0000_0001h	Load transmit data on first shift and stop bit disabled.
SHIFTCTL*n*	0003_0002h	Configure transmit using timer 0 on the rising edge of clock with output data on pin 0.
SHIFTCFG(*n* + 1)	0000_0000h	Start and stop bit disabled.
SHIFTCTL(*n* + 1)	0080_0101h	Configure receive using timer 0 on the falling edge of clock with input data on pin 1.
TIMCMP*n*	0000_3F01h	Configure 32-bit transfer with baud rate of divide by 4 of the FLEXIO clock. Set TIMCMP[15:8] as (number of bits × 2)

Table continues on the next page...

Table 483. I2S controller configuration (continued)

Register	Value	Configuration
		- 1, and set TIMCMP[7:0] as (baud rate divider ÷ 2) - 1.
TIMCFG*n*	0000_0202h	Configure start bit, enable on trigger high and never disable. Initial clock state is logic 1.
TIMCTL*n*	01C3_0281h	Configure dual 8-bit counter using inverted pin 2 output (bit clock), with shifter 0 flag as the inverted trigger. Write 0 to the PINPOL field to invert the polarity of the output shift clock.
TIMCMP(*n* + 1)	0000_007Fh	Configure 32-bit transfer with baud rate of divide by 4 of the FLEXIO clock. Set TIMCMP[15:0] as (number of bits × baud rate divider) ÷ 1.
TIMCFG(*n* + 1)	0000_0100h	Enable when timer 0 is enabled and never disable.
TIMCTL(*n* + 1)	0003_0383h	Configure 16-bit counter using inverted pin 3 output (as frame sync). Write 0 to the PINPOL field to invert the polarity of the output frame sync.
SHIFTBUF*n*	Data to transmit	Transmit data can be written to Shifter Buffer Bit Swapped (SHIFTBUFBIS0 - SHIFTBUFBIS7). Use the shifter status flag to indicate when data can be written using interrupt or DMA request. Write to Shifter Buffer (SHIFTBUF0 - SHIFTBUF7) instead to support LSB first transfer.
SHIFTBUF(*n* + 1)	Data to receive	Received data can be read from Shifter Buffer Bit Swapped (SHIFTBUFBIS0 - SHIFTBUFBIS7). Use the shifter status flag to indicate when data can be read using interrupt or DMA request. Read from Shifter Buffer (SHIFTBUF0 - SHIFTBUF7) instead to support LSB first transfer.

72.6.7 I2S target

I2S Target mode can be supported using three timers, two shifters, and four pins. For single transmit and single receive, other combinations of transmit and receive are possible.

The transmit data must be written to the transmit buffer register before the external frame sync asserts, otherwise the shifter error flag is set.

NOTE

Because of synchronization delays, the output valid time for the serial output data is 2.5 FLEXIO clock cycles. This means the maximum baud rate is divide by 6 of the FLEXIO clock frequency.

The output valid time of I2S target is maximum 2.5 cycles because there is a maximum 1.5 cycle delay on the clock synchronization, plus one cycle to output the data.

Timer 2 detects the falling edge of frame sync (start of new frame) and asserts output until the rising edge of bit clock (when the frame sync is normally sampled). Timer 0 detects the rising edge of bit clock with timer 2 output asserted and asserts output for length of frame. Timer 1 detects the falling edge of bit clock with timer 0 output asserted and controls shift registers for 32-bit transfers.

Table 484. I2S target configuration

Register	Value	Configuration
SHIFTCFG*n*	0000_0000h	Start and stop bit disabled.
SHIFTCTL*n*	0103_0002h	Configure transmit using timer 1 on the rising edge of shift clock with output data on pin 0.
SHIFTCFG(*n* + 1)	0000_0000h	Start and stop bit disabled.
SHIFTCTL(*n* + 1)	0180_0101h	Configure receive using timer 1 on the falling edge of shift clock with input data on pin 1.
TIMCMP*n*	0000_007Fh	Configure two 32-bit transfers per frame. Set TIMCMP[15:0] as (number of bits × 4) - 1.
TIMCFG*n*	0020_2500h	Configure enable on pin rising edge (inverted bit clock) with trigger high (timer 2) and disable on compare. Initial clock state is logic 1 and decrements on pin input (bit clock).
TIMCTL*n*	0B40_0203h	Configure 16-bit counter using pin 2 input (bit clock), with timer 2 output as the trigger.
TIMCMP(*n* + 1)	0000_003Fh	Configure 32-bit transfers. Set TIMCMP[15:0] as (number of bits × 2) - 1.
TIMCFG(*n* + 1)	0020_2500h	Configure enable on pin (bit clock) rising edge with trigger (timer 0) high and disable on compare. Initial clock state is logic 1 and decrement on pin input (bit clock).
TIMCTL(*n* + 1)	0340_0283h	Configure 16-bit counter using inverted pin 2 input (bit clock), with timer 0 output as the trigger.
TIMCMP(*n* + 2)	0000_0000h	Compare on zero (first edge).
TIMCFG(*n* + 2)	0020_6400h	Configure enable on inverted pin (frame sync) rising edge and disable on trigger falling edge (bit clock). Initial clock state is logic 1 and decrement on inverted pin input (frame sync).

Table continues on the next page...

Table 484. I2S target configuration (continued)

Register	Value	Configuration
TIMCTL(*n* + 2)	04C0_0383h	Configure 16-bit counter using inverted pin 3 input (frame sync), with pin 2 inverted input (bit clock) as the trigger.
SHIFTBUF*n*	Data to transmit	Transmit data can be written to Shifter Buffer Bit Swapped (SHIFTBUFBIS0 - SHIFTBUFBIS7). Use the shifter status flag to indicate when data can be written using interrupt or DMA request. Write to the SHIFTBUF register instead to support LSB first transfer.
SHIFTBUF(*n* + 1)	Data to receive	Received data can be read from Shifter Buffer Bit Swapped (SHIFTBUFBIS0 - SHIFTBUFBIS7). Use the shifter status flag to indicate when data can be read using interrupt or DMA request. Read from the SHIFTBUF register instead to support LSB first transfer.

72.6.8 SENT receiver

The SENT receiver can be supported by using one timer and one pin. The timer is configured as Input-Capture mode, and captures the counter value at the falling edge of the pin input. After the counter value is captured, the counter is automatically restarted from counter value 0. Therefore, the captured value always indicates the period between the previous falling edge and current falling edge. You can configure the CPU interrupt or DMA trigger at each capture of the counter. The CPU software performs the entire SENT frame decoding with the latest tick width adjustment.

Table 485. SENT receiver configuration

Register	Value	Configuration
TIMCMP*n*	—	Stores counter value at the falling edge of the pin.
TIMCFG*n*	0000_6000h	Timer is always enabled. It is disabled on trigger falling edge, decrement counter on FLEXIO clock.
TIMCTL*n*	0040_0007h	Single 16-bit input capture mode. The PINSEL field selects the timer pin input and output. Timer pin output disabled, internal trigger selected, select pin 0 as trigger.

72.6.9 Camera interface

Camera interface can be supported using one timer, one or more shifters, and multiple pins. Multiple transfers can be supported using the DMA controller.

The example configuration shown in the following table describes the FLEXIO configuration for interfacing with an 8-bit CMOS sensor with PCLK, VSYNC, HREF, and D[7:0] outputs. The example uses a 128-bit buffer to capture 16 pixels of image data before an interrupt or DMA transfer. You can use a bigger or smaller buffer depending on system DMA performance and FLEXIO resource usage by other applications.

NOTE

You can use additional timers to track the number of pixels per row and number of rows per frame, or HREF or VSYNC can be assigned as GPIO interrupts for software tracking.

Table 486. Camera interface configuration for 8-bit CMOS sensor

Register	Value	Configuration
SHIFTCFG*n*...*n*+2[1]	0007_0100h	Configure 8-bit parallel shift in from adjacent shifter.
SHIFTCFG*n*+3	0007_0000h	Configure 8-bit parallel shift in from pins FXIO_D[7:0] (D[7:0]).
SHIFTCTL*n*...*n*+3	0080_0001h	Configure receive using timer 0 on the negative edge of clock.
TIMCMP*n*	0000_001Fh	Configure 16 pixel (8 bits/pixel × 16 pixels = 128 bits) transfer. Set TIMCMP[15:0] as (number of pixels × 2) - 1.
TIMCFG*n*	0120_6600h	Configure enable on trigger (HREF) rising edge and disable on trigger falling edge. Initial shift clock state is logic 0 and decrement on PCLK input.
TIMCTL*n*	12C0_0803h	Configure 16-bit counter using FXIO_D[8] input (PCLK), with FXIO_D[9] input (HREF) as the inverted trigger.
SHIFTBUF*n*...*n*+3	Data to receive	Received data can be read from SHIFTBUF*n*...*n*+3. Use the shifter status flag to indicate when data can be read using interrupt or DMA request.

1. *n* = 0 or 4

72.6.10 Motorola 68K and Intel 8080 bus interface

Motorola 68K and Intel 8080 bus are two parallel interfaces commonly used by smart and asynchronous LCD controllers. With GPIO, FLEXIO can drive these interfaces using one timer and one shifter. Additional shifters can be used to support large transfers via the DMA controller.

Example configurations are provided for 1-beat and 16-beat transfers. Other lengths are also supported.

The following table provides an example of how to drive a 16-bit 68K or 8080 bus. For an 8080 bus, two GPIOs are used to drive the nCS and RS pins. For a 68K bus, an additional GPIO is required to drive the RDWR pin.

Table 487. Motorola 68K and Intel 8080 write configuration

Register	Value	Configuration
SHIFTCFG0...7	000F_0100h	Configure 16-bit parallel shift in from adjacent shifter.
SHIFTCTL0	0003_0002h	Configure transmit using timer 0 on the positive edge of clock with data output to FXIO_D[15:0].

Table continues on the next page...

Table 487. Motorola 68K and Intel 8080 write configuration (continued)

Register	Value	Configuration
SHIFTCTL1...7	0000_0002h	Configure transmit using timer 0 on the positive edge of clock.
TIMCMP0	0000_0101h (1 beat) 0000_1F01h (16 beats)	Configure 1 or 16-beat transfer with baud rate of divide by 4 of the FLEXIO clock. Set TIMCMP[15:8] as (number of beats × 2) - 1, and set TIMCMP[7:0] as (baud rate divider ÷ 2) - 1.
TIMCFG0	0000_2200h	Configure enable on trigger high and disable on compare. Timer output high on enable.
TIMCTL0	01C3_1001h (Motorola 68K, 1 beat) 1DC3_1001h (Motorola 68K, 16 beats) 01C3_1081h (Intel 8080, 1 beat) 1DC3_1081h (Intel 8080, 16 beats)	Configure dual 8-bit counter using FXIO_D[16] output (EN pin for 68K, nWR pin for 8080), with shifter 0 (1-beat) or shifter 7 (16 beats) flag as the inverted trigger.
SHIFTBUF0...7	Data to transmit	Transmit data can be written to SHIFTBUF0 (1 beat) or SHIFTBUF0...7 (16-beats) to initiate a transfer; use the shifter status flag to indicate when data can be written using interrupt or DMA request.

Table 488. Motorola 68K and Intel 8080 read configuration

Register	Value	Configuration
SHIFTCFG0...6	000F_0100h	Configure 16-bit parallel shift in from the adjacent shifter.
SHIFTCFG7	000F_0000h	Configure 16-bit parallel shift in from pin.
SHIFTCTL0...7	0080_0001h	Configure receive using timer 0 on the negative edge of clock with data input from FXIO_D[15:0].
TIMCMP0	0000_0101h (1 beat) 0000_1F01h (16 beats)	Configure 1 or 16-beat transfer with baud rate of divide by 4 of the FLEXIO clock. Set TIMCMP[15:8] = (number of beats × 2) - 1, and set TIMCMP[7:0] = (baud rate divider ÷ 2) - 1.
TIMCFG0	0000_2220h	Configure stop_bit. Enable on trigger high and disable on compare. Timer output high on enable.
TIMCTL0	1DC3_1001h (Motorola 68K, 1 beat) 01C3_1001h (Motorola 68K, 16 beats)	Configure dual 8-bit counter using either FXIO_D[16] output (EN pin for 68K) or FXIO_D[17] output (nRD pin for 8080),

Table continues on the next page...

Table 488. Motorola 68K and Intel 8080 read configuration (continued)

Register	Value	Configuration
	1DC3_1181h (Intel 8080, 1 beat) 01C3_1181h (Intel 8080, 16 beats)	with shifter 7 flag (1-beat) or shifter 0 flag (16 beats) as the inverted trigger.
SHIFTBUF0...7	Data received	Received data can be read from SHIFTBUF0 (1 beat) or SHIFTBUF0...7 (16 beats). Use the shifter status flag to indicate when data can be read using interrupt or DMA request.

In general, any operation to a 68K or 8080 bus target begins with a register write cycle followed by one or more data read or write cycles. To accomplish this, perform the following procedure:

1. Configure FLEXIO with 1-beat write configuration.
2. Configure GPIO to assert the nCS and RS pins (and deassert the RDWR pin for 68K).
3. Write register index data to SHIFTBUF0[15:0].
4. Configure GPIO to deassert the RS pin (and assert the RDWR pin for 68K data read).
5. Configure FLEXIO with the desired read or write configuration (1 or 16 beats).
6. Use the shifter status flag to trigger interrupt or DMA driven data transfers to and from Shifter Buffer (SHIFTBUF0 - SHIFTBUF7).
7. Configure GPIO to deassert the nCS pin.

72.6.11 Low-power state machine

Table 489 shows an example of a hypothetical state machine to illustrate the flexibility allowed in Shifter State mode.

In this example, FLEXIO waits for the FXIO_D[2] pin to assert and then drives a complementary square wave output at a frequency of FLEXIO_CLK/131072 on the FXIO_D[1:0] pins while the comparator output is asserted. This assumes that the comparator is connected to external trigger 15. See the chip-specific FLEXIO information for actual FLEXIO trigger mappings. Figure 398 shows the states and transitions implemented by this example.

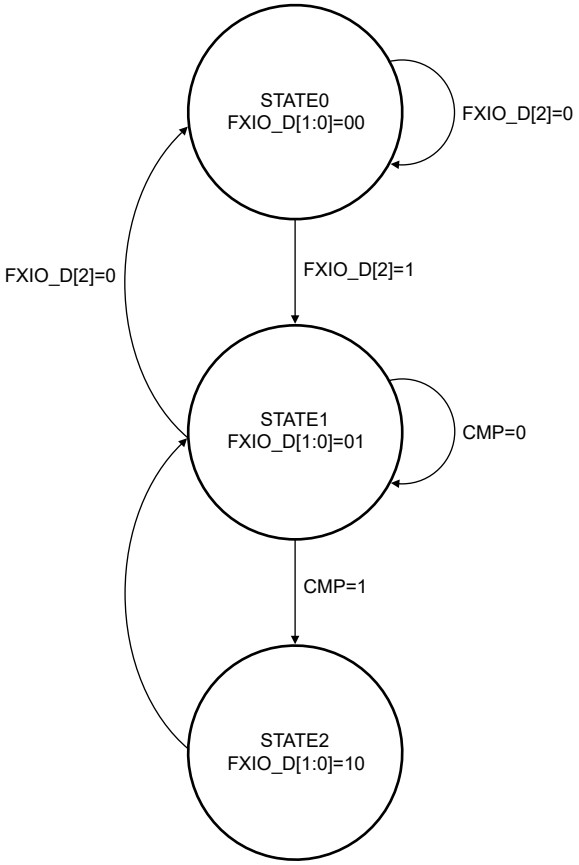


Figure 398. State diagram

Table 489. State machine configuration

Register	Value	Configuration
SHIFTCFG0...2	0000_0003h	Enable FXIO_D[1:0] as outputs.
SHIFTCTL0	0080_0206h	Configure for State mode using FXIO_D[4:2] as inputs to select next state and timer 0 output low to trigger next state.
SHIFTBUF0	0020_8208h	State0: Drive FXIO_D[1:0] = 00; transition to state0 if FXIO_D[2] = 0, state1 if FXIO_D[2] = 1.
SHIFTCTL1	0000_0206h	Configure for State mode using FXIO_D[4:2] as inputs to select next state and timer 0 output high to trigger next state.
SHIFTBUF1	0140_8408h	State1: Drive FXIO_D[1:0]=01; transition to state0 if FXIO_D[2]=0, state1 if CMP = 0, state2 if CMP = 1 (FXIO_D[3]=1).

Table continues on the next page...

Table 489. State machine configuration (continued)

Register	Value	Configuration
SHIFTCTL2	0080_0206h	Configure for State mode using FXIO_D[4:2] as inputs to select next state and timer 0 output low to trigger next state.
SHIFTBUF2	0224_9249h	State2: Drive FXIO_D[1:0] = 10, transition to state1 with timer 0 output low.
TIMCMP0	0000_FFFFh	Configure baud rate of divide by 131072 of the FLEXIO clock. Set TIMCMP[7:0] as (baud rate divider ÷ 2) - 1.
TIMCFG0	0000_0000h	Configure timer always enabled.
TIMCTL0	0000_0003h	Configure single 16-bit counter.
TIMCFG1	0010_7600h	Configure timer enabled on trigger rising edge, disabled on trigger falling edge, decrement on trigger edge.
TIMCTL1	0F03_0303h	Configure timer output enabled on FXIO_D[3], external trigger 15 (comparator output) selected.

72.6.12 Keypad interface

The keypad interface can support a 3 × 4 keypad matrix using three timers and three shifters, although a larger matrix can be supported using additional shifters. The configuration is designed for four columns configured as active low open-drain pins and three rows configured as input pins with pull-up resistors enabled.

One shifter is configured in Logic mode to assert its output when any of the row inputs are low, indicating a key is pressed. Use a timer to filter the shifter output to ensure that the key is pressed for a minimum amount of time before performing the column scan.

A different shifter is configured for parallel transmit. Use this shifter to scan each column when a keypress is detected. When not scanning, the shifter output is configured to assert all active low open-drain column outputs to detect any keypress. Use a dedicated timer to control the transmit shifter.

The last shifter is configured for parallel receive. Use this shifter to capture the result of the column scanning so that you can decode which key (or keys) was pressed. This configuration captures the state of both row and column pins for each scan, although the row state can also be deduced by the shift order. Use a dedicated timer to control the receive shifter, which shifts at half the frequency of the transmit shifter.

When the result of the key scan is available in the receive shifter register, FLEXIO continues to monitor the row inputs and can trigger multiple scans from a single keypress. To support debouncing, you can decide how many consecutive scans must be considered as a single keypress.

NOTE

Because the pins used in Logic mode are fixed per shifter and the shifters that support parallel shifts are limited, this configuration is restricted to what pins and shifters it can use. Increasing the matrix beyond four-row inputs requires multiple shifters in Logic mode. Increasing beyond four-column outputs requires concatenating transmit shifters to create a larger shift register.

Table 490. Keypad interface configuration

Register	Value	Configuration
SHIFTCFG0	0003_0032h	Start bit enabled (logic 0) and stop bit enabled (logic 1), configured for 4-bit shift.
SHIFTCTL0	0101_0402h	Configure transmit using timer 1 on the rising edge of clock generating open-drain output on pins[7:4] (column outputs).
SHIFTBUF0	0804_0201h	Static data containing the column scan pattern; each column is scanned one-hot with dead time in between.
SHIFTCFG1	0000_0020h	In Logic mode, mask input 3.
SHIFTCTL1	0000_0007h	Configure Logic mode.
SHIFTBUF1	07FF_07FFh	Static data configuring Logic mode LUT. Output asserts if pins [3:1] are logic 0.
SHIFTCFG3	0007_0000h	Configured for 8-bit shift.
SHIFTCTL3	0280_0001h	Configure receive using timer 2 on falling edge of clock with input data on pins [7:0] (both rows and columns; pin 0 is don't care).
TIMCMP0	0000_00ffh	Configure prescanning glitch filter to 256 FLEXIO clock cycles. For different filter cycles, configure TIMCMP[15:0] as (filter cycles) - 1.
TIMCFG0	0103_6600h	Configure enable on trigger rising edge and disable on trigger falling edge. Initial clock state is logic 0 and is not affected by reset.
TIMCTL0	0540_0003h	Configure 16-bit counter using the shifter 1 flag (logic state) as trigger.
TIMCMP1	0000_0F3Fh	Configure eight shifts (twice for each column) at column scan rate of divide by 128. For a different scan frequency, define TIMCMP[7:0] as (scan divider ÷ 2) - 1.
TIMCFG1	0020_2622h	Enable on trigger rising edge, disable on compare. Start and stop bits are enabled.
TIMCTL1	0340_0001h	Configure dual 8-bit counter using timer 0 output as the trigger.
TIMCMP2	0000_0801h	Configure four shifts at half the frequency of the timer 1 trigger.

Table continues on the next page...

Table 490. Keypad interface configuration (continued)

Register	Value	Configuration
TIMCFG2	0110_2100h	Enable on timer 1 enable, disable on compare, decrement on trigger input with output initially negated, and not affected by reset.
TIMCTL2	0740_0001h	Configure dual 8-bit counter using timer 1 output as the trigger.
SHIFTBUF3	Keypad scan result	Keypad scan result can be read from SHIFTBUF3. Each byte is the result of one scan with both row and column pin state from the scan (pin 0 is not used). Use the shifter status flag to indicate when data can be written using interrupt or DMA request.

72.7 Memory map and registers

72.7.1 FLEXIO register descriptions

NOTE

Invalid register accesses, which include reading a write-only register, writing to a read-only register, or accessing an invalid address, result in a bus error.

72.7.1.1 FLEXIO memory map

FlexIO base address: 4032_4000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID (VERID)	32	R	0201_0003h
4h	Parameter (PARAM)	32	R	0420_0808h
8h	FLEXIO Control (CTRL)	32	RW	0000_0000h
Ch	Pin State (PIN)	32	R	0000_0000h
10h	Shifter Status (SHIFTSTAT)	32	RW	0000_0000h
14h	Shifter Error (SHIFTERR)	32	RW	0000_0000h
18h	Timer Status Flag (TIMSTAT)	32	RW	0000_0000h
20h	Shifter Status Interrupt Enable (SHIFTSIEN)	32	RW	0000_0000h
24h	Shifter Error Interrupt Enable (SHIFTEIEN)	32	RW	0000_0000h
28h	Timer Interrupt Enable (TIMIEN)	32	RW	0000_0000h
30h	Shifter Status DMA Enable (SHIFTSDEN)	32	RW	0000_0000h
38h	Timer Status DMA Enable (TIMERSDEN)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
40h	Shifter State (SHIFTSTATE)	32	RW	0000_0000h
48h	Trigger Status (TRGSTAT)	32	RW	0000_0000h
4Ch	External Trigger Interrupt Enable (TRIGIEN)	32	RW	0000_0000h
50h	Pin Status (PINSTAT)	32	RW	0000_0000h
54h	Pin Interrupt Enable (PINIEN)	32	RW	0000_0000h
58h	Pin Rising Edge Enable (PINREN)	32	RW	0000_0000h
5Ch	Pin Falling Edge Enable (PINFEN)	32	RW	0000_0000h
60h	Pin Output Data (PINOUTD)	32	RW	0000_0000h
64h	Pin Output Enable (PINOUTE)	32	RW	0000_0000h
68h	Pin Output Disable (PINOUTDIS)	32	RW	0000_0000h
6Ch	Pin Output Clear (PINOUTCLR)	32	RW	0000_0000h
70h	Pin Output Set (PINOUTSET)	32	RW	0000_0000h
74h	Pin Output Toggle (PINOUTTOG)	32	RW	0000_0000h
80h - 9Ch	Shifter Control (SHIFTCTL0 - SHIFTCTL7)	32	RW	0000_0000h
100h - 11Ch	Shifter Configuration (SHIFTCFG0 - SHIFTCFG7)	32	RW	0000_0000h
200h - 21Ch	Shifter Buffer (SHIFTBUF0 - SHIFTBUF7)	32	RW	0000_0000h
280h - 29Ch	Shifter Buffer Bit Swapped (SHIFTBUFBIS0 - SHIFTBUFBIS7)	32	RW	0000_0000h
300h - 31Ch	Shifter Buffer Byte Swapped (SHIFTBUFBYS0 - SHIFTBUFBYS7)	32	RW	0000_0000h
380h - 39Ch	Shifter Buffer Bit Byte Swapped (SHIFTBUFBBS0 - SHIFTBUFBBS7)	32	RW	0000_0000h
400h - 41Ch	Timer Control (TIMCTL0 - TIMCTL7)	32	RW	0000_0000h
480h - 49Ch	Timer Configuration (TIMCFG0 - TIMCFG7)	32	RW	0000_0000h
500h - 51Ch	Timer Compare (TIMCMP0 - TIMCMP7)	32	RW	0000_0000h
680h - 69Ch	Shifter Buffer Nibble Byte Swapped (SHIFTBUFNBS0 - SHIFTBUFNBS7)	32	RW	0000_0000h
700h - 71Ch	Shifter Buffer Halfword Swapped (SHIFTBUFHWS0 - SHIFTBUFHWS7)	32	RW	0000_0000h
780h - 79Ch	Shifter Buffer Nibble Swapped (SHIFTBUFNIS0 - SHIFTBUFNIS7)	32	RW	0000_0000h
800h - 81Ch	Shifter Buffer Odd Even Swapped (SHIFTBUFOES0 - SHIFTBUFOES7)	32	RW	0000_0000h
880h - 89Ch	Shifter Buffer Even Odd Swapped (SHIFTBUFEOS0 - SHIFTBUFEOS7)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
900h - 91Ch	Shifter Buffer Halfword Byte Swapped (SHIFTBUFHBS0 - SHIFTBUFHBS7)	32	RW	0000_0000h

72.7.1.2 Version ID (VERID)

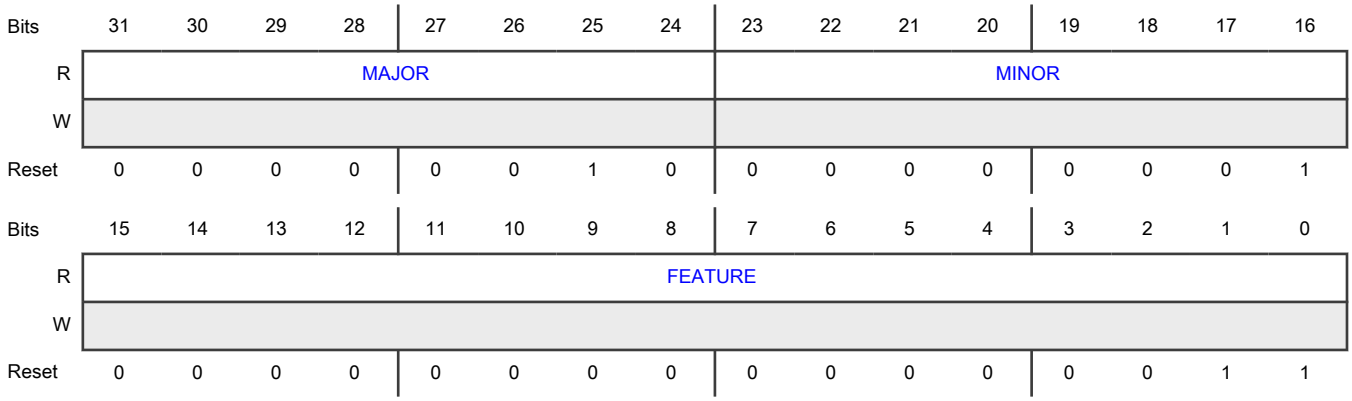
Offset

Register	Offset
VERID	0h

Function

Indicates the version of FLEXIO.

Diagram



Fields

Field	Function
31-24 MAJOR	Major Version Number Indicates the major version number of the module specification.
23-16 MINOR	Minor Version Number Indicates the minor version number of the module specification.
15-0 FEATURE	Feature Specification Number Indicates the feature set number. 0000_0000_0000_0000b - Standard features implemented

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0000_0000_0000_0001b - State, logic, and parallel modes supported 0000_0000_0000_0010b - Pin control registers supported 0000_0000_0000_0011b - State, logic, and parallel modes, plus pin control registers supported

72.7.1.3 Parameter (PARAM)

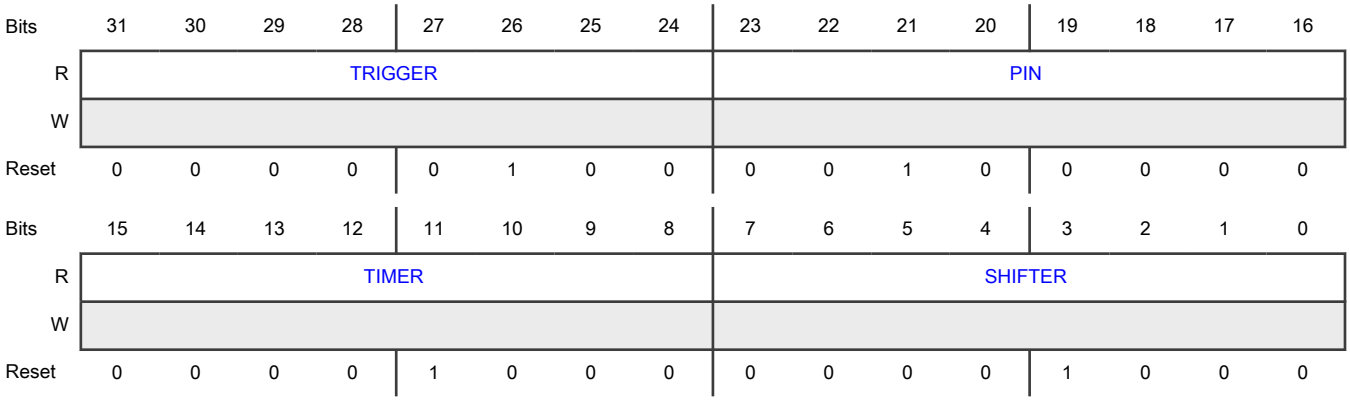
Offset

Register	Offset
PARAM	4h

Function

Contains the number of shifters, timers, pins, and triggers.

Diagram



Fields

Field	Function
31-24 TRIGGER	Trigger Number Indicates the number of external triggers implemented.
23-16 PIN	Pin Number Indicates the number of pins implemented.
15-8 TIMER	Timer Number Indicates the number of timers implemented.
7-0 SHIFTER	Shifter Number Indicates the number of shifters implemented.

72.7.1.4 FLEXIO Control (CTRL)

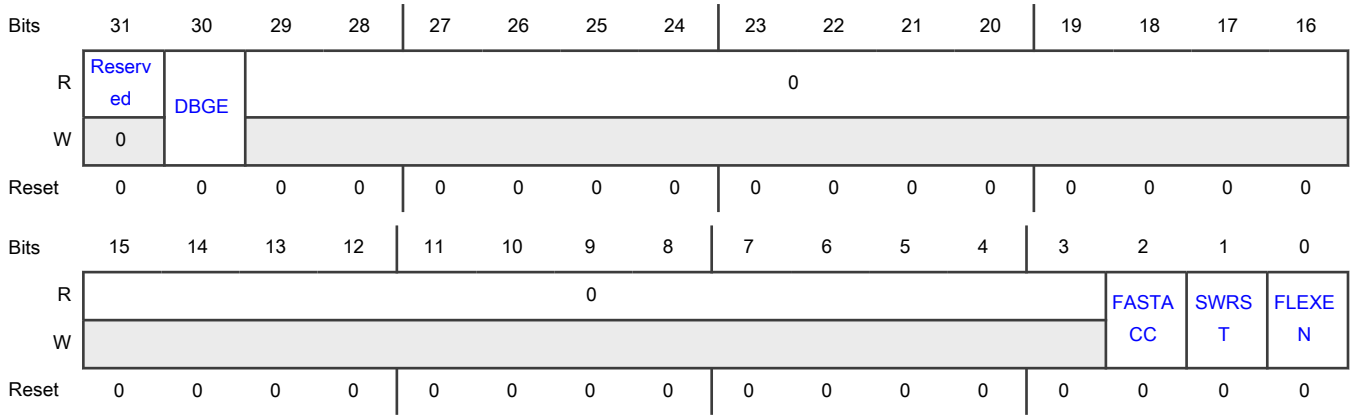
Offset

Register	Offset
CTRL	8h

Function

Controls various aspects of the FLEXIO operation.

Diagram



Fields

Field	Function
31 —	Any write value to this bit must be zero.
30 DBGE	Debug Enable Enables the FLEXIO operation in Debug mode. 0b - Disable 1b - Enable
29-3 —	Reserved
2 FASTACC	Fast Access Configures fast or normal register accesses to FLEXIO registers, but requires the FLEXIO functional clock to be at least equal to the frequency of the bus clock. 0b - Normal 1b - Fast

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 SWRST	Software Reset Specifies whether software reset is enabled. The software reset does not affect this register but it affects all other logic in FLEXIO. All other register accesses are ignored until this field is cleared. The field remains 1 until software clears it and the reset has cleared in the FLEXIO clock domain. If you write 1 to this field, all FLEXIO registers except the Control register are reset. 0b - Disabled 1b - Enabled
0 FLEXEN	FLEXIO Enable Enables FLEXIO. 0b - Disable 1b - Enable

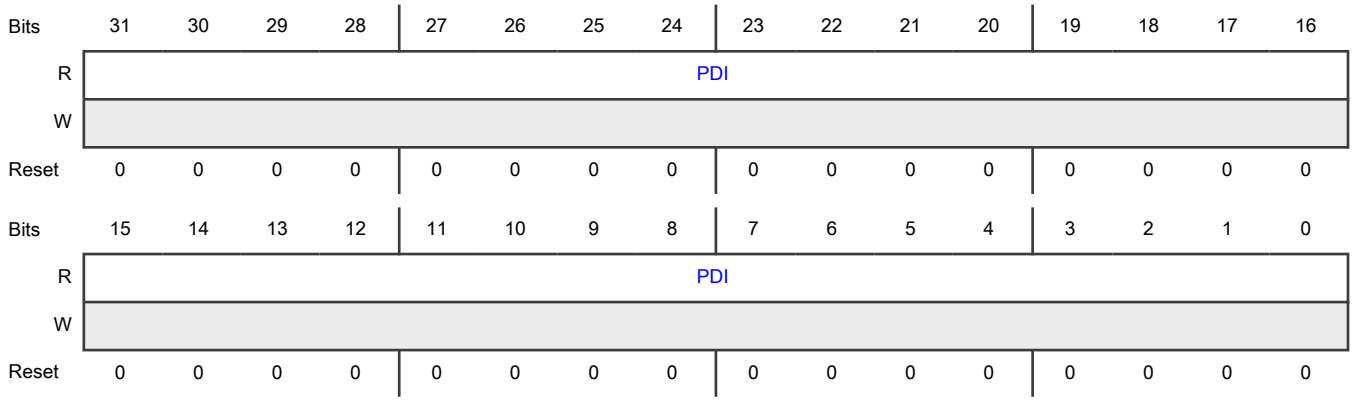
72.7.1.5 Pin State (PIN)

Offset

Register	Offset
PIN	Ch

Function
Indicates the status of the pin data input.

Diagram



Fields

Field	Function
31-0 PDI	Pin Data Input Indicates the input data on each of the FLEXIO pins.

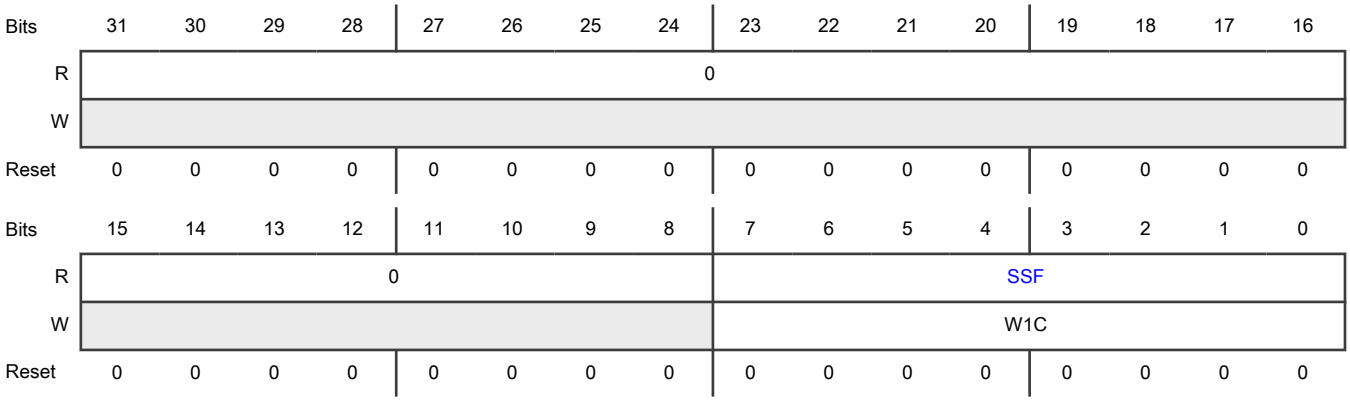
72.7.1.6 Shifter Status (SHIFTSTAT)

Offset

Register	Offset
SHIFTSTAT	10h

Function
Contains shifter status flags.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 SSF	Shifter Status Flag Indicates the shifter status. This flag is updated in one of the following cases: • If SHIFTCTL*n*[SMOD] = 001b (Receive mode), the status flag is set when SHIFTBUF is loaded with data from the shifter (SHIFTBUF is full). The status flag is cleared when you read Shifter Buffer (SHIFTBUF0 - SHIFTBUF7).

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• If SHIFTCTL*n*[SMOD] = 010b (Transmit mode), the status flag is set when SHIFTBUF data is transferred to the shifter (SHIFTBUF is empty) or when SHIFTCTL*n*[SMOD] is initially configured as 010b (Transmit mode). The status flag is cleared when you write to the SHIFTBUF register. • If SHIFTCTL*n*[SMOD] = 100b (Match Store mode), the status flag is set when a match occurs between SHIFTBUF and the shifter. The status flag is cleared when you read the SHIFTBUF register. • If SHIFTCTL*n*[SMOD] = 101b (Match Continuous mode), the status flag returns the current match result between SHIFTBUF and the shifter. You cannot clear the status flag by reading the SHIFTBUF register. • If SHIFTCTL*n*[SMOD] = 110b (State mode), the status flag for a shifter sets when it is selected by the current state pointer. • If SHIFTCTL*n*[SMOD] = 111b (Logic mode), the status flag returns the current value of the programmable logic block output. You can clear this status flag by writing a logic one to the flag for all modes except Match Continuous mode, State mode, and Logic mode. **NOTE** This field behaves differently for register reads and writes. When reading 0000_0000b - Clear 0000_0001b - Set When writing 0000_0000b - No effect 0000_0001b - Clear the flag

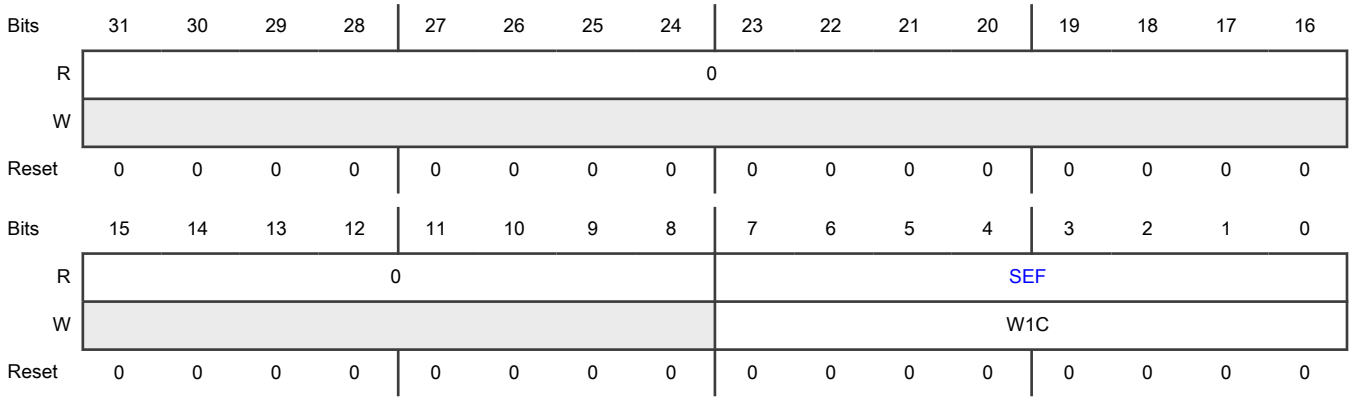
72.7.1.7 Shifter Error (SHIFTERR)

Offset

Register	Offset
SHIFTERR	14h

Function
Reports shifter errors.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 SEF	Shifter Error Flag Indicates shifter error flag status. This flag is set when one of the following events occurs: • If SHIFTCTL*n*[SMOD] = 001b (Receive mode), it indicates that either the shifter is ready to store new data into SHIFTBUF before the previous data is read from SHIFTBUF (SHIFTBUF overrun), or the received start or stop bit does not match the expected value. • If SHIFTCTL*n*[SMOD] = 010b (Transmit mode), it indicates that the shifter is ready to load new data from SHIFTBUF before new data is written into SHIFTBUF (SHIFTBUF underrun). • If SHIFTCTL*n*[SMOD] = 100b (Match Store mode), it indicates the occurrence of a match event before the previous match data is read from SHIFTBUF (SHIFTBUF overrun). • If SHIFTCTL*n*[SMOD] = 101b (Match Continuous mode), the error flag is set when a match occurs between SHIFTBUF and the shifter. • If SHIFTCTL*n*[SMOD] = 111b (Logic mode), the error flag is set when the output of the programmable logic block is asserted. For SHIFTCTL*n*[SMOD] = 101b (Match Continuous mode), the flag can also be cleared when you read Shifter Buffer (SHIFTBUF0 - SHIFTBUF7). **NOTE** This field behaves differently for register reads and writes. When reading 0000_0000b - Clear 0000_0001b - Set When writing 0000_0000b - No effect 0000_0001b - Clear the flag

72.7.1.8 Timer Status Flag (TIMSTAT)

Offset

Register	Offset
TIMSTAT	18h

Function
Reports timer status.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								TSF							
W									W1C							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-8 —	Reserved
7-0 TSF	Timer Status Flag Indicates timer status. This flag is set depending on Timer mode: • In 8-bit baud counter mode, this flag is set when the upper 8-bit counter equals zero and decrements. • In 8-bit high PWM mode, this flag is set when the upper 8-bit counter equals zero and decrements. • In 16-bit counter mode, this flag is set when the 16-bit counter equals zero and decrements. • In 16-bit counter disable mode, TSF is set when a timer disable event is detected. • In 8-bit word counter mode, TSF is set when the upper 8-bit counter equals zero and decrements. • In 8-bit low PWM mode, TSF is set when the upper 8-bit counter equals zero and decrements. • In 16-bit input capture mode, TSF is set when a timer disable event is detected and the flag is clear. In this mode, you must read Timer Control (TIMCTL0 - TIMCTL7) only when TSF is set. **NOTE** This field behaves differently for register reads and writes.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When reading 0000_0000b - Clear 0000_0001b - Set When writing 0000_0000b - No effect 0000_0001b - Clear the flag

72.7.1.9 Shifter Status Interrupt Enable (SHIFTSIEN)

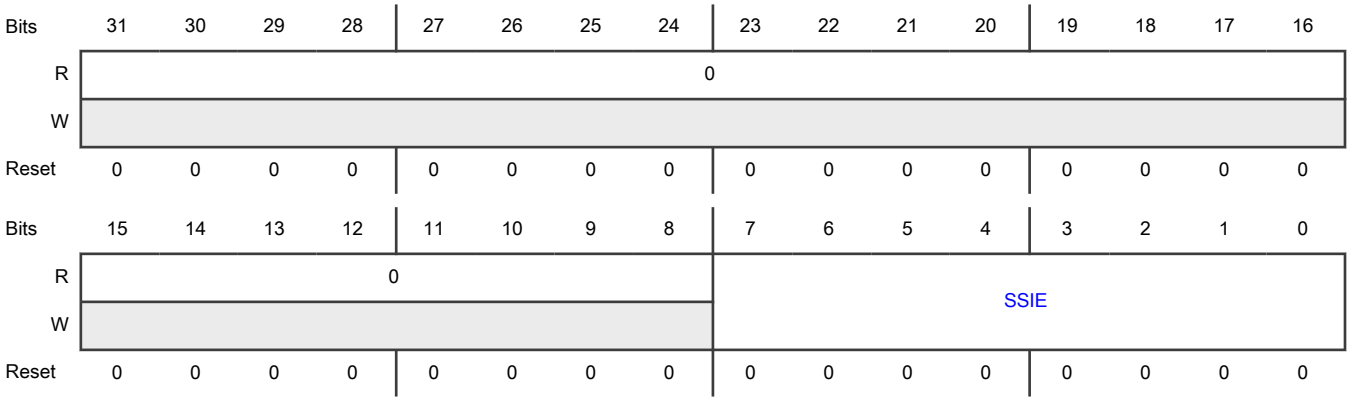
Offset

Register	Offset
SHIFTSIEN	20h

Function

Enables shifter status interrupts.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 SSIE	Shifter Status Interrupt Enable Enables interrupt generation when the corresponding SHIFTSTAT[SSF] flag is set. If you write 0 to this field, SHIFTSTAT[SSF] is disabled; and if you write 1 to this field, SHIFTSTAT[SSF] is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disable 1b - Enable

72.7.1.10 Shifter Error Interrupt Enable (SHIFTEIEN)

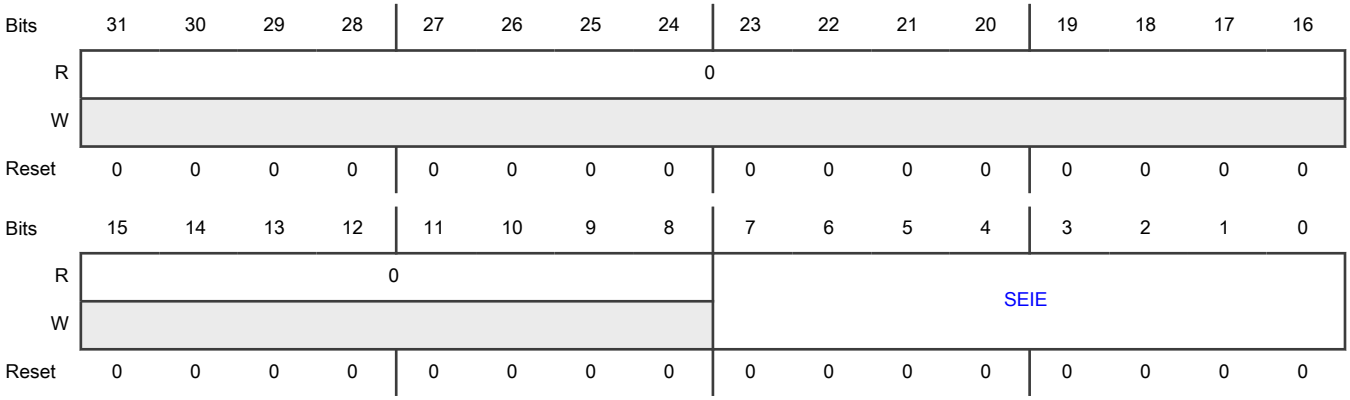
Offset

Register	Offset
SHIFTEIEN	24h

Function

Enables shifter error interrupts.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 SEIE	Shifter Error Interrupt Enable Enables interrupt generation when the corresponding SHIFTERR[SEF] flag is set. If you write 0 to this field, SHIFTERR[SEF] is disabled; and if you write 1 to this field, SHIFTERR[SEF] is enabled. 0b - Disable 1b - Enable

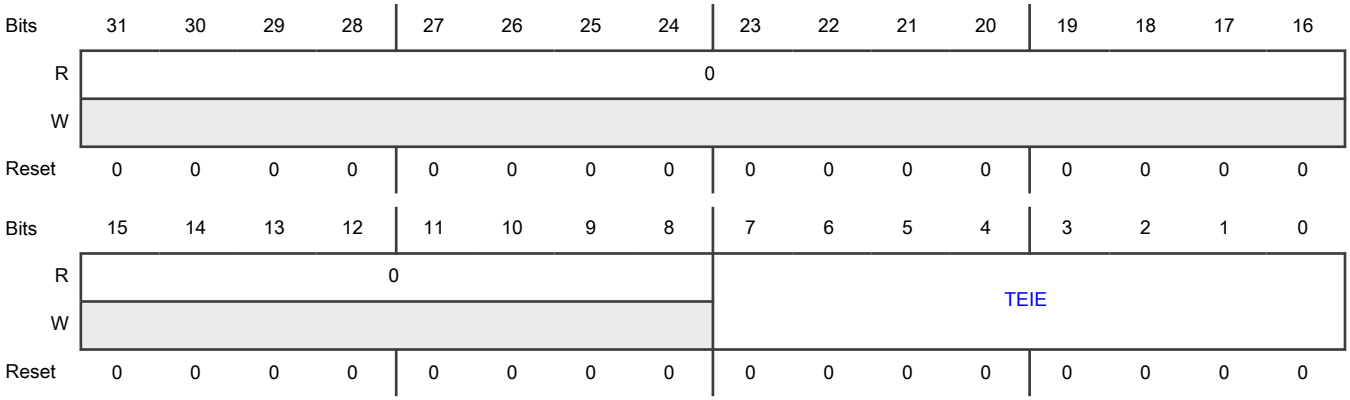
72.7.1.11 Timer Interrupt Enable (TIMIEN)

Offset

Register	Offset
TIMIEN	28h

Function
Enables timer status interrupts.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 TEIE	Timer Status Interrupt Enable Enables interrupt generation when the corresponding TIMSTAT[TSF] flag is set. If you write 0 to this field, TIMSTAT[TSF] is disabled; and if you write 1 to this field, TIMSTAT[TSF] is enabled. 0b - Disable 1b - Enable

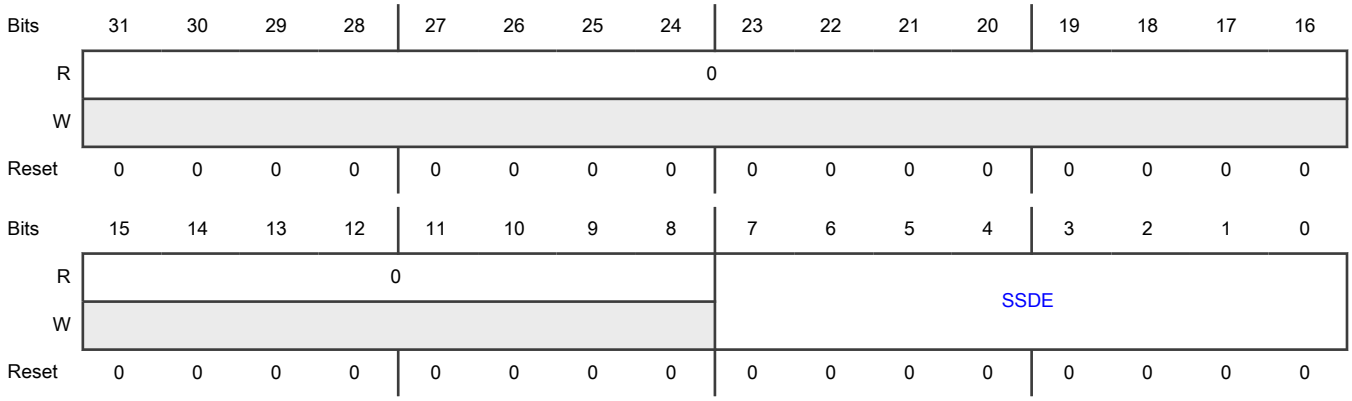
72.7.1.12 Shifter Status DMA Enable (SHIFTSDEN)

Offset

Register	Offset
SHIFTSDEN	30h

Function
Enables shifter DMA requests.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 SSDE	Shifter Status DMA Enable Enables DMA request generation when the corresponding SHIFTSTAT[SSF] flag is set. If you write 0 to this field, SHIFTSTAT[SSF] is disabled; and if you write 1 to this field, SHIFTSTAT[SSF] is enabled. 0b - Disable 1b - Enable

72.7.1.13 Timer Status DMA Enable (TIMERSDEN)

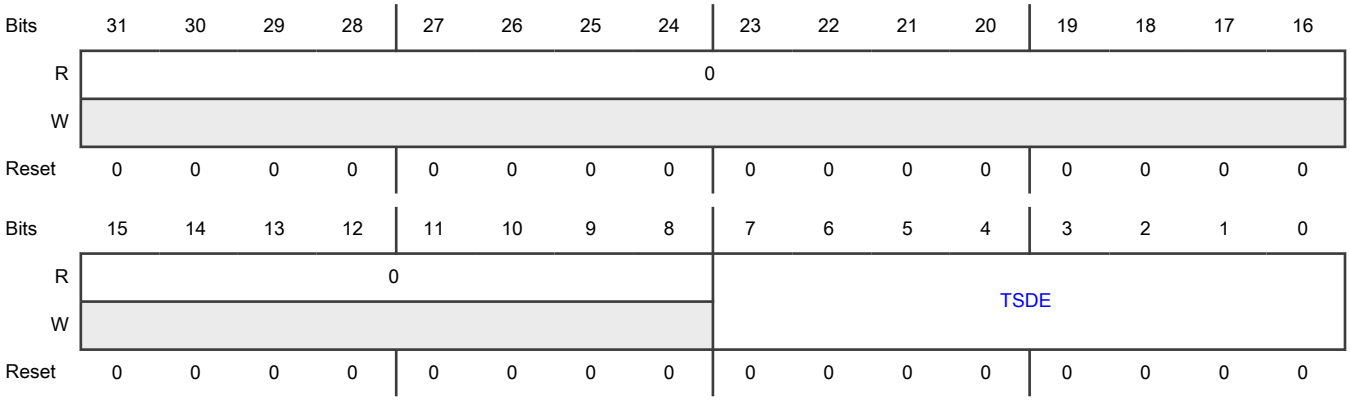
Offset

Register	Offset
TIMERSDEN	38h

Function

Enables DMA requests when the timer status flag is set.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 TSDE	Timer Status DMA Enable Enables DMA request generation when the corresponding TIMSTAT[TSF] flag is set. When the timer status DMA request is enabled, reading or writing to a timer compare register clears the corresponding timer status register. The DMA must therefore read or write to the timer compare register as part of the DMA transfer; otherwise, the DMA request remains asserted. 0b - Disable 1b - Enable

72.7.1.14 Shifter State (SHIFTSTATE)

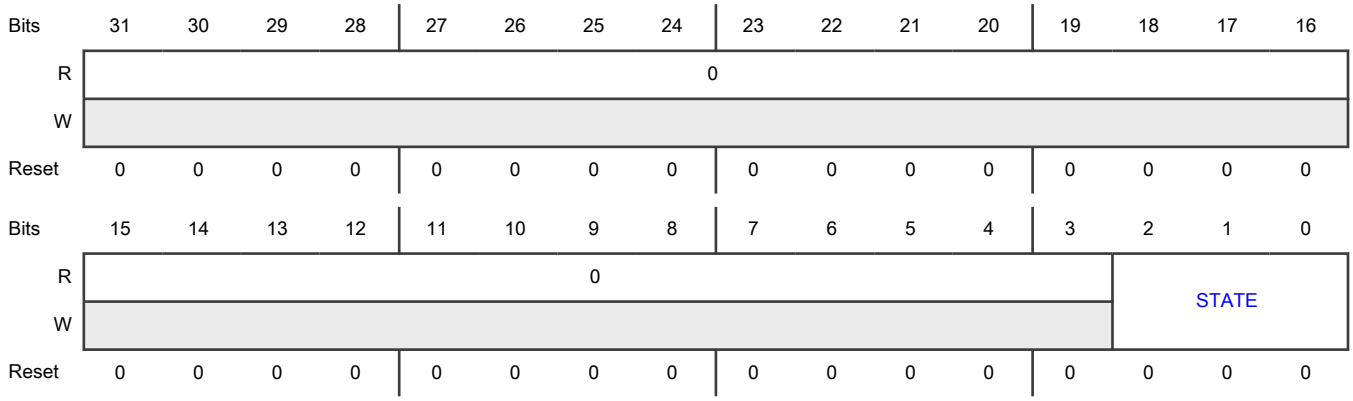
Offset

Register	Offset
SHIFTSTATE	40h

Function

Contains a pointer to track the current shifter.

Diagram



Fields

Field	Function
31-3 —	Reserved
2-0 STATE	Current State Pointer Maintains a pointer to track the current shifter (configured for State mode) enabled to drive outputs and compute the next state. Reading this register when the state pointer is updating can result in the return of an incorrect state. The value that you write to this field overrides the current state.

72.7.1.15 Trigger Status (TRGSTAT)

Offset

Register	Offset
TRGSTAT	48h

Function

Contains external trigger status flags.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												ETSF			
W													W1C			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-4 —	Reserved
3-0 ETSF	External Trigger Status Flag Specifies whether the external trigger status flag is set when a rising edge is detected on the corresponding external trigger input. **NOTE** This field behaves differently for register reads and writes. When reading 0000b - Clear 0001b - Set When writing 0000b - No effect 0001b - Clear the flag

72.7.1.16 External Trigger Interrupt Enable (TRIGIEN)

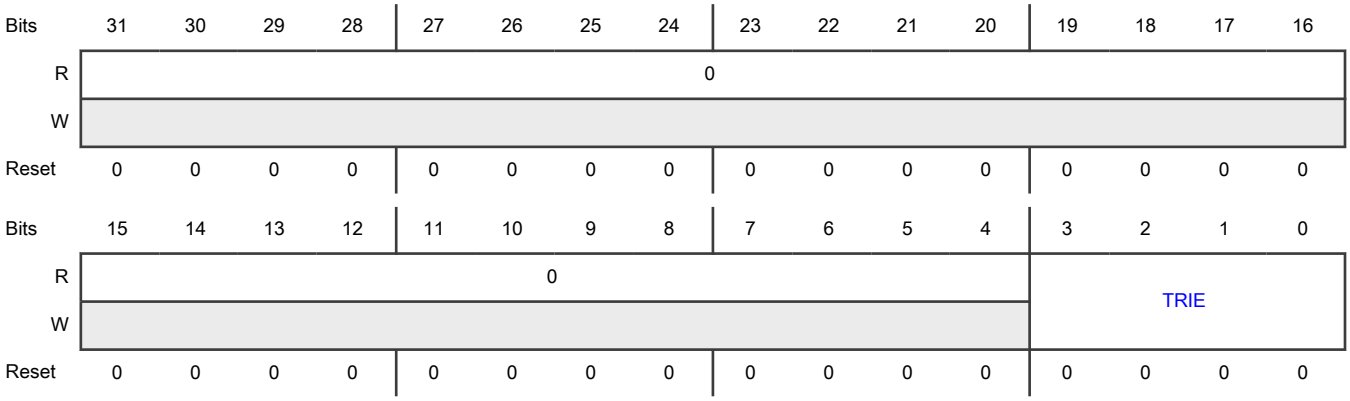
Offset

Register	Offset
TRIGIEN	4Ch

Function

Enables external trigger interrupts.

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 TRIE	External Trigger Interrupt Enable Enables interrupt generation when the corresponding TRGSTAT[ETSF] flag is set. If you write 0 to this field, TRGSTAT[ETSF] is disabled, and if you write 1 to this field, TRGSTAT[ETSF] is enabled. 0b - Disable 1b - Enable

72.7.1.17 Pin Status (PINSTAT)

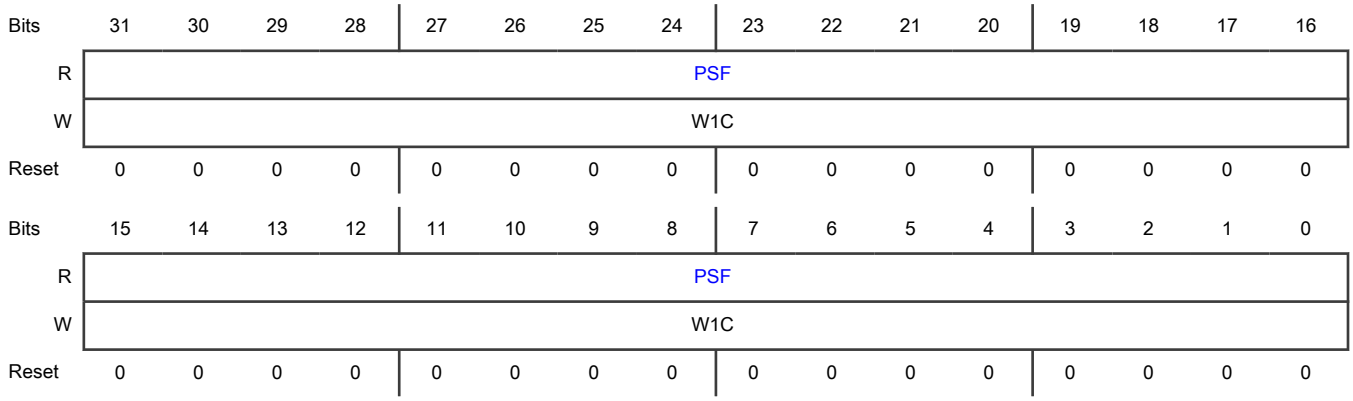
Offset

Register	Offset
PINSTAT	50h

Function

Contains pin status flags.

Diagram



Fields

Field	Function
31-0 PSF	Pin Status Flag Indicates whether the pin status flag is set when a rising edge or falling edge (if configured) is detected on the corresponding pin, as configured by the pin. **NOTE** This field behaves differently for register reads and writes. When reading 0000_0000_0000_0000_0000_0000_0000_0000b - Clear 0000_0000_0000_0000_0000_0000_0000_0001b - Set When writing 0000_0000_0000_0000_0000_0000_0000_0000b - No effect 0000_0000_0000_0000_0000_0000_0000_0001b - Clear the flag

72.7.1.18 Pin Interrupt Enable (PINIEN)

Offset

Register	Offset
PINIEN	54h

Function

Enables pin status interrupts.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PSIE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PSIE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PSIE	Pin Status Interrupt Enable Enables interrupt generation when the corresponding PINSTAT[PSF] flag is set. If you write 0 to this field, PINSTAT[PSF] is disabled, and if you write 1 to this field, PINSTAT[PSF] is enabled. 0b - Disable 1b - Enable

72.7.1.19 Pin Rising Edge Enable (PINREN)

Offset

Register	Offset
PINREN	58h

Function
Enables the pin status flag on a rising edge.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PRE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PRE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PRE	Pin Rising Edge Specifies whether the pin status flag is set whenever a rising edge is detected on the pin. 0b - Not set 1b - Set

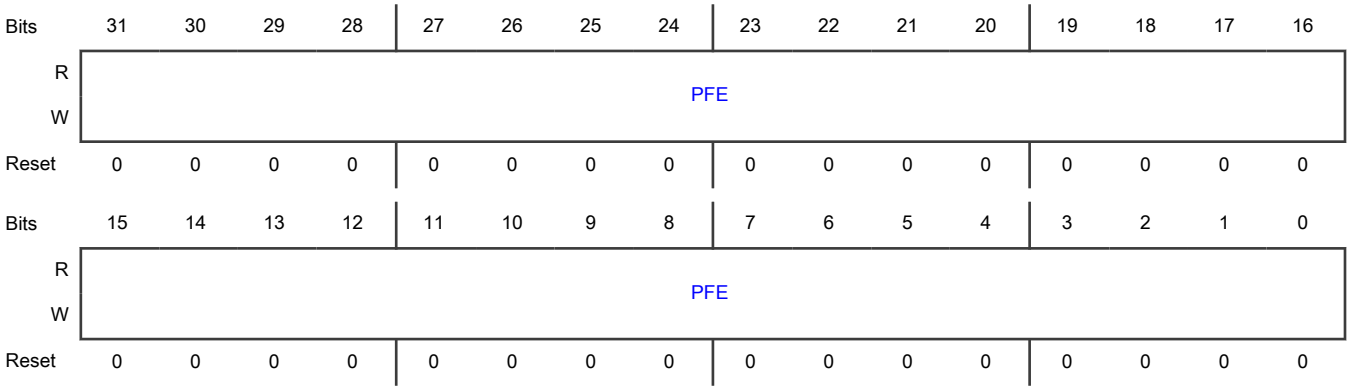
72.7.1.20 Pin Falling Edge Enable (PINFEN)

Offset

Register	Offset
PINFEN	5Ch

Function
Enables the pin status flag on a falling edge.

Diagram



Fields

Field	Function
31-0 PFE	Pin Falling Edge Specifies whether the pin status flag is set whenever a falling edge is detected on the pin. 0b - Not set 1b - Set

72.7.1.21 Pin Output Data (PINOUTD)

Offset

Register	Offset
PINOUTD	60h

Function
Contains data output when direct pin output is enabled.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	OUTD															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	OUTD															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 OUTD	Output Data Configures the value driven on the corresponding pin when direct pin output is enabled. 0b - Logic zero 1b - Logic one

72.7.1.22 Pin Output Enable (PINOUTE)

Offset

Register	Offset
PINOUTE	64h

Function
Enables pin output.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	OUTE															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	OUTE															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 OUTE	Output Enable Enables direct output on the corresponding pin. If this field is 0, the pin is controlled by timer/shifter configuration, and if this field is 1, pin is an output and driven with the value of Pin Output Data (PINOUTD). 0b - Controlled by timer/shifter configuration 1b - Output; driven with value of PINOUTD

72.7.1.23 Pin Output Disable (PINOUTDIS)

Offset

Register	Offset
PINOUTDIS	68h

Function
Disables pin output.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W	OUTDIS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W	OUTDIS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 OUTDIS	Output Disable Configures the corresponding pins to disable direct output. If this field is 1, the corresponding fields in Pin Output Data (PINOUTD) and Pin Output Enable (PINOUTE) become 0. 0b - No effect 1b - Corresponding fields become 0

72.7.1.24 Pin Output Clear (PINOUTCLR)

Offset

Register	Offset
PINOUTCLR	6Ch

Function
Clears pin output.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W	OUTCLR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W	OUTCLR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 OUTCLR	Output Clear Configures the corresponding pins to output zero. If this field is 1, the corresponding field in Pin Output Data (PINOUTD) becomes 0 and the one in Pin Output Enable (PINOUTE) becomes 1. 0b - No effect 1b - Corresponding field in PINOUTD becomes 0; corresponding field in PINOUTE becomes 1

72.7.1.25 Pin Output Set (PINOUTSET)

Offset

Register	Offset
PINOUTSET	70h

Function
Sets pin output.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W	OUTSET															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W	OUTSET															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 OUTSET	Output Set Configures the corresponding pins to output logic one. If this field is 1, the corresponding fields in Pin Output Data (PINOUTD) and Pin Output Enable (PINOUTE) become 1. 0b - No effect 1b - Corresponding fields become 1

72.7.1.26 Pin Output Toggle (PINOUTTOG)

Offset

Register	Offset
PINOUTTOG	74h

Function
Toggles pin output.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W	OUTTOG															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W	OUTTOG															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 OUTTOG	Output Toggle Configures the corresponding pins to toggle. If this field is 1, the corresponding field in Pin Output Data (PINOUTD) is inverted and the one in Pin Output Enable (PINOUTE) becomes 1. 0b - No effect 1b - Corresponding field in PINOUTD is inverted; corresponding field in PINOUTE becomes 1

72.7.1.27 Shifter Control (SHIFTCTL0 - SHIFTCTL7)

Offset

For n = 0 to 7:

Register	Offset
SHIFTCTLn	80h + (n × 4h)

Function

Provides shifter controls.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				TIMSEL				TIMPOL	0			PINCFG			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0			PINSEL					PINPOL	0			SMOD			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-27 —	Reserved
26-24 TIMSEL	Timer Select Selects which timer is used for controlling the logic or shift register and generating the shift clock. TIMSEL = i selects TIMERi.
23 TIMPOL	Timer Polarity Determines whether the shift occurs on the positive edge or negative edge of the shift clock. 0b - Positive edge 1b - Negative edge
22-18 —	Reserved
17-16 PINCFG	Shifter Pin Configuration Specifies shifter pin configuration. For pins configured as an output (PINCFG = 11b), this field takes effect when you write to the register. **NOTE** When initially configuring PINCFG as 11b, FLEXIO may briefly drive the pin low. To avoid this, you can configure PINCFG as 10b along with the rest of the Control register and then perform a subsequent write to set PINCFG as 11b. Likewise, when changing the value of PINCFG from 11b to 00b, you must perform an initial write to set PINCFG as 10b and then perform a subsequent write to update the rest of the Control register with the value of PINCFG as 00b. 00b - Shifter pin output disabled 01b - Shifter pin open-drain or bidirectional output enable 10b - Shifter pin bidirectional output data 11b - Shifter pin output
15-13 —	Reserved
12-8 PINSEL	Shifter Pin Select Selects the pin that is used by the shifter input or output. PINSEL = i selects the FXIO_Di pin. For pins configured as an output (PINCFG = 11b), this field takes effect when you write to the register.
7 PINPOL	Shifter Pin Polarity Specifies the shifter pin polarity. For pins configured as an output (PINCFG = 11b), this field takes effect when you write to this register.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Active high 1b - Active low
6-3 —	Reserved
2-0 SMOD	Shifter Mode Configures the mode of the shifter. 000b - Disable 001b - Receive mode; capture the current shifter content into SHIFTBUF on expiration of the timer 010b - Transmit mode; load SHIFTBUF contents into the shifter on expiration of the timer 011b - Reserved 100b - Match Store mode; shifter data is compared to SHIFTBUF content on expiration of the timer 101b - Match Continuous mode; shifter data is continuously compared to SHIFTBUF contents 110b - State mode; SHIFTBUF contents store programmable state attributes 111b - Logic mode; SHIFTBUF contents implement programmable logic lookup table

72.7.1.28 Shifter Configuration (SHIFTCFG0 - SHIFTCFG7)

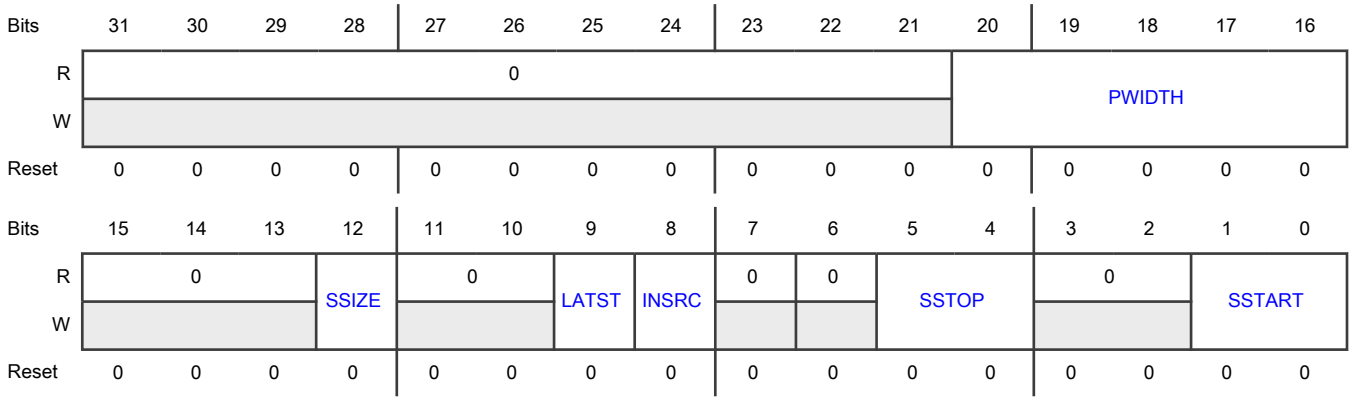
Offset

For n = 0 to 7:

Register	Offset
SHIFTCFGn	100h + (n × 4h)

Function
Provides fields for shifter configuration.

Diagram



Fields

Field	Function
31-21 —	Reserved
20-16 PWIDTH	Parallel Width Configures the number of bits to be shifted on each shift clock for all shifters: • 1-bit shift for PWIDTH = 0 • 2-bit shift for PWIDTH = 1 • 4-bit shift for PWIDTH = 2...3 • 8-bit shift for PWIDTH = 4...7 • 16-bit shift for PWIDTH = 8...15 • 32-bit shift for PWIDTH = 16...31 For shifters that support parallel transmit (SHIFTER0, SHIFTER4, ...) or parallel receive (SHIFTER3, SHIFTER7, ...), this field, together with PINSEL, also selects the pins to be driven or sampled on each shift clock: FXIO_D[PINSEL+PWIDTH]:FXIO_D[PINSEL]. Shifters that do not support parallel transmit or parallel receive only support parallel shift when SHIFTCFG*n*[INSRC] = 1. If SHIFTCTL*n*[SMOD] = 110b (State mode), use this field to disable state outputs (see State mode).
15-13 —	Reserved
12 SSIZE	Shifter Size Configures the size of the Shift registers. A 24-bit Shift register shifts data only into bits [23:0] and does not update bits [31:24] during shift operations. When the Shift register is configured for a 24-bit shift, configuring PWIDTH as 8..15 performs a 12-bit shift and PWIDTH as 16..31 performs a 24-bit shift.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - 32-bit 1b - 24-bit
11-10 —	Reserved
9 LATST	Late Store Configures what happens when a receive or match Shift register is configured to both shift and store on the same cycle. 0b - Store the pre-shift register state 1b - Store the post-shift register state
8 INSRC	Input Source Selects the input source for the shifter. Configuring this field as 1 is not supported for the last shifter. 0b - Pin 1b - Shifter n+1 output
7 —	Reserved
6 —	Reserved
5-4 SSTOP	Shifter Stop Allows automatic stop bit insertion, if the selected timer has also enabled a stop bit, when SHIFTCTL*n*[SMOD] is 10b (Transmit mode). If SHIFTCTL*n*[SMOD] is 1b or 100b (Receive mode or Match Store mode), this field allows automatic stop bit checking if the selected timer has also enabled a stop bit. If SHIFTCTL*n*[SMOD] is 110b (State mode), this field disables state outputs (see State mode). If SHIFTCTL*n*[SMOD] is 111b (Logic mode), this field masks logic pin inputs (see Logic mode). 00b - Stop bit disabled for Transmitter, Receiver, and Match Store modes 01b - Stop bit disabled for Transmitter, Receiver, and Match Store modes; when timer is in stop condition, Receiver and Match Store modes store receive data on the configured shift edge 10b - Transmitter mode outputs stop bit value 0 in Match Store mode; if stop bit is not 0, Receiver and Match Store modes set error flag (when timer is in stop condition, these modes also store receive data on the configured shift edge) 11b - Transmitter mode outputs stop bit value 1 in Match Store mode; if stop bit is not 1, Receiver and Match Store modes set error flag (when timer is in stop condition, these modes also store receive data on the configured shift edge)
3-2	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
1-0 SSTART	Shifter Start Allows automatic start bit insertion, if the selected timer has also enabled a start bit, when SHIFTCTL*n*[SMOD] is 10b (Transmit mode). If SHIFTCTL*n*[SMOD] = 1b (Receive mode) or 100b (Match Store mode), this field allows automatic start bit checking if the selected timer has also enabled a start bit. If SHIFTCTL*n*[SMOD] is 110b (State mode), this field disables state outputs (see State mode). If SHIFTCTL*n*[SMOD] = 111b (Logic mode), this field masks logic pin inputs (see Logic mode). 00b - Start bit disabled for Transmitter, Receiver, and Match Store modes; Transmitter mode loads data on enable 01b - Start bit disabled for Transmitter, Receiver, and Match Store modes; Transmitter mode loads data on first shift 10b - Transmitter mode outputs start bit value 0 before loading data on first shift; if start bit is not 0, Receiver and Match Store modes set error flag 11b - Transmitter mode outputs start bit value 1 before loading data on first shift; if start bit is not 1, Receiver and Match Store modes set error flag

72.7.1.29 Shifter Buffer (SHIFTBUF0 - SHIFTBUF7)

Offset

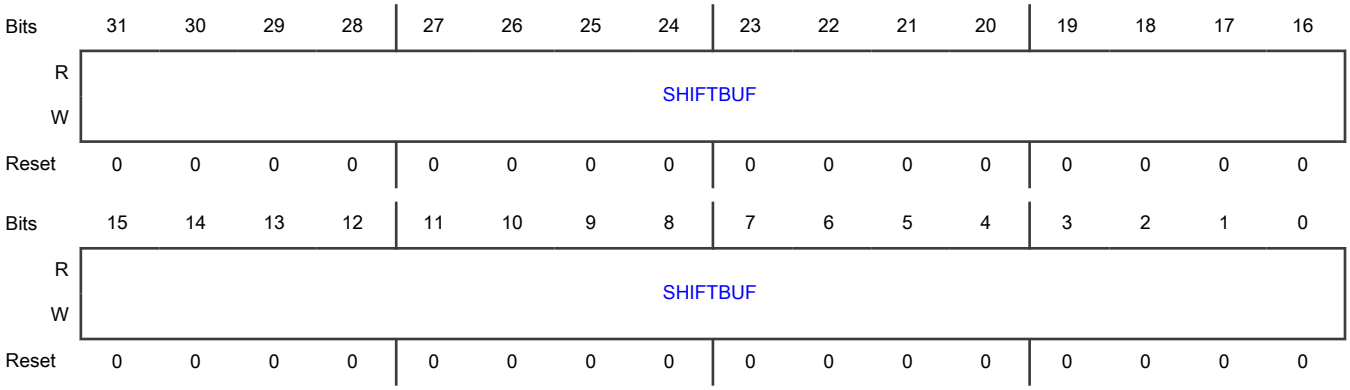
For n = 0 to 7:

Register	Offset
SHIFTBUFn	200h + (n × 4h)

Function

Contains shift buffer data.

Diagram



Fields

Field	Function
31-0 SHIFTBUF	Shift Buffer Contains the data to be matched with the shifter contents and is used for various other functions, depending on the setting of SHIFTCTL*n*[SMOD]: • If SHIFTCTL0[SMOD] is 1b (Receive mode), shifter data is transferred into SHIFTBUF at the expiration of the timer. You must read this register only when the corresponding SHIFTSTAT[SSF] flag is set, indicating that new shifter data is available. • If SHIFTCTL0[SMOD] is 10b (Transmit mode), SHIFTBUF data is transferred into the shifter before the timer begins. • If SHIFTCTL0[SMOD] is 100b (Match Store mode), SHIFTBUF[31:16] contains the data to be matched with the shifter contents and SHIFTBUF[15:0] can be used to mask the match result (1 = mask, 0 = no mask). The match is checked when the timer expires. Shifter data [31:16] is written to SHIFTBUF[31:16] whenever a match event occurs. You must read this register only when the corresponding shifter status flag is set, indicating that new shifter data is available. • If SHIFTCTL0[SMOD] is 101b (Match Continuous mode), SHIFTBUF[31:16] contains the data to be matched with the shifter contents, and SHIFTBUF[15:0] can be used to mask the match result (1 = mask, 0 = no mask). • If SHIFTCTL0[SMOD] is 111b (Logic mode), SHIFTBUF[31:0] implements a 5-input, 32-bit programmable logic lookup table (see Logic mode). • If SHIFTCTL0[SMOD] is 110b (State mode), use SHIFTBUF[31:24] to drive the output value when this shifter is selected by the current state pointer and use SHIFTBUF[23:0] to configure the value of the next state transition (see State mode).

72.7.1.30 Shifter Buffer Bit Swapped (SHIFTBUFBIS0 - SHIFTBUFBIS7)

Offset

For n = 0 to 7:

Register	Offset
SHIFTBUFBISn	280h + (n × 4h)

Function

Contains Shifter Buffer (SHIFTBUF0 - SHIFTBUF7) content, but it is bit-swapped.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SHIFTBUFBIS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SHIFTBUFBIS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 SHIFTBUFBIS	Shift Buffer Acts as an alias to Shifter Buffer (SHIFTBUF0 - SHIFTBUF7), but reads or writes to this register are bit-swapped. Reads return SHIFTBUF[0:31].

72.7.1.31 Shifter Buffer Byte Swapped (SHIFTBUFBYS0 - SHIFTBUFBYS7)

Offset

For n = 0 to 7:

Register	Offset
SHIFTBUFBYSn	300h + (n × 4h)

Function

Contains Shifter Buffer (SHIFTBUF0 - SHIFTBUF7) content, but it is byte-swapped.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SHIFTBUFBYS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SHIFTBUFBYS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 SHIFTBUFBYS	Shift Buffer Acts as an alias to Shifter Buffer (SHIFTBUF0 - SHIFTBUF7), but reads or writes to this register are byte-swapped. Reads return {SHIFTBUF[7:0], SHIFTBUF[15:8], SHIFTBUF[23:16], SHIFTBUF[31:24]}.

72.7.1.32 Shifter Buffer Bit Byte Swapped (SHIFTBUFBBS0 - SHIFTBUFBBS7)

Offset

For n = 0 to 7:

Register	Offset
SHIFTBUFBBSn	380h + (n × 4h)

Function

Contains the register data for Shifter Buffer (SHIFTBUF0 - SHIFTBUF7), but it is bit-swapped within each byte.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SHIFTBUFBBS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SHIFTBUFBBS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 SHIFTBUFBBS	Shift Buffer Acts as an alias to Shifter Buffer (SHIFTBUF0 - SHIFTBUF7), except that reads or writes to this register are bit-swapped within each byte. Reads return {SHIFTBUF[24:31], SHIFTBUF[16:23], SHIFTBUF[8:15], SHIFTBUF[0:7]}.

72.7.1.33 Timer Control (TIMCTL0 - TIMCTL7)

Offset

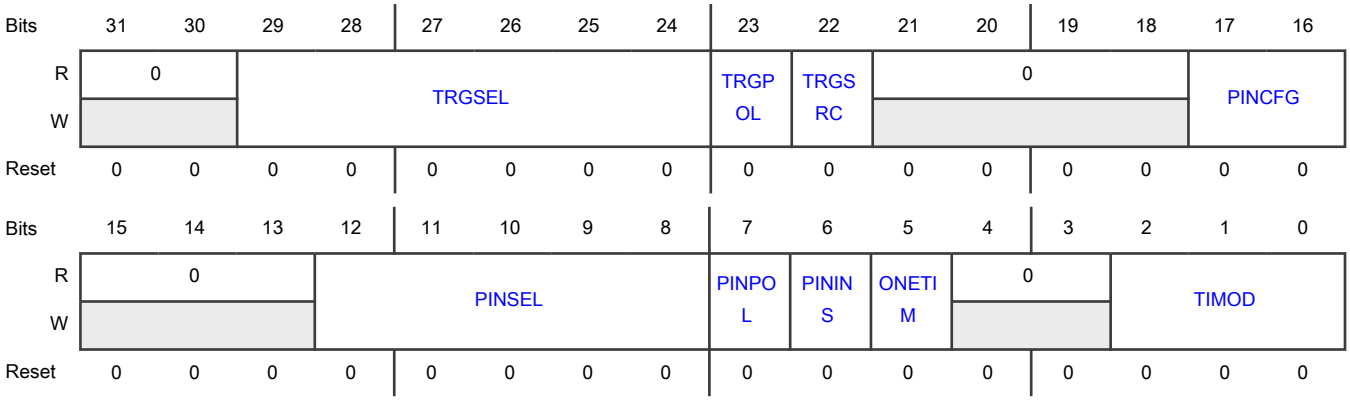
For n = 0 to 7:

Register	Offset
TIMCTLn	400h + (n × 4h)

Function

Controls various settings for timer *n*.

Diagram



Fields

Field	Function
31-30 —	Reserved
29-24 TRGSEL	Trigger Select Selects the trigger. The valid values for TRGSEL depend on the configuration of Parameter (PARAM): • If TRGSRC = 1, the valid values for *n* depend on the settings of PARAM[PIN], PARAM[TIMER], and PARAM[SHIFTER]. • If TRGSRC = 0, the valid values for *n* depend on PARAM[TRIGGER]. See the chip-specific FLEXIO information for external trigger selection. **NOTE** For a pin, *n* = 0 to 31, for a shifter, *n* = 0 to 7, and for a timer, *n* = 0 to 7. If TRGSRC = 0, configure the trigger selection as *n* = external trigger *n* input. If TRGSRC = 1, you can configure the internal trigger to select an input pin as 2×*n* = pin *n* input. If TRGSRC = 1, you can configure the internal trigger to select a shifter or timer signal as: • 4×*n* + 1 = shifter *n* status flag • 4×*n* + 3 = timer *n* trigger output Following are the values for expanded internal trigger selection (TRGSRC = 1):

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• 0000 = Pin 0 • 0001 = Shifter 0 flag • 0010 = Pin 1 • 0011 = Timer 0 trigger • 0100 = Pin 2 • 0101 = Shifter 1 flag • 0110 = Pin 3 • 0111 = Timer 1 trigger • ... • This continues up to pin 31, shifter 7, and timer 7.
23 TRGPOL	Trigger Polarity Specifies whether the trigger is active high or active low. 0b - Active high 1b - Active low
22 TRGSRC	Trigger Source Specifies whether the selected trigger source is external or internal. 0b - External 1b - Internal
21-18 —	Reserved
17-16 PINCFG	Timer Pin Configuration Configures the direction of the timer pin. For pins configured as an output (PINCFG = 11b), this field takes effect when you write to the register. **NOTE** When you initially configure PINCFG as 11b, FLEXIO may briefly drive the pin low. To avoid this, configure PINCFG as 10b along with the rest of the Control register and then perform a subsequent write to set the value of PINCFG as 11b. Likewise, when changing the value of PINCFG from 11b to 00b, you must perform an initial write to set PINCFG as 10b, and then perform a subsequent write to update the rest of the Control register with PINCFG as 00b. 00b - Timer pin output disabled 01b - Timer pin open-drain or bidirectional output enable 10b - Timer pin bidirectional output data 11b - Timer pin output

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-13 —	Reserved
12-8 PINSEL	Timer Pin Select Selects the pin that is used by the timer input or output. PINSEL = i selects the FXIO_Di pin. For pins configured as an output (PINCFG = 11b), this field takes effect when you write to the register.
7 PINPOL	Timer Pin Polarity Specifies the timer pin polarity. For pins configured as an output (PINCFG = 11b), this field takes effect when you write to the register. 0b - Active high 1b - Active low
6 PININS	Timer Pin Input Select Specifies what selects the timer pin input. If this field is 1, the timer input pin is different from the timer output pin. PINSEL must select an even-numbered pin when this field is 1, which means that the output pin is even-numbered and input pin is odd-numbered. 0b - PINSEL selects timer pin input and output 1b - PINSEL + 1 selects the timer pin input; timer pin output remains selected by PINSEL
5 ONETIM	Timer One Time Operation Configures the timer to perform a single enable or disable iteration. Clear the timer status flag for the timer to be enabled again. 0b - Generate the timer enable event as normal 1b - Block the timer enable event unless the timer status flag is clear
4-3 —	Reserved
2-0 TIMOD	Timer Mode Specifies the timer mode: • In 8-bit baud counter mode, the lower 8 bits of the counter and compare register are used to configure the baud rate of the timer shift clock. The upper 8 bits are used to configure the shifter bit count. • In 8-bit PWM high mode, the lower 8 bits of the counter and compare register are used to configure the high period of the timer shift clock. The upper 8 bits are used to configure the low period of the timer shift clock. The shifter bit count is configured using another timer or external signal. • In 16-bit counter mode, the full 16 bits of the counter and compare register are used to configure either the baud rate of the shift clock or the shifter bit count. • In 16-bit counter disable mode, the full 16 bits of the counter and compare register are used to configure either the baud rate of the shift clock or the shifter bit count.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• In 8-bit word counter mode, the lower 8 bits of the counter and compare register are used to configure the shifter bit count. The upper 8 bits are used to configure the shifter word count. • In 8-bit PWM low mode, the lower 8 bits of the counter and compare register are used to configure the low period of the timer shift clock. The upper 8 bits are used to configure the high period of the timer shift clock. Use another timer or external signal to configure the shifter bit count. • In 16-bit input capture mode, the inverted value of the 16-bit counter is latched into the compare register when a timer counter disable condition is detected (as configured by TIMCFG*n*[TIMDIS]). This also sets the timer status flag. The timer counter is immediately restarted from FFFFh. 000b - Timer disabled 001b - Dual 8-bit counters baud mode 010b - Dual 8-bit counters PWM high mode 011b - Single 16-bit counter mode 100b - Single 16-bit counter disable mode 101b - Dual 8-bit counters word mode 110b - Dual 8-bit counters PWM low mode 111b - Single 16-bit input capture mode

72.7.1.34 Timer Configuration (TIMCFG0 - TIMCFG7)

Offset

For n = 0 to 7:

Register	Offset
TIMCFGn	480h + (n × 4h)

Function

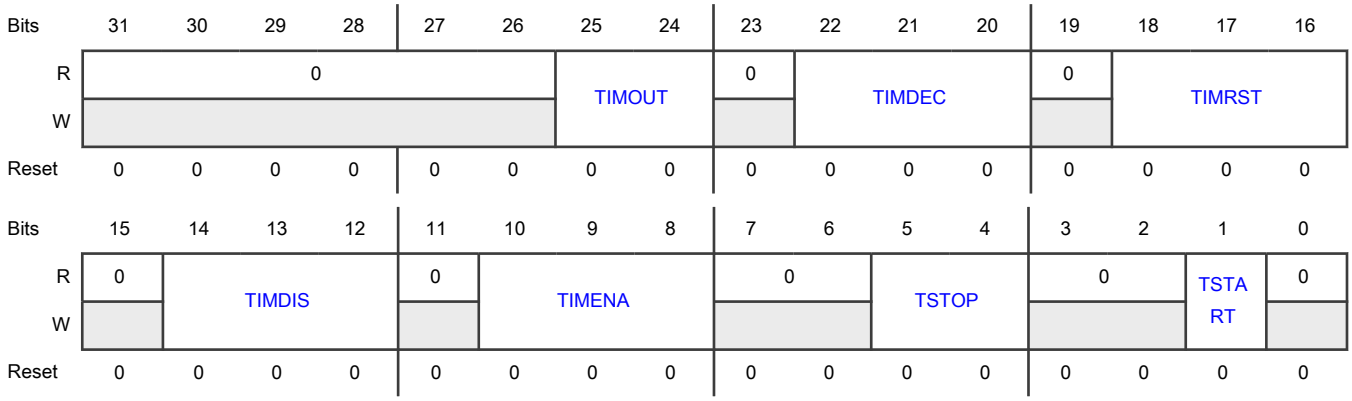
Controls various aspects of timer configuration.

The options to enable or disable the timer using the timer *n* - 1 enable or disable are reserved when *n* is evenly divisible by 4 (timer 0, for example).

NOTE

The pin and trigger level and edges specified in this register refer to the signal state after being modified by the settings of TIMCTL*n*[PINPOL] and TIMCTL*n*[TRGPOL]. For example, "trigger low" means that a trigger is actually at logic level 1 if TIMCTL*n*[TRGPOL] is 1 (active low).

Diagram



Fields

Field	Function
31-26 —	Reserved
25-24 TIMOUT	Timer Output Configures the initial state of the timer output and whether it is affected by the timer reset. 00b - Logic one when enabled; not affected by timer reset 01b - Logic zero when enabled; not affected by timer reset 10b - Logic one when enabled and on timer reset 11b - Logic zero when enabled and on timer reset
23 —	Reserved
22-20 TIMDEC	Timer Decrement Configures the source of the timer decrement and that of the shift clock. 000b - Decrement counter on FLEXIO clock; shift clock equals timer output 001b - Decrement counter on trigger input (both edges); shift clock equals timer output 010b - Decrement counter on pin input (both edges); shift clock equals pin input 011b - Decrement counter on trigger input (both edges); shift clock equals trigger input 100b - Decrement counter on FLEXIO clock divided by 16; shift clock equals timer output 101b - Decrement counter on FLEXIO clock divided by 256; shift clock equals timer output 110b - Decrement counter on pin input (rising edge); shift clock equals pin input 111b - Decrement counter on trigger input (rising edge); shift clock equals trigger input
19 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
18-16 TIMRST	Timer Reset Configures the condition that causes the timer counter (and optionally the timer output) to be reset. In 8-bit counter mode, the timer reset only resets the lower 8 bits that configure the baud rate. In all other modes, the timer reset resets full 16 bits of the counter. 000b - Never reset timer 001b - Timer reset on timer output high. 010b - Timer reset on timer pin equal to timer output 011b - Timer reset on timer trigger equal to timer output 100b - Timer reset on timer pin rising edge 101b - Reserved 110b - Timer reset on trigger rising edge 111b - Timer reset on trigger rising or falling edge
15 —	Reserved
14-12 TIMDIS	Timer Disable Configures the condition that causes the timer to be disabled and stop decrementing. 000b - Timer never disabled 001b - Timer disabled on timer n-1 disable 010b - Timer disabled on timer compare (upper 8 bits match and decrement) 011b - Timer disabled on timer compare (upper 8 bits match and decrement) and trigger low 100b - Timer disabled on pin rising or falling edge 101b - Timer disabled on pin rising or falling edge provided trigger is high 110b - Timer disabled on trigger falling edge 111b - Reserved
11 —	Reserved
10-8 TIMENA	Timer Enable Configures the condition that causes the timer to be enabled and start decrementing. 000b - Timer always enabled 001b - Timer enabled on timer n-1 enable 010b - Timer enabled on trigger high 011b - Timer enabled on trigger high and pin high

Table continues on the next page...

Table continued from the previous page...

Field	Function
	100b - Timer enabled on pin rising edge 101b - Timer enabled on pin rising edge and trigger high 110b - Timer enabled on trigger rising edge 111b - Timer enabled on trigger rising or falling edge
7-6 —	Reserved
5-4 TSTOP	Timer Stop Specifies whether the stop bit is enabled. The stop bit can be added on a timer compare (between each word) or on a timer disable. When stop bit is enabled, configured shifters output the contents of the stop bit when the timer is disabled. When stop bit is enabled on timer disable, the timer remains disabled until the next rising edge of the shift clock. If configured for both timer compare and timer disable, only one stop bit is inserted on timer disable. 00b - Disabled 01b - Enabled on timer compare 10b - Enabled on timer disable 11b - Enabled on timer compare and timer disable
3-2 —	Reserved
1 TSTART	Timer Start Specifies whether the start bit is enabled. If it is enabled, configured shifters output the contents of the start bit when the timer is enabled. The timer counter reloads from the compare register on the first rising edge of the shift clock. 0b - Disabled 1b - Enabled
0 —	Reserved

72.7.1.35 Timer Compare (TIMCMP0 - TIMCMP7)

Offset

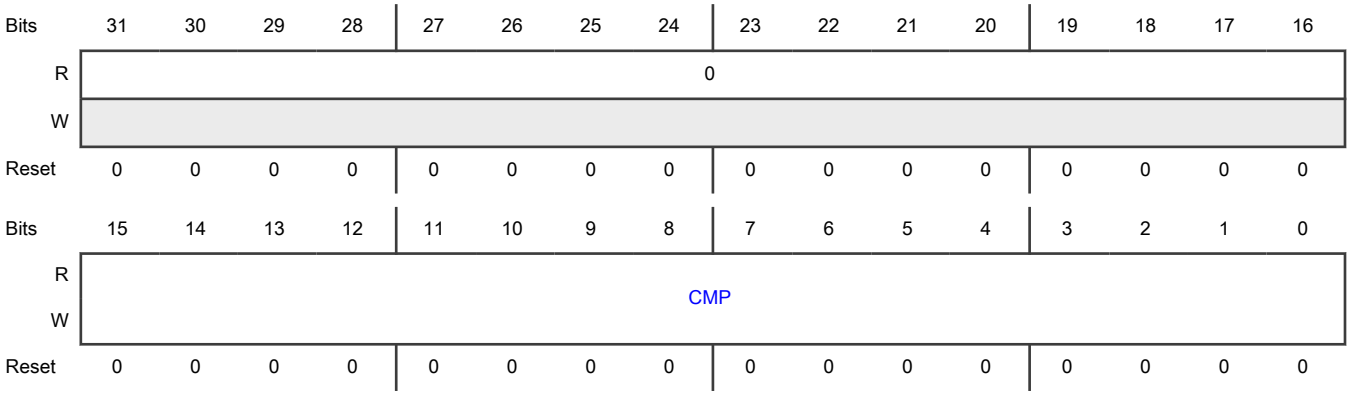
For n = 0 to 7:

Register	Offset
TIMCMPn	500h + (n × 4h)

Function

Contains the timer compare value.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 CMP	Timer Compare Value Loads into the timer counter when the timer is first enabled, when the timer is reset, and when the timer decrements down to zero. In 8-bit baud counter mode, the lower 8 bits configure the baud rate divider as (CMP[7:0] + 1) × 2. The upper 8 bits configure the number of bits in each word as (CMP[15:8] + 1) ÷ 2. In 8-bit PWM high mode, the lower 8 bits configure the high period of the output to (CMP[7:0] + 1) and the upper 8 bits configure the low period of the output to (CMP[15:8] + 1). In 16-bit counter mode, the compare value can be used to generate the baud rate divider (if shift clock source is timer output) as (CMP[15:0] + 1) × 2. When the shift clock source is a pin or trigger input, the compare register is used to set the number of bits in each word as (CMP[15:0] + 1) ÷ 2. In 16-bit counter disable mode, the compare value can be used to generate the baud rate divider (if shift clock source is timer output) as (CMP[15:0] + 1) × 2. When the shift clock source is a pin or trigger input, the compare register is used to set the number of bits in each word as (CMP[15:0] + 1) ÷ 2. In 8-bit word counter mode, the lower 8 bits configure the number of bits in each word as (CMP[7:0] + 1) ÷ 2. The upper 8 bits configure the number of words to transfer equal to (CMP[15:8] + 1) ÷ 2. In 8-bit PWM low mode, the lower 8 bits configure the low period of the output to (CMP[7:0] + 1) and the upper 8 bits configure the high period of the output to (CMP[15:8] + 1). In 16-bit input capture mode, the compare register is updated with the inverse of the timer counter value whenever the timer status flag is set. You must read this register only when the timer status flag is set.

72.7.1.36 Shifter Buffer Nibble Byte Swapped (SHIFTBUFNBS0 - SHIFTBUFNBS7)

Offset

For n = 0 to 7:

Register	Offset
SHIFTBUFNBSn	680h + (n × 4h)

Function

Contains Shifter Buffer (SHIFTBUF0 - SHIFTBUF7) content, but it is nibble-swapped within each byte.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SHIFTBUFNBS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SHIFTBUFNBS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 SHIFTBUFNBS	Shift Buffer Acts as an alias to Shifter Buffer (SHIFTBUF0 - SHIFTBUF7), but reads or writes to this register are nibble-swapped within each byte. Reads return {SHIFTBUF[27:24], SHIFTBUF[31:28], SHIFTBUF[19:16], SHIFTBUF[23:20], SHIFTBUF[11:8], SHIFTBUF[15:12], SHIFTBUF[3:0], SHIFTBUF[7:4]}.

72.7.1.37 Shifter Buffer Halfword Swapped (SHIFTBUFHWS0 - SHIFTBUFHWS7)

Offset

For n = 0 to 7:

Register	Offset
SHIFTBUFHWSn	700h + (n × 4h)

Function

Contains Shifter Buffer (SHIFTBUF0 - SHIFTBUF7) content, but it is halfword-swapped.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SHIFTBUFHWS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SHIFTBUFHWS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 SHIFTBUFHWS	Shift Buffer Acts as an alias to Shifter Buffer (SHIFTBUF0 - SHIFTBUF7), but reads or writes to this register are halfword-swapped. Reads return {SHIFTBUF[15:0], SHIFTBUF[31:16]}.

72.7.1.38 Shifter Buffer Nibble Swapped (SHIFTBUFNIS0 - SHIFTBUFNIS7)

Offset

For n = 0 to 7:

Register	Offset
SHIFTBUFNISn	780h + (n × 4h)

Function

Contains Shifter Buffer (SHIFTBUF0 - SHIFTBUF7) content, but it is nibble-swapped.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SHIFTBUFNIS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SHIFTBUFNIS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 SHIFTBUFNIS	Shift Buffer Acts as an alias to Shifter Buffer (SHIFTBUF0 - SHIFTBUF7), but reads or writes to this register are nibble-swapped. Reads return {SHIFTBUF[3:0], SHIFTBUF[7:4], SHIFTBUF[11:8], SHIFTBUF[15:12], SHIFTBUF[19:16], SHIFTBUF[23:20], SHIFTBUF[27:24], SHIFTBUF[31:28]}.

72.7.1.39 Shifter Buffer Odd Even Swapped (SHIFTBUFOES0 - SHIFTBUFOES7)

Offset

For n = 0 to 7:

Register	Offset
SHIFTBUFOESn	800h + (n × 4h)

Function

Contains Shifter Buffer (SHIFTBUF0 - SHIFTBUF7) content, but it has odd and even bits partitioned separately.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SHIFTBUFOES															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SHIFTBUFOES															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 SHIFTBUFOES	Shift Buffer Acts as an alias to Shifter Buffer (SHIFTBUF0 - SHIFTBUF7), but reads or writes to this register have the odd and even bits partitioned separately. Only 32-bit accesses are supported for this register. Reads return {SHIFTBUF[31], SHIFTBUF[29], SHIFTBUF[27],SHIFTBUF[25],SHIFTBUF[23],SHIFTBUF[21],SHIFTBUF[19],SHIFTBUF[17],SHIFTBUF[15],SHIFTBUF[13],SHIFTBUF[11],SHIFTBUF[9], SHIFTBUF[7], SHIFTBUF[5], SHIFTBUF[3], SHIFTBUF[1], SHIFTBUF[30], SHIFTBUF[28], SHIFTBUF[26], SHIFTBUF[24], SHIFTBUF[22], SHIFTBUF[20], SHIFTBUF[18], SHIFTBUF[16], SHIFTBUF[14], SHIFTBUF[12], SHIFTBUF[10], SHIFTBUF[8], SHIFTBUF[6], SHIFTBUF[4], SHIFTBUF[2], SHIFTBUF[0]}.

72.7.1.40 Shifter Buffer Even Odd Swapped (SHIFTBUFEOS0 - SHIFTBUFEOS7)

Offset

For n = 0 to 7:

Register	Offset
SHIFTBUFEOSn	880h + (n × 4h)

Function

Contains Shifter Buffer (SHIFTBUF0 - SHIFTBUF7) content, with even and odd bits partitioned separately.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SHIFTBUFEOS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SHIFTBUFEOS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 SHIFTBUFEOS	Shift Buffer Acts as an alias to Shifter Buffer (SHIFTBUF0 - SHIFTBUF7), but reads or writes to this register have the even and odd bits partitioned separately. Only 32-bit accesses are supported for this register. Reads return {SHIFTBUF[30], SHIFTBUF[28], SHIFTBUF[26], SHIFTBUF[24], SHIFTBUF[22], SHIFTBUF[20], SHIFTBUF[18], SHIFTBUF[16], SHIFTBUF[14], SHIFTBUF[12], SHIFTBUF[10], SHIFTBUF[8], SHIFTBUF[6], SHIFTBUF[4], SHIFTBUF[2], SHIFTBUF[0], SHIFTBUF[31], SHIFTBUF[29], SHIFTBUF[27],SHIFTBUF[25],SHIFTBUF[23],SHIFTBUF[21],SHIFTBUF[19],SHIFTBUF[17],SHIFTBUF[15],SHIFTBUF[13],SHIFTBUF[11],SHIFTBUF[9], SHIFTBUF[7], SHIFTBUF[5], SHIFTBUF[3], SHIFTBUF[1]}.

72.7.1.41 Shifter Buffer Halfword Byte Swapped (SHIFTBUFHBS0 - SHIFTBUFHBS7)

Offset

For n = 0 to 7:

Register	Offset
SHIFTBUFHBSn	900h + (n × 4h)

Function

Contains Shifter Buffer (SHIFTBUF0 - SHIFTBUF7) content, but it is halfword byte-swapped.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SHIFTBUFHBS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SHIFTBUFHBS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 SHIFTBUFHBS	Shift Buffer Acts as an alias to Shifter Buffer (SHIFTBUF0 - SHIFTBUF7), but reads or writes to this register are halfword byte-swapped. Reads return {SHIFTBUF[23:16], SHIFTBUF[31:24], SHIFTBUF[7:0], SHIFTBUF[15:8]}.

72.8 Glossary

Baud rate Number of bits that LPUART transmits or receives per second

Break character Break character is generated when the transmitter is holding the data line at the space level for at least 1 character time

Idle character Idle character is generated when the transmitter is holding the data line at logic 1 for at least 1 character time

PWM Pulse width modulation

Chapter 73
CAN (FlexCAN)

73.1 Chip-specific FlexCAN information

73.1.1 FlexCAN instances and configuration

This chip contains up to 12 instances of FlexCAN:

- FlexCAN_0
- FlexCAN_1
- FlexCAN_2
- FlexCAN_3
- FlexCAN_4
- FlexCAN_5
- FlexCAN_6
- FlexCAN_7
- FlexCAN_8
- FlexCAN_9
- FlexCAN_10
- FlexCAN_11

NOTE

- Message buffer, Enhanced RX FIFO, and Legacy RXFIFO structure of Flexcan is big endian whereas Cortex-M7 cores support little endian format. So, to align with the IP, there is an extra software overhead for endianness correction.
- The availability of 'Interrupt Masks 3 Register (IMASK3)' and 'Interrupt Flags 3 Register (IFLAG3)' registers for an instance depends upon the number of message buffers supported. See Table 491 for details.

Table 491. FlexCAN configuration and instances

Chip	Instances	Flexible Data Rate (CAN FD) protocol specification and CAN protocol.	Number of message buffers (assuming 8B payload)	Rx FIFO ID Filtering	ECC	External tick time	DMA	Enhanced RX FIFO feature (size)
S32K358, S32K356, S32K348, S32K338, S32K328, S32K388, S32K389	FlexCAN_0	Yes	96	Yes	Yes	Yes (PIT_RTI_2)	Yes	Yes (20)
	FlexCAN_1	Yes	96	Yes	Yes	Yes (PIT_RTI_2)	Yes	Yes (20)
	FlexCAN_2	Yes	96	Yes	Yes	Yes (PIT_RTI_2)	Yes	Yes (20)

Table continues on the next page...

Table 491. FlexCAN configuration and instances (continued)

Chip	Instances	Flexible Data Rate (CAN FD) protocol specification and CAN protocol.	Number of message buffers (assuming 8B payload)	Rx FIFO ID Filtering	ECC	External tick time	DMA	Enhanced RX FIFO feature (size)
	FlexCAN_3	Yes	64	Yes	Yes	Yes (PIT_RTI_2)	Yes	No
	FlexCAN_4	Yes	64	Yes	Yes	Yes (PIT_RTI_2)	Yes	No
	FlexCAN_5	Yes	64	Yes	Yes	Yes (PIT_RTI_2)	Yes	No
	FlexCAN_6	Yes	64	Yes	Yes	Yes (PIT_RTI_2)	Yes	No
	FlexCAN_7	Yes	64	Yes	Yes	Yes (PIT_RTI_2)	Yes	No
	FlexCAN_8 [1]	Yes	64	Yes	Yes	Yes (PIT_RTI_2)	Yes	No
	FlexCAN_9 [1]	Yes	64	Yes	Yes	Yes (PIT_RTI_2)	Yes	No
	FlexCAN_10 [1]	Yes	64	Yes	Yes	Yes (PIT_RTI_2)	Yes	No
	FlexCAN_11 [1]	Yes	64	Yes	Yes	Yes (PIT_RTI_2)	Yes	No
S32K314, S32K324, S32K344	FlexCAN_0	Yes	96	Yes	Yes	Yes (PIT_RTI_2)	Yes	Yes (20)
	FlexCAN_1	Yes	64	Yes	Yes	Yes (PIT_RTI_2)	Yes	No
	FlexCAN_2	Yes	64	Yes	Yes	Yes (PIT_RTI_2)	Yes	No
	FlexCAN_3	Yes	32	Yes	Yes	Yes (PIT_RTI_2)	Yes	No
	FlexCAN_4	Yes	32	Yes	Yes	Yes (PIT_RTI_2)	Yes	No
	FlexCAN_5	Yes	32	Yes	Yes	Yes (PIT_RTI_2)	Yes	No
S32K322, S32K342, S32K341	FlexCAN_0	Yes	64	Yes	Yes	Yes (PIT_RTI_2)	Yes	Yes (20)
	FlexCAN_1	Yes	64	Yes	Yes	Yes (PIT_RTI_2)	Yes	No

Table continues on the next page...

Table 491. FlexCAN configuration and instances (continued)

Chip	Instances	Flexible Data Rate (CAN FD) protocol specification and CAN protocol.	Number of message buffers (assuming 8B payload)	Rx FIFO ID Filtering	ECC	External tick time	DMA	Enhanced RX FIFO feature (size)
	FlexCAN_2	Yes	64	Yes	Yes	Yes (PIT_RTI_2)	Yes	No
	FlexCAN_3	Yes	32	Yes	Yes	Yes (PIT_RTI_2)	Yes	No
S32K312	FlexCAN_0	Yes	64	Yes	Yes	Yes (PIT_RTI_1)	Yes	Yes (20)
	FlexCAN_1	Yes	64	Yes	Yes	Yes (PIT_RTI_1)	Yes	No
	FlexCAN_2	Yes	64	Yes	Yes	Yes (PIT_RTI_1)	Yes	No
	FlexCAN_3	Yes	32	Yes	Yes	Yes (PIT_RTI_1)	Yes	No
	FlexCAN_4	Yes	32	Yes	Yes	Yes (PIT_RTI_1)	Yes	No
	FlexCAN_5	Yes	32	Yes	Yes	Yes (PIT_RTI_1)	Yes	No
S32K311, S32K310	FlexCAN_0	Yes	64	Yes	Yes	Yes (PIT_RTI_1)	Yes	Yes (20)
	FlexCAN_1	Yes	64	Yes	Yes	Yes (PIT_RTI_1)	Yes	No
	FlexCAN_2	Yes	64	Yes	Yes	Yes (PIT_RTI_1)	Yes	No

1. Only applicable for S32K389.

NOTE

See attached interrupt map file for the vector numbers for the CAN instances and their corresponding interrupt sources.

73.1.2 FlexCAN error injection

Error injection address mapping

Use the following table to convert from the memory map address to the location in the physical FlexCAN RAM (where pairs of values are provided, the first is the address for MCR[FDEN] negated, the second is for MCR[FDEN] asserted):

RAM contents	Memory map	Injection address							
—	—	FlexCAN0	FlexCAN0	FlexCAN0 /1/2	FlexCAN1 /2	FlexCAN3	FlexCAN4 /5	FlexCAN3 /4/5/6/7	FlexCAN 8/9/10/11
		S32K344	S32K342	S32K388/ S32K389	S32K3xx, except S32K358, S32K356, S32K348, S32K338, S32K338, S32K388, and S32K389	S32K344	S32K344	S32K388/ S32K389	S32K389
		S32K324	S32K341	S32K358/ S32K356		S32K324	S32K324	S32K358/ S32K356	
		S32K314	S32K322	S32K348		S32K314	S32K314	S32K348	
			S32K312	S32K338		S32K342	S32K312	S32K338	
			S32K311	S32K328		S32K341		S32K328	
						S32K322			
						S32K312			
FlexCAN Registers	—	Not mapped	Not mapped	Not mapped	Not mapped	Not mapped	Not mapped	Not mapped	Not mapped
MBs	0080h	0000h	0000h	0000h	0000h	0000h	0000h	0000h	0000h
RXIMRs	0880h	600h	400h	600h	400h	200h	200h	400h	400h
RXIFR_0	0A80h	600h	500h	600h	500h	280h	280h	500h	500h
RXIFR_1	0A84h	784h	504h	784h	504h	284h	284h	504h	504h
RXIFR_2	0A88h	788h	508h	788h	508h	288h	288h	508h	508h
RXIFR_3	0A8Ch	78Ch	50Ch	78Ch	50Ch	28Ch	28Ch	50Ch	50Ch
RXIFR_4	0A90h	790h	510h	790h	510h	290h	290h	510h	510h
RXIFR_5	0A94h	794h	514h	794h	514h	294h	294h	514h	514h
Reserved	0A98h	—	—	—	—	—	—	—	—
RXMGMASK	0AA0h	7A0h	520h	7A0h	520h	2A0h	2A0h	520h	520h
RXFGMASK	0AA4h	7A4h	524h	7A4h	524h	2A4h	2A4h	524h	524h
RX14MASK	0AA8h	7A8h	528h	7A8h	528h	2A8h	2A8h	528h	528h
RX15MASK	0AACh	7ACh	52Ch	7ACh	52Ch	2ACh	2Ach	52Ch	52Ch
Tx_SMB	0AB0h/ 0F28h	7B0h	530h	7B0h	530h	2B0h	2B0h	530h	530h
Rx_SMB0	0AC0h/ 0F70h	7C0h/ 7F8h	540h/ 578h	7C0h/ 7F8h	540h/ 578h	2C0h/ 2F8h	2C0h/ 2F8h	540h/ 578h	540h/ 578h
Rx_SMB1	0AD0h/ 0FB8h	7D0h/ 840h	550h/ 5C0h	7D0h/ 840h	550h/ 5C0h	2D0h/ 340h	2D0h/ 340h	550h/ 5C0h	550h/ 5C0h
Rx_SMB0_TIME_STAMP	0C20h	888h	608h	888h	608h	388h	388h	608h	608h

Table continues on the next page...

Table continued from the previous page...

RAM contents	Memory map	Injection address							
Rx_SMB1_TIME_STAMP	0C24h	88Ch	60Ch	88Ch	60Ch	38Ch	38Ch	60Ch	60Ch
HR_TIME_STAMP	0C30h	890h	610h	890h	610h	390h	390h	610h	610h
Enhanced RX FIFO	2000h	A10h	710h	A10h	—	—	—	—	—
ERFFEL	3000h	1050h	D50h	1050h	—	—	—	—	—

FlexCAN memory initialization

All FlexCAN memory must be initialized before starting its operation in order to have the parity bits in memory properly updated. CTRL2[WRMFRZ] grants write access to all memory positions that require initialization.

These locations are as listed in this table from 080h–ADFh and from C20h–31FFh.:

Chip	Instance	Offset address ranges to be initialized
S32K358/S32K356/S32K348/S32K338/ S32K328/S32K388/S32K389	FlexCAN0–FlexCAN2	080h–ADFh; C20h–31FFh
	FlexCAN3–FlexCAN7	080h–ADFh; C20h–FFFh
S32K389	FlexCAN8–FlexCAN11	080h–ADFh; C20h–FFFh
S32K344, S32K324, S32K314	FlexCAN0	080h–ADFh; C20h–31FFh
	FlexCAN1–FlexCAN5	080h–ADFh; C20h–FFFh
S32K342, S32K341, S32K322	FlexCAN0	080h–ADFh; C20h–31FFh
	FlexCAN1–FlexCAN3	080h–ADFh; C20h–FFFh
S32K312	FlexCAN0	080h–DFh; C20h–31FFh
	FlexCAN1–FlexCAN5	080h–ADFh; C20h–FFFh
S32K311	FlexCAN0	080h–ADFh; C20h–31FFh
	FlexCAN1–FlexCAN2	080h–ADFh; C20h–FFFh

NOTE

The reset value of CAN0_CTRL2 register for S32K312/S32K311 and S32K342/S32K322/S32K341 is 0080_0000h.

73.1.3 Reset value of MDIS bit

The CAN_MCR[MDIS] bit is set to 1 when the module is reset. Therefore, FlexCAN module is disabled following a reset. For details of CAN_MCR[MDIS] register refer to "CAN register description" section.

73.1.4 CAN Timestamp Implementation

FlexCAN uses Ethernet Media Access Controller (EMAC) and System Timer Module (STM_0) counters as time source. For details see 'FlexCAN timestamp implementation' section in Clocking chapter.

73.1.4.1 Considerations while using FlexCAN timestamp in S32K344

While using FlexCAN timestamp feature, the message buffers should be unlocked within each 20 CAN bits. A failure to do so while using FlexCAN timestamp feature alongwith FlexCAN RX FIFO or enhanced RX FIFO might result in timestamp information being missed.

73.1.5 CAN clock setup

For clock setup, see Clocking chapter.

73.2 Overview

FlexCAN is a communication controller implementing the CAN protocol according to the ISO 11898-1:2015 standard and CAN 2.0 Part B protocol specifications.

The CAN protocol was primarily designed to be used as a vehicle serial data bus, meeting the specific real-time processing and reliable operation requirements in the electromagnetic interference (EMI) environment of a vehicle. FlexCAN is a full implementation of the CAN protocol specification, the CAN with flexible data rate (CAN FD) protocol, and the CAN version 2.0 Part B protocol. It supports both standard and extended message frames and long payloads.

NOTE

Legacy Receive (RX) FIFO cannot be used in CAN Flexible Data (FD) mode. In CAN FD mode, use the Enhanced Receive FIFO feature instead of the Legacy Receive FIFO.

73.2.1 Block diagram

Figure 399 shows the main submodules implemented in FlexCAN.

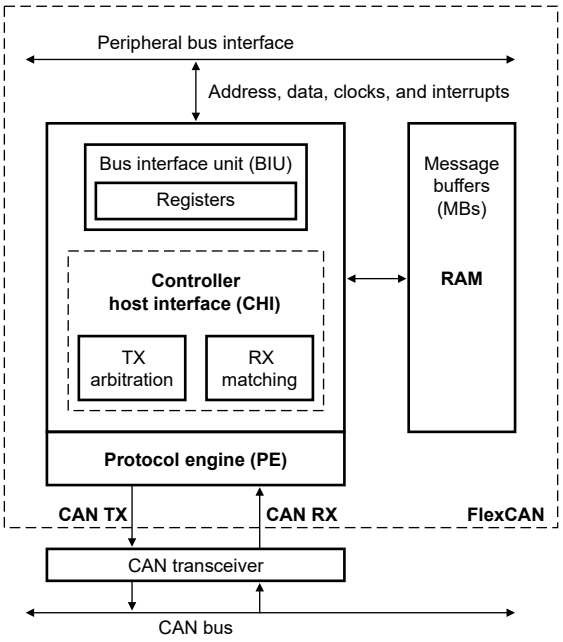


Figure 399. Block diagram

The Protocol Engine (PE) submodule manages the serial communication on the CAN bus:

- RAM access for receiving and transmitting message frames
- Receive message validation
- Error handling
- CAN FD message detection

Controller Host Interface (CHI) manages:

- Message buffer (MB) selection for reception and transmission
- Arbitration and ID matching algorithms for both CAN FD and non-CAN FD message formats

Bus Interface Unit (BIU) controls access to and from the internal interface bus to establish connection to the CPU and to other blocks. The BIU manages access to:

- Clocks
- Address and data buses
- Interrupt outputs
- DMA
- Test signals

73.2.2 Features

- Full implementation of *CAN with Flexible data rate (CAN FD) protocol specification* and *CAN Specification Version 2.0, Part B*:
  - Standard data frames
  - Extended data frames
  - Data length of 0–64 bytes
  - Programmable bit rate (see chip-specific FlexCAN information for specific maximum rate configuration)
  - Content-related addressing
- Compliance with the ISO 11898-1:2015 standard
- Flexible message buffers that can be configured to store a payload of 0, 8, 16, 32, or 64 bytes
  - Increasing the payload size decreases the number of supported message buffers (see FlexCAN memory partition for CAN FD).
  - Message buffers are configurable as receive or transmit, supporting standard and extended messages.
- Individual Receive Mask registers for each message buffer
- Full-featured Legacy RX FIFO with storage capacity for six frames and automatic internal pointer handling with DMA support
- Full-featured Enhanced RX FIFO with storage capacity of 20 CAN FD frames and automatic internal pointer handling with DMA support
- Transmission abort capability
- Optional general purpose RAM space, using RAM not used by reception or transmission structures
- Listen-Only mode
- Programmable loopback mode supporting self-test operation
- Programmable transmission priority scheme: lowest ID, lowest buffer number, or highest priority
- Timestamp based on 32-bit free-running timer, with optional external time tick
- Global network time, synchronized by specific message

- Maskable interrupts
- Independence from transmission medium (external transceiver is assumed)
- Short latency time due to arbitration scheme for high-priority messages
- Low-power mode
- Transceiver delay compensation when transmitting CAN FD messages at faster data rates
- Management of remote request frames, automatically or by software
- Restriction only to write CAN bit time settings and configuration bits in Freeze mode
- Transmit message buffer status (lowest-priority buffer or empty buffer)
- Identifier Acceptance Filter Hit Indicator (IDHIT) register for received frames
- ESR1[SYNCH] to indicate module is synchronous with CAN bus
- CRC status for transmitted message
- Legacy RX FIFO Global Mask register
- Selectable priority between message buffers and receive FIFO during matching process
- Powerful legacy RX FIFO ID filtering, capable of matching incoming IDs against either 128 extended IDs, 256 standard IDs, or 512 partial (8-bit) IDs, with 32 individual masking capability
- Powerful enhanced RX FIFO ID filtering, capable of matching incoming IDs against either 64 extended or 128 standard ID filter elements with three filtering schemes: mask plus filter, range, and two filters without mask
- Complete backward compatibility with previous FlexCAN version
- Detection and correction of errors in memory read accesses:
  - Each byte of FlexCAN memory is associated to five parity bits.
  - The error correction mechanism ensures that errors in one bit of this 13-bit word can be corrected (correctable errors).
  - Errors in two bits can be detected but not corrected (noncorrectable errors).

73.3 Functional description

FlexCAN is a CAN protocol engine with a flexible message buffer system for transmitting and receiving CAN frames. The system is a set of message buffers (MBs) that stores configuration and control data, timestamp, message ID, and data (see Message buffer structure). The memory corresponding to the first 38 message buffers can be configured to support a Legacy FIFO reception scheme with a powerful ID filtering mechanism. This scheme can check incoming frames against a table of IDs (up to 128 extended IDs or 256 standard IDs or 512 8-bit ID slices), with an individual mask register for up to 32 ID filter table elements.

For classical CAN frames, FlexCAN supports simultaneous reception through Legacy FIFO and message buffer. For CAN FD frames, FlexCAN supports reception through message buffer and enhanced receive FIFO.

For message buffer reception, a matching algorithm makes it possible to store received frames only into MBs that have the same ID programmed in the ID field. A masking scheme makes it possible to match the ID programmed on the message buffer with a range of IDs on received CAN frames. For transmission, an arbitration algorithm decides the prioritization of message buffers to be transmitted based on the message ID (optionally augmented by three local priority bits) or the message buffer ordering.

A message buffer is active at a given time if it can participate in both the matching and arbitration processes. A receive message buffer with a 0b code is inactive (see Table 528). A transmit message buffer with a 1000b or 1001b code is also inactive (see Table 529).

FlexCAN can receive and transmit messages in CAN FD format. The message buffers are sized to store the quantity of data bytes selected by FDCTRL[MBDSRn]. The quantity of FD message buffers available for a given quantity of data bytes is described in the CAN FD Control (FDCTRL) register. See also FlexCAN memory partition for CAN FD.

73.3.1 Modes of operation

FlexCAN has these functional modes:

Table 492. Functional modes

Mode	Description
Normal (User or Supervisor)	In Normal mode, FlexCAN receives and transmits message frames, manages errors normally, and enables all CAN Protocol functions. User and Supervisor modes differ in the access to some restricted control registers.
Freeze	Freeze mode is enabled when MCR[FRZ] = 1. If enabled, FlexCAN enters Freeze mode when MCR[HALT] is 1 or when is requested at chip level and FlexCAN writes 1 to MCR[FRZACK]. In this mode, no transmission or reception of frames is done, and synchronicity to the CAN bus is lost. See Freeze mode.
Loopback	FlexCAN enters this mode when CTRL1[LPB] becomes 1. In this mode, FlexCAN performs an internal loopback that can be used for self-test. The bit stream output of the transmitter is internally fed back to the receiver input. The receiving CAN input pin is ignored and the transmitting CAN output goes to the recessive state (logic 1). FlexCAN behaves as it normally does when transmitting and treats its own transmitted message as a message received from a remote node. To ensure proper reception of its own message, FlexCAN ignores the bit sent during the ACK slot in the CAN frame acknowledge field. Both transmit and receive interrupts are generated.
Listen-Only	FlexCAN enters this mode when CTRL1[LOM] becomes 1. In this mode, transmission is disabled, all error counters are frozen, and the module operates in a CAN Error Passive mode. Only messages acknowledged by another CAN station are received. If FlexCAN detects an unacknowledged message, it flags a BIT0 error (without changing the REC), as if it was trying to acknowledge the message.
CAN FD Active	In this mode, FlexCAN can transmit and receive all messages formatted according to the CAN FD Standard (2.0) and 2.0B Protocol in an interleaved fashion. The CPU can put FlexCAN into CAN FD Active mode by configuring MCR[FDEN] in Freeze mode.

Some features available in classical CAN are unavailable in CAN FD mode.

Table 493. Differences between classical CAN and CAN FD

Feature	Classical CAN	CAN FD
Legacy RX FIFO DMA	Yes	No
Legacy RX FIFO	Yes	No
Enhanced RX FIFO DMA	Yes	Yes
Enhanced RX FIFO	Yes	Yes

FlexCAN can operate in these low-power modes:

Table 494. Low-power modes

Mode	Description
Module Disable	FlexCAN enters this mode when the CPU writes 1 to MCR[MDIS] and FlexCAN writes 1 to MCR[LPMACK]. After FlexCAN is disabled, it issues a request to disable the clocks to the CAN Protocol Engine and Controller Host Interface submodules. Writing 0 to MCR[MDIS] exits this mode. See Module Disable mode.

73.3.1.1 Modes of operation details

FlexCAN has functional modes and low-power modes. See Modes of operation for an introductory description of all the modes of operation. The following subsections contain functional details about Freeze mode and low-power modes.

CAUTION

FlexCAN does not support "Permanent Dominant" failure on the CAN bus line. If a Low-Power request or Freeze mode request occurs during a "Permanent Dominant" condition, the corresponding acknowledgment field can never be 1.

73.3.1.1.1 Freeze mode

This mode is requested either by the CPU writing 1 to MCR[HALT] or when the chip is put into Debug mode. MCR[FRZ] must be 1 and the module must not be in a low-power mode.

When MECR[NCEFAFRZ] becomes 1 and a noncorrectable error is detected in a memory read access performed by FlexCAN internal processes (see Error response), FlexCAN also requests Freeze mode. This request occurs via both MCR[HALT] and MCR[FRZ] automatically becoming 1.

To obtain acknowledgment, FlexCAN writes 1 to MCR[FRZACK]. The CPU must only consider FlexCAN to be in Freeze mode when both request and acknowledgment conditions are satisfied.

When Freeze mode is requested, FlexCAN:

1. Waits to be in either Intermission, Error Passive, Bus Off, or Idle state.
2. Waits for all internal activities like arbitration, matching, move-in, and move-out to finish. A pending move-in does not prevent entering Freeze mode.
3. Ignores the receive input pin and drives the transmit pin as recessive.
4. Stops the prescaler, halting all CAN protocol activities.
5. Grants write access to the Error Counters register, which is read-only in other modes.
6. Writes 1 to MCR[NOTRDY] and MCR[FRZACK].

After requesting Freeze mode, you must wait for MCR[FRZACK] to become 1 before executing any other action, otherwise FlexCAN may operate unpredictably. In Freeze mode, all memory-mapped registers are accessible.

Freeze mode is exited in one of these conditions:

- CPU writes 0 to MCR[FRZ].
- The chip is removed from Debug mode or the MCR[HALT] becomes 0.

MCR[FRZACK] becomes 0 after the protocol engine recognizes the negation of the freeze request. After leaving Freeze mode, FlexCAN tries to resynchronize to the CAN bus by waiting for 11 consecutive recessive bits.

73.3.1.1.2 Module Disable mode

This low-power mode is normally used to disable a complete FlexCAN block temporarily, with no power consumption. The CPU requests this mode by writing 1 to MCR[MDIS], and FlexCAN acknowledges the request by writing 1 to MCR[LPMACK]. The CPU must only consider FlexCAN to be in Disable mode when both the request and acknowledgment conditions are satisfied.

If FlexCAN is disabled during Freeze mode, the module requests to disable the clocks to the PE and CHI submodules, writes 1 to MCR[LPMACK] and writes 0 to MCR[FRZACK].

If the module is disabled during transmission or reception, FlexCAN:

1. Waits to be in either Idle or Bus Off state, or waits for the third bit of Intermission and then checks that it is recessive.
2. Waits for all internal activities like arbitration, matching, move-in, and move-out to finish. FlexCAN does not take a pending move-in into account.
3. Ignores its receive input pin and drives its transmit pin as recessive.

4. Shuts down the clocks to the PE and CHI submodules.
5. Writes 1 to MCR[NOTRDY] and MCR[LPMACK].

In this mode, the Bus Interface Unit continues to operate, enabling the CPU to access memory-mapped registers, except for:

- The Receive Message Buffers Global Mask registers
- The Receive Buffer 14 Mask register
- The Receive Buffer 15 Mask register
- The Legacy Receive FIFO Global Mask register

When in Disable mode, these items may not be accessible:

- The Legacy Receive FIFO Information register
- The message buffers
- The Receive Individual Mask registers
- The reserved words within RAM

To exit this mode, the CPU writes 0 to MCR[MDIS], causing FlexCAN to request to resume the clocks and write 0 to MCR[LPMACK]. This write occurs after the CAN protocol engine recognizes the negation of disable mode requested by the CPU.

73.3.2 Transmission process

NOTE

Instances of MB_CS in this topic refer to items in message buffers. See Message buffer structure for details.

To transmit a CAN frame, the CPU must prepare a message buffer for transmission by executing the following procedure:

1. Check whether the respective interrupt flag is set, and clear it if necessary.
2. If the message buffer is active (transmission pending), request an abort of the transmission. Write the ABORT code (1001b) to the CODE field of the Control and Status word.
3. Poll the IFLAG register until the corresponding IFLAG flag is set, or wait for the interrupt request (if enabled by the respective IMASK field).
4. Read the CODE field to identify whether the transmission was aborted or transmitted (see Transmission abort mechanism).
5. Clear the corresponding interrupt flag.
6. Write the ID register (containing the local priority if enabled via MCR[LPRIOEN]).
7. Write payload data bytes.
8. Configure the Control and Status word as needed.
    a. Set ID type via MB_CS[IDE].
    b. Set Remote Transmission Request (if needed) via MB_CS[RTR].
    c. If MCR[FDEN] = 1, configure MB_CS[EDL] and MB_CS[BRS]. For details about the relationship between the written value and transmitted value of MB_CS[ESI], see Table 505.[12]
    d. Configure Data Length Code in bytes via MB_CS[DLC]. See Table 532 for detailed information.
    e. To transmit the CAN frame, activate the message buffer by writing Ch to MB_CS[CODE].

[12] There is no need to write the ESI field; it is automatically transmitted as dominant by error active nodes and as recessive by error passive nodes. If FlexCAN is operating as a network gateway, however, the CPU writes MB_CS[ESI] according to the error status of the node that sent the message.

NOTE

To maximize software performance, configure all the fields in the MB_CS word in a single 32-bit write operation. If the fields are configured in separate writes, MB_CS[CODE] must be the last write in the Control and Status word.

When the MB is activated, it participates in the arbitration process and its contents are eventually transmitted according to its priority. When the DLC value stored in the MB selected for transmission exceeds the MB payload size, FlexCAN completes the expected DLC by adding a constant CCh pattern.

After a successful transmission:

1. The value of the free-running timer is written into the timestamp field.
2. The CODE field in the Control and Status word is updated.
3. Both Cyclic Redundancy Check (CRCR) and CAN FD CRC (FDCRC) are updated.
4. A status flag is set in the Interrupt Flag register.
5. If allowed by the corresponding Interrupt Mask Register field, an interrupt is generated.

After transmission, the new CODE field depends on the code used to activate the message buffer (see Table 528 and Table 529 in Message buffer structure).

When the Abort feature is enabled (MCR[AEN] is 1), after the Interrupt flag is set for an MB configured as transmit buffer, the message buffer is blocked. The CPU cannot update the message buffer until the Interrupt Flag is cleared by the CPU. The CPU must clear the corresponding IFLAG flag before preparing this MB for a new transmission or reception.

NOTE

For backward compatibility (MCR[AEN] is 0), write the INACTIVE code (1000b) to the CODE field to deactivate the MB. In this case, the pending frame may be transmitted without notification (see Message buffer inactivation).

73.3.3 Arbitration process

The arbitration process scans the message buffers, searching for the transmission MB that holds the message to be sent at the next opportunity. This MB is called the arbitration winner. The scan starts from the lowest number MB and continues to the higher ones.

The arbitration process is triggered when at least one of the following occur:

- CRC field of the CAN frame arrives. The starting point depends on the value of CTRL2[TASD].
- Error Delimiter field of a CAN frame is in progress.
- Overload Delimiter field of a CAN frame is in progress.
- The winner of the arbitration is deactivated, and the CAN bus has not reached the first bit of the Intermission field.
- CPU writes to the Control and Status word of a winner MB, and the CAN bus has not reached the first bit of the Intermission field.
- CHI is in the Idle state and the CPU writes to the Control and Status word of any message buffer.
- FlexCAN exits Bus Off state.
- FlexCAN leaves Freeze mode or a low-power mode.

If the arbitration process does not evaluate all message buffers before the CAN bus reaches the first bit of the Intermission field, the temporary arbitration winner is invalidated. FlexCAN will not compete for the CAN bus at the next opportunity.

The arbitration process selects the winner among the active transmission message buffers at the end of the scan according to the values of CTRL1[LBUF] and MCR[LPRIOEN].

See Arbitration process (continued) for more information about this process.

73.3.3.1 Lowest-number message buffer first

If CTRL1[LBUF] is 1, the first (lowest number) active transmission message buffer found is the arbitration winner. MCR[LPRIOEN] has no effect when CTRL1[LBUF] is 1.

73.3.3.2 Highest-priority message buffer first

If CTRL1[LBUF] is 0, the arbitration process searches for the active transmission message buffer with the highest priority. The frame of this message buffer has a higher probability of winning the arbitration on the CAN bus when multiple external nodes compete for the bus simultaneously.

The sequence of bits considered for this arbitration is called the arbitration value of the message buffer. The transmission message buffer with the lowest arbitration value among all transmission message buffers has the highest priority.

If two or more message buffers have equivalent arbitration values, the message buffer with the lowest number is the arbitration winner.

The composition of the arbitration value depends on MCR[LPRIOEN].

73.3.3.2.1 Local priority disabled

If MCR[LPRIOEN] = 0, the arbitration value is built using the exact sequence of bits that would be transmitted in a CAN frame where local priority is disabled.

Table 495. Composition of the arbitration value when local priority is disabled

Format	Message buffer arbitration value (32 bits)				
Standard (IDE = 0)	Standard ID (11 bits)	RTR (1 bit)	IDE (1 bit)	— (18 bits)	— (1 bit)
Extended (IDE = 1)	Extended ID[28:18] (11 bits)	SRR (1 bit)	IDE (1 bit)	Extended ID[17:0] (18 bits)	RTR (1 bit)

73.3.3.2.2 Local priority enabled

To enable local priority, MCR[LPRIOEN] must be 1. In this case, the message buffer PRIO field (see Message buffer structure) is included at the very left of the arbitration value.

Table 496. Composition of the arbitration value when local priority is enabled

Format	Message buffer arbitration value (35 bits)					
Standard (IDE = 0)	PRIO (3 bits)	Standard ID (11 bits)	RTR (1 bit)	IDE (1 bit)	— (18 bits)	— (1 bit)
Extended (IDE = 1)	PRIO (3 bits)	Extended ID[28:18] (11 bits)	SRR (1 bit)	IDE (1 bit)	Extended ID[17:0] (18 bits)	RTR (1 bit)

Because the PRIO field is the most significant part of the arbitration value, message buffers with low PRIO values have higher priority than message buffers with high PRIO values. This priority is maintained regardless of the rest of their arbitration values.

The PRIO field is not part of the frame on the CAN bus. Its purpose is only to affect the internal arbitration process.

73.3.3.3 Arbitration process (continued)

After the arbitration winner is found (see Arbitration process), its content is copied to a hidden auxiliary message buffer called a transmit serial message buffer (TX SMB). The TX SMB has the same structure as a normal message buffer, but is not user-accessible. This copy operation is called move-out. After it is done, write access to the Control and Status word of the corresponding MB is blocked (if MCR[AEN] = 1). Write access is restored in one of the following events:

- The CPU clears the corresponding IFLAG flag after the message buffer is transmitted.
- FlexCAN enters Freeze mode or Bus Off state.
- FlexCAN loses the bus arbitration, or there is an error during the transmission.

At the first opportunity window on the CAN bus, the message on the TX SMB is transmitted according to the CAN protocol rules.

The arbitration process can be triggered under the following conditions:

- During RX and TX frames from CAN CRC field to end of frame. The value of CTRL2[TASD] may be changed to optimize the arbitration start point.
- During CAN Bus Off state from TX_ERR_CNT = 124 to 128. The value of CTRL2[TASD] may be changed to optimize the arbitration start point.
- During Control and Status write by CPU in Bus Idle. The first Control and Status write starts the arbitration process, and a second Control and Status write during this same arbitration restarts the process. If other Control and Status writes are performed, the transmission arbitration process is pending. If there is no arbitration winner after the arbitration process has finished, then the TX arbitration machine begins a new arbitration process. If there is a pending arbitration and Bus Idle state starts, then an arbitration process is triggered. In this case, the first and second Control and Status writes in Bus Idle do not restart the arbitration process. If there is not enough time to finish arbitration in the Wait for Bus Idle state and the next state is Idle, the scan is not interrupted. That scan is completed during Bus Idle state. During this arbitration, a Control and Status write does not cause an arbitration restart.
- Deactivation of an arbitration winner during a valid arbitration window.
- Upon exiting Freeze mode (first bit of the Wait for Bus Idle state). If a resynchronization occurs during the Wait for Bus Idle state, the arbitration process is restarted.

The arbitration process stops when:

- All message buffers are scanned.
- A transmission message buffer is found (if lowest buffer feature is enabled).
- An arbitration winner deactivates or aborts during any arbitration process.
- There is not enough time to finish the transmission arbitration process (for instance, when a deactivation is performed near the end of frame). In this case, the arbitration process is pending.
- An error or overload flag occurs in the bus.
- A low-power or Freeze mode request occurs in Idle state.

Arbitration is considered pending when:

- It is not possible to finish arbitration process in time.
- A Control and Status write occurs during arbitration, when that write is performed in an MB whose number is lower than the transmission arbitration pointer.
- Any Control and Status write occurs when there is no transmission arbitration process in progress.
- RX Match has updated an RX code to TX code.
- FlexCAN enters the Bus Off state.

If a Control and Status write is performed in the arbitration winner, a new process is restarted immediately.

If a Control and Status write is performed in an MB whose number is higher than the transmission arbitration pointer, the ongoing arbitration process scans this MB as normal.

73.3.4 Receive process

To be able to receive CAN frames into a message buffer, the CPU must prepare it for reception by executing the following steps:

1. If the message buffer is active (either TX or RX), deactivate it (see Message buffer inactivation), preferably with a safe deactivation (see Transmission abort mechanism).

2. Write the ID word into the message buffer.
3. To activate the message buffer, write the EMPTY code (0100b) to the CODE field of the Control and Status word. No setup is required for EDL, BRS, and ESI fields. The respective fields in the received message overwrite these fields.

After the MB is activated, it can receive frames that match the programmed filter. At the end of a successful reception, the move-in process updates the message buffer (see Move-in) as follows:

1. The received data field (up to eight bytes for Classical CAN message format and up to 64 bytes for CAN FD message format) is stored.
2. The received Identifier field is stored.
3. The value of the free-running timer when the second bit of the Identifier field of the frame is written into the Timestamp field of the MB.
4. The received SRR, IDE, RTR, EDL, BRS, ESI, and DLC fields are stored.
5. The CODE field in the Control and Status word is updated (see Table 528 and Table 529 in Section Message buffer structure).
6. If allowed by the corresponding Interrupt Mask field, a status flag is set in the Interrupt Flag register and an interrupt is generated.

The recommended way for the CPU to service (read) the frame received in a message buffer is:

1. Read the Control and Status word of that message buffer.
2. Verify that the BUSY bit is 0, indicating that the message buffer is locked. If it is not 0, repeat step 1 until it becomes 0. See Message buffer lock mechanism.
3. Read the contents of the message buffer. After the message buffer is locked, FlexCAN move-in processes do not modify its contents. See Move-in.
4. Acknowledge the proper flag in the IFLAG registers.
5. Unlock the message buffer by reading the free-running timer.

To verify frame reception, the CPU should poll the status flag bit for that specific message buffer in one of the IFLAG registers, not the CODE field of that message buffer. Polling the CODE field does not work in this case. After a frame is received and the CPU services the message buffer (by reading the Control and Status word and unlocking the message buffer), the CODE field does not return to EMPTY. It remains FULL, as explained in Table 528. If the CPU tries to work around this behavior by writing to the Control and Status word to force an EMPTY code after reading the message buffer without a prior safe deactivation, a newly received frame matching the filter of that message buffer may be lost.

CAUTION

In summary: never poll by reading the Control and Status word of the message buffers directly. Instead, read the IFLAG registers.

The Identifier field of the received frame is always stored in the matching message buffer. If the match was due to masking, the contents of the ID field in a message buffer may change. When MCR[SRXDIS] becomes 1, FlexCAN does not store frames transmitted by itself in any MB, even if it contains a matching receive MB. Also, no interrupt flag or interrupt signal is generated. Otherwise, when MCR[SRXDIS] becomes 0, if a matching receive MB exists, FlexCAN can receive frames transmitted by itself.

To be able to receive CAN frames through the Legacy RX FIFO, the CPU must enable and configure the Legacy RX FIFO during Freeze mode (see Legacy RX FIFO). Upon receiving the Frames Available in the Legacy RX FIFO interrupt (see MCR[BUF5I]), the CPU must service the received frame using the following procedure:

1. If a mask was used for IDE and RTR bits, read the Control and Status word.
2. If a mask was used, read the ID field.
3. Read the data field.
4. If the Identifier Acceptance Filter Hit Indicator (IDHIT) is required in the application, read Legacy RX FIFO Information (RXFIR).

5. Clear the Frames Available in Legacy RX FIFO interrupt by writing one to IFLAG1[BUF5I]. This step is mandatory; it releases the MB and allows the CPU to read the next RX FIFO entry.

When MCR[DMA]) becomes 1, upon receiving a frame in the Legacy FIFO, IFLAG1[BUF5I] generates a DMA request and does not generate a CPU interrupt (see Legacy RX FIFO in DMA operation). The IMASK1 fields in the Legacy RX FIFO region are not used.

The DMA controller must service the received frame using the following procedure:

1. Read the Control and Status word (read 80h address, optional).
2. Read the ID field (read 84h address, optional).
3. Read all data bytes (start read at 88h address, optional).
4. Read the last data bytes (read 8Ch address, mandatory).

73.3.5 Matching process

The matching process scans the message buffer memory for RX MBs programmed with the same ID as the one received from the CAN bus. If the Legacy RX or Enhanced RX FIFO is enabled, the priority of scanning can be selected between MBs and FIFO filters. The matching starts from the lowest number message buffer and continues toward the higher-numbered ones. If no match is found within the first structure, the other is scanned afterward. If the FIFO is full, the matching algorithm always looks for a matching MB outside the FIFO region.

For enhanced RX FIFO, see Enhanced RX FIFO.

For legacy RX FIFO, see Legacy RX FIFO.

As the frame is received, it is stored in a hidden auxiliary MB called Receive Serial Message Buffer (RX SMB).

The starting point of the matching process depends on the following conditions:

- If the received frame is a remote frame, the starting point is the CRC field of the frame.
- If the received frame is a data frame with DLC field equal to zero, the starting point is the CRC field of the frame.
- If the received frame is a data frame and the DLC field has a nonzero value, the starting point is the DATA field of the frame.

If a matching ID is found in the FIFO table or in one of the message buffers, the move-in process transfers the contents of the RX SMB to the FIFO or to the matched MB. If any CAN protocol error is detected, no match results are transferred to the FIFO or to the matched MB at the end of reception.

The matching process scans all matching elements of the RX FIFO (if enabled) and active receive MBs (CODE is EMPTY, FULL, OVERRUN, or RANSWER). The process searches for a successful comparison to the matching elements of the RX SMB that is receiving the frame on the CAN bus. The RX SMB has the same structure as a message buffer. The reception structures (RX FIFO or message buffers) associated with the matching elements that had a successful comparison are the matched structures. The matching winner is selected at the end of the scan among those matched structures. The matching winner depends on conditions described in Table 497.

Table 497. Matching architecture

Structure	SMB[RTR]	CTRL2[RRS]	CTRL2[EACEN]	MB[IDE]	MB[RTR]	MB[ID[1]]	MB[CODE]
Message buffer	0	—	0	cmp[2]	no_cmp[3]	cmp_msk[4]	EMPTY, FULL, or OVERRUN
Message buffer	0	—	1	cmp_msk	cmp_msk	cmp_msk	EMPTY, FULL, or OVERRUN

Table continues on the next page...

Table 497. Matching architecture (continued)

Structure	SMB[RTR]	CTRL2[RRS]	CTRL2[EACEN]	MB[IDE]	MB[RTR]	MB[ID[1]]	MB[CODE]
Message buffer	1	0	—	cmp	no_cmp	cmp	RANSWER
Message buffer	1	1	0	cmp	no_cmp	cmp_msk	EMPTY, FULL, or OVERRUN
Message buffer	1	1	1	cmp_msk	cmp_msk	cmp_msk	EMPTY, FULL, or OVERRUN
Legacy RX FIFO[5]	—	—	—	cmp_msk	cmp_msk	cmp_msk	—

1. For message buffer structure, if SMB[IDE] is 1, the ID is 29 bits (ID Standard plus ID Extended). If SMB[IDE] is 0, the ID is 11 bits (ID Standard). For Legacy RX FIFO structure, the ID depends on IDAM.
2. cmp: Compares the RX SMB contents to the MB contents regardless of masks.
3. no_cmp: The RX SMB contents are not compared to the MB contents.
4. cmp_msk: Compares the RX SMB contents to MB contents, accounting for masks.
5. SMB[IDE] and SMB[RTR] are not considered when IDAM is type C.

NOTE

For Enhanced RX FIFO, see Enhanced RX FIFO matching process.

A reception structure is free-to-receive when any of the following conditions is satisfied:

- The CODE field of the message buffer is EMPTY.
- The CODE field of the message buffer is either FULL or OVERRUN, and it has already been serviced (the CPU has read the Control and Status word and unlocked it as described in Message buffer lock mechanism).
- The CODE field of the message buffer is either FULL or OVERRUN and an inactivation (see Message buffer inactivation) is performed.
- The Legacy RX FIFO or Enhanced RX FIFO is not full.

The scan order for message buffers and Legacy Receive FIFO is from the matching element with the lowest number to the higher ones.

73.3.5.1 Matching priority

MCR[IRMQ] affects the matching winner search for MBs. If the field is 0, the matching winner is the first matched MB whether it is free-to-receive or not. If it is 1, the matching winner is selected according to this priority:

1. The first free-to-receive matched message buffer
2. The last non-free-to-receive matched message buffer

It is possible to select the priority of scan between MBs and Legacy RX FIFO or Enhanced RX FIFO with CTRL2[MRP].

If the selected priority is RX FIFO first:

- If the RX FIFO is a matched structure and is free-to-receive, the RX FIFO is the matching winner regardless of the scan for MBs.
- Otherwise, the matching winner is searched for among MBs as described above.

If the selected priority is MBs first:

- If a free-to-receive matched MB is found, it is the matching winner regardless of the scan for RX FIFO.

- If no matched MB is found, then the matching winner is searched for in the scan for the RX FIFO.
- If both conditions above are not satisfied and a non-free-to-receive matched MB is found, then the matching winner depends on the value of MCR[IRMQ]:
  - If MCR[IRMQ] = 0, the matching winner is the first matched MB.
  - If MCR[IRMQ] = 1, the matching winner is the RX FIFO if it is a free-to-receive matched structure. Otherwise, the matching winner is the last non-free-to-receive matched MB.

Table 498. Matching possibilities and resulting reception structures

RFEN or ERFEN	IRMQ	MRP	Matched in MB	Matched in FIFO	Reception structure	Description
No FIFO, only MB, match is always MB first						
0	0	X[1]	None[2]	—[3]	None	Frame lost by no match
0	0	X	Free[4]	—	First MB	—
0	1	X	None	—	None	Frame lost by no match
0	1	X	Free	—	First MB	—
0	1	X	Not free	—	Last MB	Overrun
FIFO enabled, no match in FIFO; is as if FIFO does not exist						
1	0	X	None	None[5]	None	Frame lost by no match
1	0	X	Free	None	First MB	—
1	1	X	None	None	None	Frame lost by no match
1	1	X	Free	None	First message buffer	—
1	1	X	Not free	None	Last message buffer	Overrun
FIFO enabled; queue disabled						
1	0	0	X	Not full[6]	FIFO	—
1	0	0	None	Full[7]	None	Frame lost by FIFO full (FIFO overflow)
1	0	0	Free	Full	First MB	—
1	0	0	Not free	Full	First MB	
1	0	1	None	Not full	FIFO	
1	0	1	None	Full	None	Frame lost by FIFO full (FIFO overflow)
1	0	1	Free	X	First message buffer	—

Table continues on the next page...

Table 498. Matching possibilities and resulting reception structures (continued)

RFEN or ERFEN	IRMQ	MRP	Matched in MB	Matched in FIFO	Reception structure	Description
1	0	1	Not free	X	First message buffer	Overrun
FIFO enabled; queue enabled						
1	1	0	X	Not full	FIFO	—
1	1	0	None	Full	None	Frame lost by FIFO full (FIFO overflow)
1	1	0	Free	Full	First message buffer	—
1	1	0	Not free	Full	Last message buffer	Overrun
1	1	1	None	Not full	FIFO	—
1	1	1	Free	X	First message buffer	—
1	1	1	Not free	Not full	FIFO	—
1	1	1	Not free	Full	Last message buffer	Overrun

1. Indicates a don't care condition.
2. Matched in message buffer "None" means that the frame has not matched any message buffer (free-to-receive or non-free-to-receive).
3. This condition is forbidden.
4. Matched in message buffer "Free" means that the frame matched at least one message buffer free-to-receive regardless of whether it has matched MBs that are non-free-to-receive.
5. Matched in FIFO, "None" means that the frame has not matched any filter in FIFO. It is as if the FIFO did not exist (CTRL2[RFFN] = 0 and ERFCR[ERFEN] = 0).
6. Matched in FIFO, "Not full" means that the frame has matched a FIFO filter and has empty slots to receive it.
7. Matched in FIFO, "Full" means that the frame matched a FIFO filter but could not store it, because it has no empty slots to receive it.

73.3.5.2 Special cases

If a non-safe MB inactivation (see Message buffer inactivation) occurs during the matching process and the inactivated MB is the temporary matching winner, the temporary matching winner is invalidated. The matching elements scan is not stopped and not restarted; it continues normally. The consequence is that the current matching process works as if the matching elements compared before the inactivation did not exist. In this case, a message may be lost.

Consider an example where:

- The FIFO is disabled.
- MCR[IRMQ] is 1.
- There are two message buffers with the same ID: the second and fifth MBs in the array.
- FlexCAN starts receiving messages with that ID.

When the first message arrives, the matching algorithm finds the first match in message buffer number 2. The code of this message buffer is EMPTY, so the message is stored in that MB. When the second message arrives, the matching algorithm finds MB number 2 again, but it is not "free-to-receive." It continues looking, finds MB number 5, and stores the message in that MB.

If yet another message with the same ID arrives, the matching algorithm finds no matching free-to-receive MBs, so it overwrites the last matched message buffer (MB number 5). In doing so, it updates the CODE field of the message buffer to OVERRUN.

The ability to match the same ID in more than one MB can be used to implement a reception queue (in addition to the full-featured FIFO) to allow more time for the CPU to service the MBs. By programming more than one MB with the same ID, received messages are queued into the message buffers. The CPU can examine the Timestamp field of the message buffers to determine the order in which the messages arrived.

Matching a range of IDs is possible via ID acceptance masks. FlexCAN supports individual masking per message buffer (see Receive Individual Mask (RXIMR0 - RXIMR95)). During the matching algorithm, if a mask field is 1, the corresponding ID bit is compared. If the mask field is 0, the corresponding ID bit is a "don't care". Individual Mask Registers are implemented in RAM, so they are not initialized out of reset. Also, they can only be programmed when the module is in Freeze mode; otherwise, the module blocks them.

FlexCAN also supports an alternate masking scheme with only Legacy RX FIFO Global Mask (RXFGMASK), RX Message Buffers Global Mask (RXMGMASK), Receive 14 Mask (RX14MASK), and Receive 15 Mask (RX15MASK) for backward compatibility with legacy applications. This alternate masking scheme is enabled when the MCR[IRMQ] = 0.

73.3.6 Move process

There are two types of move process: move-in and move-out.

73.3.6.1 Move-in

The move-in process is the copying of a message received by an RX SMB to an RX message buffer or FIFO that has matched it. If the move destination is the Legacy RX FIFO, attributes of the message are also copied to the CAN_RXFIR FIFO. Each RX SMB has its own move-in process, but only one is performed at a given time. The move-in starts only when the message held by the RX SMB has a corresponding match (see Matching process) and all of the following conditions are true:

- The CAN bus has reached or already gone past:
  - The second bit of Intermission field next to the frame that carried the message that is in the RX SMB.
  - The first bit of an overload frame next to the frame that carried the message in the RX SMB.
- There is no ongoing matching process.
- The CPU is not locking the destination message buffer.
- No move-in process from another RX SMB is ongoing. If more than one move-in process is to be started at the same time, both are performed and the newest process substitutes for the oldest.

The term "pending move-in" is used throughout the documentation and stands for a move-to-be that does not satisfy all of the above conditions.

If any of the following conditions is satisfied, the move-in is canceled and the RX SMB is able to receive another message:

- The destination message buffer is inactivated after the CAN bus has reached the first bit of the Intermission field next to the frame that carried the message. Also, its matching process has finished.
- There is a previous pending move-in to the same destination message buffer.
- The RX SMB is receiving a frame transmitted by FlexCAN itself and self-reception is disabled (MCR[SRXDIS] = 1).
- Any CAN protocol error is detected.

If the module enters Freeze or Low-Power mode, the pending move-in is not canceled. It remains on hold, waiting for Freeze and Low-Power mode to be exited and for the module to be unlocked. If a message buffer is unlocked during Freeze mode, the move-in occurs immediately.

The move-in process is FlexCAN executing the following steps:

1. Push IDHIT into the RXFIR FIFO if the message is destined for the Legacy RX FIFO.
2. Read all data words from the RX SMB in accordance with the selected payload size for the RX storage element.

3. Write all data words to the RX message buffer according to the selected payload size for the RX storage element. If the data size of the storage element is smaller than the original payload size described in the DLC field of the message, the payload is truncated. The high-order bytes that do not fit the destination size are lost.
4. Read the Control and Status and ID words from the RX SMB.
5. Write the Control and Status and ID words to the RX message buffer, and update the CODE field.

The move-in process is not atomic; the inactivation of the destination message buffer immediately cancels it (see Message buffer inactivation). In this case, the message buffer may remain partially updated, and therefore incoherent. When the move-in destination is a Legacy or Enhanced RX FIFO message buffer, however, the process cannot be canceled.

To alert the CPU that the message buffer content is temporarily incoherent, the BUSY Bit (least significant bit of the CODE field) of the destination message buffer becomes 1 during move-in.

73.3.6.2 Move-out

The move-out process is the copying of content from a TX message buffer to the TX SMB when a message for transmission is available (see Arbitration process). The move-out occurs in the following conditions:

- In the first bit of the Intermission field
- During the Bus Off state, when the TX error counter is in the 124 to 128 range
- During the Bus Idle state
- During the Wait for Bus Idle state

The move-out process is not atomic. Only the CPU has priority to access the memory concurrently outside the Bus Idle state. In Bus Idle, the move-out has the lowest priority of the concurrent memory accesses.

73.3.7 Data coherence

In order to maintain data coherency and proper FlexCAN operation, the CPU must obey the rules described in Transmission process and Receive process.

73.3.7.1 Transmission abort mechanism

The abort mechanism provides a safe way to request the abortion of a pending transmission. A feedback mechanism is provided to inform the CPU whether the transmission was aborted or the frame could not be aborted and was transmitted instead.

These primary conditions must be fulfilled in order to abort a transmission:

- MCR[AEN] must be 1.
- The first CPU action must be the writing of abort code (1001b) into the CODE field of the Control and Status word.

Active message buffers configured for transmission must be aborted before they can be updated. The write operation is blocked and the transmission is not disturbed when the abort code is written to:

- A message buffer currently being transmitted.
- A message buffer that was already loaded into the TX SMB for transmission.

In this case, the abort request is captured and kept pending until one of the following conditions is satisfied:

- The module loses the bus arbitration.
- There is an error during the transmission.
- The module is put into Freeze mode.
- The module enters the Bus Off state.
- There is an overload frame.

If none of the conditions above are reached:

1. The message buffer is transmitted correctly.
2. The interrupt flag is set in the proper IFLAG register.
3. If enabled, an interrupt to the CPU is generated.

The abort request is automatically cleared when the interrupt flag is set. If only one of the above conditions is reached, the frame is not transmitted. In this case:

1. The abort code is written into the CODE field.
2. The interrupt flag is set in the proper IFLAG register.
3. Optionally, an interrupt is generated to the CPU.

If the CPU writes the abort code before the transmission begins internally, the write operation is not blocked. The MB is updated and the interrupt flag is set. In this way, the CPU only needs to read the abort code to verify that the active MB was safely inactivated. In this case, although MCR[AEN] = 1 and the CPU wrote the abort code, the MB is inactivated and not aborted, because the transmission did not start yet. A message buffer is aborted only when the abort request is captured and kept pending until one of the previous conditions is satisfied.

73.3.7.2 Message buffer inactivation

Inactivation protects the message buffer against updates by FlexCAN internal processes. It allows the CPU to rely on message buffer data coherence after having updated it, even in Normal mode.

Inactivation of transmission message buffers must be performed only when MCR[AEN] = 0.

If a message buffer is inactivated, it does not participate in the arbitration process or the matching process until it is reactivated. See Transmission process and Receive process for detailed instructions on how to inactivate and reactivate a message buffer.

To inactivate a message buffer, the CPU must update its CODE field to INACTIVE (either 0b or 1000b).

Because you cannot synchronize the CODE field update with FlexCAN internal processes, an inactivation can have the following consequences:

- A frame in the bus that matches the filtering of an inactivated RX message buffer may be lost without notice. This loss can occur even if there are other message buffers with the same filter.
- A frame containing the message within an inactivated TX message buffer may be transmitted without setting the respective IFLAG flag.

To perform a safe inactivation and avoid the above consequences for TX message buffers, the CPU must use the transmission abort mechanism (see Transmission abort mechanism).

The inactivation automatically unlocks the message buffer (see Message buffer lock mechanism).

NOTE

Message buffers that are part of the Legacy RX FIFO or Enhanced RX FIFO cannot be inactivated. There is no write protection on the Legacy FIFO region by FlexCAN. The CPU must maintain data coherency in the Legacy FIFO region when MCR[RFEN] = 1.

73.3.7.3 Message buffer lock mechanism

In addition to message buffer inactivation, FlexCAN uses a message buffer lock mechanism to maintain data coherence for the receive process. When the CPU reads the Control and Status word of an RX message buffer with codes FULL or OVERRUN, FlexCAN is configured to allow the CPU to read the whole message buffer in an atomic operation. FlexCAN sets an internal lock flag for that message buffer.

The lock is released in any of the following events:

- The CPU reads the free-running timer (global unlock operation).
- The CPU reads the Control and Status word of another message buffer, regardless of its code.

- The CPU writes into the Control and Status word. This procedure is not recommended for normal unlocking, because it cancels a pending move and may lose a received message.

The message buffer lock prevents a new frame from being written into the message buffer when the CPU is reading it.

NOTE

The locking mechanism applies only to RX message buffers that are not part of the Legacy RX FIFO and have a code other than INACTIVE (0b) or EMPTY[1] (0100b). TX message buffers cannot be locked.

Consider an example where:

- The Legacy RX FIFO is disabled.
- The second and the fifth message buffers of the array are programmed with the same ID.
- FlexCAN has already received and stored messages into these two message buffers.
- The CPU reads message buffer number 5 while another message with the same ID is arriving.

When the CPU reads the Control and Status word of message buffer number 5, this message buffer is locked. The new message arrives and the matching algorithm finds no free-to-receive message buffers, so it overrides message buffer number 5. This message buffer is locked, so the new message cannot be written to it. The message remains in the RX SMB until the message buffer is unlocked, and only then is it written to the message buffer.

If the message buffer remains locked and another new message with the same ID arrives, the new message overwrites the one in the RX SMB. There is no indication of lost messages in the CODE field of the message buffer or in Error and Status 1 (ESR1).

When the message is moved from the RX SMB to the message buffer, the BUSY bit on the CODE field becomes 1. If the CPU reads the Control and Status word and identifies that the BUSY bit is 1, it must wait until the BUSY bit becomes 0 to access the MB.

NOTE

If the BUSY bit is 1 or the message buffer is empty, reading the Control and Status word does not lock the message buffer.

Inactivation takes precedence over locking. If the CPU inactivates a locked receive message buffer, then its lock status is negated and the message buffer is marked as invalid for the current matching round. Any pending message on the RX SMB is not transferred to the message buffer. A message buffer is unlocked when the CPU reads Free-Running Timer (TIMER) or the Control and Status word of another message buffer.

The lock and unlock mechanisms have the same functionality in Normal and Freeze modes.

An unlock during Normal or Freeze mode results in the move-in of the pending message. If unlocking occurs during a low-power mode, however, the move-in is postponed (see Modes of operation). Move-in takes place only when the module returns to Normal or Freeze mode.

73.3.8 Enhanced RX FIFO

FlexCAN supports an enhanced RX FIFO engine which can store up to 20 CAN FD messages. The region 2000h–204Fh contains the output of the FIFO, which the CPU should read. To enable the enhanced RX FIFO, write 1 to ERFCR[ERFEN]. FlexCAN has two FIFO options, Legacy RX FIFO and Enhanced RX FIFO, but both options cannot be enabled at the same time. See Legacy RX FIFO for additional information.

To configure the enhanced RX FIFO watermark, write a value to ERFCR[ERFWM]. If ERFCR[ERFWM] is configured, the CPU is notified only if a minimum number of messages is stored in the FIFO. When the number of stored messages is greater than the value in ERFCR[ERFWM], the module sets ERFSR[ERFWMI]. Optionally, if MCR[DMA] or ERFIER[ERFWMIIE] are enabled, a DMA transfer or an interrupt can be triggered, respectively.

For the enhanced RX FIFO to receive, the CPU must execute the configuration procedure below. If the CPU must change any configurations of the Enhanced RX FIFO, the same procedure must be followed.

[1] In previous FlexCAN versions, reading the Control and Status word locked the message buffer even when CODE indicates it is EMPTY. This behavior is maintained when the IRMQ bit is 0.

Prerequisites

MCR[RFEN] must be 0.

Procedure

Step	Purpose	Programming	Notes
1	Enter Freeze mode.	See Freeze mode.	—
2	If enhanced RX FIFO is not already enabled, enable it.	Write 1 to ERFCR[ERFEN].	—
3	Reset enhanced RX FIFO engine.	Write 1 to ERFSR[ERFCLR].	—
4	If the enhanced RX FIFO error flags are set, clear them.	Write 1 to these flags: • ERFSR[ERFUFW] • ERFSR[ERFOVF] • ERFSR[ERFWMI] • ERFSR[ERFDA]	—
5	Specify the total number of enhanced RX FIFO filter elements to be used in Enhanced RX FIFO reception.	Write the number to ERFCR[NFE].	—
6	Specify the number of extended ID and standard ID filter elements to be used in Enhanced RX FIFO reception.	Write the number to ERFCR[NEXIF].	ERFCR[NEXIF] ≤ ERFCR[NFE] + 1.
7	If you are using DMA, enable DMA.	Write 1 to MCR[DMA].	—
8	If you are using DMA, specify the number of words to transfer for each Enhanced RX FIFO data element.	Write the number to ERFCR[DMALW].	—
9	Specify the Enhanced RX FIFO watermark.	Write the number to ERFCR[ERFWM].	If MCR[DMA] = 1, ERFCR[ERFWM] should be 0h.
10	If you are using interrupts, enable the interrupts.	Write 1 to the interrupt enables in Enhanced RX FIFO Interrupt Enable (ERFIER).	—
11	Configure the filter elements.	Write to the ERFFEL*n* registers.	ERFFEL*n* registers are implemented in RAM; you must explicitly initialize them before prior any reception.
12	Exit Freeze mode.	See Freeze mode.	—

There are two types of enhanced RX FIFO filter elements that can be stored in ERFFEL*n* registers: extended-ID filter elements and standard-ID filter elements. Each extended-ID filter element is stored in two ERFFEL*n* registers, and each standard-ID filter element is stored in one ERFFEL*n* register. ERFCR[NFE] defines the total number of Enhanced RX FIFO filter elements.

In addition, the filter memory space can be split into two regions: one for extended-ID filter elements and another for standard-ID filter elements, according to ERFCR[NEXIF]. Figure 400 shows how the enhanced RX filter elements are defined. See Enhanced RX FIFO matching process for information about the Enhanced RX FIFO matching process and filter element formats.

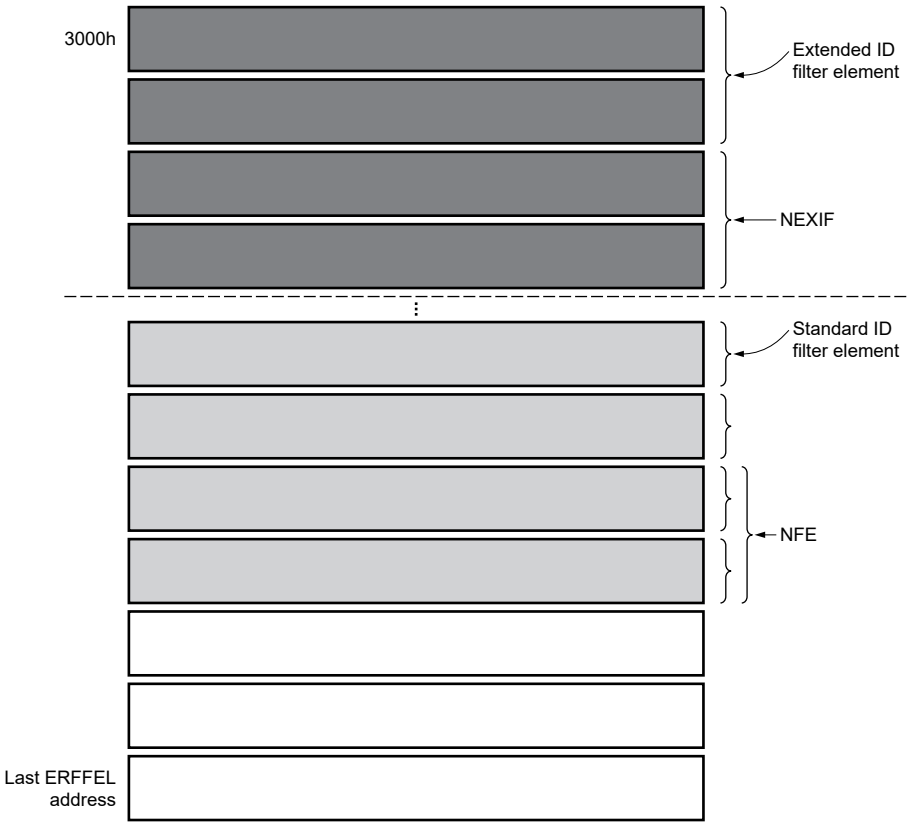


Figure 400. Number of enhanced RX FIFO filter elements

73.3.8.1 Enhanced RX FIFO matching process

When ERFCR[ERFEN] = 1, FlexCAN scans the ERFFEL*n* memory region. If at least one filter element satisfies the matching criteria, the CAN message content is transferred to the enhanced RX FIFO memory. If multiple filters match the incoming message ID, the first matching filter found by the matching process must be indicated in IDHIT.

Each ERFFEL*n* register can store one standard filter element. ERFFEL*n*[FEL][31:30], also called FSCH, determines the matching criteria in this way:

- If FSCH = b00, the filter scheme is based on mask and filter. A CAN message matches a standard ID filter element only if these criteria are reached:
  1. CAN message is base-frame format (IDE = 0).
  2. (ID[*n*] = STD ID filter [*n*]) or (STD ID Mask[*n*] = 0) for each bit *n* from 0 to 10.
  3. (RTR = RTR Filter) or (RTR MASK = 0).

In this explanation, RTR and ID are the Remote Transmit Request field and the ID from a CAN message, respectively.

If FSCH = b00, the filters and masks are defined as shown in Table 499.

Table 499. Standard ID filter element with filter and mask scheme (FSCH = b00)

31	30	29	28	27	26	16	15	12	11	10	0
FSCH = b00		Reserved		RTR Filter	STD ID Filter		Reserved		RTR MASK	STD ID MASK	

- If FSCH = b01, the filter scheme is based on range. A CAN message matches a standard ID filter element only if these criteria are reached:
  1. CAN message is base frame format (IDE = 0).
  2. ID ≥ STD ID Filter1.
  3. ID ≤ STD ID Filter2.
  4. (RTR = RTR filter) or (RTR MASK = 0).

RTR and ID are the Remote Transmit Request bit and ID from a CAN message, respectively. If FSCH = b01, the filters and mask are defined as shown in Table 500.

Table 500. Standard ID filter element with range scheme (FSCH = b01)

31	30	29	28	27	26	16	15	12	11	10	0
FSCH = b01		Reserved		RTR Filter	STD ID Filter2		Reserved		RTR MASK	STD ID Filter1	

- If FSCH = b10, the filter scheme is based on two filters without masks. A CAN message matches a standard ID filter element only if these criteria are reached:
  1. CAN message is base frame format (IDE = 0).
  2. (ID[*n*] = STD ID Filter1[*n*]) or (ID[*n*] = STD ID Filter2[*n*]) for each bit *n* from 0 to 10.
  3. (RTR = RTR Filter1) or (RTR = RTR Filter2).

RTR and ID are the Remote Transmit Request bit and ID from a CAN message, respectively. If FSCH = b10, the filters are defined as shown in Table 501.

Table 501. Standard ID filter element with two-filter scheme (FSCH = b10)

31	30	29	28	27	26	16	15	12	11	10	0
FSCH = b10		Reserved		RTR Filter 2	STD ID Filter2		Reserved		RTR Filter 1	STD ID Filter1	

Each pair of ERFFEL*n* registers can store one extended filter element. ERFFEL*n*[FSCH] determines the matching criteria in this way:

- If FSCH = b00, the filter scheme is based on mask and filter. A CAN message matches an extended ID filter element only if these criteria are reached:
  1. CAN message is extended frame format (IDE = 1).
  2. (ID[*n*] = EXT ID filter [*n*]) or (EXT ID Mask[*n*] = 0) for each bit *n* from 0 to 28.
  3. (RTR = RTR Filter) or (RTR MASK = 0).

If FSCH = b00, the filters and masks are defined as shown in Table 502.

Table 502. Extended ID filter element with filter + mask scheme (FSCH = b00)

31	30	29	28	0
FSCH		RTR Filter	EXT ID Filter	
Reserved		RTR MASK	EXT ID MASK	

- If FSCH = b01, the filter scheme is based on range. A CAN message matches an extended ID filter element only if the following criteria are reached:
  1. CAN message is extended frame format (IDE = 1).
  2. ID ≥ EXT ID Filter1.
  3. ID ≤ EXT ID Filter2.
  4. (RTR = RTR Filter) or (RTR MASK = 0).

If FSCH = b01, the filters and masks are defined as shown in Table 503.

Table 503. Extended ID filter element with range scheme (FSCH = b01)

31	30	29	28	0
FSCH		RTR Filter	EXT ID Filter2	
Reserved		RTR MASK	EXT ID Filter 1	

- If FSCH = b10, the filter scheme is based on two filters without masks. A CAN message matches an extended ID filter element only if these criteria are reached:
  1. CAN message is extended frame format (IDE = 1).
  2. (ID[*n*] = EXT ID Filter1[*n*]) or (ID[*n*] = EXT ID Filter2[*n*]) for each bit *n* from 0 to 28.
  3. (RTR = RTR Filter1) or (RTR = RTR Filter2).

If FSCH = b10, the filters are defined as shown in Table 504.

Table 504. Extended ID filter element with two-filter scheme (FSCH = b10)

31	30	29	28	0
FSCH		RTR Filter2	EXT ID Filter2	
Reserved		RTR Filter1	EXT ID Filter 1	

73.3.8.2 Enhanced RX FIFO under DMA operation

You can enable the DMA feature by writing 1 to both ERFCR[ERFEN] and MCR[DMA]. The DMA controller can read the received message by reading a message buffer structure at the enhanced FIFO output port at the address range defined in Enhanced RX FIFO structure.

NOTE

FlexCAN supports 32-bit access only for DMA transfers.

For proper FIFO engine operation, the CPU should not access the Enhanced FIFO output port address range during DMA operation. Before writing 1 to MCR[DMA], the CPU must service Enhanced RX FIFO status bits. Otherwise, these bits may show that the FIFO has data to be serviced, and mistakenly generate a DMA request. Before writing 0 to MCR[DMA], the CPU must first clear the ERFSR[ERFUFW], ERFSR[ERFOVF], ERFSR[ERFWMI], and ERFSR[ERFDA] flags. It must then clear the enhanced RX FIFO engine by writing one to ERFSR[ERFCLR].

When there is one frame available to be read from the Enhanced RX FIFO, FlexCAN sets ERFSR[ERFDA]. Upon receiving the request, the DMA controller can read the message in the Enhanced RX FIFO output. Each message reading process must end by the address defined in ERFCR[DMALW].

Follow these rules for Enhanced RX FIFO DMA operation:

- Because a DMA transfer cannot be changed dynamically, program ERFCR[DMALW] so the enhanced RX FIFO element can store the largest CAN message present on the CAN bus.
- Data bytes are valid according to the DLC field. See Table 532.

Each time the DMA controller reads one message from the FIFO, FlexCAN clears ERFSR[ERFDA]. If there is at least one message stored in the FIFO, FlexCAN sets it again.

Consider an example where the maximum number of bytes in the data field of a CAN frame for a certain application is eight, and high-resolution timestamp is enabled. In that case, the last enhanced RX FIFO address offset can be found in Table 542 and Table 543. Using this address offset, ERFCR[DMALW] can be determined in this way:

- Maximum number of data bytes = 8
- HR_TIME_STAMP enabled
- Last address offset = TS_OFF = 2014h
- DMALW = 5

73.3.8.3 Enhanced RX FIFO clear operation

When ERFCR[ERFEN] is 1, the CPU can clear the Enhanced RX FIFO by writing 1 to ERFSR[ERFCLR]. The clear operation resets the internal FIFO pointers, but the FIFO content stored in RAM is not changed. This operation can only be performed in Freeze mode; the module blocks the operation in other modes. This operation does not clear ERFSR[ERFUFW], ERFSR[ERFOVF], ERFSR[ERFDA], or ERFSR[ERFWMI]. The CPU must service all these fields before executing the clear FIFO operation.

73.3.9 Legacy RX FIFO

The Legacy RX FIFO is receive-only. To enable it, write 1 to MCR[RFEN]. To maintain software backward compatibility with previous versions of FlexCAN that did not have the Legacy FIFO feature, the reset value of this field is zero.

CAUTION

Do not enable Legacy RX FIFO when the CAN FD feature is enabled.

The Legacy FIFO is six messages deep. The memory region the Legacy FIFO structure occupies (both message buffers and Legacy FIFO engine) is described in Legacy RX FIFO structure. The CPU can read the received messages sequentially, in the order they were received, by repeatedly reading a message buffer structure at the output of the Legacy FIFO.

IFLAG1[BUF5I] (Frames Available in Legacy RX FIFO) is set when at least one frame is available to be read from the Legacy FIFO. If the corresponding mask bit enables it, an interrupt is generated. Upon receiving the interrupt, the CPU can read the message (accessing the output of the Legacy FIFO as a message buffer) and Legacy RX FIFO Information (RXFIR), then clear the interrupt. If there are more messages in the Legacy FIFO, clearing the interrupt:

1. Updates the output of the Legacy FIFO with the next message.
2. Updates RXFIR with the attributes of that message.
3. Reissues the interrupt to the CPU.

Otherwise, the flag remains cleared. The output of the Legacy FIFO is valid only when IFLAG1[BUF5I] is set.

IFLAG1[BUF6I] (Legacy RX FIFO Warning) is set when the Legacy RX FIFO receives a new message that increases the number of unread messages from four to five. This change means that the Legacy RX FIFO is almost full. The flag remains set until the CPU clears it.

IFLAG1[BUF7I] (Legacy RX FIFO Overflow) is set when an incoming message is lost because the Legacy RX FIFO is full. The flag is not set when the Legacy RX FIFO is full and a message buffer captures the message. The flag remains set until the CPU clears it.

Clearing one of the three flags above does not affect the state of the other two.

If an IFLAG flag is set and the corresponding mask bit is 1, an interrupt is generated.

A powerful filtering scheme is provided to accept only frames intended for the target application, reducing the interrupt servicing workload. The filtering criteria are specified by programming a table of up to 128 32-bit registers, according to CTRL2[RFFN]. This table can be configured to one of the following formats (see also Legacy RX FIFO structure):

- Format A: 128 Identifier Acceptance Filters (IDAFs) — extended or standard IDs including IDE and RTR
- Format B: 256 IDAFs — standard IDs or extended 14-bit ID slices including IDE and RTR
- Format C: 512 IDAFs — standard or extended 8-bit ID slices

NOTE

A chosen format is applied to all entries of the filter table. It is not possible to mix formats within the table.

Every frame available in the Legacy RX FIFO has a corresponding Identifier Acceptance Filter Hit Indicator (IDHIT). The IDHIT can be read in the IDHIT field in the Control and Status word, as shown in the Legacy RX FIFO Structure description. The CPU can also obtain this information by accessing Legacy RX FIFO Information (RXFIR). RXFIR[IDHIT] refers to the message at the output of the Legacy FIFO, and is valid when IFLAG1[BUF5I] is set. RXFIR must be read only before clearing the flag, guaranteeing that the information refers to the correct frame within the Legacy FIFO.

The Individual Mask Registers (RXIMR*n*) individually affect up to 32 elements of the filter table, according to the value of CTRL2[RFFN]. This configuration allows very powerful filtering criteria to be defined. If MCR[IRMQ] is 0, the Legacy RX FIFO filter table is affected by Legacy RX FIFO Global Mask (RXFGMASK).

NOTE

See Table 493 for information about the difference between FD and non-FD regarding this feature.

73.3.9.1 Legacy RX FIFO in DMA operation

The receive-only Legacy FIFO can support DMA. To enable this feature, write 1 to both MCR[RFEN] and MCR[DMA]. To maintain backward compatibility with previous versions of the module that did not have the DMA feature, the reset value of MCR[DMA] is zero.

The DMA controller can read the received message by reading a message buffer structure at the Legacy FIFO output port in the 80h–8Ch address range.

NOTE

FlexCAN supports 32-bit access only for DMA transfers.

When MCR[DMA] = 1, the CPU must not access the Legacy FIFO output port address range. Before writing 1 to MCR[DMA], the CPU must service the IFLAG flags set in the Legacy RX FIFO region. Otherwise, these flags may indicate that the FIFO has data to be serviced, and mistakenly generate a DMA request. Before writing 0 to MCR[DMA], the CPU must perform a clear Legacy FIFO operation.

When at least one frame available to be read from the FIFO, IFLAG1[BUF5I] (Frames available in Legacy RX FIFO) is set. A DMA request is generated simultaneously. Upon receiving the request, the DMA controller can read the message (accessing the output of the Legacy FIFO as a message buffer). The DMA reading process must end by reading address 8Ch. This read operation:

- Clears IFLAG1[BUF5I].
- Updates the FIFO output with the next message (if the FIFO is not empty).
- Updates Legacy RX FIFO Information (RXFIR) with the attributes of the new message.

If there are more messages stored in the FIFO, IFLAG1[BUF5I] is reasserted and another DMA request is issued. Otherwise, the flag remains cleared.

NOTE

RXFIR contents cannot be read after DMA completes the Legacy FIFO read. The IDHIT information is also available in the Control and Status word at address 080h (see Legacy RX FIFO structure).

IFLAG1[BUF6I] and IFLAG1[BUF7I] are not used when the DMA feature is enabled.

When FlexCAN is working with DMA, the CPU does not receive any Legacy RX FIFO interruption and must not clear the related IFLAG flags. The related IMASK bits are not used to mask the generation of DMA requests.

NOTE

See Table 493 for information about the difference between FD and non-FD regarding this feature.

73.3.9.2 Clear legacy FIFO

When MCR[RFEN] = 1, you can use the clear Legacy FIFO operation to empty Legacy FIFO contents. When the CPU writes 1 to IFLAG1[BUF0I], the clear FIFO operation occurs. This operation can only be performed in Freeze mode; FlexCAN blocks it in other modes. This operation does not clear the FIFO IFLAG flags; the CPU must service all FIFO IFLAG flags before executing the clear FIFO operation.

When Legacy RX FIFO is working with DMA, the clear FIFO operation clears IFLAG1[BUF5I], and the DMA request is canceled.

CAUTION

The clear Legacy FIFO operation does not clear IFLAG flags, except when MCR[DMA] = 1; in this case, only IFLAG1[BUF5I] is cleared.

73.3.10 CAN protocol-related features

73.3.10.1 CAN FD ISO compliance

The CAN FD protocol has been improved to increase the failure-detection capability that was in the original CAN FD protocol. This original protocol is also called non-ISO CAN FD, by CAN in Automation (CiA). A three-bit stuff counter and a parity bit have been introduced in the improved CAN FD protocol, now called ISO CAN FD. The CRC calculation has also been modified. All these improvements make the ISO CAN FD protocol incompatible with the non-ISO CAN FD protocol. FlexCAN still supports non-ISO CAN FD, so it can be used during an intermediate phase, for evaluation and development purposes.

It is recommended that you configure FlexCAN with the ISO CAN FD protocol by writing 1 to CTRL2[ISOCANFDEN].

73.3.10.2 CAN FD frames

ISO 11898-1:2015 specifies the Classical Frame format compliant to ISO 11898-1:2003 (2003) and introduces the CAN Flexible Data Rate Frame format (CAN FD). The Classical Frame format allows bit rates up to one Mbit/s and payloads up to eight bytes per frame. The Flexible Data Rate Frame format allows bit rates faster than one Mbit/s and payloads longer than eight bytes per frame. FlexCAN can receive and transmit CAN FD messages interleaved with Classical CAN messages.

There are additional control bits in the CAN FD frame:

- The Extended Data Length (EDL) bit enables a longer data payload with different data length coding.
- The Bit Rate Switch (BRS) bit decides whether the bit rate is switched inside a CAN FD format frame.
- The Error State Indicator (ESI) flag is transmitted dominant by error active nodes, and recessive by error passive nodes.

There are no Remote Frames (see Remote frames) in the CAN FD format. A message configured to transmit a Remote Frame is always sent out in Classical CAN format. When an FD frame is received and matches a message buffer, the RTR bit in the receiving message buffer becomes 0. The RTR bit must be considered in classical frames only.

73.3.10.2.1 CAN FD messages

CAN FD messages may be formatted as long frames where the data field exceeds eight bytes, and may range from 12 up to 64 bytes. They can also be configured to support bit rate switching. In this case, the control field, data field, and CRC field of a CAN frame are transmitted with a higher bit rate than the beginning and end of the frame. Messages in the classical CAN format are limited to transport a maximum payload of eight bytes at nominal rate. Figure 401 illustrates the message formats for classical and FD frames with either standard or extended ID.

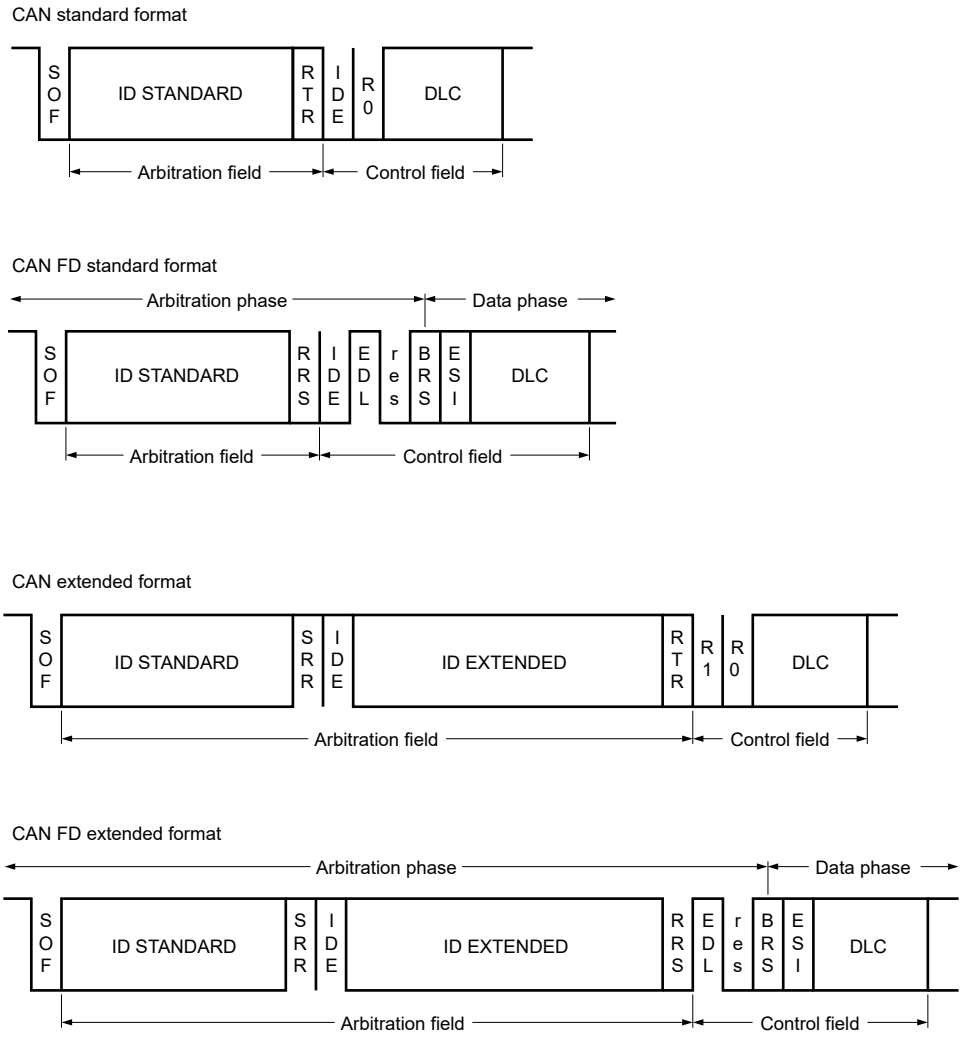


Figure 401. CAN message formats

MCR[FDEN] enables the ability to receive and transmit CAN FD messages. A recessive R0 bit in CAN frames with 11-bit identifiers, or a recessive R1 bit in CAN frames with 29-bit identifiers, is decoded as an EDL bit (not a reserved one). A recessive EDL bit identifies a CAN FD frame, and a dominant EDL bit identifies a Classical CAN frame. The BRS bit specifies whether this frame switches the bit rate in its data phase. A long frame is decoded according to the DLC field value (see DLC definition in Message buffer structure).

CAN FD messages can be transmitted with two different bit rates. The first part of a CAN FD frame, from the Start of Frame (SOF) bit until the Bit Rate Switch (BRS) bit is called the arbitration phase. This part is transmitted with the nominal bit rate based on a set of nominal CAN bit timing configuration values. The second part, from the BRS bit until the CRC Delimiter bit, is called the data phase. When this second part is transmitted, a second set of CAN data bit timing configuration values determines the data bit rate. Finally, from the CRC delimiter until the Intermission bits, the transmission returns to nominal bit rate.

73.3.10.2.2 BRS in CAN FD

In CAN FD frames with bit rate switching, the bit timing changes inside the frame at the sample point of the BRS bit if this bit is recessive. Before the BRS bit, in the CAN FD arbitration phase, the nominal CAN bit timing is used as defined by CAN Bit Timing (CBT). (Control 1 (CTRL1) also defines this timing for backward compatibility.) Upon detecting a recessive BRS bit, the CAN data bit timing is used as defined by CAN FD Bit Timing (FDCBT).

NOTE

If the time quantum length in nominal bit timing and in the data bit timing are not identical, a quantization error of up to one time quantum of the arbitration phase may be present as a phase error. This situation can occur after the switch from arbitration to data phase, and it lasts until the next synchronization event. The length of the time quantum should be the same in nominal and data bit timing. This configuration minimizes the chance of error frames on the CAN bus, and optimizes the clock tolerance in networks that use FD frames.

If BRS = 1 in the selected TX MB, FDCTRL[FDRATE] enables the transmission of all frames with bit rate switching. If FDCTRL[FDRATE] = 0, the transmission is performed at nominal rate regardless of the BRS bit value. FDCTRL[FDRATE] can be written at any time but takes effect only for the next message transmitted or received.

Nominal bit timing is resumed at the sample point of the CRC Delimiter bit or when an error is detected, whichever occurs first. Figure 402 describes the mechanism for entering and leaving the data phase when the BRS bit is recessive.

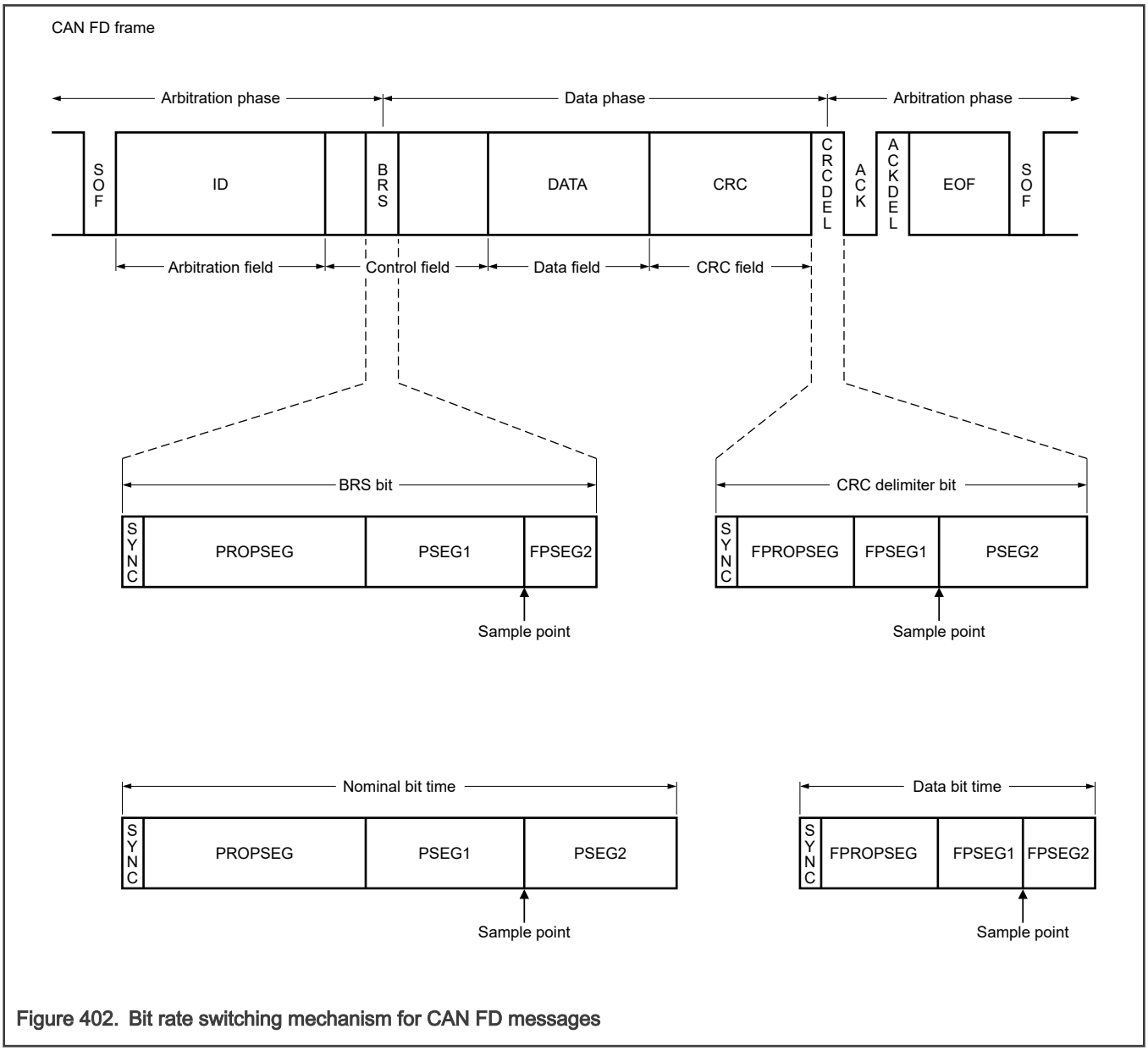


Figure 402. Bit rate switching mechanism for CAN FD messages

NOTE

In Classical CAN frames, the CRC delimiter is one recessive bit. In CAN FD frames, the CRC delimiter may consist of one or two recessive bits. FlexCAN sends only one recessive bit as the CRC delimiter. It accepts two recessive bits before the recessive-to-dominant edge that starts the acknowledge slot. As a receiver, FlexCAN sends its acknowledge bit after the first CRC delimiter bit. In CAN FD frames, FlexCAN accepts a two-bit dominant ACK slot as a valid ACK to compensate for phase shifts between the receivers.

The maximum configurable bit rate in the CAN FD data phase depends on the clock frequency of the CAN_PE subblock. For example, for a CAN_PE clock frequency of 40 MHz and the shortest configurable bit time of 5 time quanta, the bit rate in the data phase is 8 Mbit/s.

NOTE

The frequency used in this example may not be supported on this chip. It is shown only to demonstrate how the maximum configurable bit rate is calculated.

73.3.10.2.3 ESI in CAN FD

The value of the ESI field is determined by either of the following:

- The error state of the transmitter at the start of the transmission, if the frame is originated in the FlexCAN node
- The original transmitting node when FlexCAN is acting as a gateway for the message

If the transmitter is error-passive, ESI is transmitted recessive; otherwise, it is transmitted dominant. The permutations of the relationship between the written value and the transmitted value of the ESI are shown in Table 505.

Table 505. Written versus transmitted values of the ESI field

FlexCAN fault confinement status at start of frame	ESI field of TX MB	Transmitted ESI
Error active	0	0 (Error Active)
Error passive	0	1 (Error Passive)
Error active	1	1 (Error Passive)
Error passive	1	1 (Error Passive)

73.3.10.2.4 CRC calculations in CAN FD

Different CAN frame formats have different CRC polynomials. The first polynomial, CRC_15, is used for all frames in Classical CAN format. The second, CRC_17, is used for frames in CAN FD format with a data field up to 16 bytes long. The third, CRC_21, is used for frames in CAN FD format with a data field longer than 16 bytes. Each polynomial results in a Hamming distance of 6. At the start of the frame, all three CRC polynomials are calculated concurrently. The values of the EDL bit and the DLC field select the CRC sequence to be transmitted. When receiving a message, FlexCAN decodes EDL and DLC to select the adequate CRC polynomial to check for a CRC error.

In CAN FD format frames, stuff bits are included in the bit stream for CRC calculation. In Classical CAN format frames, stuff bits are not included. After the transmission of the last bit relevant to the CRC calculation, CAN FD CRC (FDCRC) stores the calculated CRC for the transmitted message. This storage is performed with adequate length for the type of message, for CAN FD and non-FD messages. Cyclic Redundancy Check (CRCR) reports a valid CRC for Classical CAN messages only.

In CAN FD format frames, the CAN bit stuffing method changes for the CRC sequence, so the stuff bits are inserted at fixed positions. When FlexCAN is transmitting a CAN FD frame, a fixed stuff bit is inserted just before the first bit of the CRC sequence. This insertion occurs even if the last bits of the preceding field do not fulfill the CAN stuff condition. Additional stuff bits are inserted after each fourth bit of the CRC sequence. The value of any fixed stuff bit is the inverse value of its preceding bit. When FlexCAN receives a CAN FD frame, it discards the fixed stuff bits from the bit stream for the CRC check. A stuff error is detected if the fixed stuff bit has the same value as its preceding bit.

73.3.10.2.5 CAN FD errors

FlexCAN detects errors in CAN FD frames the same way as in Classical CAN frames. The error counters ECR[RXERRCNT] and ECR[TXERRCNT] accumulate the counts of RX and TX errors, respectively, for both FD and non-FD frames indiscriminately. Two extra error counters, ECR[RXERRCNT_FAST] and ECR[TXERRCNT_FAST], accumulate RX and TX errors occurring in the data phase of CAN FD frames with BRS = 1 only. The rules for updating the error counters are the same for both CAN FD and non-FD frames (see Error Counter (ECR)).

These error flags report errors in both CAN FD and non-FD frames:

- ESR1[BIT1ERR]
- ESR1[BIT0ERR]
- ESR1[ACKERR]
- ESR1[CRCERR]
- ESR1[FRMERR]
- ESR1[STFERR]

If CTRL1[ERRMSK] = 1, they also generate the ERRINT interrupt.

These additional error flags indicate the occurrence of errors in the data phase of CAN FD frames with BRS = 1:

- ESR1[BIT1ERR_FAST]
- ESR1[BIT0ERR_FAST]
- ESR1[CRCERR_FAST]
- ESR1[FRMERR_FAST]
- ESR1[STFERR_FAST]

No ACKERR is detected in the data phase of a CAN FD frame. Fault confinement status reported in ESR1[FLTCONF] is the same for both CAN FD and Classical CAN frames, and is based on ECR[RXERRCNT] and ECR[TXERRCNT] only. Information in ECR[RXERRCNT_FAST] and ECR[TXERRCNT_FAST] may be considered as status to help detect the error nature related to the bit rate value.

When FlexCAN detects an error while transmitting or receiving a CAN FD message in the data phase, it immediately switches:

- Back to the arbitration phase, and
- Back to the nominal rate to start an error flag.

73.3.10.2.6 CAN FD synchronization

Resynchronization and hard synchronization occur in CAN FD frames in the same way as in Classical CAN ones. A hard synchronization is also performed at the recessive-to-dominant edge from EDL to R0 in CAN FD format frames. FlexCAN does not resynchronize when transmitting in the CAN FD data phase.

73.3.10.3 Transceiver delay compensation

The CAN FD protocol allows the transmission and reception of data at a higher bit rate than the nominal rate used in the arbitration phase, when BRS = 1 in the message. This feature enables the use of rates up to 8 Mbit/s.

During the data phase of a CAN FD frame, if the transmitter cannot receive its own latest transmitted bit at the sample point of that bit, it detects a bit error. When bit rate switching is enabled (BRS = 1), the CAN bit time in the data phase can become shorter than the loop delay of the transceiver. This condition impedes the correct comparison between the transmitted bit and the received bit within the current CAN bit time interval.

The transceiver delay compensation (TDC) process defines a secondary sample point where the transmitted bit is correctly compared to the received bit to check for bit errors.

You can enable the TDC mechanism via FDCTRL[TDCEN] or ETDC[ETDCEN]. The TDC mechanism is effective only during the data phase of FD frames with BRS = 1. It has no effect on either non-FD frames or FD frames transmitted at the normal bit rate. When the transmitted message has BRS = 1, TDC is active from the sample point of the BRS bit until the sample point of the CRC Delimiter bit. When TDC is active, the real received bit is compared to the delayed transmitted bit, where the delay is calculated based on the measured transceiver loop delay.

NOTE

The transmitters using TDC disregard the value of the CRC delimiter bit. A global error at the end of the CRC field causes the receivers to send error frames that the transmitter detects during Acknowledge or End of Frame.

For every transmitted FD frame with BRS = 1, the transition from the recessive EDL bit to the dominant R0 bit triggers the delay measurement (as shown in Figure 403). The loop delay is measured in Protocol Engine (PE) clock periods (CANCLK, see Protocol timing), from the transmitted EDL-R0 edge to the received EDL-R0 edge. The measured loop delay time added to an offset value specified in FDCTRL[TDCOFF] or ETDC[ETDCOFF] determines the position of the secondary sample point. FDCTRL[TDCVAL] or ETDC[ETDCVAL] stores the result of this calculation. The TDCVAL and ETDCVAL value saturates at its maximum value of 63 CANCLK and 255 CANCLK when the delay measurement is too long.

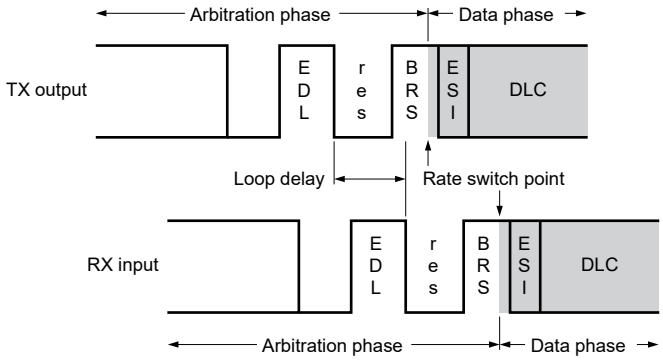


Figure 403. Transceiver loop delay measurement

The measured loop delay is not enough to define the secondary sample point, because it relates to the CAN bit edges. The transceiver delay compensation offset FDCTRL[TDCOFF] or ETDC[ETDCOFF] is used to shift the secondary sample point to an intermediate point inside the bit time, far away from its edges. The value of FDCTRL[TDCOFF] or ETDC[ETDCOFF] cannot be larger than the CAN bit duration in the data phase.

If the secondary sample point is set very near the CAN bit edge (SYNC field), problems may occur during the bit sampling in the data phase. For the TDC to work reliably, the offset must use optimal settings. To ensure that bit sampling is performed in the best region, configure the TDC offset as shown in this equation:

$$Offset = (FPSEG1 + FPROPSEG + 2) \times (FPRESDIV + 1)$$
or
$$Offset = (DTSEG1 + 2) \times (EDPRESDIV + 1), \textit{ if ETDCEN}$$

Equation 12. TDC offset calculation

Figure 404 shows the SSP position when these settings are used.

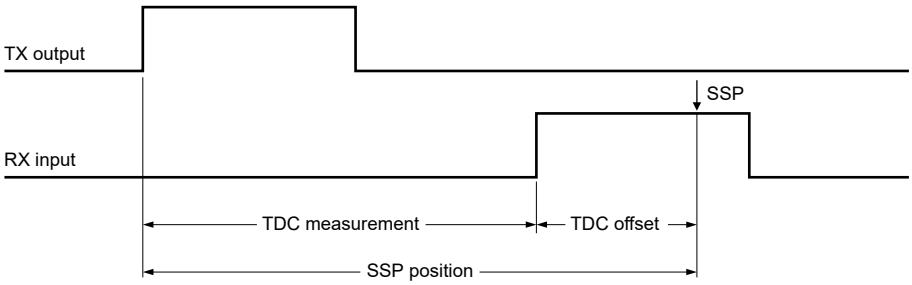


Figure 404. SSP position with optimal values

Alternatively, if CTRL2[BTE] and ETDC[ETDCEN] are 1, you can write 1 to ETDC[TDMDIS] to disable the transceiver delay measurement. In this case, only ETDC[ETDCOFF] defines the SSP position. Figure 405 shows the secondary sample point position when the transceiver delay measurement is disabled.

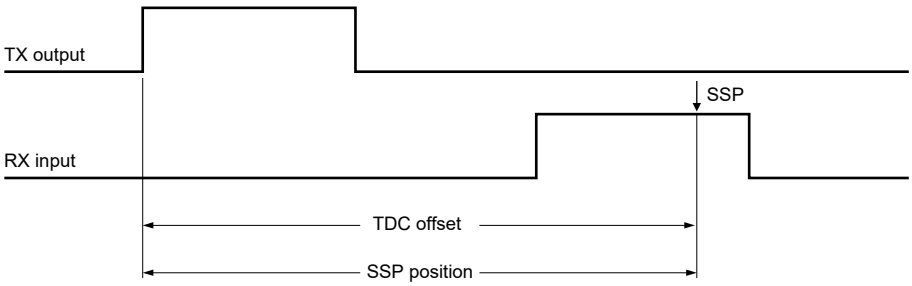


Figure 405. SSP position with TDC measurement disabled by ETDCEN and TDMDIS

During the data phase of CAN FD frames with bit rate switching enabled, at the onset of every TX CAN bit:

- The transmitted TX bit value is temporarily stored in a buffer.
- A time countdown based on FDCTRL[TDCVAL] or ETDC[ETDCVAL] is started. This countdown ends with the comparison of the received RX bit (delayed by the external loop delay plus the specified offset) to the stored TX bit.

If a bit error is detected at the secondary sample point, FlexCAN issues an error flag to the CAN bus at the next sample point.

During the arbitration phase, delay compensation is always disabled. During the data phase, the TDC mechanism of FlexCAN can compensate a maximum delay of 3 CAN bit times − 2 Tq. Beyond this limit, the FDCTRL[TDCFAIL] or ETDC[ETDCFAIL] flag is set. The flag indicates when the TDC mechanism is out of range and is unable to compensate the transceiver loop delay.

73.3.10.4 Remote frames

A remote frame is a special type of frame. You can program a message buffer to be a remote request frame by configuring the message buffer as Transmit with the RTR = 1. After the remote request frame is transmitted successfully, the message buffer becomes a receive message buffer, with the same ID as before.

When FlexCAN receives a remote request frame, the frame can be treated in different ways, depending on remote request storing (CTRL2[RRS]) and RX FIFO Enable (MCR[RFEN]):

- If RRS = 0, the ID of the frame is compared to the IDs of the transmit message buffers with the CODE field 1010b. If a matching ID exists, this message buffer frame is transmitted. If the matching message buffer has the RTR = 1, FlexCAN transmits a remote frame as a response. The received remote request frame is not stored in a receive buffer. It is only used to trigger a transmission of a frame in response.

  The mask registers are not used in remote frame matching, and all ID bits (except RTR) of the incoming received frame should match. If a remote request frame is received and matches a message buffer, this message buffer immediately enters the internal arbitration process. However, it is considered a normal TX message buffer, with no higher priority. The data length of this frame is independent of the DLC field in the remote frame that initiated its transmission.

- If CTRL2[RRS] = 1, the ID of the frame is compared to the IDs of the receive message buffers with the CODE field 0100b, 0010b, or 0110b. If a matching ID exists, this message buffer stores the remote frame in the same fashion as a data frame. No automatic remote response frame is generated. The mask registers are used in the matching process.
- If MCR[RFEN] = 1, FlexCAN does not generate an automatic response for remote request frames that match the Legacy FIFO filtering criteria. If the remote frame matches one of the target IDs, it is stored in the Legacy FIFO and presented to the CPU.

  For filtering formats A and B (see Legacy RX FIFO structure), it is possible to select whether remote frames are accepted or not. For format C, remote frames are always accepted if they match the ID. Remote request frames are considered as normal frames. They generate a Legacy FIFO overflow when a successful reception occurs and the Legacy FIFO is already full.
- If ERFCR[ERFEN] = 1, FlexCAN does not generate an automatic response for remote request frames that match the Enhanced RX FIFO filtering criteria. Remote Request Frames are considered normal frames. They generate an Enhanced RX FIFO overflow when a successful reception occurs and the enhanced RX FIFO is already full.

NOTE

There is no remote frame in the CAN FD format. A fixed dominant RRS bit replaces the RTR bit. FlexCAN receives and transmits remote frames in the Classical CAN format.

73.3.10.5 Overload frames

When a dominant bit is detected on the CAN bus in these locations, FlexCAN transmits overload frames:

- The first or second bit of Intermission.
- The seventh bit (last) of End of Frame field (RX frames).
- The eighth bit (last) of Error Frame Delimiter or Overload Frame Delimiter.

73.3.10.6 Message buffer timestamp

The value of the free-running timer is sampled at the beginning of the Identifier field on the CAN bus. This value is stored at the end of move-in in the TIME_STAMP field of a message buffer, providing network behavior regarding time.

When CTRL2[TIMER_SRC] = 1, an external time tick continuously clocks the free-running timer.

When CTRL2[TIMER_SRC] = 0, the FlexCAN bit clock clocks the free-running timer, which defines the baud rate on the CAN bus. During a message transmission or reception, it increments by one for each bit that is received or transmitted. When there is no message on the bus, it counts using the previously programmed baud rate.

The free-running timer is not incremented during Disable and Freeze modes. It can be reset upon a specific frame reception, enabling network time synchronization. See CTRL1[TSYN].

Alternatively, by configuring CTRL2[MBTSBASE], the timestamp of the message buffer can capture the lower or higher 16 bits of the high-resolution dedicated counter. See the chip-specific FlexCAN timestamp implementation.

73.3.10.7 High-resolution timestamp

The high-resolution timestamp (HR_TIME_STAMP) uses a dedicated timer with a 32-bit counter operating in free-running mode. CTRL2[TSTAMPCAP] enables the high-resolution timestamp. When this field is not zero, the dedicated 32-bit counter value is captured during a valid CAN frame and stored in an HR_TIME_STAMP*n* register.

Each HR_TIME_STAMP*n* corresponds to a specific message buffer. For example, HR_TIME_STAMP0 stores the 32-bit timestamp associated with message buffer 0. HR_TIME_STAMP1 stores the 32-bit timestamp associated with message buffer 1, and so on.

The counter value is captured according to CTRL2[TSTAMPCAP]. For classical CAN frames, the capture points can be the start of frame bit or the point in time a CAN frame is considered valid. This valid point is the seventh bit of end of frame for transmission and the sixth bit of end of frame for reception.

For CAN FD frames, the capture points can be:

- The start of frame

- The point in time a CAN FD frame is considered valid
- The res bit of a CAN FD frame

The 16-bit timestamp of the message buffer can be configured to capture the lower or higher 16 bits of the high-resolution timer. This configuration is made by CTRL2[MBTSBASE]. See the chip-specific FlexCAN timestamp implementation.

73.3.10.8 Protocol timing

Figure 406 shows the structure of the clock generation circuitry that feeds the CAN Protocol Engine (PE) submodule.

NOTE

To identify the proper clock source, see the clock distribution chapter (module clocks table).

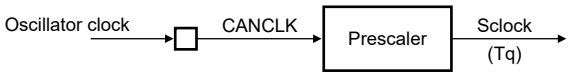


Figure 406. CAN engine clocking scheme

73.3.10.8.1 Bit timing configuration

FlexCAN supports various means to configure bit timing parameters required by the CAN protocol. Control 1 (CTRL1) has various fields to control bit timing parameters:

- CTRL1[PRESDIV]
- CTRL1[PROPSEG]
- CTRL1[PSEG1]
- CTRL1[PSEG2]
- CTRL1[RJW]

CAN Bit Timing (CBT) extends the range of the CAN bit timing variables in CTRL1. Enhanced Data Phase CAN Bit Timing (EDCBT) provides a second set of CAN bit timing variables to be applied at the data phase of CAN FD frames with the Bit Rate Switch (BRS) = 1.

Enhanced Nominal CAN Bit Timing (ENCBT) extends the range of CAN bit timing variables in CBT. Enhanced Nominal CAN Bit Timing (ENCBT) extends the range of CAN bit timing variables in FDCBT. When using ENCBT and EDCBT, you must program the nominal bit timing and data phase serial clock (Sclock) dividers in Enhanced CAN Bit Timing Prescalers (EPRS).

NOTE

When the CAN FD feature is enabled, always write 1 to CBT[BTF] or CTRL2[BTE] and specify the CAN bit timing variables in CBT or ENCBT. See CAN Bit Timing (CBT) or Enhanced Nominal CAN Bit Timing (ENCBT).

CTRL1[PRESDIV], and its extended range CBT[EPRESDIV] (or EPRS[ENPRESDIV]) and FDCBT[FPRESDIV] (or EPRS[EDPRESDIV]) for the data phase bits of CAN FD messages, defines the prescaler value that generates the serial clock (Sclock). (See Equation 13.) The period of Sclock defines the time quantum used to compose the CAN waveform. A time quantum (Tq) is the atomic unit of time managed by the CAN engine. It is the smallest time unit for all configuration values.

$$Tq = \frac{(PRESDIV + 1)}{f_{CANCLK}}$$

Equation 13. Time quantum

The bit rate, which defines the rate the CAN message is received or transmitted, is calculated with the formula:

$$CAN\ bit\ time = (Number\ of\ time\ quanta\ in\ 1\ bit\ time) \times Tq$$

$$Bit\ rate = \frac{1}{CAN\ bit\ time}$$

Equation 14. CAN bit time and baud rate

73.3.10.8.2 Bit time segments

A bit time is subdivided into three segments as shown in Figure 407. See also Figure 408 , Figure 409, and Table 506.

NOTE

For further explanation of the underlying concepts, see ISO 11898-1:2015. See also *CAN Specification Version 2.0, Part A and Part B* for bit timing.

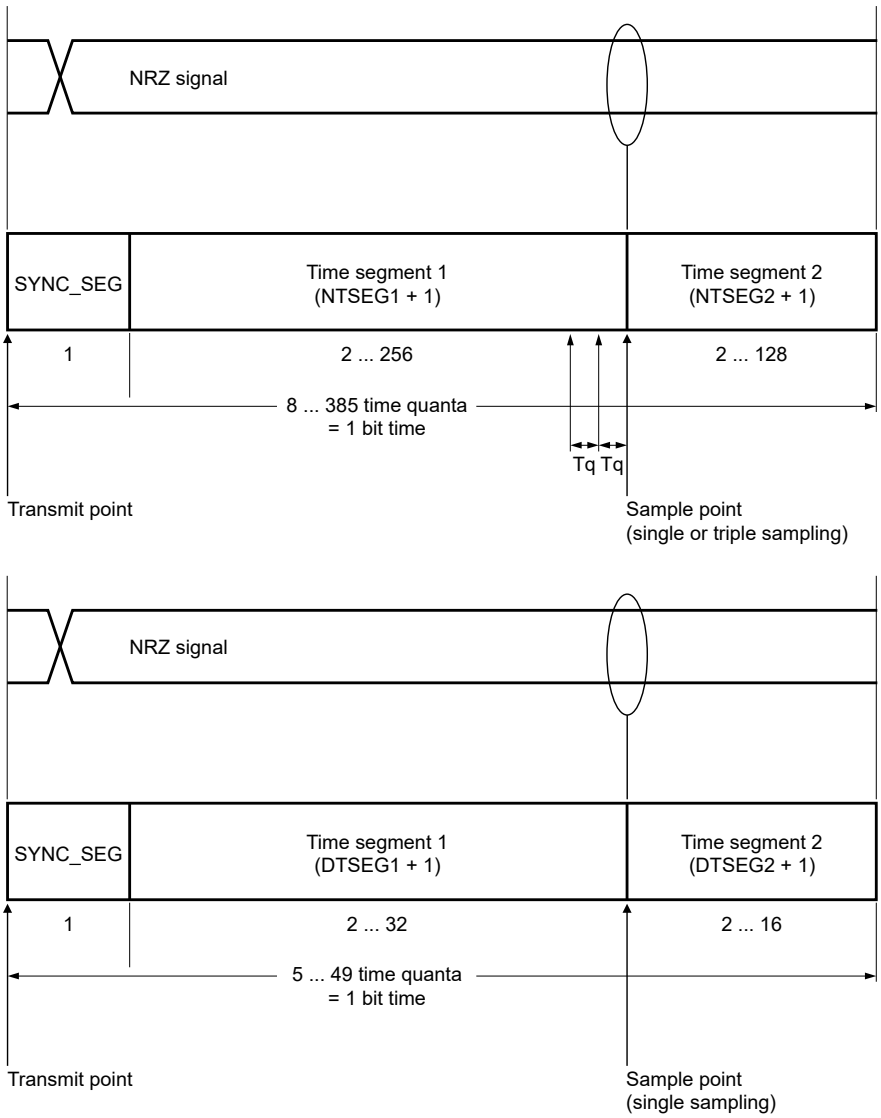


Figure 407. Segments within the bit timing (example using ENCBT and EDCBT bit timing variables)

The three bit time segments are:

- SYNC_SEG—this segment has a fixed length of one time quantum. Signal edges are expected to occur within this section.
- Time Segment 1—this segment includes the propagation segment and the phase segment 1 of the CAN standard.

  It can be programmed by configuring CTRL1[PROPSEG] and CTRL1[PSEG1] so that the sum (plus 2) is 2–16 time quanta. When CBT[BTF] = 1, FlexCAN uses CBT[EPROPSEG] and CBT[EPSEG1] so that the sum (plus 2) is 2–96 time quanta. For messages in CAN FD format with the BRS = 1, FlexCAN uses FDCBT[FPROPSEG] and FDCBT[FPSEG1] so that the sum (plus 1) is 2–39 time quanta.

  If CTRL2[BTE] = 1, FlexCAN uses ENCBT[NTSEG1] to configure time segment 1 to 2–256 time quanta. For the data phase in CAN FD messages with BRS = 1, EDCBT[DTSEG1] must be used for configuring time segment 1 to 2–32 time quanta.
- Time Segment 2—this segment represents the phase segment 2 of the CAN standard.

  It can be programmed by configuring CTRL1[PSEG2] (plus 1) to be 2–8 time quanta. When CBT[BTF] = 1, FlexCAN uses CBT[EPSEG2] so that its value (plus 1) is 2–32 time quanta. For messages in CAN FD format with the BRS = 1, FlexCAN uses FDCBT[FPSEG2] instead, so that its value (plus 1) is 2–8 time quanta. Time segment 2 cannot be smaller than the Information Processing Time (IPT), which is 2 time quanta in FlexCAN.

  If CTRL2[BTE] = 1, FlexCAN uses ENCBT[NTSEG2] to configure time segment 2 to 2–128 time quanta. For the data phase in CAN FD messages with BRS = 1, EDCBT[DTSEG2] must configure time segment 2 to 2–16 time quanta.

NOTE

The bit time defined by the above time segments must not be smaller than five time quanta. For bit time calculations, use an Information Processing Time (IPT) of two, which is the value implemented in the FlexCAN module.

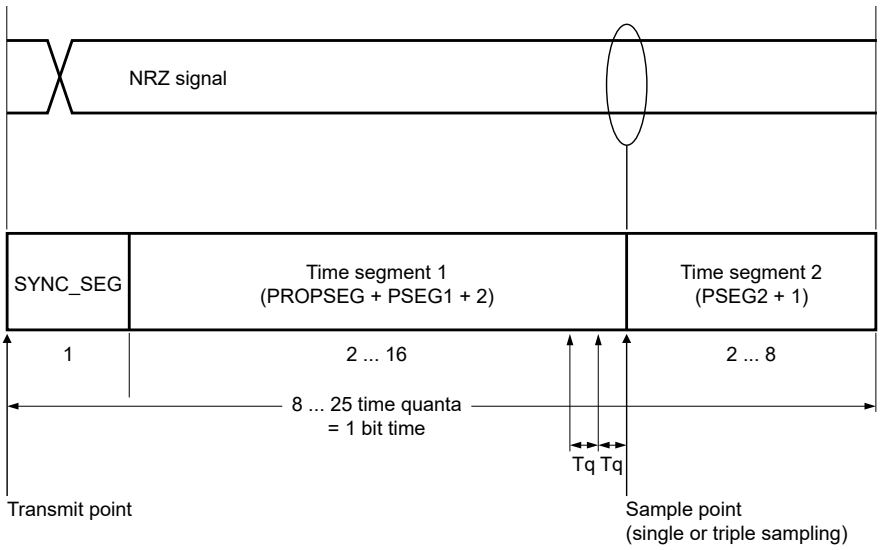


Figure 408. Segments within the bit time (example using CTRL1 bit timing variables for Classical CAN format)

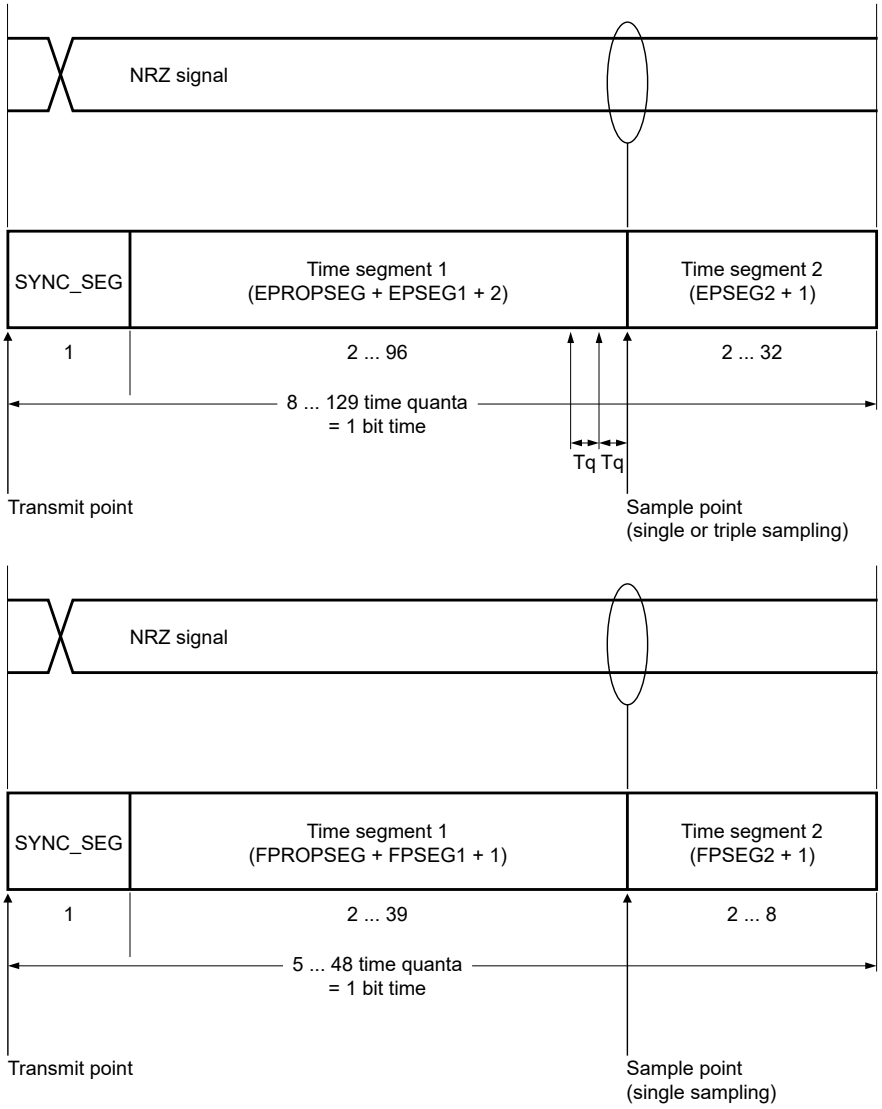


Figure 409. Segments within the bit time (example using CBT and FDCBT bit timing variables for CAN FD format)

Table 506. Time segment syntax

Syntax	Description
SYNC_SEG	Period during which the system expects transitions to occur on the bus
TSEG1	Period corresponding to the sum of PROPSEG and PSEG1
TSEG2	Period corresponding to the PSEG2 value
Transmit point	Point at which a node in Transmit mode transfers a new value to the CAN bus
Sample point	Point at which a node samples the bus. If the option of three samples per bit is selected, this point marks the position of the third sample.

Table 507 gives some examples of the CAN-compliant segment settings for Classical CAN format (Bosch CAN 2.0B) (non-FD) messages.

Table 507. Bosch CAN 2.0B standard compliant bit time segment settings

Time segment 1	Time segment 2	Resynchronization jump width
5 to 10	2	1 to 2
4 to 11	3	1 to 3
5 to 12	4	1 to 4
6 to 13	5	1 to 4
7 to 14	6	1 to 4
8 to 15	7	1 to 4
9 to 16	8	1 to 4

NOTE

You must ensure the bit time settings comply with the CAN Protocol standard ISO 11898-1:2015.

73.3.10.8.3 Calculating peripheral clocks

A CAN bit can be used as a measure of duration (for example, estimating the occurrence of a CAN bit event in a message). When a CAN bit is used in this way, the number of peripheral clocks in one CAN bit (NumClkBit) can be calculated as:

$$NumClkBit = \frac{f_{SYS}}{f_{CANCLK}} \times (PRESDIV + 1) \times (PROPSEG + PSEG1 + PSEG2 + 4)$$

Equation 15. Number of peripheral clocks per CAN bit when CTRL2[BTE] = 0

Or, if CTRL2[BTE] = 1:

$$NumClkBit = \frac{f_{SYS}}{f_{CANCLK}} \times (ENPRESDIV + 1) \times (NTSEG1 + NTSEG2 + 3)$$

Equation 16. Number of peripheral clocks per CAN bit when CTRL2[BTE] = 1

Where:

- NumClkBit is the number of peripheral clocks in one CAN bit.
- f_{CANCLK} is the Protocol Engine (PE) Clock (see Figure 406), in Hz.
- f_{SYS} is the frequency of operation of the system (CHI) clock, in Hz.
- PSEG1 is the value of CTRL1[PSEG1].
- PSEG2 is the value of CTRL1[PSEG2].
- PROPSEG is the value of CTRL1[PROPSEG].
- PRESDIV is the value in CTRL1[PRESDIV].
- ENPRESDIV is the value of EPRS[ENPRESDIV].
- NTSEG1 is the value of ENCBT[NTSEG1].
- NTSEG2 is the value of ENCBT[NTSEG2].

The formula above is also applicable to the alternative CAN bit timing variables described in:

- CAN Bit Timing (CBT)

- Enhanced Nominal CAN Bit Timing (ENCBT)
- CAN FD Bit Timing (FDCBT)
- Enhanced Nominal CAN Bit Timing (ENCBT)

For example, 180 CAN bits = (180 × NumClkBit) peripheral clock periods.

73.3.10.9 Arbitration and matching timing

During normal reception and transmission, the matching, arbitration, move-in, and move-out processes are executed during certain time windows inside the CAN frame. These windows are shown in the following figures.

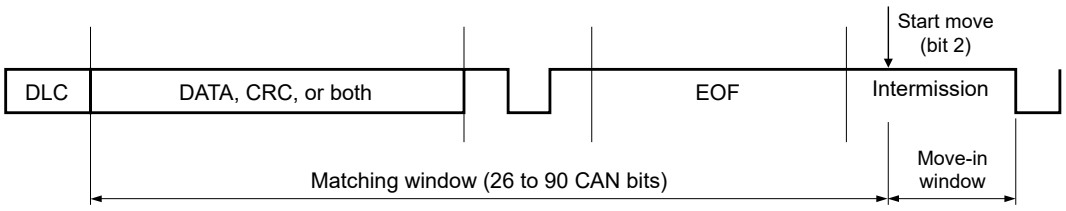


Figure 410. Matching and move-in time windows

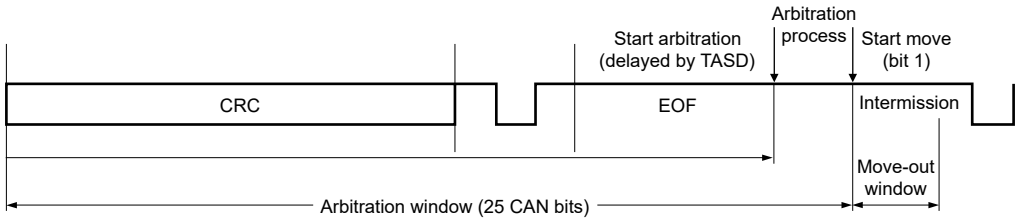


Figure 411. Arbitration and move-out time windows

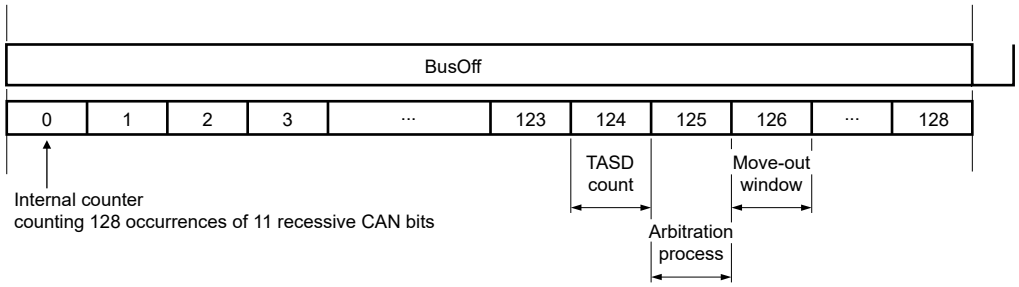


Figure 412. Arbitration at the end of bus off and move-out time windows

NOTE

In these figures, the matching and arbitration timing do not consider delays caused by concurrent memory access due to the CPU or other internal FlexCAN subblocks.

73.3.10.10 TX arbitration start delay

TX Arbitration Start Delay (CTRL2[TASD]) indicates the number of CAN bits that FlexCAN uses to delay the TX arbitration process starting point from the first bit of the CRC field of the current frame. This variable can be written only in Freeze mode; FlexCAN blocks it in other modes.

The ability of the CPU to reconfigure message buffers for transmission after the end of the internal arbitration process impacts transmission performance. In the arbitration process, FlexCAN finds the winner MB for transmission (see Arbitration process). If

the arbitration ends too early (before the first bit of the Intermission field) the CPU may reconfigure some TX message buffers. It is possible that the winning message buffer is no longer the best candidate to be transmitted.

TASD can optimize the transmission performance by defining the arbitration start point, as shown in Figure 413, based on factors such as:

- Peripheral-to-oscillator clock ratio
- CAN bit timing variables that determine the CAN bit rate
- Number of message buffers in use by the matching and arbitration processes

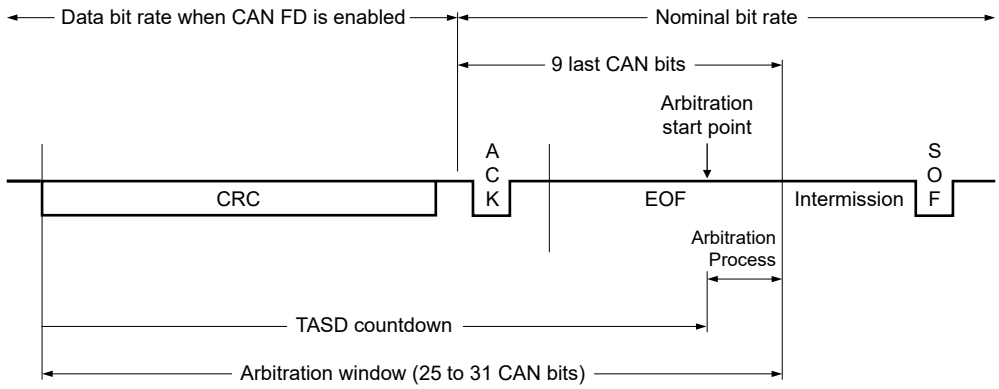


Figure 413. Optimal TX arbitration start point

The duration of an arbitration process, in terms of CAN bits, is:

- Directly proportional to the number of available message buffers
- Directly proportional to the CAN bit rate
- Inversely proportional to the peripheral clock frequency

The optimal arbitration timing occurs when the last message buffer is scanned immediately before the first bit of the Intermission field of a CAN frame. For instance, if the following are true:

- There are few message buffers.
- The peripheral-to-oscillator clock ratio is high.
- The CAN baud rate is low.

Then the arbitration can be placed closer to the end of the frame, adding more delay to its starting point, and vice versa.

If CTRL2[TASD] = 0, the arbitration start is not delayed, and more time is reserved for arbitration. Alternatively, if CTRL2[TASD] is close to 24, the CPU can configure a TX message buffer later, and less time is reserved for arbitration. If too little time is reserved for arbitration, FlexCAN may not be able to find a winner MB in time. The transmitted arbitration winner may not have the best chance to win the bus arbitration against external nodes on the CAN bus.

The optimal TASD value can be calculated as follows:

For CAN FD frames and (MAXMB + 1) ≤ NMB_{END}

$$TASD = 31 - \frac{2 * (MAXMB + 1) + 4}{CPCB_N}$$

For CAN FD frames and (MAXMB + 1) > NMB_{END}

$$TASD = 22 - \frac{2 * (MAXMB + 1) - NMB_{END}}{CPCB_F}$$

For non-FD frames

$$TASD = 25 - \frac{2 * (MAXMB + 1) + 4}{CPCB}$$

Equation 17. Optimal value for TASD

Where:

$$NMB_{END} = \frac{(9 \times CPCB_N) - 4}{2}$$

$$BITRATE_N = \left(\frac{f_{CANCLK}}{[1 + (EPSEG1 + 1) + (EPSEG2 + 1) + (EPROPSEG + 1)] \times (EPRESDIV + 1)} \right)$$

$$BITRATE_F = \left(\frac{f_{CANCLK}}{[1 + (FPSEG1 + 1) + (FPSEG2 + 1) + FPROPSEG] \times (FPRESDIV + 1)} \right)$$

$$CPCB_N = \frac{f_{SYS}}{BITRATE_N}$$

$$CPCB_F = \frac{f_{SYS}}{BITRATE_F}$$

$$CPCB = CPCB_N$$

Equation 18. Variables used in TASD calculation

- MAXMB is the value in MCR[MAXMB].
- NMB_{END} is the number of message buffers that the arbitration process can scan during the last nine CAN bits at the end of a frame. (See Equation 18.)
- $BITRATE_N$ is the CAN bit rate in bits per second calculated by the nominal CAN bit time variables.
- $BITRATE_F$ is the CAN bit rate in bits per second calculated by the data CAN bit time variables.
- $CPCB_N$ is the number of peripheral clocks per CAN bit in nominal bit rate for CAN FD frames.
- $CPCB_F$ is the number of peripheral clocks per CAN bit in data bit rate for CAN FD frames.
- CPCB is the number of peripheral clocks per CAN bit for non-FD frames.
- f_{CANCLK} is the oscillator clock, in Hz.
- f_{SYS} is the peripheral clock, in Hz.
- EPSEG1 is the value in CBT[EPSEG1] (CTRL1[PSEG1] can also be used).
- EPSEG2 is the value in CBT[EPSEG2] (CTRL1[PSEG2] can also be used).
- EPROPSEG is the value in CBT[EPROPSEG] (CTRL1[PROPSEG] can also be used).
- EPRESDIV is the value in CBT[EPRESDIV] (CTRL1[PRESDIV] can also be used).
- FPSEG1 is the value in FDCBT[FPSEG1].

- FPSEG2 is the value in FDCBT[FPSEG2].
- FPROPSEG is the value in FDCBT[FPROPSEG].
- FPRESDIV is the value in FDCBT[FPRESDIV].
- NTSEG1 is the value in ENCBT[NTSEG1].
- NTSEG2 is the value in ENCBT[NTSEG2].
- ENPRESDIV is the value in EPRS[ENPRESDIV].
- DTSEG1 is the value in EDCBT[DTSEG1].
- DTSEG2 is the value in EDCBT[DTSEG2].
- EDPRESDIV is the value in EPRS[EDPRESDIV].

If CTRL2[BTE] = 1, then:

$$BITRATE_N = \frac{f_{CANCLK}}{[1 + (NTSEG1 + 1) + (NTSEG2 + 1)] \times (ENPRESDIV + 1)}$$

Equation 19. Nominal baud rate when CTRL2[BTE] = 1

$$BITRATE_F = \frac{f_{CANCLK}}{[1 + (DTSEG1 + 1) + (DTSEG2 + 1)] \times (EDPRESDIV + 1)}$$

Equation 20. Fast baud rate when CTRL2[BTE] = 1

See also Protocol timing for more details.

73.3.10.10.1 TASD configuration examples

The following tables show the TASD value calculated for some configuration cases.

Case 1:

- Clock ratio = 2:1 (for example, peripheral clock 80 MHz and oscillator clock 40 MHz)
- Bit rate in arbitration phase = 1 Mbaud

Table 508. TASD values in Case 1

Number of message buffers	TASD value	Maximum bit rate in data phase (MBd)
16	24	Invalid
32	24	8.0
64	23	8.0
96	22	8.0

Case 2:

- Clock ratio = 1:1 (for example, peripheral clock 40 MHz and oscillator clock 40 MHz)
- Bit rate in arbitration phase = 1 Mbaud

Table 509. TASD values in Case 2

Number of message buffers	TASD value	Maximum bit rate in data phase (MBd)
16	24	Invalid

Table continues on the next page...

Table 509. TASD values in Case 2 (continued)

Number of message buffers	TASD value	Maximum bit rate in data phase (MBd)
32	23	6.67
54	22	5.0
64	21	3.33
96	20	1.6

Case 3:

- Clock ratio = 2:1 (for example, peripheral clock 40 MHz and oscillator clock 20 MHz)
- Bit rate in arbitration phase = 1 Mbaud

Table 510. TASD values in Case 3

Number of message buffers	TASD value	Maximum bit rate in data phase (MBd)
16	24	Invalid
32	23	4.0
54	22	4.0
64	21	3.33
96	20	1.54

73.3.11 Clocks

The following table describes the clock sources for FlexCAN. See the chip clocking chapter for clock setting, configuration, and gating information.

Table 511. FlexCAN clocks

Clock name	Description
MODULE_CLK (system_clk)	Peripheral clock
MODULE_CLK_CHI (host_clock)	Control Host Interface (CHI) clock
MODULE_CLK_PE (protocol_engine_clock)	Protocol Engine (PE) clock
MODULE_CLK_PE_NOGATE (protocol_engine_clock_nogate)	Protocol Engine clock (no gating)
MODULE_CLK_S (system_clock_nogate)	Peripheral access clock

73.3.11.1 Clock domains and restrictions

FlexCAN has two clock domains asynchronous to each other:

- The bus domain feeds the Control Host Interface (CHI) submodule.
- The oscillator domain feeds the CAN Protocol Engine (PE) submodule.

When the two domains are connected to clocks with different frequencies or phases, the frequency relationship between the two clock domains is restricted. In asynchronous operation, the bus domain clock frequency must always be greater than the oscillator domain clock frequency.

NOTE

Asynchronous operation with a 1:1 ratio between peripheral and oscillator clocks is not allowed.

When performing matching and arbitration, FlexCAN must scan the whole message buffer memory during the time slot of one CAN frame, comprised of a number of CAN bits. To provide sufficient time for the scan, observe the following requirements:

- The peripheral clock frequency cannot be less than the oscillator clock frequency.
- There must be a minimum number of peripheral clocks per CAN bit, as specified in Table 512.

Table 512. Minimum number of peripheral clocks per CAN bit for Classical CAN format

Number of message buffers	Value of MCR[RFEN]	Value of ERFCR[ERFEN]	Minimum number of peripheral clocks per CAN bit
16	0	0	16
32	0	0	16
64	0	0	25
96	0	0	37
16	1	0	16
32	1	0	17
64	1	0	30
96	1	0	42
16	0	1	16
32	0	1	19
64	0	1	31
96	0	1	42

For classical frame format, the minimum number of peripheral clocks per CAN bit specified in Table 512 determines the minimum peripheral clock frequency for a given number of message buffers and for an expected CAN bit rate. The CAN bit rate depends on the number of time quanta in a CAN bit. This number can be defined by adjusting one or more of the bit timing values contained in:

- Control 1 (CTRL1)
- CAN Bit Timing (CBT)
- Enhanced Nominal CAN Bit Timing (ENCBT)

The time quantum (Tq) is defined in Protocol timing. The minimum number of time quanta per CAN bit must be eight, so the oscillator clock frequency should be at least eight times the CAN bit rate.

73.3.11.1.1 Clock restrictions for CAN FD

For CAN FD frame format, some constraints must be satisfied. The equation below calculates the number of peripheral clocks per CAN bit in nominal bit rate (NumClkNomBit).

$$NumClkNomBit = \frac{f_{SYS}}{f_{CANCLK}} \times (PRESDIV + 1) \times (PROPSEG + PSEG1 + PSEG2 + 4)$$

$$= \frac{f_{SYS}}{NomBitRate}$$

Equation 21. Number of peripheral clocks per nominal CAN bit

Where PRESDIV, PSEG1, and PSEG2 are CAN bit time values in Control 1 (CTRL1). Alternatively, EPRESDIV, EPSEG1, and EPSEG2 values in CAN Bit Timing (CBT) or the values of EPRS[ENPRESDIV], ENCBT[NTSEG1], and ENCBT[NTSEG2] can be used instead. NumClkNomBit can also be calculated as a function of the expected nominal bit rate used in the arbitration phase (NomBitRate), as shown in the equation above.

The number of CAN bits in the data phase of an FD frame with BRS = 1 (fast CAN bits) depends on the number of data bytes in the payload. The number of fast CAN bits (NumOfFastBits) can be determined in Table 513. Having fewer data bytes means having fewer fast CAN bits. It also means that less time is available for FlexCAN to scan the whole message buffer memory during the internal matching and arbitration processes.

Table 513. Number of fast CAN bits in a CAN FD frame

Minimum number of data bytes	DLC field	NumOfFastBits
0	0h	21
1	1h	29
2	2h	37
3	3h	45
4	4h	53
5	5h	61
6	6h	69
7	7h	77
8	8h	85
12	9h	117
16	Ah	149
20	Bh	186
24	Ch	218
32	Dh	282
48	Eh	410
64	Fh	538

The critical part of a CAN FD frame is during the data phase, where the CAN bit rate is faster than in the arbitration phase. The minimum number of peripheral clocks per fast CAN bit (MinNumClkFastBit) can be calculated to guarantee that enough time is available for FlexCAN to scan the message buffer memory during reception and transmission. The equation below calculates this constraint.

$$MinNumClkFastBit_A = \frac{(8.5 \times MaxNumOfMb) + [ERFEN \times (2 \times NFE + 4)] + 64 - (9 \times NumClkNomBit)}{NumOfFastBits}$$

Equation 22. Minimum number of peripheral clocks per fast CAN bit for FlexCAN scan process

Where MaxNumOfMb is the maximum number of available message buffers defined in MCR[MAXMB]. NFE and ERFEN are the fields defined in Enhanced RX FIFO Control (ERFCR).

The clock-domain-crossing circuit between the CHI and PE subblocks also imposes a minimum number of peripheral clocks per fast CAN bit. This minimum is required for the handshake mechanism to work properly without losing status information through the interface, as shown in the equation below.

$$MinNumClkFastBit_B = 3 \times \left(1 + \frac{f_{SYS}}{f_{CANCLK}}\right)$$

Equation 23. Minimum number of peripheral clocks per fast CAN bit for FlexCAN clock domain interface

Therefore, the larger of the two values calculated above determines the minimum number of peripheral clocks per fast CAN bit (MinNumClkFastBit).

$$MinNumClkFastBit = Maximum\ (MinNumClkFastBitA, MinNumClkFastBitB)$$

Equation 24. Minimum number of peripheral clocks per fast CAN bit

Then, the maximum CAN bit rate in the data phase of CAN FD frames (DataBitRateMAX) can be calculated as below.

$$DataBitrate_{MAX} = \frac{f_{CANCLK}}{ROUNDUP\left(\frac{MinNumClkFastBit \times f_{CANCLK}}{f_{SYS}}\right)}$$

Equation 25. Maximum achievable baud rate for data phase

These factors affect the maximum data bit rate attainable by FlexCAN in CAN FD mode:

- The peripheral and oscillator clock frequencies
- The maximum number of message buffers
- The expected nominal bit rate

Also, the data bit rate depends on the minimum payload size of FD frames used in a given application.

To illustrate how the configuration of FlexCAN variables affects the CAN FD bit rate, consider this application example:

- The peripheral clock frequency is set to 50 MHz
- The oscillator clock frequency is set to 40 MHz

1. Considering the nominal bit rate as 1 Mbit/s, the number of peripheral clocks per CAN bit in nominal bit rate is calculated as below.

   $$NumClkNomBit = \frac{50 \times 10^6}{1 \times 10^6} = 50$$

   Equation 26. Calculation example for number of peripheral clocks per nominal CAN bit

2. The number of fast CAN bits (NumOfFastBits) is determined in Table 513. For example, if the minimum payload in FD frames is 8 bytes, there are 85 CAN bits in the data phase.
3. Assuming the maximum number of message buffers is 96, and Enhanced RX FIFO is disabled, the minimum number of peripheral clocks per fast CAN bit (MinNumClkFastBit) can be calculated.

   $$MinNumClkFastBit_A = \frac{(8.5 \times 96) + 64 - (9 \times 50)}{85} = 5.06$$

   Equation 27. Calculation example for number of peripheral clocks per fast CAN bit for FlexCAN scan process

$$MinNumClkFastBit_B = 3 \times \left(1 + \frac{50}{40}\right) = 6.75$$

Equation 28. Calculation example for number of peripheral clocks per fast CAN bit for FlexCAN clock domain interface

$$MinNumClkFastBit = Maximum\ (\ 5.06, 6.75\ ) = 6.75$$

Equation 29. Calculation example for number of peripheral clocks per fast CAN bit

4. The maximum CAN bit rate in the data phase can finally be found.

$$DataBitRate_{MAX} = \frac{40 \times 10^6}{ROUNDUP\left(\frac{6.75 \times 40 \times 10^6}{50 \times 10^6}\right)} = 6.667\ Mbps$$

Equation 30. Calculation example for maximum achievable baud rate

Even though the oscillator clock frequency (40 MHz) is adequate to generate a data rate of 8 Mbit/s in CAN FD mode, the specific FlexCAN configuration limits this rate to 6.667 Mbit/s. This limitation is mainly due to the low peripheral clock frequency that imposes the MinNumClkFastBitB bound.

Table 514 shows the maximum data rate for CAN FD with Enhanced RX FIFO disabled according to clock frequencies, payload size, and number of available message buffers. For some cases, if the number of available message buffers is reduced, FlexCAN can then achieve a data rate up to 8 Mbit/s.

Table 514. Maximum CAN bit rate in data phase on CAN FD frames with Enhanced RX FIFO disabled

Peripheral clock frequency (MHz)	Minimum payload size	Number of available message buffers	Maximum data rate (Mbit/s)
40	8	94	6.667
40	8	114	>5.0
40	12	>117	6.667
40	12	128	5.714
50	12–64	128	6.667
60	8	126	8.0
60	12	128	8.0
67	6	128	8.0
80	3	128	8.0
100	0	128	8.0

73.3.12 Reset

You can reset FlexCAN in the following ways:

- Chip-level hard reset, which resets all memory-mapped registers asynchronously.
- Soft reset:
  - MCR[SOFTRST], which resets some of the memory-mapped registers synchronously. To see which registers soft reset affects, see Table 517.

— Chip-level soft reset, which has the same effect as MCR[SOFTRST].

Soft reset is synchronous and must follow an internal request-and-acknowledge procedure across clock domains. Therefore, it may take some time to propagate its effects fully. MCR[SOFTRST] remains 1 when soft reset is pending, so software can poll this field to identify when the reset has completed. Soft reset cannot be applied when clocks are shut down in a low-power mode. The low-power mode should be exited and the clocks resumed before applying soft reset.

When the module is enabled (MCR[MDIS] becomes 0), FlexCAN automatically enters Freeze mode. In Freeze mode:

1. FlexCAN is unsynchronized to the CAN bus.
2. MCR[HALT] and MCR[FRZ] become 1.
3. The internal state machines are disabled.
4. MCR[FRZACK] and MCR[NOTRDY] become 1.

The TX pin is in the recessive state and FlexCAN does not initiate any transmission or reception of CAN frames. Reset does not affect the message buffers and the RX Individual Mask registers, so they are not automatically initialized.

73.3.13 Interrupts

FlexCAN has many interrupt sources:

- Interrupts due to message buffers
- Interrupts due to interrupts combined via an OR operator from:
  - Message buffers
  - Bus Off
  - Bus Off Done
  - Error
  - Error Fast (errors detected in the data phase of CAN FD format messages with BRS = 1)
  - TX Warning
  - RX Warning

If its corresponding IMASK bit is 1, each message buffer can be an interrupt source. There is no distinction between TX and RX interrupts for a particular buffer, under the assumption that the buffer is initialized for either transmission or reception. Each buffer has an assigned flag bit in the IFLAG registers. When the corresponding buffer completes a successful transfer, the flag is set. When the CPU writes 1 to it, the flag is cleared (unless another interrupt is generated at the same time).

NOTE

The CPU must clear only the bit causing the current interrupt. For this reason, do not use bit manipulation instructions (BSET) to clear interrupt flags. These instructions may cause accidental clearing of interrupt flags which are set after entering the current interrupt handler.

If the Legacy RX FIFO is enabled (MCR[RFEN] = 1) and DMA is disabled (MCR[DMA] = 0), the interrupts corresponding to message buffers 0–7 have different meanings.

- Bit 7 of Interrupt Flags 1 (IFLAG1) becomes the Legacy FIFO Overflow flag
- Bit 6 becomes the Legacy FIFO Warning flag
- Bit 5 becomes the Frames Available in Legacy FIFO flag
- Bits 4–0 are unused.

See Interrupt Flags 1 (IFLAG1) for more information.

If both Legacy RX FIFO and DMA are enabled (MCR[RFEN] = 1 and MCR[DMA] = 1), FlexCAN does not generate any Legacy FIFO interrupt. Bit 5 of IFLAG1 still indicates Frames Available in Legacy FIFO and generates a DMA request. Bits 7, 6, and 4–0 are unused.

CAUTION

Legacy FIFO cannot be enabled when CAN FD is enabled.

When multiple message buffer interrupt sources are combined via an OR operator into a single interrupt, the interrupt is generated when any associated message buffer (or FIFO, if applicable) generates an interrupt. In this case, the CPU must read the IFLAG registers to determine which message buffer or FIFO source caused the interrupt.

These interrupt sources generate interrupts like the message buffer interrupt sources, and can be read from Error and Status 1 (ESR1):

- Bus Off
- Bus Off Done
- Error
- Error Fast
- TX Warning
- RX Warning

The Bus Off, Error, TX Warning, and RX Warning interrupt masks are located in Control 1 (CTRL1).

73.3.14 Bus interface

CPU access to FlexCAN registers is subject to the following rules:

- Unrestricted read and write access to supervisor registers results in an access error. In Table 517, supervisor registers are registers identified with an S, and registers that are identified with S/U that are in Supervisor Mode
- Read and write access to implemented reserved address space results in an access error.
- Write access to positions whose bits are all currently read-only results in an access error. If at least one of the bits is not read-only, no access error is issued. Write permission to specific positions or some of their bits can change depending on the mode of operation or transitory state. See register and field descriptions for details.
- Read and write access to unimplemented address space results in an access error.
- Read and write access to RAM-located positions during Low-Power mode results in an access error.
- The RXIMR memory region can be considered as general-purpose memory and available for access via these methods:
  - If you write 0 to MCR[IRMQ], the individual masks (RXIMR) are disabled. In this case, the RXIMR memory region is considered general-purpose memory.
  - If MCR[MAXMB] is programmed with a value smaller than the available number of message buffers, the unused memory space can be used as general-purpose RAM space. Reserved words within RAM cannot be used.
    For example, suppose the RAM in FlexCAN can support up to 16 message buffers, CTRL2[RFFN] = 0h, and MCR[MAXMB] = 0.
    - In this case, the maximum number of message buffers becomes one.
    - The RAM starts at 0080h, and the space 0080h–008Fh is used by the one message buffer.
    - The memory space 0090h–017Fh is available.
    - The space 0180h–087Fh is reserved.
    - The space 0880h–0883h is used by the one individual mask and the available memory in the mask register space is 0884h–08BFh.
    - In the space from 08C0h–09DFh, there are reserved words for internal use which cannot be used as general-purpose RAM.

    As a rule, free memory space for general purpose depends only on MCR[MAXMB].
  - If MCR[FDEN] = 1, general-purpose memory can be used only outside Freeze mode.

Table 515. Access permissions

Supervisor access	0				1		
Supervisor mode MCR[SUPV] = 1	0			1	Any		
Modes of operation	Normal	Freeze	Low-power	Any	Normal	Freeze	Low-power
MCR	Bus error	Bus error	Bus error	Bus error	Read and write	Read and write	Read and write
CTRL1	Read and write	Read and write	Read and write	Bus error	Read and write	Read and write	Read and write
TIMER	Read and write	Read and write	Read and write	Bus error	Read and write	Read and write	Read and write
TCR	Bus error	Bus error	Bus error	Bus error	Bus error	Bus error	Bus error
RXMGMASK[1]	Bus error for write operation	Read and write	Bus error for write operation	Bus error	Bus error for write operation	Read and write	Bus error for write operation
RX14MASK[1]	Bus error for write operation	Read and write	Bus error for write operation	Bus error	Bus error for write operation	Read and write	Bus error for write operation
RX15MASK[1]	Bus error for write operation	Read and write	Bus error for write operation	Bus error	Bus error for write operation	Read and write	Bus error for write operation
ECR	Bus error for write operation	Read and write	Bus error for write operation	Bus error	Bus error for write operation	Read and write	Bus error for write operation
ESR1	Read and write	Read and write	Read and write	Bus error	Read and write	Read and write	Read and write
IMASK2	Read and write	Read and write	Read and write	Bus error	Read and write	Read and write	Read and write
IMASK1	Read and write	Read and write	Read and write	Bus error	Read and write	Read and write	Read and write
IFLAG2	Read and write	Read and write	Read and write	Bus error	Read and write	Read and write	Read and write
IFLAG1	Read and write	Read and write	Read and write	Bus error	Read and write	Read and write	Read and write
CTRL2	Read and write	Read and write	Read and write	Bus error	Read and write	Read and write	Read and write
ESR2	Read and write	Read and write	Read and write	Bus error	Read and write	Read and write	Read and write
CRCR	Bus error for write operation	Bus error for write operation	Bus error for write operation	Bus error	Bus error for write operation	Bus error for write operation	Bus error for write operation

Table continues on the next page...

Table 515. Access permissions (continued)

Supervisor access	0				1		
Supervisor mode MCR[SUPV] = 1	0			1	Any		
Modes of operation	Normal	Freeze	Low-power	Any	Normal	Freeze	Low-power
RXFGMASK[1]	Bus error for write operation	Read and write	Bus error for write operation	Bus error	Bus error for write operation	Read and write	Bus error for write operation
RXFIR[1]	Bus error for write operation	Bus error for write operation	Bus error for write operation	Bus error	Bus error for write operation	Bus error for write operation	Bus error for write operation
CBT	Bus error for write operation	Read and write	Bus error for write operation	Bus error	Bus error for write operation	Read and write	Bus error for write operation
DBG1	Bus error for write operation	Bus error for write operation	Bus error for write operation	Bus error for write operation	Bus error for write operation	Bus error for write operation	Bus error for write operation
DBG2	Bus error for write operation	Bus error for write operation	Bus error for write operation	Bus error for write operation	Bus error for write operation	Bus error for write operation	Bus error for write operation
IMASK3	Read and write	Read and write	Read and write	Bus error	Read and write	Read and write	Read and write
IFLAG3	Read and write	Read and write	Read and write	Bus error	Read and write	Read and write	Read and write
MB[1] [2]	Read and write	Read and write	Read and write	Bus error	Read and write	Read and write	Read and write
Legacy FIFO header[1]	Read and write	Read and write	Read and write	Bus error	Read and write	Read and write	Read and write
Legacy FIFO reserved space[1]	Bus error	Bus error	Bus error	Bus error	Bus error	Bus error	Bus error
Legacy FIFO filters[1]	Read and write	Read and write	Read and write	Bus error	Read and write	Read and write	Read and write
RXIMR[1]	Bus error	Read and write	Bus error	Bus error	Bus error	Read and write	Bus error
MECR	Read and write	Read and write	Read and write	Bus error	Read and write	Read and write	Read and write
ERRIAR	Read and write	Read and write	Read and write	Bus error	Read and write	Read and write	Read and write
ERRIDPR	Read and write	Read and write	Read and write	Bus error	Read and write	Read and write	Read and write
ERRIPPR	Read and write	Read and write	Read and write	Bus error	Read and write	Read and write	Read and write

Table continues on the next page...

Table 515. Access permissions (continued)

Supervisor access	0			1			
Supervisor mode MCR[SUPV] = 1	0			1	Any		
Modes of operation	Normal	Freeze	Low-power	Any	Normal	Freeze	Low-power
RERRAR [3]	Read and write	Read and write	Read and write	Bus error	Read and write	Read and write	Read and write
RERRDR[3]	Read and write	Read and write	Read and write	Bus error	Read and write	Read and write	Read and write
RERRSYNR[3]	Read and write	Read and write	Read and write	Bus error	Read and write	Read and write	Read and write
ERRSR	Read and write	Read and write	Read and write	Bus error	Read and write	Read and write	Read and write
EPRS	Bus error for write operation	Read and write	Bus error for write operation	Bus error	Bus error for write operation	Read and write	Bus error for write operation
ENCBT	Bus error for write operation	Read and write	Bus error for write operation	Bus error	Bus error for write operation	Read and write	Bus error for write operation
EDCBT	Bus error for write operation	Read and write	Bus error for write operation	Bus error	Bus error for write operation	Read and write	Bus error for write operation
ETDC	Bus error for write operation	Read and write	Bus error for write operation	Bus error	Bus error for write operation	Read and write	Bus error for write operation
FDCTRL	Read and write	Read and write	Read and write	Bus error	Read and write	Read and write	Read and write
FDCBT	Read and write[3]	Read and write	Read and write[3]	Bus error	Read and write[3]	Read and write	Read and write[3]
FDCRC	Bus error for write operation	Bus error for write operation	Bus error for write operation	Bus error	Bus error for write operation	Bus error for write operation	Bus error for write operation
ERFCR	Read and write[3]	Read and write	Read and write[3]	Bus error	Read and write[3]	Read and write	Read and write[3]
ERFIER	Read and write	Read and write	Read and write	Bus error	Read and write	Read and write	Read and write
ERFSR	Read and write	Read and write	Read and write	Bus error	Read and write	Read and write	Read and write
HR_TIME_STAMP	Read and write	Read and write	Read and write	Bus error	Read and write	Read and write	Read and write

Table continues on the next page...

Table 515. Access permissions (continued)

Supervisor access	0				1		
Supervisor mode MCR[SUPV] = 1	0			1	Any		
Modes of operation	Normal	Freeze	Low-power	Any	Normal	Freeze	Low-power
Enhanced Rx FIFO header[4]	Bus error for write operation	Bus error for write operation	Bus error for write operation	Bus error	Bus error for write operation	Bus error for write operation	Bus error for write operation
Enhanced Rx FIFO reserved space[4]	Bus error	Bus error	Bus error	Bus error	Bus error	Bus error	Bus error
ERFFEL	Bus error for write operation	Read and write	Bus error for write operation	Bus error	Bus error for write operation	Read and write	Bus error for write operation
General purpose RAM[1]	Read and write	Read and write	Read and write	Bus error	Read and write	Read and write	Read and write
Reserved space (used)[1]	Bus error	Bus error	Bus error	Bus error	Bus error	Bus error	Bus error
Reserved space (empty)[1]	Bus error	Bus error	Bus error	Bus error	Bus error	Bus error	Bus error

1. Access in low power is only possible if RAM_clk and MODULE_CLK are enabled.
2. If MCR[RFEN] = 1, see Legacy FIFO access rules in the rows below.
3. Write operation has no effect.
4. If MCR[RFEN] = 1, do not access Enhanced RX FIFO.

73.3.15 Detection and correction of memory errors

To update the parity bits in memory properly, all FlexCAN memory must be initialized before starting its operation. CTRL2[WRMFRZ] grants write access to all memory positions that require initialization, from 080h–ADFh and from C20h–31FFh. You must also initialize these registers:

- RX Message Buffers Global Mask (RXMGMASK)
- Receive 14 Mask (RX14MASK)
- Receive 15 Mask (RX15MASK)
- Legacy RX FIFO Global Mask (RXFGMASK)

MCR[RFEN] and ERFCR[ERFEN] must not be 1 during memory initialization.

FlexCAN supports detection and correction of errors in memory read accesses. Each byte of FlexCAN memory is associated with five parity bits that ensure a Hamming distance of 4. The error correction mechanism ensures that in this 13-bit word, errors in one bit can be corrected (correctable errors). Errors in two bits can be detected but not corrected (uncorrectable errors). Errors in more than two bits may not be detected. For uncorrectable errors, the error correction logic does not change the corrupted data. When a read access is performed, the parity bits are used to calculate a syndrome, which indicates the error in each byte.

When an all-zeroes or an all-ones read occurs, FlexCAN detects an uncorrectable error. See Error Report Syndrome (RERRSYNR).

Memory errors are indicated to the host via status register (Error Status (ERRSR)) and bus transfer errors. Memory errors are reported via these report registers:

- Error Report Address (RERRAR)
- Error Report Data (RERRDR)

- Error Report Syndrome (RERRSYNR)

MECR[ECCDIS] determines whether the error detection and correction mechanism can be activated. When disabled, updates on indications and reporting registers are stopped. To ensure that memory has consistent parity bits associated with the data, the parity bits are still calculated and written with data in memory write operations.

To avoid accidentally changing the critical error correction configuration, follow this protocol to enable the update of Memory Error Control (MECR):

1. Write 1 to CTRL2[ECRWRE]. (By default, CTRL2[ECRWRE] is 0 and MECR[ECRWRDIS] is 1.)
2. Write 0 to MECR[ECRWRDIS].
3. All writes to Memory Error Control (MECR) must keep MECR[ECRWRDIS] = 0.
4. After configuration is done, lock MECR by writing 1 to MECR[ECRWRDIS] or writing 0 to CTRL2[ECRWRE].

73.3.15.1 Sources of memory access

These major sources (or requestors) can access FlexCAN memory:

- Host (CPU). The largest word accessed is 32-bit.
- FlexCAN internal processes:
  - RX matching
  - TX arbitration
  - Move-in on reception
  - Move-out on transmission

  The largest word accessed is 64-bit.

The source of access determines the way that uncorrectable errors are indicated and reported.

73.3.15.2 Error indication

These flags indicate memory errors:

- ERRSR[HANCEIF]
- ERRSR[FANCEIF]
- ERRSR[CEIF]

Uncorrectable errors detected in memory reads requested by the host are indicated separately from the errors detected in requests by FlexCAN internal processes. When correctable errors are detected, FlexCAN makes no distinction of the source of the access. There are three independent flags for these three cases, and each flag raises an interrupt unless a mask in Memory Error Control (MECR) masks it. If both uncorrectable and correctable errors are found in different bytes in the same read operation, both flags are set.

An uncorrectable error detected in host access is also indicated as a bus transfer error. A bus wait request may be asserted to extend the memory transaction to the moment the report registers are updated. This indication cannot be masked. If ERRSR[HANCEIF] is not masked, the same uncorrectable error raises a bus transfer error and an interrupt request.

Each indication flag has one overrun flag in Error Status (ERRSR). The overrun flags do not request interrupts. Overrun flags for uncorrectable errors indicate that other errors of the same nature were detected after the current error being treated. Overrun flags for correctable errors indicate that other errors of the same nature were detected before the current error being treated.

The recommended handling sequence for error indication is:

1. Get error report information from report registers.
2. Use this information to take proper measures in the application.
3. Clear the ERRSR[HANCEIF], ERRSR[FANCEIF], and ERRSR[CEIF] flags.

4. If the overrun flag is active:
    a. Alert the application that at least one error could not be managed.
    b. Clear the overrun flag.

FlexCAN internal processes can access memory in transactions larger than 32 bits. For the indication, this kind of access is considered a consecutive sequence of 32-bit accesses. If errors are found in two or more 32-bit words, the interrupt and overrun flags are set simultaneously.

73.3.15.3 Error reporting

The error report registers provide detailed information about the address read, raw data, and syndrome read with error. The flags described in Error indication indicate these report registers:

- Error Report Address (RERRAR)
- Error Report Data (RERRDR)
- Error Report Syndrome (RERRSYNR)

The address, data, and syndrome registers are updated simultaneously with the error flags, according to these rules:

1. If either of the uncorrectable error flags is set, the report registers are not updated. The previous uncorrectable error reporting is preserved.
2. Otherwise, either no error flag is set or only the correctable error flag is set. The report registers are updated according to the new error, or according to the most severe new error, if uncorrectable and correctable errors are simultaneously detected.

Reporting of errors detected in accesses larger than 32 bits follows the rules described in Error indication and in the description of Error Report Address (RERRAR).

The addresses reported in RERRAR and defined in ERRIAR are not the same as the addresses listed in the module memory map. The relation between the reported addresses and the respective ones in the module memory map is shown in the description of Error Injection Address (ERRIAR).

Addresses reported when reading memory portions organized as FIFOs refer to the address of the specific entry accessed in the FIFO, not to the FIFO base address. Such memory portions include:

- Legacy RX FIFO Structure
- Enhanced RX FIFO Structure
- Legacy RX FIFO Information (RXFIR)

To ensure coherence of error report registers, disable the report update by writing 1 to MECR[RERRDIS] before reading the report registers.

73.3.15.4 Error response

Correctable errors have no consequence to FlexCAN operation because the host or FlexCAN internal processes corrects affected data before its use.

For host-initiated reads, an uncorrectable error may affect the host, but does not affect FlexCAN operation.

Uncorrectable errors detected on memory read operations requested by FlexCAN internal processes may result in incorrect operation depending on the state of MECR[NCEFAFRZ], as follows:

- During reception (either matching or move-in process), when an uncorrectable error occurs:
    — An incorrect destination may be selected to store the incoming frame.
    — A corrupted frame may be stored in the correct destination.
    — Both of these events may occur.

If MECR[NCEFAFRZ] = 1, FlexCAN stops operation automatically and enters Freeze mode to prevent corrupted data from being treated as valid by FlexCAN internal processes. If MECR[NCEFAFRZ] = 0, FlexCAN continues working, and a corrupted frame is received.

- During arbitration, when an uncorrectable error occurs:
  - A non-highest-priority TX message buffer may be mistakenly selected for transmission.
  - Its data may be corrupted.

  If MECR[NCEFAFRZ] = 1, FlexCAN stops operation automatically and enters Freeze mode before starting the move-out. If MECR[NCEFAFRZ] = 0, FlexCAN proceeds to move-out with a corrupted frame that will be transmitted on the CAN bus.
- During move-out, when an uncorrectable error occurs, a corrupted frame is copied from the selected TX MB that won the arbitration to the TX SMB for transmission. If MECR[NCEFAFRZ] = 1, FlexCAN stops operation automatically and enters Freeze mode before starting the transmission. When MECR[NCEFAFRZ] = 0, the corrupted frame is transferred from the TX SMB to the Protocol Engine (PE) subblock and is transmitted on the CAN bus.
- An uncorrectable error can also be detected beyond move-out, when TX data is read from TX SMB (buffer located in RAM) to be transferred to the PE subblock for transmission. In this case, a frame with corrupted ID or data is transmitted on the CAN bus.

  To prevent the external nodes from successfully receiving the frame, FlexCAN inverts all bits in the CRC field (CRC sequence plus CRC delimiter). Also, it transmits an error flag just after CRC delimiter due to self-detecting a Bit1 error and a form error due to the CRC field inversion. When MECR[NCEFAFRZ] = 1, FlexCAN stops operation automatically and enters Freeze mode just after the error frame. When MECR[NCEFAFRZ] = 0, FlexCAN may attempt to retransmit the same frame, as long as no other higher-priority TX MB is configured for transmission after.

  If the uncorrectable error persists, FlexCAN eventually reaches the Bus Off state due to consecutive error detections. ECR[TXERRCNT] is updated every time FlexCAN inverts the CRC field, causing errors as described above.

When MECR[NCEFAFRZ] = 1 and FlexCAN enters Freeze mode, only the CPU can cause FlexCAN to exit Freeze mode and resume Normal mode. MECR[NCEFAFRZ] becoming 1 is the only way to prevent corrupted frames from being transmitted on the CAN bus to the move-out internal process.

The error report registers can provide information to the application for customized management of these situations.

73.3.15.5 Error injection

These error injection registers are used to inject errors in memory reads to force errors and update the indication and reporting registers:

- Error Injection Address (ERRIAR)
- Error Injection Data Pattern (ERRIDPR)
- Error Injection Parity Pattern (ERRIPPR)

The relation between the error injection addresses and the corresponding addresses in the module memory map is shown in Error Injection Address (ERRIAR).

The injection is done by flipping the data and parity bits corresponding to the bits set to 1 in ERRIDPR and ERRIPPR. You can select injection specifically for memory accesses requested by the host or by FlexCAN internal processes.

For accesses larger than 32 bits, MECR[EXTERRIE] extends the injection pattern, replicating it in 32-bit words to fill the width of the access.

NOTE

It is possible, but very unlikely, for error injection to correct a bit with error. This correction does not raise the error flags and reports as expected.

To ensure coherence among error injection registers and avoid spurious error injections, you must clear MECR[HAERRIE] and MECR[FAERRIE] when configuring the memory injection registers.

73.4 External signal descriptions

FlexCAN has two I/O signals connected to the external chip pins. These signals are summarized in Table 516 and described in the following subsections.

Table 516. FlexCAN signal descriptions

Signal	Description	I/O
CAN RX	CAN receive pin	Input
CAN TX	CAN transmit pin	Output

73.4.1 CAN RX

This pin is the receive pin from the CAN bus transceiver. Logic level 0 represents its dominant state. Logic level 1 represents its recessive state.

73.4.2 CAN TX

This pin is the transmit pin to the CAN bus transceiver. Logic level 0 represents its dominant state. Logic level 1 represents its recessive state.

73.5 Initialization and application information

73.5.1 FlexCAN initialization sequence

For any configuration change or initialization, you must put FlexCAN into Freeze mode (see Freeze mode). The module must be initialized after every reset. FlexCAN memory must be initialized before switching to functional mode.

The following is a generic initialization sequence applicable to FlexCAN:

1. Initialize Module Configuration (MCR).
    a. Enable the individual filtering per message buffer and reception queue features by writing 1 to MCR[IRMQ].
    b. Enable the warning interrupts by writing 1 to MCR[WRNEN].
    c. If required, disable frame self-reception by writing 1 to MCR[SRXDIS].
    d. Enable the Legacy RX FIFO by writing 1 to MCR[RFEN] or enable the Enhanced RX FIFO by writing 1 to ERFCR[ERFEN].
    e. If Legacy RX FIFO or Enhanced RX FIFO is enabled and DMA is required, write 1 to MCR[DMA].
    f. Enable the abort mechanism by writing 1 to MCR[AEN].
    g. Enable the local priority feature by writing 1 to MCR[LPRIOEN].
2. Initialize Control 1 (CTRL1) and CAN FD Bit Timing (FDCBT). Optionally initialize CAN Bit Timing (CBT).
    a. Determine the bit timing parameters: CTRL1[PROPSEG], CTRL1[PSEG1], CTRL1[PSEG2], and CTRL1[RJW].
    b. Optionally determine the bit timing parameters: CBT[EPROPSEG], CBT[EPSEG1], CBT[EPSEG2], and CBT[ERJW].
    c. Determine the CAN FD bit timing parameters: FDCBT[FPROPSEG], FDCBT[FPSEG1], FDCBT[FPSEG2], and FDCBT[FRJW].
    d. Determine the bit rate by programming CTRL1[PRESDIV] and optionally programming CBT[EPRESDIV].
    e. Determine the CAN FD bit rate by programming FDCBT[FPRESDIV].
    f. Determine the internal arbitration mode by programming CTRL1[LBUF].

3. If Error Code Correction (ECC) is enabled, you must initialize all FlexCAN memory. See Detection and correction of memory errors.
4. Initialize the message buffers. (See Message buffer structure for message buffer details.)
    a. The control and status word of all message buffers must be initialized.
    b. If RX FIFO is enabled, the ID filter table must be initialized.
    c. Other entries in each message buffer should be initialized as required.
5. Initialize Receive Individual Mask (RXIMR0 - RXIMR95).
6. Write 1 to required interrupt mask bits in:
    - IMASK registers (for all message buffer interrupts)
    - Control 1 (CTRL1) and Control 2 (CTRL2) (for Bus Off and Error interrupts)
7. Write 0 to MCR[HALT].

After the last step listed above, FlexCAN attempts to synchronize to the CAN bus.

73.6 Memory map and register definition

This section describes the registers and data structures in FlexCAN. The base address of the module depends on the particular memory map of the chip.

73.6.1 FlexCAN memory mapping

The address space occupied by FlexCAN has 128 bytes for registers starting at the module base address, followed by embedded RAM starting at address offset 0080h.

Each individual register is identified by its complete name and corresponding mnemonic. The access type can be Supervisor (S) or Unrestricted (U). Most registers can be configured to have either Supervisor or Unrestricted access by programming MCR[SUPV]. These registers are identified as S/U in the Access column of Table 517.

NOTE

An invalid register access results in a bus error. Invalid accesses include reading a write-only register, writing a read-only register, and accessing an invalid address.

NOTE

To update the parity bits in memory properly, all FlexCAN memory must be initialized before reading registers which are implemented in memory. CTRL2[WRMFRZ] grants write access to all memory positions that require initialization, from 080h–ADFh and from C20h–31FFh. You must also initialize RX Message Buffers Global Mask (RXMGMASK), Receive 14 Mask (RX14MASK), Receive 15 Mask (RX15MASK) and Legacy RX FIFO Global Mask (RXFGMASK). MCR[RFEN] and ERFCR[ERFEN] must not be 1 during memory initialization.

Table 517. Register access and reset information

Register	Access type	Affected by hard reset	Affected by soft reset
Module Configuration (MCR)	S	Yes	Yes
Control 1 (CTRL1)	S/U	Yes	No
Free-Running Timer (TIMER)	S/U	Yes	Yes
RX Message buffers Global Mask (RXMGMASK)	S/U	No	No

Table continues on the next page...

Table 517. Register access and reset information (continued)

Register	Access type	Affected by hard reset	Affected by soft reset
RX Buffer 14 Mask (RX14MASK)	S/U	No	No
RX Buffer 15 Mask (RX15MASK)	S/U	No	No
Error Counter (ECR)	S/U	Yes	Yes
Error and Status 1 (ESR1)	S/U	Yes	Yes
Interrupt Masks 2 (IMASK2)	S/U	Yes	Yes
Interrupt Masks 1 (IMASK1)	S/U	Yes	Yes
Interrupt Flags 2 (IFLAG2)	S/U	Yes	Yes
Interrupt Flags 1 (IFLAG1)	S/U	Yes	Yes
Control 2 (CTRL2)	S/U	Yes	No
Error and Status 2 (ESR2)	S/U	Yes	Yes
Cyclic Redundancy Check (CRCR)	S/U	Yes	Yes
RX FIFO Global Mask (RXFGMASK)	S/U	No	No
RX FIFO Information (RXFIR)	S/U	No	No
CAN Bit Timing (CBT)	S/U	Yes	No
Interrupt Masks 3 (IMASK3)	S/U	Yes	Yes
Interrupt Flags 3 (IFLAG3)	S/U	Yes	Yes
Message buffers	S/U	No	No
RX Individual Masks	S/U	No	No
Memory Error Control (MECR)	S/U	Yes	Yes
Error Injection Address (ERRIAR)	S/U	Yes	Yes
Error Injection Data Pattern (ERRIDPR)	S/U	Yes	Yes
Error Injection Parity Pattern (ERRIPPR)	S/U	Yes	Yes
Error Report Address (RERRAR)	S/U	Yes	Yes
Error Report Data (RERRDR)	S/U	Yes	Yes
Error Report Syndrome (RERRSYNR)	S/U	Yes	Yes

Table continues on the next page...

Table 517. Register access and reset information (continued)

Register	Access type	Affected by hard reset	Affected by soft reset
Error Status (ERRSR)	S/U	Yes	Yes
Enhanced CAN Bit Timing Prescalers (EPRS)	S/U	Yes	No
Enhanced Nominal CAN Bit Timing (ENCBT)	S/U	Yes	No
Enhanced Data Phase CAN bit Timing (EDCBT)	S/U	Yes	No
Enhanced Transceiver Delay Compensation (ETDC)	S/U	Yes	No
CAN FD Control (FDCTRL)	S/U	Yes	No
CAN FD Bit Timing (FDCBT)	S/U	Yes	No
CAN FD CRC (FDCRC)	S/U	Yes	Yes
Enhanced RX FIFO Control (ERFCR)	S/U	Yes	Yes
Enhanced RX FIFO Interrupt Enable (ERFIER)	S/U	Yes	Yes
Enhanced RX FIFO Status (ERFSR)	S/U	Yes	Yes
High-Resolution Timestamp (HR_TIME_STAMP)	S/U	No	No
Enhanced RX FIFO	S/U	No	No
Enhanced RX FIFO Filter Element (ERFFEL)	S/U	No	No

FlexCAN can store CAN messages for transmission and reception using message buffers and RX FIFO structures.

73.6.2 FlexCAN register descriptions

The following table shows the FlexCAN memory map.

The address range from offset 80h–67Fh allocates the ninety-six 128-bit message buffers. The memory maps for the message buffers are in FlexCAN message buffer memory map.

The address range from offset 2000h–204Ch allocates the Enhanced RX FIFO output, and the address range from offset 2050h–263Ch allocates the rest of Enhanced RX FIFO 19 elements. The memory map for the Enhanced RX FIFO is in Enhanced RX FIFO structure.

73.6.2.1 FlexCAN memory map

CAN_0 base address: 4030_4000h

CAN_1 base address: 4030_8000h

CAN_2 base address: 4030_C000h

CAN_3 base address: 4031_0000h

CAN_4 base address: 4031_4000h

CAN_5 base address: 4031_8000h

CAN_6 base address: 4031_C000h

CAN_7 base address: 4032_0000h

CAN_8 base address: 4057_0000h

CAN_9 base address: 4057_4000h

CAN_10 base address: 4057_8000h

CAN_11 base address: 4057_C000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Module Configuration (MCR)	32	RW	D890_000Fh
4h	Control 1 (CTRL1)	32	RW	0000_0000h
8h	Free-Running Timer (TIMER)	32	RW	0000_0000h
10h	RX Message Buffers Global Mask (RXMGMASK)	32	RW	See section
14h	Receive 14 Mask (RX14MASK)	32	RW	See section
18h	Receive 15 Mask (RX15MASK)	32	RW	See section
1Ch	Error Counter (ECR)	32	RW	0000_0000h
20h	Error and Status 1 (ESR1)	32	RW	0000_0000h
24h	Interrupt Masks 2 (IMASK2)	32	RW	0000_0000h
28h	Interrupt Masks 1 (IMASK1)	32	RW	0000_0000h
2Ch	Interrupt Flags 2 (IFLAG2)	32	RW	0000_0000h
30h	Interrupt Flags 1 (IFLAG1)	32	RW	0000_0000h
34h	Control 2 (CTRL2)	32	RW	See section
38h	Error and Status 2 (ESR2)	32	R	0000_0000h
44h	Cyclic Redundancy Check (CRCR)	32	R	0000_0000h
48h	Legacy RX FIFO Global Mask (RXFGMASK)	32	RW	See section
4Ch	Legacy RX FIFO Information (RXFIR)	32	R	See section
50h	CAN Bit Timing (CBT)	32	RW	0000_0000h
6Ch	Interrupt Masks 3 (IMASK3)	32	RW	0000_0000h
74h	Interrupt Flags 3 (IFLAG3)	32	RW	0000_0000h
880h - 9FCh	Receive Individual Mask (RXIMR0 - RXIMR95)	32	RW	See section
AE0h	Memory Error Control (MECR)	32	RW	800C_0080h
AE4h	Error Injection Address (ERRIAR)	32	RW	0000_0000h
AE8h	Error Injection Data Pattern (ERRIDPR)	32	RW	0000_0000h
AECh	Error Injection Parity Pattern (ERRIPPR)	32	RW	0000_0000h
AF0h	Error Report Address (RERRAR)	32	R	0000_0000h
AF4h	Error Report Data (RERRDR)	32	R	0000_0000h
AF8h	Error Report Syndrome (RERRSYNR)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
AFCh	Error Status (ERRSR)	32	RW	0000_0000h
BF0h	Enhanced CAN Bit Timing Prescalers (EPRS)	32	RW	0000_0000h
BF4h	Enhanced Nominal CAN Bit Timing (ENCBT)	32	RW	0000_0000h
BF8h	Enhanced Data Phase CAN Bit Timing (EDCBT)	32	RW	0000_0000h
BFCh	Enhanced Transceiver Delay Compensation (ETDC)	32	RW	0000_0000h
C00h	CAN FD Control (FDCTRL)	32	RW	8000_0100h
C04h	CAN FD Bit Timing (FDCBT)	32	RW	0000_0000h
C08h	CAN FD CRC (FDCRC)	32	R	0000_0000h
C0Ch	Enhanced RX FIFO Control (ERFCR)	32	RW	0000_0000h
C10h	Enhanced RX FIFO Interrupt Enable (ERFIER)	32	RW	0000_0000h
C14h	Enhanced RX FIFO Status (ERFSR)	32	RW	0000_0000h
C30h - DACh	High-Resolution Timestamp (HR_TIME_STAMP0 - HR_TIME_STAMP95)	32	RW	See section
3000h - 31FCh	Enhanced RX FIFO Filter Element (ERFFEL0 - ERFFEL127)	32	RW	See section

73.6.2.2 Module Configuration (MCR)

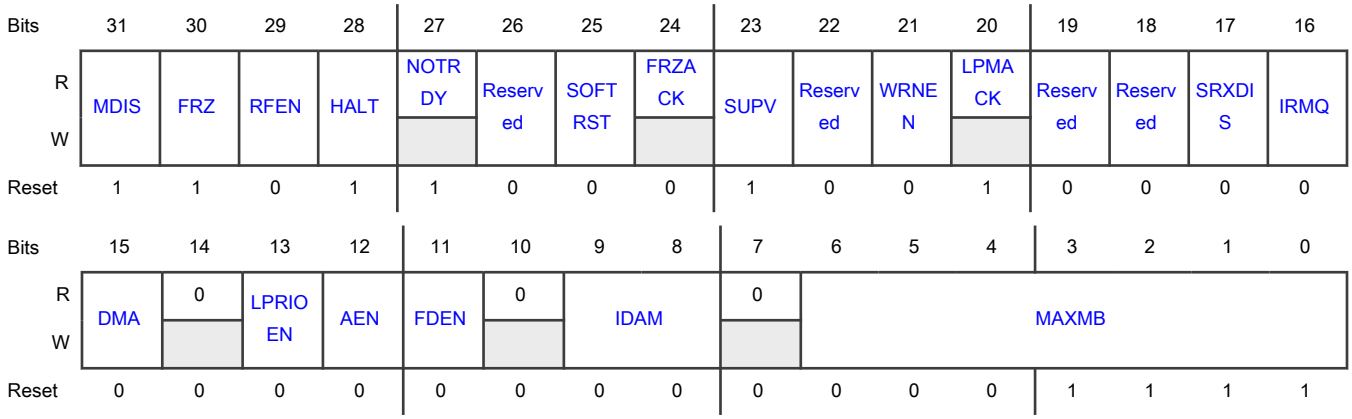
Offset

Register	Offset
MCR	0h

Function

Defines global system configurations, including the module operation modes and the maximum message buffer configuration.

Diagram



Fields

Field	Function
31 MDIS	Module Disable Disables FlexCAN. When disabled, FlexCAN disables the clocks to the CAN Protocol Engine and Controller Host Interface submodules. Soft reset does not affect this field. 0b - Enable 1b - Disable
30 FRZ	Freeze Enable Specifies FlexCAN behavior when MCR[HALT] = 1 or when Debug mode is requested at chip level. When this field becomes 1, FlexCAN can enter Freeze mode. Writing 0 to this field causes FlexCAN to exit from Freeze mode. The chip writes 1 to this field when a noncorrectable error is detected (MECR[NCEFAFRZ] becomes 1). 0b - Disable 1b - Enable
29 RFEN	Legacy RX FIFO Enable Enables the Legacy RX FIFO feature. When this field is 1, message buffers 0–5 cannot be used for normal reception and transmission. The corresponding memory region (80h–DCh) is used by the FIFO engine and additional message buffers (up to 32, depending on CTRL2[RFFN]). These message buffers are used as Legacy RX FIFO ID filter table elements. This field also impacts the definition of the minimum number of peripheral clocks per CAN bit as described in Table 512. This field can be written in Freeze mode only; the module blocks it in other modes. **NOTE** When CAN FD operation is enabled (see MCR[FDEN]), you cannot write 1 to this field. **NOTE** This field must not be 1 if ERFCR[ERFEN] = 1. 0b - Disable 1b - Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
28 HALT	Halt FlexCAN Puts FlexCAN into Freeze mode. The CPU should write 0 to this field after initializing the message buffers and Control 1 (CTRL1) and Control 2 (CTRL2). FlexCAN performs no reception or transmission before this field becomes 0. Freeze mode cannot be entered when FlexCAN is in a low-power mode. The module writes 1 to this field when a noncorrectable error is detected and MECR[NCEFAFRZ] = 1. 0b - No request 1b - Enter Freeze mode, if MCR[FRZ] = 1.
27 NOTRDY	FlexCAN Not Ready Indicates whether FlexCAN is in Disable mode or Freeze mode. When FlexCAN has exited these modes, this field becomes 0. Soft reset does not affect this field. 0b - FlexCAN is in Normal mode, Listen-Only mode, or Loopback mode. 1b - FlexCAN is in Disable mode or Freeze mode.
26 —	Reserved When writing to this field, always write the reset value.
25 SOFTRST	Soft Reset Resets internal state machines of FlexCAN and some memory-mapped registers. The CPU can write 1 to this field directly. Because soft reset is synchronous and must follow a request-and-acknowledge procedure across clock domains, it may take some time to propagate its effect fully. When reset is pending, this field remains 1; it automatically becomes 0 when reset completes. You can poll this field to know when the soft reset has completed. Soft reset cannot be applied when clocks are shut down in a low-power mode. Transfer the module out of the low-power mode before applying soft reset. Soft reset does not affect this field. **NOTE** This field becomes 0 within 2 CAN bits after assertion of this bit. 0b - No reset 1b - Soft reset affects reset registers
24 FRZACK	Freeze Mode Acknowledge Indicates whether FlexCAN is in Freeze mode and its prescaler is stopped. The Freeze mode request cannot be granted until current transmission or reception processes have finished. Therefore, you can poll this field to know when FlexCAN has entered Freeze mode. If the Freeze mode request is negated, this field becomes 0 after the FlexCAN prescaler is running again. If Freeze mode is requested when FlexCAN is in a low-power mode, this field becomes 1 only when the low-power mode is exited. See Freeze mode. Soft reset does not affect this field.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field becomes 1 within 180 CAN bits if current transmission or reception is ongoing for classical frames and 730 CAN bits if current transmission or reception is ongoing for FD frames after the low-power mode requested by CPU. This field becomes 0 within 2 CAN bits after the Freeze mode request removal (see Protocol timing). 0b - Not in Freeze mode, prescaler running. 1b - In Freeze mode, prescaler stopped.
23 SUPV	Supervisor Mode Configures FlexCAN to be either in Supervisor or User mode. The registers that this field affects are marked as S/U in the Access type column of Table 517. The affected registers start with Supervisor access allowance only. In User mode, affected registers allow both Supervisor and Unrestricted accesses. In Supervisor mode, affected registers allow only Supervisor access. Unrestricted access behaves as though the access is performed on an unimplemented register location. This field can be written only in Freeze mode; the module blocks it in other modes. 0b - User mode 1b - Supervisor mode
22 —	Reserved When writing to this field, always write the reset value.
21 WRNEN	Warning Interrupt Enable Enables the generation of the flags ESR1[TWRNINT] and ESR1[RWRNINT]. When this field is 1, TWRNINT and RWRNINT flags are set when the respective error counter transitions from less than 96 to greater than or equal to 96. When this field is 0, the TWRNINT and RWRNINT flags are always zero, independent of the values of the error counters. No warning interrupt is generated. This field can be written in Freeze mode only; the module blocks it in other modes. 0b - Disable 1b - Enable
20 LPMACK	Low-Power Mode Acknowledge Indicates whether FlexCAN is in a low-power mode (Disable mode). A low-power mode cannot be entered until all current transmission and reception processes have finished. The CPU can poll this field to know when FlexCAN has entered low-power mode. Soft reset does not affect this field. **NOTE** This field becomes 1 within 180 CAN bits if current transmission or reception is ongoing for classical frames and 730 CAN bits if current transmission or reception is ongoing for FD frames after the low-power mode request removal (see Protocol timing). 0b - Not in a low-power mode

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - In a low-power mode
19 —	Reserved When writing to this field, always write the reset value.
18 —	Reserved When writing to this field, always write the reset value.
17 SRXDIS	Self-Reception Disable Determines whether FlexCAN can receive frames transmitted by itself. If 1, frames transmitted by the module are not stored in any MB, regardless of whether the MB is programmed with an ID that matches the transmitted frame. No interrupt flag or interrupt signal is generated due to the frame reception. This field can be written only in Freeze mode; the module blocks it in other modes. 0b - Enable 1b - Disable
16 IRMQ	Individual RX Masking and Queue Enable Indicates whether RX matching process is based on individual masking and queue, or based on a masking scheme with RX Message Buffers Global Mask (RXMGMASK), Receive 14 Mask (RX14MASK), Receive 15 Mask (RX15MASK), and Legacy RX FIFO Global Mask (RXFGMASK). When this field is disabled, for backward compatibility with legacy applications, reading the Control and Status word locks the MB even if it is empty. This field can be written in Freeze mode only. The module blocks it in other modes. 0b - Disable 1b - Enable
15 DMA	DMA Enable Enables DMA. The DMA feature can only be used in Legacy RX FIFO or Enhanced RX FIFO, so MCR[RFEN] or ERFCR[ERFEN] must be 1. When DMA and RFEN are 1, IFLAG1[BUF5I] generates the DMA request, and no RX FIFO interrupt is generated. This field can be written in Freeze mode only; the module blocks it in other modes. 0b - Disable 1b - Enable
14 —	Reserved
13 LPRIOEN	Local Priority Enable Enables the local priority feature. It is used to expand the ID used during the arbitration process. With this expanded ID concept, the arbitration process is done based on the full 32-bit word. However, the actual transmitted ID is 11 bits for standard frames and 29 bits for extended frames. This field can be written only in Freeze mode; the module blocks it in other modes.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This bit is provided for backward compatibility with legacy applications. 0b - Disable 1b - Enable
12 AEN	Abort Enable Enables the TX abort mechanism. This mechanism guarantees a safe procedure for aborting a pending transmission, so that no frame is sent in the CAN bus without notification. This field can be written only in Freeze mode; the module blocks it in other modes. **NOTE** When this field is 1, only use the abort mechanism (see Transmission abort mechanism) to update message buffers configured for transmission. **CAUTION** Writing the abort code into RX message buffers can cause unpredictable results when this field is 1. 0b - Disabled 1b - Enabled
11 FDEN	CAN FD Operation Enable Enables the CAN with flexible data rate (CAN FD) operation. This field can be written in Freeze mode only. FlexCAN can receive and transmit messages in CAN 2.0 format. If this field is enabled, FlexCAN can also receive and transmit messages in CAN FD format. FlexCAN can transmit FD frame format according to ISO 11898-1:2015. **NOTE** If the value of this field is 1, the Legacy RX FIFO Enable (MCR[RFEN]) field cannot be 1. 0b - Disable 1b - Enable
10 —	Reserved
9-8 IDAM	ID Acceptance Mode Identifies the format of the Legacy RX FIFO ID filter table elements. This field configures all elements of the table at the same time; they are all the same format. See Legacy RX FIFO structure. This field can be written only in Freeze mode; the module blocks it in other modes. 00b - Format A: One full ID (standard and extended) per ID filter table element. 01b - Format B: Two full standard IDs or two partial 14-bit (standard and extended) IDs per ID filter table element. 10b - Format C: Four partial 8-bit standard IDs per ID filter table element.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	11b - Format D: All frames rejected.
7 —	Reserved
6-0 MAXMB	Number of the Last Message Buffer Defines the number of the last message buffer that takes part in the matching and arbitration processes. The reset value (0Fh) is equivalent to a 16-MB configuration. This field can be written only in Freeze mode; the module blocks it in other modes. **NOTE** You must write a value smaller than or equal to the number of available message buffers to this field, as described in FlexCAN memory partition for CAN FD. Additionally, the MAXMB value must consider the region of message buffers occupied by Legacy RX FIFO and its ID filters table space defined by CTRL2[RFFN]. MAXMB also impacts the definition of the minimum number of peripheral clocks per CAN bit, as described in Table 512.

73.6.2.3 Control 1 (CTRL1)

Offset

Register	Offset
CTRL1	4h

Function

Contains specific FlexCAN control features related to the CAN bus. These features include bit rate, programmable sampling point within an RX bit, Loopback mode, Listen-Only mode, Bus Off recovery behavior, and interrupt enabling (Bus-Off, Error, Warning). It also determines the division factor for the clock prescaler.

The CAN bit timing variables (CTRL1[PRESDIV], CTRL1[PROPSEG], CTRL1[PSEG1], CTRL1[PSEG2], and CTRL1[RJW]) can also be configured in CAN Bit Timing (CBT), which extends the range of all these variables. If CBT[BTF] = 1, CTRL1[PRESDIV], CTRL1[PROPSEG], CTRL1[PSEG1], CTRL1[PSEG2], and CTRL1[RJW] become read-only.

If CTRL2[BTE] = 1, CTRL1[PRESDIV], CTRL1[PROPSEG], CTRL1[PSEG1], CTRL1[PSEG2], and CTRL1[RJW] are not used by the module. Instead, these fields are read as zero, and a write operation to them has no effect.

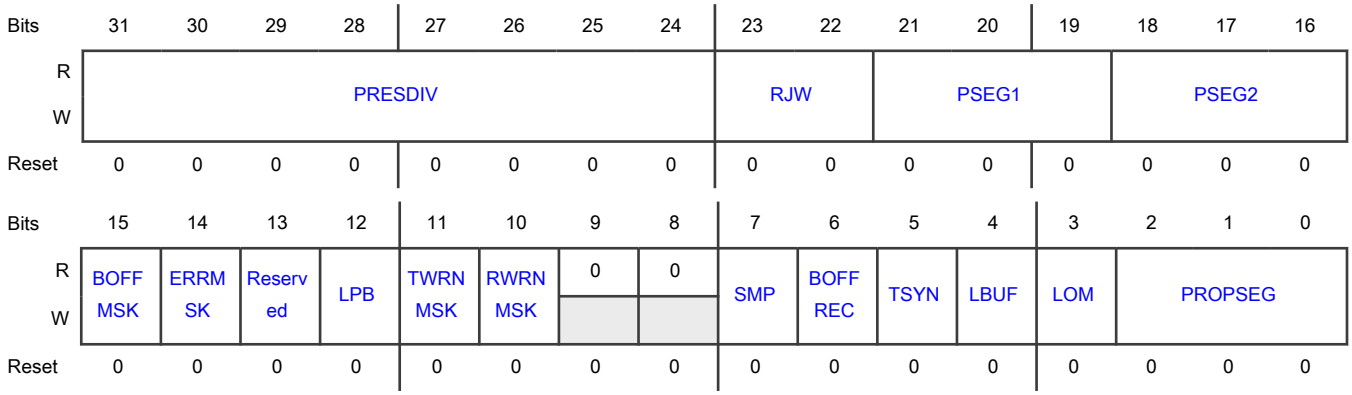
NOTE

When the CAN FD feature is enabled, do not use CTRL1[PRESDIV], CTRL1[PROPSEG], CTRL1[PSEG1], CTRL1[PSEG2], and CTRL1[RJW] for CAN bit timing. Instead use CBT[EPRESDIV], CBT[ERJW], CBT[EPSEG1], CBT[EPSEG2], and CBT[EPROPSEG].

The CAN bit variables in CTRL1 and in CBT are stored in the same internal register.

Soft reset does not affect the contents of this register.

Diagram



Fields

Field	Function
31-24 PRESDIV	Prescaler Division Factor Determines the ratio between the PE clock frequency and the serial clock (Sclock) frequency. The Sclock period defines the time quantum of the CAN protocol. For the reset value, the Sclock frequency is equal to the PE clock frequency. The maximum value of this field is FFh, which gives a minimum Sclock frequency equal to the PE clock frequency divided by 256. See Protocol timing for more information. This field can be written only in Freeze mode; the module blocks it in other modes. Sclock frequency = PE clock frequency ÷ (PRESDIV + 1).
23-22 RJW	Resync Jump Width Defines the maximum number of time quanta that one resynchronization can change a bit time. One time quantum is equal to one Sclock period. The valid programmable values are 0–3. See Protocol timing for more information. This field can be written only in Freeze mode; the module blocks it in other modes. Resync Jump Width = RJW + 1.
21-19 PSEG1	Phase Segment 1 Defines the length of phase segment 1 in the bit time. The valid programmable values are 0–7. See Protocol timing for more information. This field can be written only in Freeze mode; the module blocks it in other modes. Phase Buffer Segment 1 = (PSEG1 + 1) × Time Quanta.
18-16 PSEG2	Phase Segment 2 Defines the length of phase segment 2 in the bit time. The valid programmable values are 1–7. See Protocol timing for more information. This field can be written only in Freeze mode; the module blocks it in other modes. Phase Buffer Segment 2 = (PSEG2 + 1) × Time Quanta.
15 BOFFMSK	Bus Off Interrupt Mask Provides a mask for the Bus Off interrupt ESR1[BOFFINT]. 0b - Interrupt disabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Interrupt enabled
14 ERRMSK	Error Interrupt Mask Provides a mask for the Error interrupt ESR1[ERRINT]. 0b - Interrupt disabled 1b - Interrupt enabled
13 —	Reserved **NOTE** Writeable only if the module is disabled. Otherwise the access type is read-only.
12 LPB	Loopback Mode Configures FlexCAN to operate in Loopback mode. In this mode, FlexCAN performs an internal loopback that can be used for self-test operation. The bit stream output of the transmitter is fed back internally to the receiver input. The RX CAN input pin is ignored and the TX CAN output goes to the recessive state (logic 1). FlexCAN behaves as it normally does when transmitting, and treats its own transmitted message as a message received from a remote node. In this mode, FlexCAN ignores the bit sent during the ACK slot in the CAN frame acknowledge field. It generates an internal acknowledge bit to ensure proper reception of its own message. Both transmit and receive interrupts are generated. This field can be written only in Freeze mode; the module blocks it in other modes. **NOTE** In this mode, MCR[SRXDIS] cannot become 1, because it would impede the self-reception of a transmitted message. **NOTE** FDCTRL[TDCEN] and MCR[ETDCEN] must be 0 when this field is 1. 0b - Disabled 1b - Enabled
11 TWRNMSK	TX Warning Interrupt Mask Provides a mask for the TX Warning interrupt associated with the ESR1[TWRNINT] flag. When MCR[WRNEN] = 0, this field is read as 0. This field can be written only if MCR[WRNEN] = 1. 0b - Disabled 1b - Enabled
10 RWRNMSK	RX Warning Interrupt Mask Provides a mask for the RX Warning interrupt associated with the ESR1[RWRNINT] flag. When MCR[WRNEN] = 0, this field is read as 0. This field can be written only if MCR[WRNEN] = 1. 0b - Disabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Enabled
9 —	Reserved
8 —	Reserved
7 SMP	CAN Bit Sampling Determines the sampling mode of CAN bits at the RX input. This field can be written in Freeze mode only; the module blocks it in other modes. **NOTE** For proper operation, to write 1 to this field, you must guarantee a minimum value of two time quanta in CTRL1[PSEG1] (or CBT[EPSEG1]).This bit cannot become 1 when CAN FD is enabled (MCR[FDEN] = 1). 0b - One sample is used to determine the bit value. 1b - Three samples are used to determine the value of the received bit: the regular one (sample point) and two preceding samples. A majority rule is used.
6 BOFFREC	Bus Off Recovery Determines how FlexCAN recovers from Bus Off state. If 0, automatic recovering from Bus Off state occurs according to the CAN Specification 2.0B. If 1, automatic recovering from Bus Off is disabled. The module remains in Bus Off state until you write 1 to this field. If this field becomes 0 before 128 sequences of 11 recessive bits are detected on the CAN bus, Bus Off recovery happens as if this field had never become 1. If this field becomes 0 after 128 sequences of 11 recessive bits occurred, FlexCAN resynchronizes to the bus. It waits for 11 recessive bits before joining the bus. After this field becomes 0, it can become 1 again during Bus Off, but it will only be effective the next time the module enters Bus Off. If this field becomes 0 when the module is in Bus Off, writing 1 to this field is not effective for the current Bus Off recovery. **NOTE** See Bus Off in the CAN protocol standard ISO 11898-1:2015 for details. 0b - Enabled 1b - Disabled
5 TSYN	Timer Sync Enables a mechanism that resets the free-running timer each time a message is received in message buffer 0. This feature provides the means to synchronize multiple FlexCAN stations with a special "SYNC" message (that is, global network time). If MCR[RFEN] = 1 (Legacy RX FIFO enabled), the first available message buffer, according to CTRL2[RFFN], is used for timer synchronization instead of MB0. This field can be written in Freeze mode only; the module blocks it in other modes.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disable 1b - Enable
4 LBUF	Lowest Buffer Transmitted First Determines the ordering mechanism for message buffer transmission. When 1, MCR[LPRIOEN] does not affect the priority arbitration. This field can be written in Freeze mode only; the module blocks it in other modes. 0b - Buffer with highest priority is transmitted first. 1b - Lowest number buffer is transmitted first.
3 LOM	Listen-Only Mode Configures FlexCAN to operate in Listen-Only mode. In this mode, transmission is disabled, all error counters described in Error Counter (ECR) are frozen, and the module operates in CAN Error Passive mode. Only messages acknowledged by another CAN station are received. If FlexCAN detects an unacknowledged message, it flags a BIT0 error without changing the receive error counter (ECR[RXERRCNT]), as if it is trying to acknowledge the message. FlexCAN acknowledges Listen-Only mode when ESR1[FLTCONF] = 1, indicating the Error Passive state. There can be some delay between the Listen-Only mode request and its acknowledgment. This field can be written in Freeze mode only; the module blocks it in other modes. 0b - Listen-Only mode is deactivated. 1b - FlexCAN module operates in Listen-Only mode.
2-0 PROPSEG	Propagation Segment Defines the length of the propagation segment in the bit time. The valid programmable values are 0–7. This field can be written only in Freeze mode; the module blocks it in other modes. Propagation segment time = (PROPSEG + 1) × Time Quanta. One Time Quantum = one Sclock period.

73.6.2.4 Free-Running Timer (TIMER)

Offset

Register	Offset
TIMER	8h

Function

Represents a 16-bit free-running counter that the CPU can read and write. The timer starts from 0h after reset, counts linearly to FFFFh, and wraps around.

When CTRL2[TIMER_SRC] = 1, an external time tick continuously increments the timer. The time tick must be synchronous to the peripheral clock, with a minimum pulse width of one clock cycle.

When CTRL2[TIMER_SRC] = 0, the CAN bit clock increments the timer, which defines the baud rate on the CAN bus. During a message transmission or reception, it increments by one for each bit that is received or transmitted. When there is no message on the bus, it counts using the previously programmed baud rate. The timer is not incremented during Disable and Freeze modes.

The timer value is captured when the second bit of the identifier field of any frame is on the CAN bus. This captured value is written into the timestamp entry in a message buffer after a successful reception or transmission of a message.

If CTRL1[TSYN] = 1, the timer is reset whenever a message is received in the first available message buffer, according to CTRL2[RFFN].

The CPU can write to this register anytime. However, if the write occurs simultaneously with the timer being reset by a reception in the first message buffer, the write value is discarded.

Reading this register affects the message buffer unlocking procedure (see Message buffer lock mechanism).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TIMER															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-0 TIMER	Timer Value Contains the free-running counter value.

73.6.2.5 RX Message Buffers Global Mask (RXMGMASK)

Offset

Register	Offset
RXMGMASK	10h

Function

Masks the filter bits of all RX message buffers, excluding MB14 and MB15, which have individual mask registers.

This register is located in RAM.

RXMGMASK is provided for legacy application support.

- When MCR[IRMQ] is 0, RXMGMASK is always in effect. The bits in RXMGMASK[MG] mask the MB filter bits.
- When MCR[IRMQ] is 1, RXMGMASK has no effect. The bits in RXMGMASK[MG] do not mask the MB filter bits.

This register can only be written in Freeze mode; the module blocks it in other modes.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MG															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MG															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Fields

Field	Function
31-0 MG	Global Mask for RX Message Buffers Masks the message buffer filter bits. The alignment with the ID word of the message buffer is imperfect. The two most significant MG bits affect the fields RTR and IDE, which are located in the Control and Status word of the MB. The following table shows which MG bits mask each MB filter field.

SMB[RTR] [1]	CTRL2[RRS]	CTRL2[EACEN]	Message buffer filter fields			
			MB[RTR]	MB[IDE]	MB[ID]	Reserved
0	-	0	Note [2]	Note [3]	MG[28:0]	MG[31:29]
0	-	1	MG[31]	MG[30]	MG[28:0]	MG[29]
1	0	-	-	-	-	MG[31:0]
1	1	0	-	-	MG[28:0]	MG[31:29]
1	1	1	MG[31]	MG[30]	MG[28:0]	MG[29]

1. RTR bit of the incoming frame. It is saved into an auxiliary MB called RX serial message buffer (RX SMB).
2. If CTRL2[EACEN] is 0, the RTR bit of MB is never compared with the RTR bit of the incoming frame.
3. If CTRL2[EACEN] is 0, the IDE bit of MB is always compared with the IDE bit of the incoming frame.

0b - The corresponding bit in the filter is "don't care."

1b - The corresponding bit in the filter is checked.

73.6.2.6 Receive 14 Mask (RX14MASK)

Offset

Register	Offset
RX14MASK	14h

Function

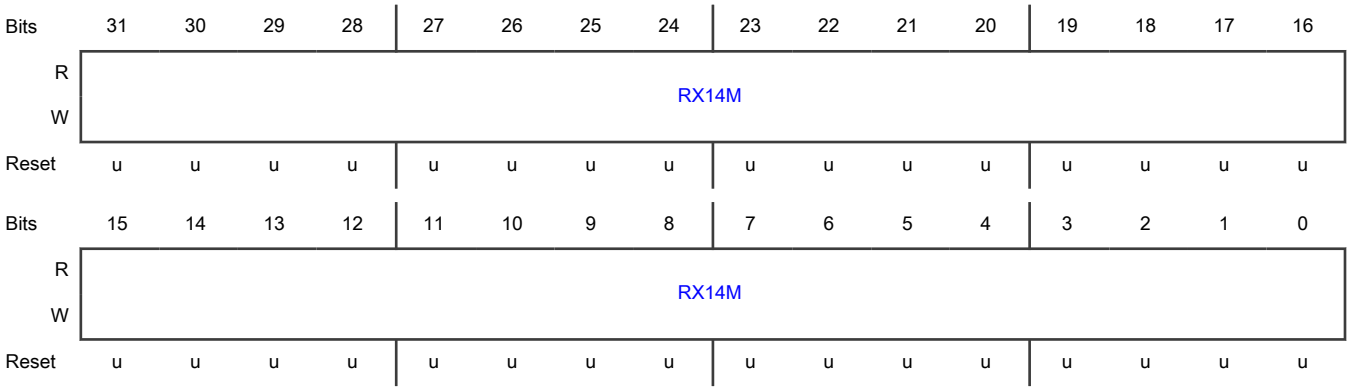
Masks the filter fields of MB14.

This register is located in RAM.

RX14MASK is provided for legacy application support. When MCR[IRMQ] = 1, RX14MASK has no effect.

This register can only be programmed when the module is in Freeze mode; the module blocks it in other modes.

Diagram



Fields

Field	Function
31-0 RX14M	RX Buffer 14 Mask Bits Masks the corresponding MB14 filter field in the same way that RX Message Buffers Global Mask (RXMGMASK) masks the filters of the other message buffers. 0b - The corresponding bit in the filter is "don't care." 1b - The corresponding bit in the filter is checked.

73.6.2.7 Receive 15 Mask (RX15MASK)

Offset

Register	Offset
RX15MASK	18h

Function

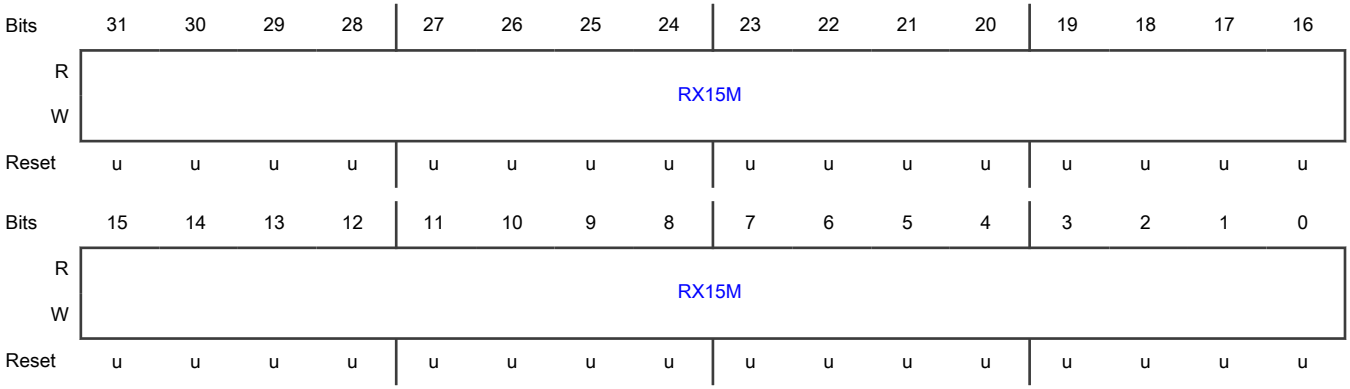
Masks the filter fields of MB15.

This register is located in RAM.

RX15MASK is provided for legacy application support. When MCR[IRMQ] = 1, RX15MASK has no effect.

This register can be programmed only when the module is in Freeze mode; the module blocks it in other modes.

Diagram



Fields

Field	Function
31-0 RX15M	RX Buffer 15 Mask Bits Masks the corresponding MB15 filter field in the same way that RX Message Buffers Global Mask (RXMGMASK) masks the filters of other message buffers. 0b - The corresponding bit in the filter is "don't care." 1b - The corresponding bit in the filter is checked.

73.6.2.8 Error Counter (ECR)

Offset

Register	Offset
ECR	1Ch

Function

Contains error counters for received and transmitted messages.

TXERRCNT and RXERRCNT consider all errors in both CAN FD and non-FD message formats. TXERRCNT_FAST and RXERRCNT_FAST count only the errors that occur in the data phase of CAN FD frames that have BRS = 1.

The Fault Confinement state (ESR1[FLTCONF]) is updated based on TXERRCNT and RXERRCNT counters only. The rules for increasing and decreasing these counters are described in the CAN protocol and are entirely implemented in FlexCAN.

The basic rules for FlexCAN bus state transitions are:

- If the value of TXERRCNT or RXERRCNT becomes greater than or equal to 128, ESR1[FLTCONF] is updated to reflect Error Passive state.
- If the state of FlexCAN is Error Passive, and TXERRCNT or RXERRCNT decrements to a value less than 128 when the other already satisfies this condition, ESR1[FLTCONF] is updated to reflect Error Active state.
- If the value of TXERRCNT becomes greater than 255, ESR1[FLTCONF] is updated to reflect Bus Off state, and an interrupt may be issued. The value of TXERRCNT is then reset to zero.
- If FlexCAN is in Bus Off, TXERRCNT is cascaded with another internal counter to count the occurrences of 11 consecutive recessive bits on the bus. TXERRCNT is reset to zero. It counts in a manner where the internal counter counts 11 such bits and then wraps around when incrementing the TXERRCNT. When TXERRCNT reaches the value of 128, ESR1[FLTCONF] is updated to Error Active, and both error counters are reset to zero. Upon any instance of a dominant bit following a stream of less than 11 consecutive recessive bits, the internal counter resets itself to zero without affecting the TXERRCNT value. The TXERRCNT_FAST counter is frozen during Bus Off.
- If only one node is operating during system startup, its TXERRCNT increases upon each attempted message transmission, as a result of acknowledge errors (indicated by ESR1[ACKERR]). After the transition to Error Passive state, TXERRCNT no longer increments upon acknowledge errors. The chip never goes into the Bus Off state.
- If RXERRCNT increases to a value greater than 127, it is not incremented further, even if more errors are detected when being a receiver. At the next successful message reception, the counter is set to a value between 119 and 127 to return to the Error Active state.
- TXERRCNT_FAST and RXERRCNT_FAST error counter values increment and decrement based on errors detected only in the data phase of CAN FD frames that have BRS = 1. These counters follow the same increment and decrement rules as TXERRCNT and RXERRCNT. These counters do not wrap around and get stuck at their maximum value (255). They stop counting and keep their values frozen when FlexCAN is in the Bus Off state. They are reset when FlexCAN leaves the Bus Off state and resume counting after FlexCAN returns to the Error Active state.

NOTE

See Fault confinement in the CAN Protocol standard ISO 11898-1:2015 for details.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RXERRCNT_FAST								TXERRCNT_FAST							
W	0								0							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXERRCNT								TXERRCNT							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 RXERRCNT_FAST	Receive Error Counter for Fast Bits Counts errors detected in the data phase of received CAN FD messages that have BRS = 1. This field is read-only except in Freeze mode, when the CPU can write an 8-bit zero value only.

Table continues on the next page...

Table continued from the previous page...

Field	Function
23-16 TXERRCNT_FAST	Transmit Error Counter for Fast Bits Counts errors detected in the data phase of transmitted CAN FD messages that have BRS = 1. This field is read-only except in Freeze mode, when the CPU can write an 8-bit zero value only.
15-8 RXERRCNT	Receive Error Counter Counts all errors detected in received messages. This field is read-only except in Freeze mode, when the CPU can write to it.
7-0 TXERRCNT	Transmit Error Counter Counts all errors detected in transmitted messages. This field is read-only except in Freeze mode, when the CPU can write to it.

73.6.2.9 Error and Status 1 (ESR1)

Offset

Register	Offset
ESR1	20h

Function

Reports various error conditions detected in the reception and transmission of a CAN frame. This register provides status information about the chip, and is the source of some interrupts to the CPU. The reported error conditions are:

NOTE

Reading host can clear these fields.

- Errors detected in CAN frames of any format:
  - BIT1ERR
  - BIT0ERR
  - ACKERR
  - CRCERR
  - FRMERR
  - STFERR
- Errors detected in the data phase of CAN FD frames with the BRS bit set only:
  - BIT1ERR_FAST
  - BIT0ERR_FAST
  - CRCERR_FAST
  - FRMERR_FAST
  - STFERR_FAST

One or more error flags may report an error detected in a single CAN frame. To account for more error events occurring in subsequent frames when the CPU does not attempt to read this register, error reporting is cumulative.

Status flags:

- TXWRN
- RXWRN
- IDLE
- TX
- FLTCONF
- RX
- SYNCH

Interrupt flags:

- BOFFINT
- BOFFDONEINT
- ERRINT
- ERRINT_FAST
- TWRNINT
- RWRNINT

The CPU should follow this procedure when servicing interrupt requests generated by these flags:

1. Read this register to capture all error condition and status flags. This action clears the respective flags that were set since the last read access.
2. Write 1 to clear the interrupt flag that triggered the interrupt request.
3. Write 1 to clear the ERROVR flag, if it is set.

Starting from all error flags cleared, a first error event sets either ERRINT or ERRINT_FAST (provided the corresponding mask bit is 1). If other error events in subsequent frames occur before the CPU serves the interrupt request, the ERROVR flag is set to indicate that errors from different frames have accumulated.

Table 518. CAN bus status

SYNCH	IDLE	TX	RX	FlexCAN state
0	0	0	0	Not synchronized to CAN bus
1	1	X	X	Idle
1	0	1	0	Transmitting
1	0	0	1	Receiving

NOTE

See Fault confinement in the CAN Protocol standard ISO 11898-1:2015 for details.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BIT1ERR...	BIT0ERR...	0	CRCERR_...	FRMERR_...	STFERR_...	0		0	0	ERROVR	ERRINT_...	BOFFDON...	SYNCH	TWRNINT	RWRNINT
W											W1C	W1C	W1C		W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BIT1ERR	BIT0ERR	ACKERR	CRCERR	FRMERR	STFERR	TXWRN	RXWRN	IDLE	TX	FLTCONF		RX	BOFFINT	ERRINT	0
W														W1C	W1C	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 BIT1ERR_FAST	Fast Bit1 Error Flag Indicates when an inconsistency occurs between the transmitted and the received bit in the data phase of CAN FD frames that have BRS = 1. After a read operation, the field's value clears to 0. 0b - No such occurrence. 1b - At least one bit transmitted as recessive is received as dominant.
30 BIT0ERR_FAST	Fast Bit0 Error Flag Indicates when an inconsistency occurs between the transmitted and the received bit in the data phase of CAN FD frames that have BRS = 1. After a read operation, the field's value clears to 0. 0b - No such occurrence. 1b - At least one bit transmitted as dominant is received as recessive.
29 —	Reserved
28 CRCERR_FAST	Fast Cyclic Redundancy Check Error Flag Indicates that the receiver node has detected a CRC error in the CRC field of CAN FD frames that have BRS = 1. This error means that the calculated CRC is different from the received CRC. After a read operation, the field's value clears to 0. 0b - No such occurrence. 1b - A CRC error occurred since last read of this register.
27	Fast Form Error Flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
FRMERR_FAST	Indicates whether the receiver node has detected a form error in the data phase of CAN FD frames that have BRS = 1. This error means that a fixed-form bit field contains at least one illegal bit. After a read operation, the field's value clears to 0. 0b - No such occurrence. 1b - A form error occurred since last read of this register.
26 STFERR_FAST	Fast Stuffing Error Flag Indicates that a stuffing error has been detected in the data phase of CAN FD frames that have BRS = 1. After a read operation, the field's value clears to 0. 0b - No such occurrence. 1b - A stuffing error occurred since last read of this register.
25-24 —	Reserved
23 —	Reserved
22 —	Reserved
21 ERROVR	Error Overrun Flag Indicates that an error condition occurred when any error flag is already set. 0b - No overrun 1b - Overrun
20 ERRINT_FAST	Fast Error Interrupt Flag Indicates that at least one error flag detected in the data phase of CAN FD frames that have BRS = 1 (BIT1ERR_FAST, BIT0ERR_FAST, CRCERR_FAST, FRMERR_FAST, or STFERR_FAST) is set. If CTRL2[ERRMSK_FAST] = 1, an interrupt is generated to the CPU. 0b - No such occurrence. 1b - Error flag set in the data phase of CAN FD frames that have BRS = 1.
19 BOFFDONEINT	Bus Off Done Interrupt Flag Indicates whether ECR[TXERRCNT] has finished counting 128 occurrences of 11 consecutive recessive bits on the CAN bus and is ready to leave Bus Off. If CTRL2[BOFFDONEMSK] = 1, an interrupt is generated to the CPU. 0b - No such occurrence 1b - FlexCAN module has completed Bus Off process.
18	CAN Synchronization Status Flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
SYNCH	Indicates whether FlexCAN is synchronized to the CAN bus and able to participate in the communication process. FlexCAN sets and clears this flag. See the table in Error and Status 1 (ESR1). 0b - Not synchronized 1b - Synchronized
17 TWRNINT	TX Warning Interrupt Flag Indicates whether TX error counter changed from less than 96 to greater than or equal to 96. If MCR[WRNEN] = 1, this flag is set when the TXWRN flag transitions from 0 to 1, meaning that the TX error counters reached 96. If CTRL1[TWRNMSK] = 1, an interrupt is sent to the CPU. When MCR[WRNEN] = 0, this flag is masked. The CPU must clear this flag before writing 0 to MCR[WRNEN]. Otherwise, this flag is set when MCR[WRNEN] = 1 again. Writing 0 has no effect. This flag is not generated when in the Bus Off state. This flag is not updated during Freeze mode. 0b - No such occurrence 1b - TX error counter changed from less than 96 to greater than or equal to 96.
16 RWRNINT	RX Warning Interrupt Flag Indicates whether the RX error counter changed from less than 96 to greater than or equal to 96. If MCR[WRNEN] = 1, this flag is set when the RXWRN flag transitions from 0 to 1, meaning that the RX error counters reached 96. If CTRL1[RWRNMSK] = 1, an interrupt is sent to the CPU. When MCR[WRNEN] = 0, this flag is masked. The CPU must clear this flag before writing 0 to MCR[WRNEN]. Otherwise, this flag is set when MCR[WRNEN] = 1 again. Writing 0 has no effect. This flag is not updated during Freeze mode. 0b - No such occurrence 1b - RX error counter changed from less than 96 to greater than or equal to 96.
15 BIT1ERR	Bit1 Error Flag Indicates when an inconsistency occurs between the transmitted and the received bit in a non-CAN FD message or in the arbitration or data phase of a CAN FD message. **NOTE** A transmitter does not set this flag for an arbitration field or ACK slot. It is not set for a node sending an error passive flag that detects dominant bits. After a read operation, the field's value clears to 0. 0b - No such occurrence. 1b - At least one bit sent as recessive is received as dominant.
14 BIT0ERR	Bit0 Error Flag Indicates when an inconsistency occurs between the transmitted and the received bit in a non-CAN FD message or in the arbitration or data phase of a CAN FD message. After a read operation, the field's value clears to 0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No such occurrence. 1b - At least one bit sent as dominant is received as recessive.
13 ACKERR	Acknowledge Error Flag Indicates whether the transmitter node has detected an acknowledge error. This error means that a dominant bit has not been detected during the ACK SLOT. After a read operation, the field's value clears to 0. 0b - No error 1b - Error occurred since last read of this register.
12 CRCERR	Cyclic Redundancy Check Error Flag Indicates whether the receiver node has detected a cyclic redundancy check (CRC) error either in a non-FD message or in the arbitration or data phase of a frame in CAN FD format. This error means that the calculated CRC is different from the received. After a read operation, the field's value clears to 0. 0b - No error 1b - Error occurred since last read of this register.
11 FRMERR	Form Error Flag Indicates whether a form error has been detected in a non-FD message or in the arbitration or data phase of an FD message by the receiver node. This error means that a fixed-form field contains at least one illegal bit. After a read operation, the field's value clears to 0. 0b - No error 1b - Error occurred since last read of this register.
10 STFERR	Stuffing Error Flag Indicates that a stuffing error has been detected in a non-FD message or in the arbitration or data phase of an FD message by the receiver node. After a read operation, the field's value clears to 0. 0b - No error 1b - Error occurred since last read of this register.
9 TXWRN	TX Error Warning Flag Indicates when repetitive errors occur during message transmission. Only the value of ECR[TXERRCNT] affects this flag. This flag is not updated during Freeze mode. After a read operation, the field's value clears to 0. 0b - No such occurrence. 1b - TXERRCNT is 96 or greater.

Table continues on the next page...

Table continued from the previous page...

Field	Function
8 RXWRN	RX Error Warning Flag Indicates when repetitive errors occur during message reception. Only the value of ECR[RXERRCNT] affects this flag. This flag is not updated during Freeze mode. After a read operation, the field's value clears to 0. 0b - No such occurrence. 1b - RXERRCNT is greater than or equal to 96.
7 IDLE	Idle Indicates whether CAN bus is in IDLE state. See Table 518. 0b - Not IDLE 1b - IDLE
6 TX	FlexCAN In Transmission Indicates whether FlexCAN is transmitting a message. See Table 518. 0b - Not transmitting 1b - Transmitting
5-4 FLTCONF	Fault Confinement State Indicates the confinement state of FlexCAN. If CTRL1[LOM] = 1, after a delay that depends on the CAN bit timing, this field indicates Error Passive. The same delay affects the way that this field reflects an update to ECR register by the CPU. It may be necessary to wait up to one CAN bit time for coherence to be restored. Soft reset affects this field, but if CTRL1[LOM] = 1, its reset value lasts for only one CAN bit. After that time, this field reports Error Passive. 00b - Error Active 01b - Error Passive 1xb - Bus Off
3 RX	FlexCAN in Reception Flag Indicates whether FlexCAN is receiving a message. See the table in Error and Status 1 (ESR1). 0b - Not receiving 1b - Receiving
2 BOFFINT	Bus Off Interrupt Flag Indicates whether FlexCAN has entered Bus Off state. If CTRL1[BOFFMSK]) = 1, an interrupt is generated to the CPU. Writing 0 to this field has no effect. 0b - No such occurrence. 1b - FlexCAN module entered Bus Off state.

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 ERRINT	Error Interrupt Flag Indicates that at least one of the error flags (ESR1[BIT1ERR], ESR1[BIT0ERR], ESR1[ACKERR], ESR1[CRCERR], ESR1[FRMERR], or ESR1[STFERR]) is set. If the corresponding mask CTRL1[ERRMSK] = 1, an interrupt is generated to the CPU. Writing 0 to this field has no effect. 0b - No such occurrence. 1b - Indicates setting of any error flag in the Error and Status register.
0 —	Reserved

73.6.2.10 Interrupt Masks 2 (IMASK2)

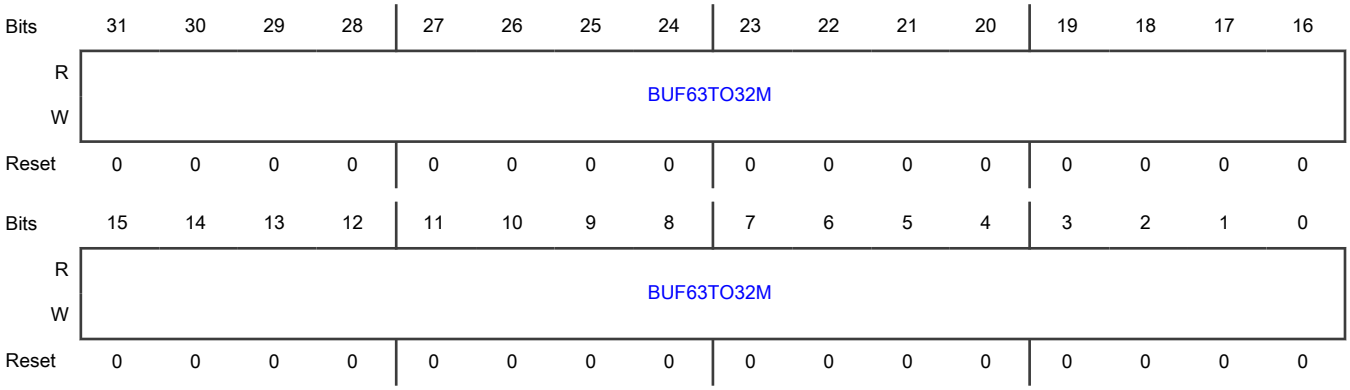
Offset

Register	Offset
IMASK2	24h

Function

Masks interrupt flags. This register allows any of the 32 message buffer interrupts to be enabled or disabled for MB63–MB32. It contains one interrupt mask bit per buffer. This configuration allows the CPU to determine which buffer generates an interrupt after a successful transmission or reception when the corresponding Interrupt Flags 2 (IFLAG2) flag is set.

Diagram



Fields

Field	Function
31-0	Buffer MBi Mask Masks the corresponding FlexCAN message buffer interrupt for MB63–MB32.

Table continues on the next page...

Field	Function
BUF63TO32M	**NOTE** If the corresponding Interrupt Flags 2 (IFLAG2) flag is set, writing 1 or 0 to a field in IMASK2 can set or clear an interrupt request. 0b - The corresponding buffer interrupt is disabled. 1b - The corresponding buffer interrupt is enabled.

73.6.2.11 Interrupt Masks 1 (IMASK1)

Offset

Register	Offset
IMASK1	28h

Function

Masks interrupt flags. This register allows any of the 32 message buffer interrupts to be enabled or disabled for MB31–MB0. It contains one interrupt mask bit per buffer. This configuration allows the CPU to determine which buffer generates an interrupt after a successful transmission or reception when the corresponding Interrupt Flags 1 (IFLAG1) flag is set.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BUF31TO0M															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BUF31TO0M															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 BUF31TO0M	Buffer MBi Mask Enables or disables the corresponding FlexCAN message buffer interrupt for MB31–MB0. **NOTE** If the corresponding Interrupt Flags 1 (IFLAG1) flag is set, writing 1 or 0 to a field in IMASK1 can set or clear an interrupt request.

Table continues on the next page...

Field	Function
	0b - The corresponding buffer interrupt is disabled. 1b - The corresponding buffer interrupt is enabled.

73.6.2.12 Interrupt Flags 2 (IFLAG2)

Offset

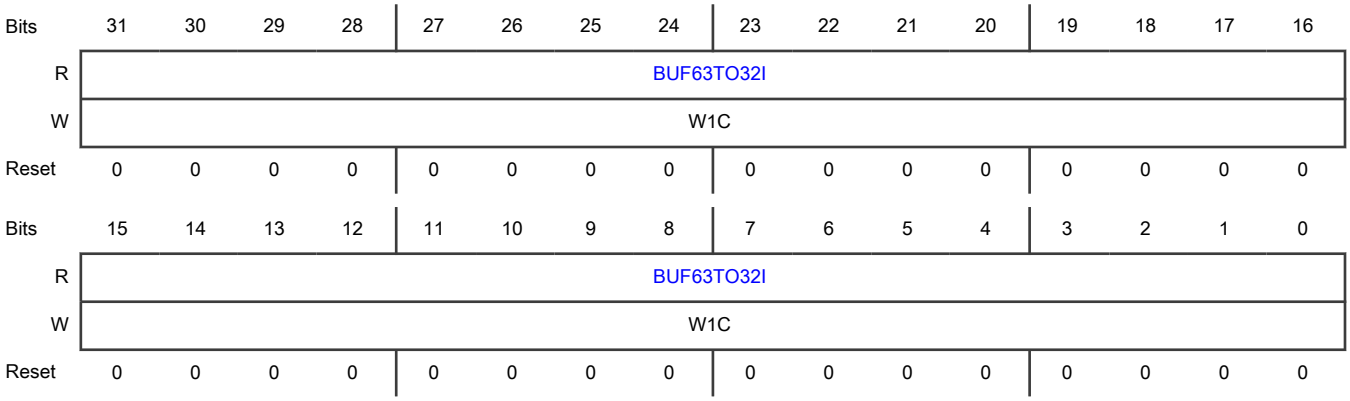
Register	Offset
IFLAG2	2Ch

Function

Contains the flags for the 32 message buffer interrupts for MB63–MB32. It contains one interrupt flag bit per buffer. Each successful transmission or reception sets the respective flag in this register. If the corresponding Interrupt Masks 2 (IMASK2) bit is set, an interrupt is generated.

Before updating MCR[MAXMB], the CPU must service the IFLAG2 flags whose MB value is greater than the MAXMB to be updated. Otherwise, those flags remain set and are inconsistent with the number of message buffers available.

Diagram



Fields

Field	Function
31-0 BUF63TO32I	Buffer MBi Interrupt Flags the corresponding FlexCAN message buffer interrupt for MB63–MB32. 0b - The corresponding buffer has no occurrence of successfully completed transmission or reception. 1b - The corresponding buffer has successfully completed transmission or reception.

73.6.2.13 Interrupt Flags 1 (IFLAG1)

Offset

Register	Offset
IFLAG1	30h

Function

Contains the flags for the 32 message buffer interrupts for MB31–MB0. It contains one interrupt flag bit per buffer. Each successful transmission or reception sets the corresponding IFLAG1 bit. If the corresponding Interrupt Masks 1 (IMASK1) bit is set, an interrupt is generated. There is an exception when DMA for Legacy RX FIFO is enabled, as described below.

The BUF7I–BUF5I flags also represent Legacy FIFO interrupts when the Legacy RX FIFO is enabled. When MCR[RFEN] is 1 and MCR[DMA] is 0, the function of the eight least significant interrupt flags changes:

- BUF7I, BUF6I, and BUF5I indicate operating conditions of the Legacy FIFO.
- BUF4I–BUF1I fields are reserved.
- BUF0I empties the Legacy FIFO.

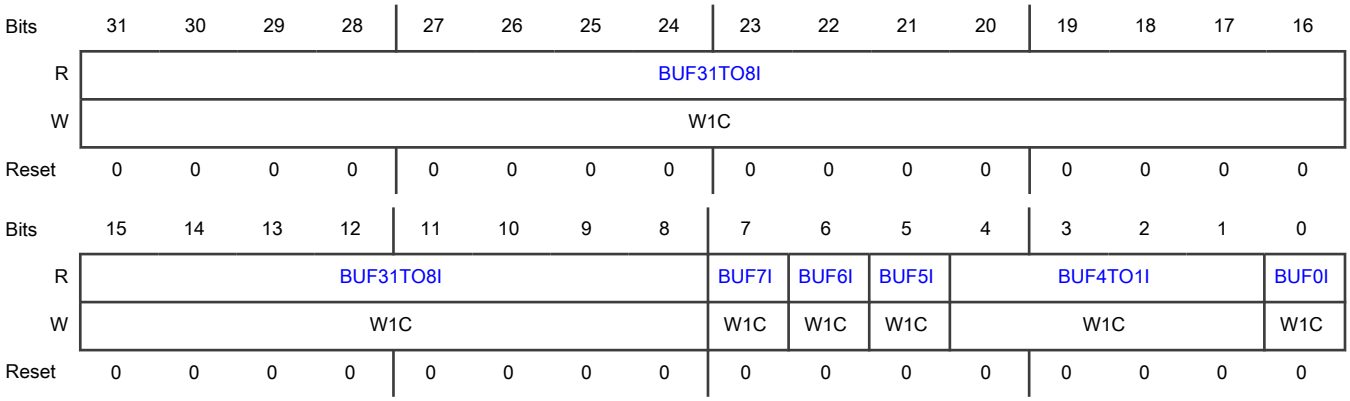
Before writing 1 to MCR[RFEN], the CPU must service the IFLAG flags set in the Legacy RX FIFO region; see Legacy RX FIFO. Otherwise, these IFLAG flags mistakenly show the related message buffers now belonging to Legacy FIFO as having contents to be serviced. When MCR[RFEN] is 0, the Legacy FIFO flags must be cleared. The same care must be taken when a CTRL2[RFFN] value is selected, extending Legacy RX FIFO filters beyond MB7. For example, when RFFN is 8h, Legacy RX FIFO filters occupy the MB23–MB0 range, and related IFLAG flags must be cleared.

When MCR[RFEN] and MCR[DMA] are 1 (DMA feature for Legacy RX FIFO enabled), the function of the eight least significant interrupt flags (BUF7I–BUF0I) changes to support DMA operation. BUF7I, BUF6I, and BUF4I–BUF1I are not used.

BUF5I indicates the operating condition of the Legacy FIFO, and BUF0I empties the Legacy FIFO. Moreover, BUF5I does not generate a CPU interrupt, but it does generate a DMA request. IMASK1 bits in the Legacy RX FIFO region are not considered when bit MCR[DMA] = 1. In addition, the CPU must not clear the BUF5I flag when DMA is enabled. Before writing 1 to MCR[DMA], the CPU must service the IFLAG flags set in the Legacy RX FIFO region. When MCR[DMA] is 0, the Legacy FIFO must be empty. Legacy FIFO must be disabled when MCR[FDEN] = 1.

Before updating MCR[MAXMB], the CPU must service the IFLAG1 flags whose MB value is greater than the MCR[MAXMB] to be updated. Otherwise, those flags remain set and are inconsistent with the number of message buffers available.

Diagram



Fields

Field	Function
31-8 BUF31TO8I	Buffer MBi Interrupt Flags the corresponding FlexCAN message buffer interrupt for MB31–MB8. 0b - The corresponding buffer has no occurrence of successfully completed transmission or reception. 1b - The corresponding buffer has successfully completed transmission or reception.
7 BUF7I	Buffer MB7 Interrupt or Legacy RX FIFO Overflow Flags the interrupt for MB7 when MCR[RFEN] = 0 (Legacy RX FIFO disabled). **NOTE** When a CPU write changes the value of MCR[RFEN], FlexCAN clears this flag. When MCR[RFEN] = 1, this flag represents a Legacy RX FIFO overflow. In this case, the flag indicates that a message was lost because the Legacy RX FIFO is full. When the Legacy RX FIFO is full and a message buffer captures the message, this flag is not set. 0b - No occurrence of MB7 completing transmission or reception, or no FIFO overflow. 1b - MB7 completed transmission or reception, or FIFO overflow.
6 BUF6I	Buffer MB6 Interrupt or Legacy RX FIFO Warning Flags the interrupt for MB6 when MCR[RFEN] = 0 (Legacy RX FIFO disabled). **NOTE** When a CPU write changes the value of MCR[RFEN], FlexCAN clears this flag. When MCR[RFEN] = 1, this flag represents a Legacy RX FIFO warning. In this case, the flag indicates when the number of unread messages within the Legacy RX FIFO is increased to five from four due to the reception of a new message. In other words, the Legacy RX FIFO is almost full. If this flag is cleared when there are more than four unread messages, it does not set again until the number of unread messages in the Legacy RX FIFO decreases to four or fewer. 0b - No occurrence of MB6 completing transmission or reception, or FIFO not almost full. 1b - MB6 completed transmission or reception, or FIFO almost full.
5 BUF5I	Buffer MB5 Interrupt or Frames available in Legacy RX FIFO Flags the interrupt for MB5 when MCR[RFEN] = 0 (Legacy RX FIFO disabled). **NOTE** When a CPU write changes the value of MCR[RFEN], FlexCAN clears this flag. When MCR[RFEN] = 1, the BUF5I flag represents frames available in Legacy RX FIFO. In this case, the flag indicates that at least one frame is available to be read from the Legacy RX FIFO. When MCR[DMA] = 1, this flag generates a DMA request. The CPU must not clear this field by writing 1 to BUF5I. 0b - No occurrence of completed transmission or reception, or no frames available

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - MB5 completed transmission or reception, or frames available
4-1 BUF4TO1I	Buffer MBi Interrupt or Reserved Flags the interrupts for MB4–MB1 when MCR[RFEN] = 0 (Legacy RX FIFO disabled). **NOTE** When a CPU write changes the value of MCR[RFEN], FlexCAN clears these flags. When MCR[RFEN] = 1, the BUF4TO1I flags are reserved. 0b - The corresponding buffer has no occurrence of successfully completed transmission or reception. 1b - The corresponding buffer has successfully completed transmission or reception.
0 BUF0I	Buffer MB0 Interrupt or Clear Legacy FIFO bit Flags the interrupt for MB0 when MCR[RFEN] = 0 (Legacy RX FIFO disabled). If MCR[RFEN] = 1, this field is used to trigger the clear Legacy FIFO operation. This operation empties the Legacy FIFO contents. Before performing this operation, the CPU must service all Legacy FIFO-related IFLAG flags. When MCR[DMA] = 1, this operation also clears the BUF5I flag, aborting the DMA request. The clear Legacy FIFO operation occurs when the CPU writes 1 to BUF0I. This operation is only allowed in Freeze mode; the module blocks it in other conditions. 0b - MB0 has no occurrence of successfully completed transmission or reception. 1b - MB0 has successfully completed transmission or reception.

73.6.2.14 Control 2 (CTRL2)

Offset

Register	Offset
CTRL2	34h

Function

Complements Control 1 (CTRL1), providing control bits for memory write-access in Freeze mode. This register extends Legacy FIFO filter quantity, and adjusts the operation of internal FlexCAN processes such as matching and arbitration.

Soft reset does not affect the contents of this register.

Table 519 shows how the value of CTRL2[RFFN] determines the Legacy RX FIFO filter structure.

Table 519. Possible Legacy RX FIFO filter structures

RFFN[3:0]	Number of Legacy RX FIFO filter elements	Message buffers occupied by Legacy RX FIFO and ID filter table	Remaining available message buffers	Legacy RX FIFO ID filter table elements affected by RX individual masks	Legacy RX FIFO ID filter table elements affected by Legacy RX FIFO global mask
0h	8	MB 0–7	MB 8–95	Elements 0–7	None
1h	16	MB 0–9	MB 10–95	Elements 0–9	Elements 10–15
2h	24	MB 0–11	MB 12–95	Elements 0–11	Elements 12–23
3h	32	MB 0–13	MB 14–95	Elements 0–13	Elements 14–31
4h	40	MB 0–15	MB 16–95	Elements 0–15	Elements 16–39
5h	48	MB 0–17	MB 18–95	Elements 0–17	Elements 18–47
6h	56	MB 0–19	MB 20–95	Elements 0–19	Elements 20–55
7h	64	MB 0–21	MB 22–95	Elements 0–21	Elements 22–63
8h	72	MB 0–23	MB 24–95	Elements 0–23	Elements 24–71
9h	80	MB 0–25	MB 26–95	Elements 0–25	Elements 26–79
Ah	88	MB 0–27	MB 28–95	Elements 0–27	Elements 28–87
Bh	96	MB 0–29	MB 30–95	Elements 0–29	Elements 30–95
Ch	104	MB 0–31	MB 32–95	Elements 0–31	Elements 32–103
Dh	112	MB 0–33	MB 34–95	Elements 0–31	Elements 32–111
Eh	120	MB 0–35	MB 36–95	Elements 0–31	Elements 32–119
Fh	128	MB 0–37	MB 38–95	Elements 0–31	Elements 32–127

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ERRMSK_...	BOFFDON...	ECRWRE	WRMFRZ	RFFN				TASD					MRP	RRS	EACEN
W																
Reset	See Register reset values.															

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TIMER_S...	PREXCEN	BTE	ISOCANF...	EDFLTDIS	0	MBTSBASE		TSTAMPCAP		0	0			0	
W																
Reset	See Register reset values.															

Register reset values

Register	Reset value
CTRL2	CAN_0–CAN_2: 0060_0000h CAN_3–CAN_11: 0080_0000h

Fields

Field	Function
31 ERRMSK_FAST	Error Interrupt Mask for Errors Detected in the Data Phase of Fast CAN FD Frames Enables the ESR1[ERRINT_FAST] interrupt. 0b - Disable 1b - Enable
30 BOFFDONEMSK	Bus Off Done Interrupt Mask Enables the Bus Off Done interrupt, ESR1[BOFFDONEINT]. 0b - Disable 1b - Enable
29 ECRWRE	Error Correction Configuration Register Write Enable Enables updates for Memory Error Control (MECR). If the protocol described in Detection and correction of memory errors is not followed, this field is automatically 0. 0b - Disable 1b - Enable
28 WRMFRZ	Write Access to Memory in Freeze Mode Enables unrestricted write access to FlexCAN memory in Freeze mode. When this field is 0, write access restrictions are maintained. This field can only be written in Freeze mode, and has no effect out of Freeze mode. **NOTE** Do not write 1 to MCR[RFEN] during FlexCAN memory initialization. 0b - Disable 1b - Enable
27-24 RFFN	Number of Legacy Receive FIFO Filters Defines the number of Receive Legacy FIFO filters, as shown in Table 519. The chip determines the maximum selectable number of filters. Do not program this field with values that cause the number of message buffers occupied by Legacy RX FIFO and Legacy RX FIFO ID Filter to exceed MCR[MAXMB]. MCR[MAXMB] defines the number of message buffers present. This field can only be written in Freeze mode; the module blocks it in other modes. Each group of eight filters occupies a memory space equivalent to two message buffers. The more filters are implemented, the fewer message buffers are available. The Legacy RX FIFO occupies the memory space originally reserved for MB5–MB0. This field should be programmed with a value corresponding to a number of filters less than the number of available memory words. The number of available words can be calculated as follows: (SETUP_MB - 6) × 4 Where SETUP_MB is the smaller of the parameter NUMBER_OF_MB and MCR[MAXMB]. The number of remaining message buffers available is:

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>(SETUP_MB - 8) - (RFFN × 2)
If the number of Legacy RX FIFO filters programmed through RFFN exceeds the SETUP_MB value (memory space available), the exceeding ones are not functional.
NOTE
• The number of the last remaining available message buffers is the smaller of (NUMBER_OF_MB - 1) and MCR[MAXMB].
• If RX Individual Mask registers are not enabled, the Legacy RX FIFO Global Mask affects all Legacy RX FIFO filters.</td></tr>
<tr><td>23-19
TASD</td><td>Transmission Arbitration Start Delay
Indicates by how many CAN bits the transmission arbitration process start point can be delayed from the first bit of CRC field on CAN bus. See TX arbitration start delay for details. This field can be written only in Freeze mode; the module blocks it in other modes.</td></tr>
<tr><td>18
MRP</td><td>Message Buffers Reception Priority
Sets the priority for the matching process.
This field can be written only in Freeze mode; the module blocks it in other modes.
NOTE
The descriptions of the field settings vary by module instance.
<table><tr><th>Instance</th><th>Field value and description</th></tr><tr><td>CAN_0
CAN_1
CAN_2</td><td>0b - Matching starts from Legacy RX FIFO or Enhanced RX FIFO and continues on message buffers.
1b - Matching starts from message buffers and continues on Legacy RX FIFO or Enhanced RX FIFO.</td></tr><tr><td>CAN_3
CAN_4
CAN_5
CAN_6
CAN_7
CAN_8
CAN_9
CAN_10
CAN_11</td><td>0b - Matching starts from RX FIFO and continues on message buffers.
1b - Matching starts from message buffers and continues on RX FIFO.</td></tr></table></td></tr>
<tr><td>17
RRS</td><td>Remote Request Storing
Determines what the module does with a remote request. The remote request frame is submitted to a matching process.
If this field is 1, the frame is stored in the corresponding message buffer in the same fashion as a data frame. No automatic remote response frame is generated.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	If this field is 0, an automatic remote response frame is generated if a message buffer with CODE = 1010b is found with the same ID. You can only write to this field in Freeze mode. The module blocks it in other modes. 0b - Generated 1b - Stored
16 EACEN	Entire Frame Arbitration Field Comparison Enable for RX Message Buffers Controls the comparison of IDE and RTR fields within RX message buffer filters with their corresponding bits in the incoming frame by the matching process. If enabled, the IDE and RTR fields of the RX message buffer are compared to their corresponding bits within the incoming frame (mask bits apply). If disabled, the IDE field of the RX message buffer filter is always compared and RTR is never compared despite mask bits. This field does not affect matching for Legacy RX FIFO or Enhanced RX FIFO. You can only write to this field in Freeze mode; the module blocks it in other modes. 0b - Disable 1b - Enable
15 TIMER_SRC	Timer Source Selects the time tick source used for incrementing the free-running timer counter. If CAN bit clock is selected, it defines the baud rate on the CAN bus. If External time tick is selected, the period can be adjusted to match the baud rate on the CAN bus. It can also be adjusted to a different value as required. See the chip-specific section for details about the external time tick. You can only write to this field in Freeze mode. 0b - CAN bit clock 1b - External time tick
14 PREXCEN	Protocol Exception Enable Enables the protocol exception feature. You can only write to this field in Freeze mode. **NOTE** See Protocol exception event in the CAN Protocol standard ISO 11898-1:2015 for details. 0b - Disabled 1b - Enabled
13 BTE	Bit Timing Expansion Enable Enables the use of Enhanced CAN Bit Timing Prescalers (EPRS), Enhanced Data Phase CAN Bit Timing (EDCBT), and Enhanced Nominal CAN Bit Timing (ENCBT) to configure the CAN bit timing segments, instead of using the bit timing fields of CAN Bit Timing (CBT), CAN FD Bit Timing (FDCBT), and Control 1 (CTRL1).

Table continues on the next page...

Table continued from the previous page...

Field	Function
	If this field is 1: • CTRL1[PRESDIV], CTRL1[PROPSEG], CTRL1[PSEG1], CTRL1[PSEG2], and CTRL1[RJW] are read as zero. A write operation to these fields has no effect. • CBT[EPRESDIV], CBT[ERJW], CBT[EPROPSEG], CBT[EPSEG1], and CBT[EPSEG2], and the corresponding fields in CAN FD Bit Timing (FDCBT), are read as zero. A write operation to these fields has no effect. • ETDC[ETDCOFF], ETDC[ETDCEN], ETDC[ETDCFAIL], and ETDC[ETDCVAL] are used by FlexCAN instead of FDCTRL[TDCOFF], FDCTRL[TDCEN], FDCTRL[TDCFAIL], and FDCTRL[TDCVAL]. These fields are read as zero, and a write operation to them has no effect. • ETDC[TDMDIS] can be used to disable transceiver delay measurement. 0b - Disable 1b - Enable
12 ISOCANFDEN	ISO CAN FD Enable Enables the CAN FD protocol according to ISO specification ISO 11898-1:2015 (see CAN FD ISO compliance). When disabled, FlexCAN operates using the non-ISO CAN FD protocol. You can only write to this field in Freeze mode. **NOTE** FlexCAN is able to transmit FD frame format according to CAN Protocol standard ISO 11898-1:2015. 0b - Disable 1b - Enable
11 EDFLTDIS	Edge Filter Disable Disables the edge filter used during the Bus Integration state. When the Edge Filter is enabled, two consecutive nominal time quanta with dominant bus states are required to detect an edge that causes synchronization. When synchronization occurs, the counting of the sequence of 11 consecutive recessive bits is restarted. The edge filter prevents dominant pulses that are shorter than a nominal bit time (present during the data phase of an FD frame) from being mistaken for an idle condition. You can only write to this field in Freeze mode. **NOTE** See Bus Integration state in the CAN Protocol standard ISO 11898-1:2015 for details. 0b - Enabled 1b - Disabled
10 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
9-8 MBTSBASE	Message Buffer Timestamp Base Selects the timebase used for capturing the 16-bit TIME_STAMP field of the message buffer register. This field can be written in Freeze Mode only. 00b - TIMER 01b - Lower 16 bits of high-resolution timer 10b - Upper 16 bits of high-resolution timer 11b - Reserved
7-6 TSTAMPCAP	Timestamp Capture Point Configures the point in time when a 32-bit timebase is captured during a CAN frame. This base is stored in the high-resolution timestamp register (HR_TIME_STAMP). For classical CAN frames, capture points can be the start-of-frame bit or the point when CAN frame is considered valid. This point is the seventh bit of the end-of-frame for transmission and the sixth bit of the end-of-frame for reception. For CAN FD frames, the high-resolution timestamp can be captured at the start of frame, when a CAN FD frame is considered valid, or the res bit. You can only write to this field in Freeze mode. 00b - Disabled 01b - End of the CAN frame 10b - Start of the CAN frame 11b - Start of frame for classical CAN frames; res bit for CAN FD frames
5 —	Reserved
4-2 —	Reserved
1-0 —	Reserved

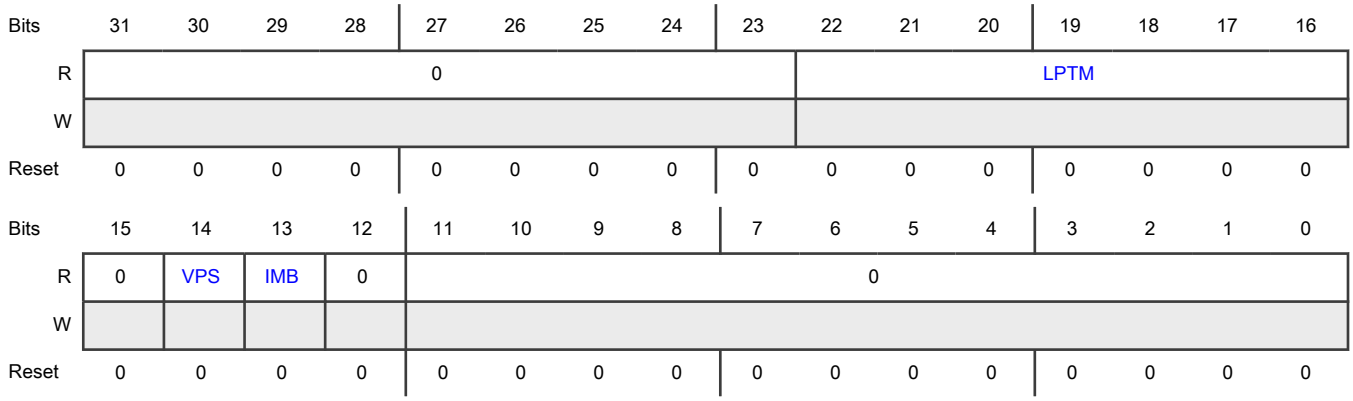
73.6.2.15 Error and Status 2 (ESR2)

Offset

Register	Offset
ESR2	38h

Function
Reports general status information.

Diagram



Fields

Field	Function
31-23 —	Reserved
22-16 LPTM	Lowest Priority TX Message Buffer Indicates the lowest number inactive message buffer when ESR2[VPS] = 1 (see ESR2[IMB]). If no message buffer is inactive, the message buffer indicated depends on the value of CTRL1[LBUF]. If CTRL1[LBUF] = 0, the message buffer indicated is the one with the greatest arbitration value (see Highest-priority message buffer first). If CTRL1[LBUF] = 1, the message buffer indicated is the active TX message buffer with the highest number. If a TX message buffer is being transmitted, it is not considered in the LPTM calculation. If ESR2[IMB] is not 0 and a frame is transmitted successfully, the value of LPTM is updated with its message buffer number.
15 —	Reserved
14 VPS	Valid Priority Status Indicates whether the contents of ESR2[IMB] and ESR2[LPTM] are valid. It becomes 1 upon every complete TX arbitration process, unless the CPU writes to the Control and Status word of a message buffer already scanned. In other words, it is behind the TX Arbitration Pointer, during the TX arbitration process. If there is no inactive message buffer and only one TX message buffer that is being transmitted, this field remains 0. This field becomes 0 upon the start of every TX arbitration process or upon a write to the Control and Status word of any message buffer. **NOTE** No CPU write to the Control and Status of a message buffer that the abort mechanism blocks affects this field. When MCR[AEN] = 1, the abort code write to the Control and Status of an MB being transmitted (pending abort) is blocked. Any write attempt to a TX MB with its IFLAG flag set is also blocked. 0b - Invalid 1b - Valid

Table continues on the next page...

Table continued from the previous page...

Field	Function
13 IMB	Inactive Message Buffer Indicates whether any message buffer is inactive (CODE field is either 1000b or 0b) when ESR2[VPS] = 1. This field becomes 1 when: • A lowest-priority TX message buffer (ESR2[LPTM]) is found and it is inactive during arbitration. • This field is not 1, and a frame is transmitted successfully. This field always becomes 0 at the start of arbitration (see Arbitration process). If a message buffer is successfully transmitted and this field is 0 (no inactive message buffer), ESR2[VPS] and this field both become 1. The index related to the MB transmitted is loaded into ESR2[LPTM]. In this case, the value of ESR2[LPTM] is the number of the first inactive message buffer. 0b - Message buffer indicated by ESR2[LPTM] is not inactive. 1b - At least one message buffer is inactive.
12 —	Reserved
11-0 —	Reserved

73.6.2.16 Cyclic Redundancy Check (CRCR)

Offset

Register	Offset
CRCR	44h

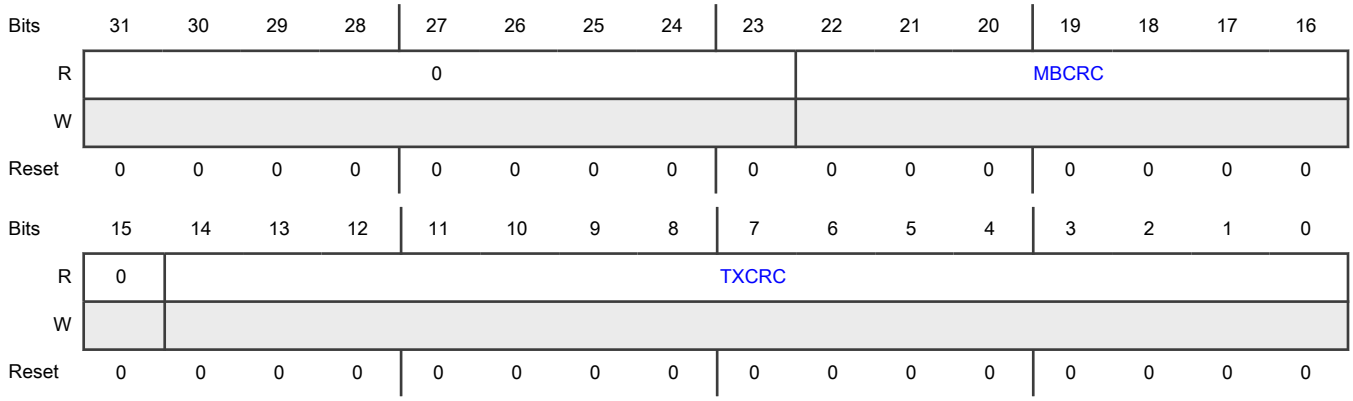
Function

Provides information about the CRC of transmitted messages for non-FD messages. This register only reports the 15 low-order bits of CRC calculations for messages in CAN FD format that require either 17 or 21 bits. For CAN FD format frames, you must use the CAN FD CRC (FDCRC). This register is updated at the same time that the TX interrupt flag is set.

NOTE

See CRC sequence calculation in the CAN Protocol standard ISO 11898-1:2015 for details.

Diagram



Fields

Field	Function
31-23 —	Reserved
22-16 MBCRC	CRC Message Buffer Indicates the number of the message buffer corresponding to the value in CRCR[TXCRC].
15 —	Reserved
14-0 TXCRC	Transmitted CRC value Indicates the CRC value of the last transmitted message for non-FD frames. For FD frames, CRC value is reported in CAN FD CRC (FDCRC).

73.6.2.17 Legacy RX FIFO Global Mask (RXFGMASK)

Offset

Register	Offset
RXFGMASK	48h

Function

Masks the Legacy RX FIFO ID filter table elements that do not have a corresponding RXIMR according to CTRL2[RFFN], when Legacy RX FIFO is enabled.

This register is located in RAM.

You can only write to this register in Freeze mode; the module blocks it in other modes.

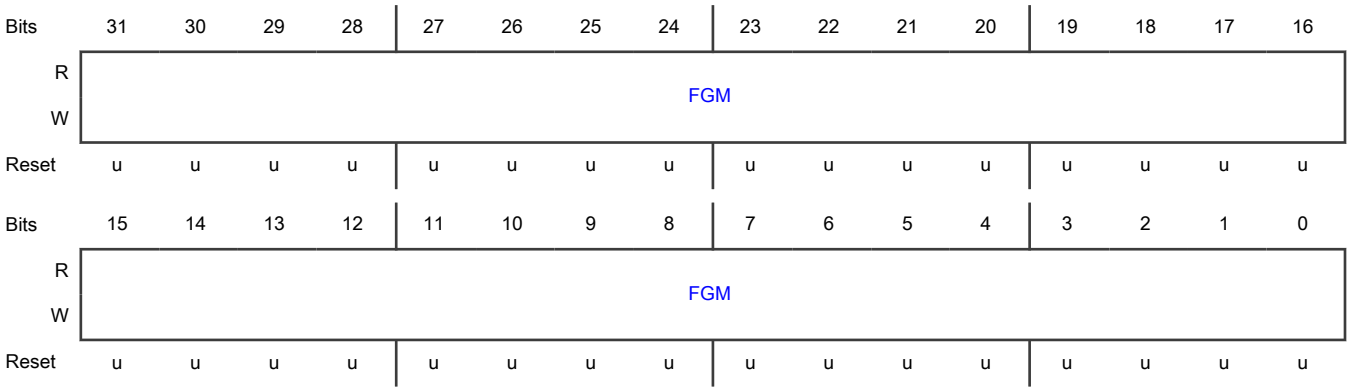
The following table shows how the FGM bits correspond to each IDAF field.

Table 520. Correspondence of Legacy RX FIFO global mask bits to IDF fields

Legacy RX FIFO ID filter table elements format (MCR[IDAM])	Identifier acceptance filter fields					
	RTR	IDE	RXIDA	RXIDB [1]	RXIDC [2]	Reserved
A	FGM[31]	FGM[30]	FGM[29:1]	—	—	FGM[0]
B	FGM[31], FGM[15]	FGM[30], FGM[14]	—	FGM[29:16], FGM[13:0]		—
C	—	—		—	FGM[31:24], FGM[23:16], FGM[15:8], FGM[7:0]	

1. If MCR[IDAM] is equivalent to format B, only the 14 most significant bits of the identifier of the incoming frame are compared with the Legacy RX FIFO filter.
2. If MCR[IDAM] is equivalent to format C, only the eight most significant bits of the identifier of the incoming frame are compared with the Legacy RX FIFO filter.

Diagram



Fields

Field	Function
31-0 FGM	Legacy RX FIFO Global Mask Bits Masks the ID filter table elements bits in a perfect alignment. 0b - The corresponding bit in the filter is "don't care." 1b - The corresponding bit in the filter is checked.

73.6.2.18 Legacy RX FIFO Information (RXFIR)

Offset

Register	Offset
RXFIR	4Ch

Function

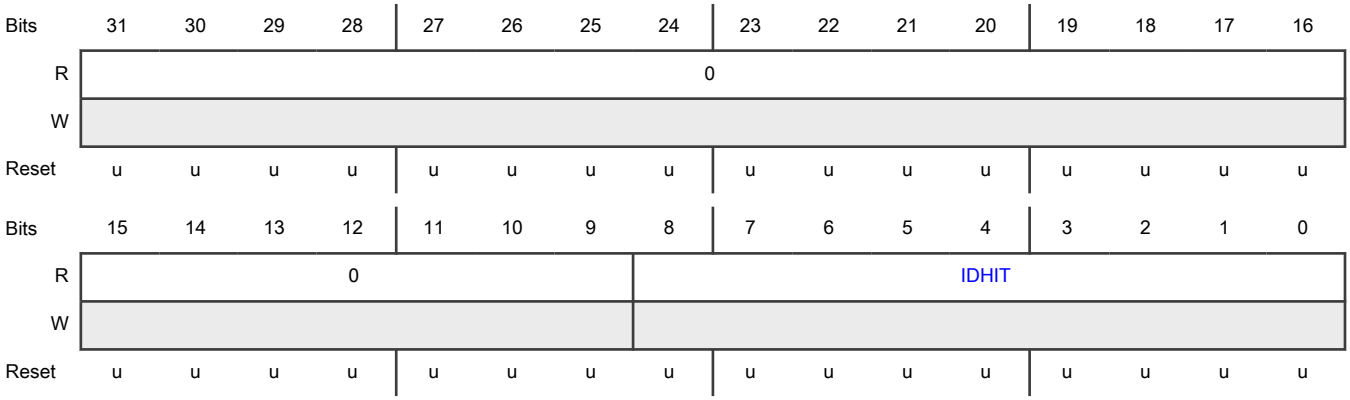
Provides information about Legacy RX FIFO.

This register is the port through which the CPU accesses the output of the Legacy RXFIR FIFO located in RAM. FlexCAN writes to the Legacy RXFIR FIFO when a new message is moved into the Legacy RX FIFO. Also, its output is updated whenever the output of the Legacy RX FIFO is updated with the next message. See Legacy RX FIFO for instructions on reading this register.

NOTE

RXFIR can be written only during memory initialization, due to the error code correction (ECC) feature. In every other case, this register is read-only.

Diagram



Fields

Field	Function
31-9 —	Reserved
8-0 IDHIT	Identifier Acceptance Filter Hit Indicator Indicates which Identifier Acceptance filter that the received message hit in the output of the Legacy RX FIFO. If multiple filters match the incoming message ID, the first matching IDAF found (lowest number) by the matching process is indicated. This field is valid only when IFLAG1[BUF5I] is set.

73.6.2.19 CAN Bit Timing (CBT)

Offset

Register	Offset
CBT	50h

Function

Provides an alternative way to store the CAN bit timing variables described in Control 1 (CTRL1). EPRESDIV, EPROPSEG, EPSEG1, EPSEG2, and ERJW are extended versions of CTRL1[PRESDIV], CTRL1[PROPSEG], CTRL1[PSEG1], CTRL1[PSEG2], and CTRL1[RJW] respectively.

NOTE

The CAN bit variables in CTRL1 and in CBT are stored in the same register.

CBT[BTF] selects the use of the timing variables defined in this register.

When the CAN FD feature is enabled (MCR[FDEN] = 1), always write 1 to CBT[BTF].

Soft reset does not affect the contents of this register.

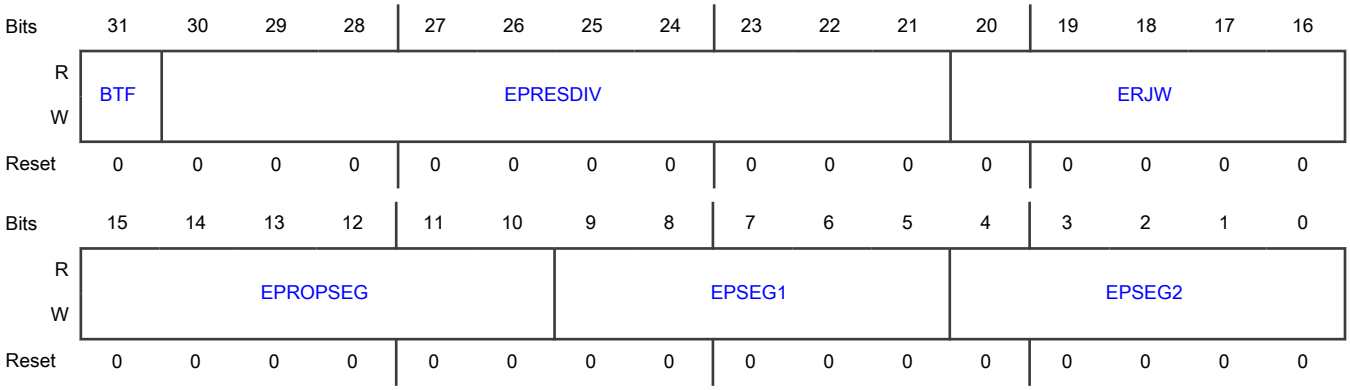
NOTE

Ensure that bit time settings and protocol engine tolerance are in compliance with the CAN Protocol standard ISO 11898-1:2015.

NOTE

If CTRL2[BTE] = 1, EPRESDIV, ERJW, EPROPSEG, EPSEG1, and EPSEG2 are read as zero. A write operation to them has no effect.

Diagram



Fields

Field	Function
31 BTF	Bit Timing Format Enable Enables the use of extended CAN bit timing fields EPRESDIV, EPROPSEG, EPSEG1, EPSEG2, and ERJW. These fields replace the CAN bit timing variables defined in Control 1 (CTRL1). This field can be written in Freeze mode only. 0b - Disable 1b - Enable
30-21 EPRESDIV	Extended Prescaler Division Factor Defines the ratio between the PE clock frequency and the serial clock (Sclock) frequency when CBT[BTF] = 1, otherwise it has no effect. It extends the CTRL1[PRESDIV] value range. The Sclock period defines the time quantum of the CAN protocol. For the reset value, the Sclock frequency is equal to the PE clock frequency (see Protocol timing). This field can be written only in Freeze mode; the module blocks it in other modes. Sclock frequency = PE clock frequency ÷ (EPRESDIV + 1)

Table continues on the next page...

Table continued from the previous page...

Field	Function
20-16 ERJW	Extended Resync Jump Width Defines the maximum number of time quanta that one resynchronization can change a bit time when CBT[BTF] = 1. Otherwise, it has no effect. It extends the CTRL1[RJW] value range. This field can be written only in Freeze mode; the module blocks it in other modes. Resync Jump Width = ERJW + 1. One Time Quantum = one Sclock period.
15-10 EPROPSEG	Extended Propagation Segment Defines the length of the propagation segment in the bit time when CBT[BTF] = 1, otherwise it has no effect. It extends the CTRL1[PROPSEG] value range. This field can be written only in Freeze mode; the module blocks it in other modes. Propagation Segment Time = (EPROPSEG + 1) × Time Quanta. One Time Quantum = one Sclock period.
9-5 EPSEG1	Extended Phase Segment 1 Defines the length of phase segment 1 in the bit time when CBT[BTF] = 1, otherwise it has no effect. It extends the CTRL1[PSEG1] value range. This field can be written only in Freeze mode; the module blocks it in other modes. Phase Buffer Segment 1 = (EPSEG1 + 1) × Time Quanta. One Time Quantum = one Sclock period.
4-0 EPSEG2	Extended Phase Segment 2 Defines the length of phase segment 2 in the bit time when CBT[BTF] = 1, otherwise it has no effect. It extends the CTRL1[PSEG2] value range. This field can be written only in Freeze mode; the module blocks it in other modes. Phase Buffer Segment 1 = (EPSEG2 + 1) × Time Quanta. One Time Quantum = one Sclock period.

73.6.2.20 Interrupt Masks 3 (IMASK3)

Offset

Register	Offset
IMASK3	6Ch

Function

Enables or disables any number of the 32 message buffer interrupts for MB95–MB64. It contains one interrupt mask bit per buffer. This configuration allows the CPU to determine which buffer generates an interrupt after a successful transmission or reception when the corresponding IFLAG3 flag is set.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
CAN_0	IMASK3	—
CAN_1	IMASK3	—
CAN_2	IMASK3	—
CAN_3	—	IMASK3
CAN_4	—	IMASK3
CAN_5	—	IMASK3
CAN_6	—	IMASK3
CAN_7	—	IMASK3
CAN_8	—	IMASK3
CAN_9	—	IMASK3
CAN_10	—	IMASK3
CAN_11	—	IMASK3

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BUF95TO64M															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BUF95TO64M															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 BUF95TO64M	Buffer MBi Mask Enables or disables the corresponding FlexCAN message buffer interrupt for MB95–MB64. When CAN FD is enabled, the MB range is defined in accordance with FDCTRL[MBDSRn].

Table continues on the next page...

Field	Function
	NOTE If the corresponding IFLAG3 flag is set, writing 1 or 0 to a field in IMASK3 can set or clear an interrupt request. 0b - The corresponding buffer interrupt is disabled. 1b - The corresponding buffer interrupt is enabled.

73.6.2.21 Interrupt Flags 3 (IFLAG3)

Offset

Register	Offset
IFLAG3	74h

Function

Defines the flags for the 32 message buffer interrupts for MB95–MB64. It contains one interrupt flag bit per buffer. Each successful transmission or reception sets the corresponding IFLAG3 flag. If the corresponding IMASK3 bit is 1, an interrupt is generated. The interrupt flag must be cleared by writing 1 to it. Writing 0 has no effect.

Before updating MCR[MAXMB], the CPU must service the IFLAG3 flags whose MB value is greater than the MAXMB to be updated. Otherwise, they remain set and are inconsistent with the number of message buffers available.

NOTE

Each module instance supports a different number of registers.

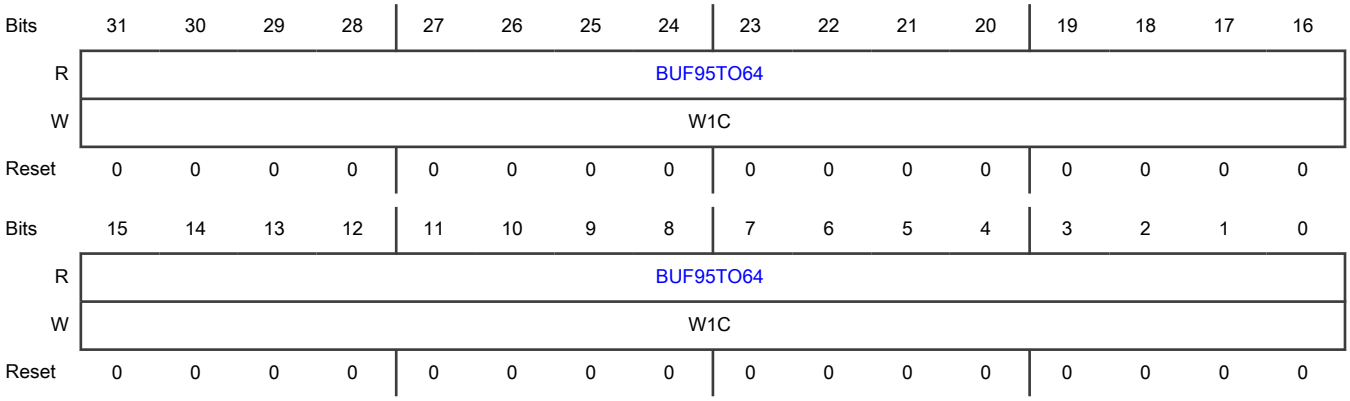
Instance	Register supported	Register not supported
CAN_0	IFLAG3	—
CAN_1	IFLAG3	—
CAN_2	IFLAG3	—
CAN_3	—	IFLAG3
CAN_4	—	IFLAG3
CAN_5	—	IFLAG3
CAN_6	—	IFLAG3
CAN_7	—	IFLAG3
CAN_8	—	IFLAG3
CAN_9	—	IFLAG3

Table continues on the next page...

Table continued from the previous page...

Instance	Register supported	Register not supported
CAN_10	—	IFLAG3
CAN_11	—	IFLAG3

Diagram



Fields

Field	Function
31-0 BUF95TO64	Buffer MBi Interrupt Flags the corresponding FlexCAN message buffer interrupt for MB95–MB64.When CAN FD is enabled, the MB range is defined in accordance with FDCTRL[MBDSRn]. 0b - The corresponding buffer has no occurrence of successfully completed transmission or reception. 1b - The corresponding buffer has successfully completed transmission or reception.

73.6.2.22 Receive Individual Mask (RXIMR0 - RXIMR95)

Offset

For n = 0 to 95:

Register	Offset
RXIMRn	880h + (n × 4h)

Function

Stores the acceptance masks for ID filtering in RX message buffers and the Legacy RX FIFO.

When the Legacy RX FIFO is disabled (MCR[RFEN] = 0), an individual mask is provided for each available RX message buffer on a one-to-one correspondence. When the Legacy RX FIFO is enabled (MCR[RFEN] = 1), an individual mask is provided for

each Legacy RX FIFO ID filter table element on a one-to-one correspondence. This correspondence depends on the setting of CTRL2[RFFN] (see Legacy RX FIFO).

RXIMR0 stores the individual mask associated with either MB0 or ID filter table element 0. RXIMR1 stores the individual mask associated with either MB1 or ID filter table element 1, and so on.

The CPU can only access the RXIMR registers when the module is in Freeze mode; otherwise, the module blocks them. Reset does not affect these registers. They are located in RAM and must be explicitly initialized prior to any reception.

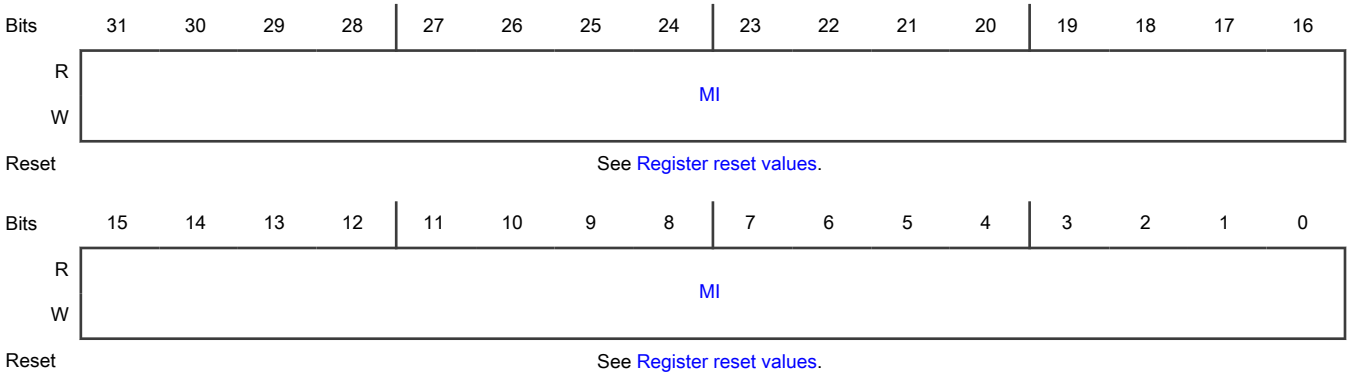
It is possible for the RXIMR memory region to be accessed as general-purpose memory. See Bus interface for more information.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
CAN_0	RXIMR0–RXIMR95	—
CAN_1	RXIMR0–RXIMR95	—
CAN_2	RXIMR0–RXIMR95	—
CAN_3	RXIMR0–RXIMR63	RXIMR64–RXIMR95
CAN_4	RXIMR0–RXIMR63	RXIMR64–RXIMR95
CAN_5	RXIMR0–RXIMR63	RXIMR64–RXIMR95
CAN_6	RXIMR0–RXIMR63	RXIMR64–RXIMR95
CAN_7	RXIMR0–RXIMR63	RXIMR64–RXIMR95
CAN_8	RXIMR0–RXIMR63	RXIMR64–RXIMR95
CAN_9	RXIMR0–RXIMR63	RXIMR64–RXIMR95
CAN_10	RXIMR0–RXIMR63	RXIMR64–RXIMR95
CAN_11	RXIMR0–RXIMR63	RXIMR64–RXIMR95

Diagram



Register reset values

Register	Reset value
RXIMR0–RXIMR63	CAN_0–CAN_11: undefined
RXIMR64–RXIMR95	CAN_0–CAN_2: undefined CAN_3–CAN_11: Register not supported

Fields

Field	Function
31-0 MI	Individual Mask Bits Masks the corresponding bit in both the message buffer filter and Legacy RX FIFO ID filter table element in distinct ways. For message buffer filters, see RX Message Buffers Global Mask (RXMGMASK). For Legacy RX FIFO ID filter table elements, see Legacy RX FIFO Global Mask (RXFGMASK). 0b - The corresponding bit in the filter is "don't care." 1b - The corresponding bit in the filter is checked.

73.6.2.23 Memory Error Control (MECR)

Offset

Register	Offset
MECR	AE0h

Function

Contains control bits for memory error detection and correction (ECC).

NOTE

When CTRL2[ECRWRE] = 0, writes to this register are blocked, except for MECR[ECRWRDIS].

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ECRWRDIS	0											HANCEI_...	FANCEI_...	0	CEI_MSK
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	HAERRIE	FAERRIE	EXTERRIE	0			RERRDIS	ECCDIS	NCEFAFRZ	0						
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

Fields

Field	Function
31 ECRWRDIS	Error Configuration Register Write Disable Disables writes on this register. This field automatically becomes 1 (disabled) when CTRL2[ECRWRE] is 1. The protocol described in Detection and correction of memory errors must be followed. 0b - Enable 1b - Disable
30-20 —	Reserved
19 HANCEI_MSK	Host Access with Noncorrectable Errors Interrupt Mask Enables the interrupt for noncorrectable errors detected in memory reads issued by the host (CPU). 0b - Disable 1b - Enable
18 FANCEI_MSK	FlexCAN Access with Noncorrectable Errors Interrupt Mask Enables the interrupt for noncorrectable errors detected in memory reads issued by FlexCAN internal processes. 0b - Disable 1b - Enable
17 —	Reserved
16 CEI_MSK	Correctable Errors Interrupt Mask Enables the interrupt for correctable errors detected in memory reads issued by the host or FlexCAN internal processes. 0b - Disable 1b - Enable
15 HAERRIE	Host Access Error Injection Enable Enables the injection of errors only in memory reads issued by the host (CPU). 0b - Disable 1b - Enable
14 FAERRIE	FlexCAN Access Error Injection Enable Enables the injection of errors only in memory reads issued by FlexCAN internal processes. 0b - Disable 1b - Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
13 EXTERRIE	Extended Error Injection Enable Extends the error injection on 32-bit memory accesses to the complementary 32-bit word. This feature uses the same 32-bit error injection data and parity words used for 64-bit memory accesses performed by internal FlexCAN processes. See Error Injection Data Pattern (ERRIDPR) and Error Injection Parity Pattern (ERRIPPR). 0b - Disable. Apply error injection only to the 32-bit word. 1b - Enable. Apply error injection to the 64-bit word.
12-10 —	Reserved
9 RERRDIS	Error Report Disable Disables the update of the error report registers. The update of error-related flags and the generation of bus transfer errors are still active. **NOTE** When reading the report registers, this field must be 1 to ensure coherence on the consecutive register reads. 0b - Enable 1b - Disable
8 ECCDIS	Error Correction Disable Completely disables the memory detection and correction mechanism. Besides disabling the error report mechanism, it also stops the update of the error-related flags and the generation of bus transfer errors. The parity bits continue to be calculated and written into memory on write transactions. 0b - Enable 1b - Disable
7 NCEFAFRZ	Noncorrectable Errors in FlexCAN Access Put Chip in Freeze Mode Determines whether to put FlexCAN into Freeze mode when a noncorrectable error is detected in a memory read performed by FlexCAN internal processes. In this case, entering Freeze mode prevents FlexCAN internal processes from treating corrupted data as valid. See Freeze mode for more information. 0b - Normal operation 1b - Freeze mode
6-0 —	Reserved

73.6.2.24 Error Injection Address (ERRIAR)

Offset

Register	Offset
ERRIAR	AE4h

Function

Contains the address where error is to be injected.

Use the following table to convert from the memory map address to the location in the physical FlexCAN RAM. Where pairs of values are provided, the first is the address for MCR[FDEN] = 0, the second is for MCR[FDEN] = 1.

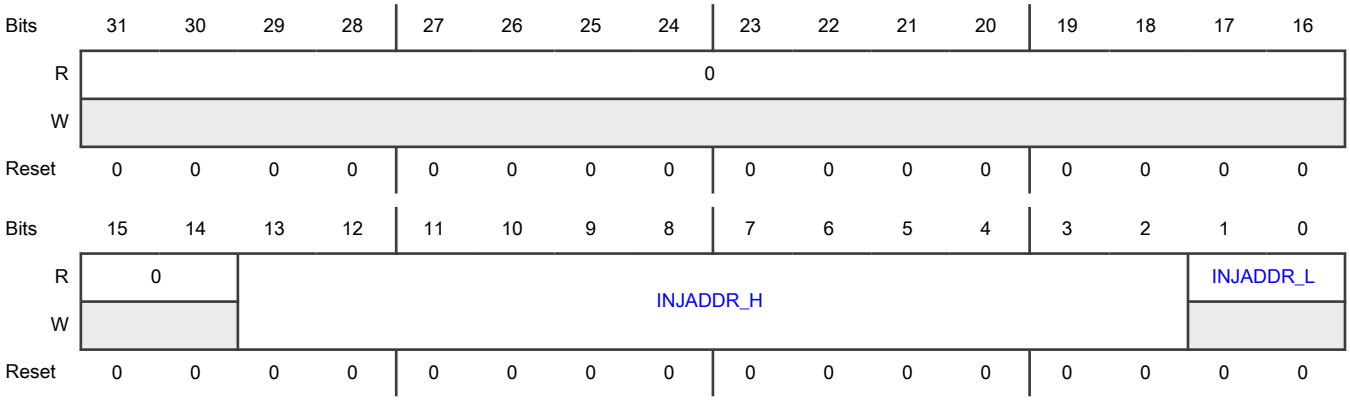
Table 521. Error injection address for classical CAN format

RAM contents	Injection address	Memory map
FlexCAN registers	Not mapped	—
Message buffers	0000h	0080h
RXIMRs	600h	0880h
RXFIR_0	780h	0A80h
RXFIR_1	784h	0A84h
RXFIR_2	788h	0A88h
RXFIR_3	78Ch	0A8Ch
RXFIR_4	790h	0A90h
RXFIR_5	794h	0A94h
Reserved	—	0A98h
RXMGMASK	7A0h	0AA0h
RXFGMASK	7A4h	0AA4h
RX14MASK	7A8h	0AA8h
RX15MASK	7ACh	0AACh
Tx_SMB	7B0h	0AB0h / 0F28h
Rx_SMB0	7C0h / 7F8h	0AC0h / 0F70h
Rx_SMB1	7D0h / 840h	0AD0h / 0FB8h
Rx_SMB0_TIME_STAMP	888h	0C20h
Rx_SMB1_TIME_STAMP	88Ch	0C24h
HR_TIME_STAMP	890h	0C30h
Enhanced RX FIFO	A10h	2000h
ERFFEL	1050h	3000h

NOTE

For RXFIR, Enhanced RX FIFO, and HR_TIME_STAMP addresses, you must inject ECC errors in host access only.

Diagram



Fields

Field	Function
31-14 —	Reserved
13-2 INJADDR_H	Error Injection Address High Defines the 12 most significant bits of the physical RAM address where error is to be injected (see table above).
1-0 INJADDR_L	Error Injection Address Low Defines the two least significant bits of the physical RAM address where error is to be injected. These bits ensure that the address is on a 32-bit boundary.

73.6.2.25 Error Injection Data Pattern (ERRIDPR)

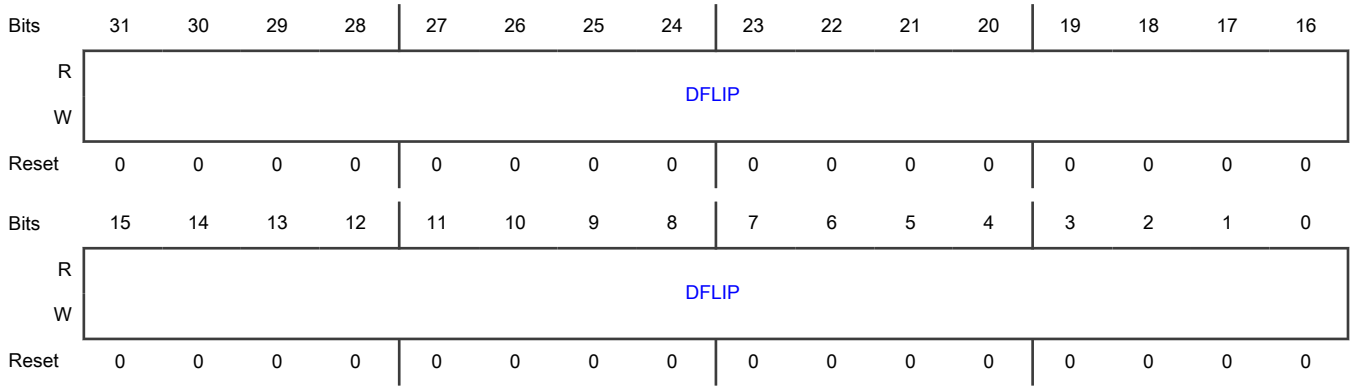
Offset

Register	Offset
ERRIDPR	AE8h

Function

Contains the error pattern to be injected in the data word read from memory.

Diagram



Fields

Field	Function
31-0 DFLIP	Data Flip Pattern Contains the flip pattern. Bits set to 1 in the flip pattern cause the corresponding data bit in the word read from memory to invert.

73.6.2.26 Error Injection Parity Pattern (ERRIPPR)

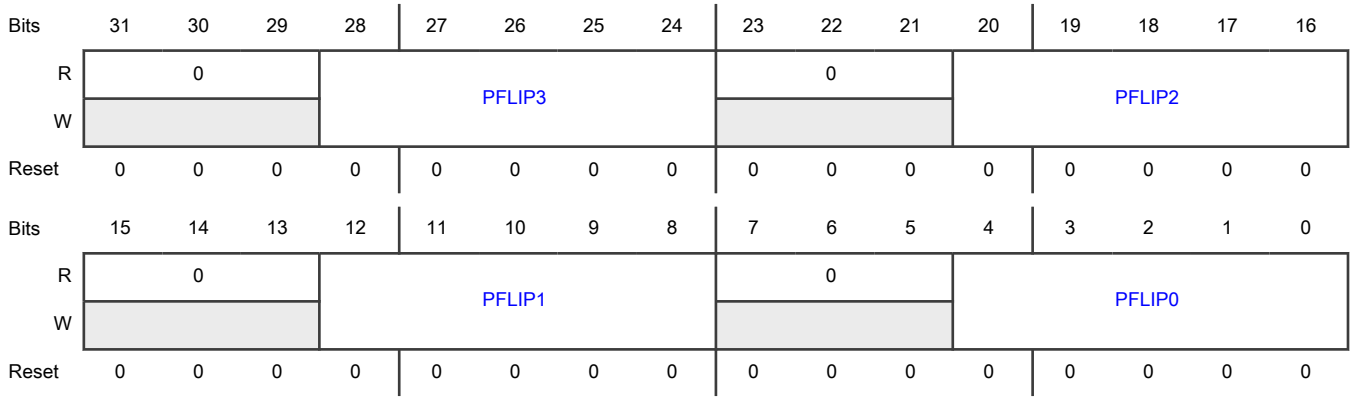
Offset

Register	Offset
ERRIPPR	AECh

Function

Contains the error pattern to be injected in parity bits read from memory with the data word. Bits set to 1 in the flip pattern cause the corresponding parity bit in the word read from memory to invert.

Diagram



Fields

Field	Function
31-29 —	Reserved
28-24 PFLIP3	Parity Flip Pattern for Byte 3 (Most Significant) Contains the error injection pattern for Byte 3.
23-21 —	Reserved
20-16 PFLIP2	Parity Flip Pattern for Byte 2 Contains the error injection pattern for Byte 2.
15-13 —	Reserved
12-8 PFLIP1	Parity Flip Pattern for Byte 1 Contains the error injection pattern for Byte 1.
7-5 —	Reserved
4-0 PFLIP0	Parity Flip Pattern for Byte 0 (Least Significant) Contains the error injection pattern for Byte 0.

73.6.2.27 Error Report Address (RERRAR)

Offset

Register	Offset
RERRAR	AF0h

Function

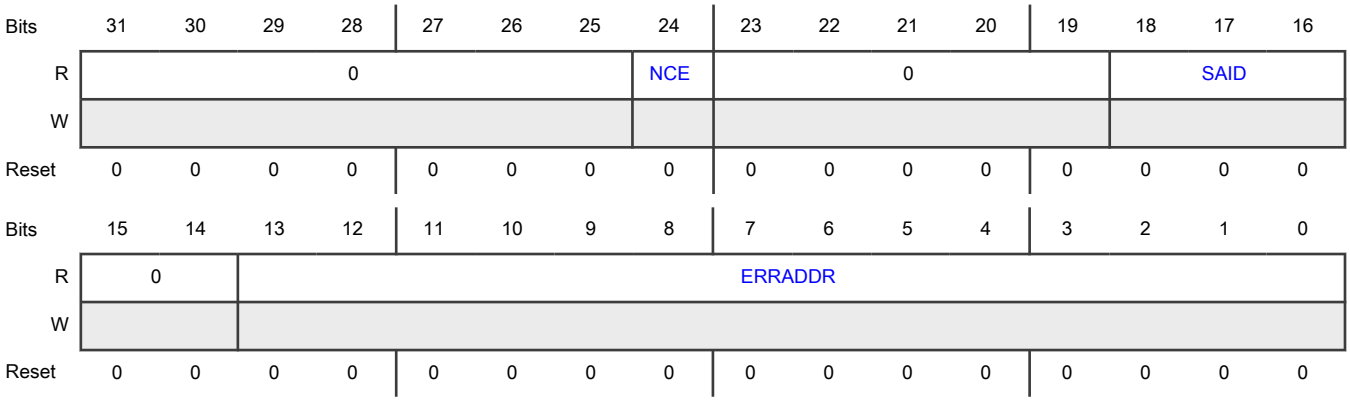
Reports the address used for an access operation in which an error (correctable or noncorrectable) was detected. Also reports the identification of the source of that access.

This address is always reported using a 32-bit alignment. Non-aligned accesses (ERRADDR[1:0] nonzero) are reported with the address aligned, and data is reported in Error Report Data (RERRDR) shifted accordingly. When errors are detected in accesses larger than 32-bit (as performed by FlexCAN internal processes), the address of the 32-bit word where the error was detected is reported. For errors detected in more than one 32-bit word, only the least significant address is reported.

Table 522. Source of memory access

SAID[2:0]	Error during...
0	Move-out FlexCAN access
1	Move-in
2	TX arbitration
3	RX matching
4	Move-out host access
5–7	Reserved

Diagram



Fields

Field	Function
31-25 —	Reserved
24 NCE	Noncorrectable Error Indicates that the report is due to a noncorrectable error. 0b - Reporting a correctable error 1b - Reporting a noncorrectable error
23-19 —	Reserved
18-16 SAID	SAID SAID[2] — Identification of the requester of the memory read request: • 0 = Requested by FlexCAN internal processes • 1 = Requested by host (CPU) SAID[1] — Details of FlexCAN operation:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• 0 = Move • 1 = Scanning SAID[0] — Operation that requested the memory read: • 0 = Transmission • 1 = Reception For more information, see Table 522.
15-14 —	Reserved
13-0 ERRADDR	Address Where Error Detected See description of Error Injection Address (ERRIAR).

73.6.2.28 Error Report Data (RERRDR)

Offset

Register	Offset
RERRDR	AF4h

Function

Reports the raw data (unmodified by the correction performed by ECC logic) read from memory with error. The value reported does not represent the transient values of the BUSY bit (see Table 528) when reading a message buffer.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RDATA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RDATA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 RDATA	Raw Data Word Read from Memory with Error

73.6.2.29 Error Report Syndrome (RERRSYNR)

Offset

Register	Offset
RERRSYNR	AF8h

Function

Contains the syndrome detected in a memory read with error. It also reports the bytes which were read in this 32-bit read transaction.

Each SYND*n* field indicates the type of error and which bit in byte (*n*) the error affects. SYND3 corresponds to the most significant byte in the data word read from memory; SYND0 corresponds to the least significant.

Table 523. Syndrome definition

SYND*n* (hex)	Type	Bit affected
00	—	None (no error)
01	Code	0
02	Code	1
04	Code	2
07	Data	5
08	Code	3
0E	Data	7
10	Code	4
13	Data	2
15	Data	6
16	Data	1
19	Data	3
1A	Data	4
1C	Data	0
06	—	All-zeroes noncorrectable error
1F	—	All-ones noncorrectable error
All others	—	Noncorrectable error

Each BE*n* field indicates which byte in the 32-bit word reported was effectively read. The syndrome bits are calculated for all bytes, including the non-read ones. Errors detected in non-read bytes are indicated (see Error indication) and reported (see Error reporting).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BE3	0			SYND3				BE2	0			SYND2			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BE1	0			SYND1				BE0	0			SYND0			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 BE3	Byte Enabled for Byte 3 (Most Significant) 0b - Byte was not read. 1b - Byte was read.
30-29 —	Reserved
28-24 SYND3	Error Syndrome for Byte 3 (Most Significant) See Table 523.
23 BE2	Byte Enabled for Byte 2 0b - Byte was not read. 1b - Byte was read.
22-21 —	Reserved
20-16 SYND2	Error Syndrome for Byte 2 See Table 523.
15 BE1	Byte Enabled for Byte 1 0b - Byte was not read. 1b - Byte was read.
14-13	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
12-8 SYND1	Error Syndrome for Byte 1 See Table 523.
7 BE0	Byte Enabled for Byte 0 (Least Significant) 0b - Byte was not read. 1b - Byte was read.
6-5 —	Reserved
4-0 SYND0	Error Syndrome for Byte 0 (Least Significant) See Table 523.

73.6.2.30 Error Status (ERRSR)

Offset

Register	Offset
ERRSR	AFCh

Function

Contains the status bits of the error correction and detection operations. These flags can be cleared by writing 1 to them. Writing 0 has no effect.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												HANCEIF	FANCEIF	0	CEIF
W													W1C	W1C		W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												HANCEIOF	FANCEIOF	0	CEIOF
W													W1C	W1C		W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-20 —	Reserved
19 HANCEIF	Host Access with Noncorrectable Error Interrupt Flag Indicates that a noncorrectable error was detected in a memory read initiated by host. A bus transfer error is asserted for that access. If MECR[HANCEI_MSK] = 1, the interrupt is 1. 0b - No errors detected 1b - Error detected
18 FANCEIF	FlexCAN Access with Noncorrectable Error Interrupt Flag Indicates that a noncorrectable error was detected in a memory read initiated by FlexCAN internal processes. If MECR[FANCEI_MSK] = 1, the interrupt is 1. 0b - No errors detected 1b - Error detected
17 —	Reserved
16 CEIF	Correctable Error Interrupt Flag Indicates that a correctable error was detected in a memory read. If MECR[CEI_MSK] = 1, the interrupt is 1. 0b - No errors detected 1b - Error detected
15-4 —	Reserved
3 HANCEIOF	Host Access with Noncorrectable Error Interrupt Overrun Flag Indicates that a noncorrectable error was detected in a memory read initiated by host when ERRSR[HANCEIF] was set. No interrupt is associated with this flag. See Error indication. 0b - No errors detected 1b - Error detected
2 FANCEIOF	FlexCAN Access with Noncorrectable Error Interrupt Overrun Flag Indicates that a noncorrectable error was detected in a memory read initiated by FlexCAN internal processes when ERRSR[FANCEIF] was set. No interrupt is associated with this flag. See Error indication. 0b - No errors detected 1b - Error detected
1 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 CEIOF	Correctable Error Interrupt Overrun Flag Indicates that a correctable error was detected in a memory read when ERRSR[CEIF] was set. No interrupt is associated with this flag. See Error indication. 0b - No errors detected 1b - Error detected

73.6.2.31 Enhanced CAN Bit Timing Prescalers (EPRS)

Offset

Register	Offset
EPRS	BF0h

Function

Defines the CAN bit timing prescalers for the nominal phase and data phase when CTRL2[BTE] = 1.

Used by the module only if CTRL2[BTE] = 1; otherwise, a write operation has no effect and all fields are read as zero.

This register can be written only in Freeze mode; the module blocks it in other modes.

Soft reset does not affect the contents of this register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0						EDPRESDIV									
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0						ENPRESDIV									
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-26 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
25-16 EDPRESDIV	Extended Data Phase Prescaler Division Factor Defines the ratio between the PE clock frequency and the Sclock frequency in the data phase of a CAN FD message when CTRL2[BTE] = 1. The Sclock period defines the time quantum of the CAN FD protocol for the data bit rate. Sclock frequency = PE clock frequency ÷ (EDPRESDIV + 1) **NOTE** To minimize errors when processing FD frames, use the same value for this field and for EPRS[ENPRESDIV]. See the first note in CAN FD frames for details.
15-10 —	Reserved
9-0 ENPRESDIV	Extended Nominal Prescaler Division Factor Defines the ratio between the PE clock frequency and the Sclock frequency when CTRL2[BTE] = 1. Otherwise, it reads as 0 and a write operation has no effect. The Sclock period defines the time quantum of the CAN protocol in the nominal phase. For the reset value, the Sclock frequency is equal to the PE clock frequency (see Protocol timing). Sclock frequency = PE clock frequency ÷ (ENPRESDIV + 1)

73.6.2.32 Enhanced Nominal CAN Bit Timing (ENCBT)

Offset

Register	Offset
ENCBT	BF4h

Function

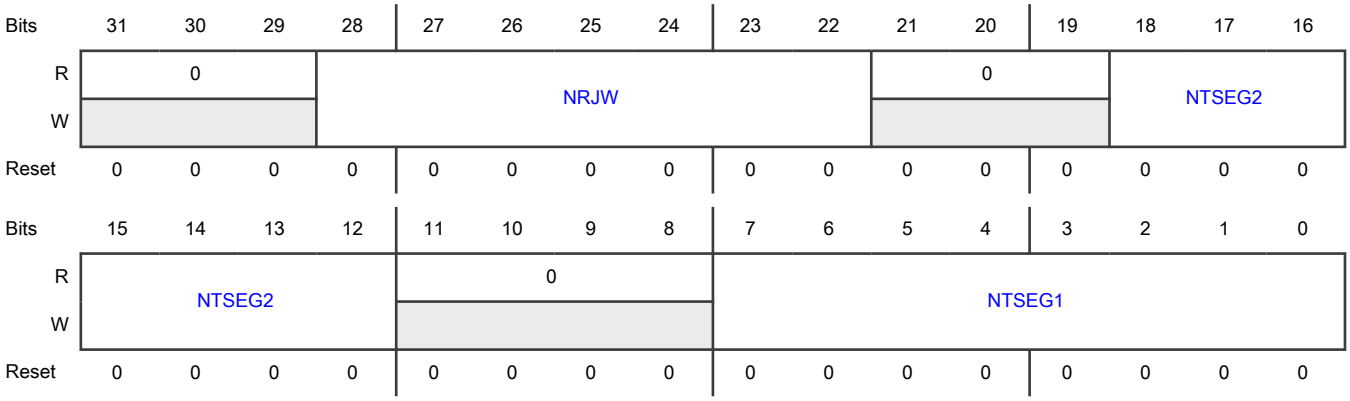
Provides an alternative way to store the CAN bit timing variables described in Control 1 (CTRL1) and CAN Bit Timing (CBT), to get higher CAN bit timing resolution.

This register is used by the module only if CTRL2[BTE] = 1. Otherwise, a write operation has no effect and all fields are read as zero.

Soft reset does not affect the contents of this register.

This register can be written only in Freeze mode; the module blocks it in other modes.

Diagram



Fields

Field	Function
31-29 —	Reserved
28-22 NRJW	Nominal Resynchronization Jump Width Defines the maximum number of time quanta that one resynchronization can change a nominal bit time when CTRL2[BTE] = 1. Otherwise, it has no effect. One time quantum = one Sclock period Nominal Resync Jump Width = NRJW + 1
21-19 —	Reserved
18-12 NTSEG2	Nominal Time Segment 2 Defines the length of Time Segment 2 in the nominal bit time when CTRL2[BTE] = 1. Otherwise, it has no effect. Nominal Time Segment 2 = (NTSEG2 + 1) × Time Quanta One time quantum = one Sclock period
11-8 —	Reserved
7-0 NTSEG1	Nominal Time Segment 1 Defines the length of Time Segment 1 in the bit time when CTRL2[BTE] = 1. Otherwise, it has no effect. Nominal Time Segment 1 = (NTSEG1 + 1) × Time Quanta One time quantum = one Sclock period

73.6.2.33 Enhanced Data Phase CAN Bit Timing (EDCBT)

Offset

Register	Offset
EDCBT	BF8h

Function

Provides an alternative way to store the data phase CAN bit timing variables described in CAN FD Bit Timing (FDCBT) to achieve higher CAN bit timing resolution.

This register is used by the module only if CTRL2[BTE] = 1; otherwise, a write operation has no effect and all fields are read as zero.

Soft reset does not affect the contents of this register.

This register can be written only in Freeze mode; the module blocks it in other modes.

NOTE

Ensure that bit time settings and protocol engine tolerance are in compliance with the CAN Protocol standard ISO 11898-1:2015.

NOTE

DTSEG1 must be at least two time quanta.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0						DRJW				0					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DTSEG2				0				DTSEG1							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-26 —	Reserved
25-22 DRJW	Data Phase Resynchronization Jump Width Defines the maximum number of time quanta that one resynchronization can change a data phase bit time when CTRL2[BTE] = 1. Otherwise, it has no effect.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Data Phase Resync Jump Width = DRJW + 1.
21-16 —	Reserved
15-12 DTSEG2	Data Phase Time Segment 2 Defines the length of time segment 2 in the data phase bit time when CTRL2[BTE] = 1. Otherwise, it has no effect. Data Phase Time Segment 2 = (DTSEG2 + 1) × Time Quanta. One Time Quantum = one Sclock period.
11-5 —	Reserved
4-0 DTSEG1	Data Phase Segment 1 Defines the length of time segment 1 in the data phase bit time when CTRL2[BTE] = 1. Otherwise, it has no effect. Data Phase Time Segment 1 = (NTSEG1 + 1) × Time Quanta. One Time Quantum = one Sclock period.

73.6.2.34 Enhanced Transceiver Delay Compensation (ETDC)

Offset

Register	Offset
ETDC	BFCh

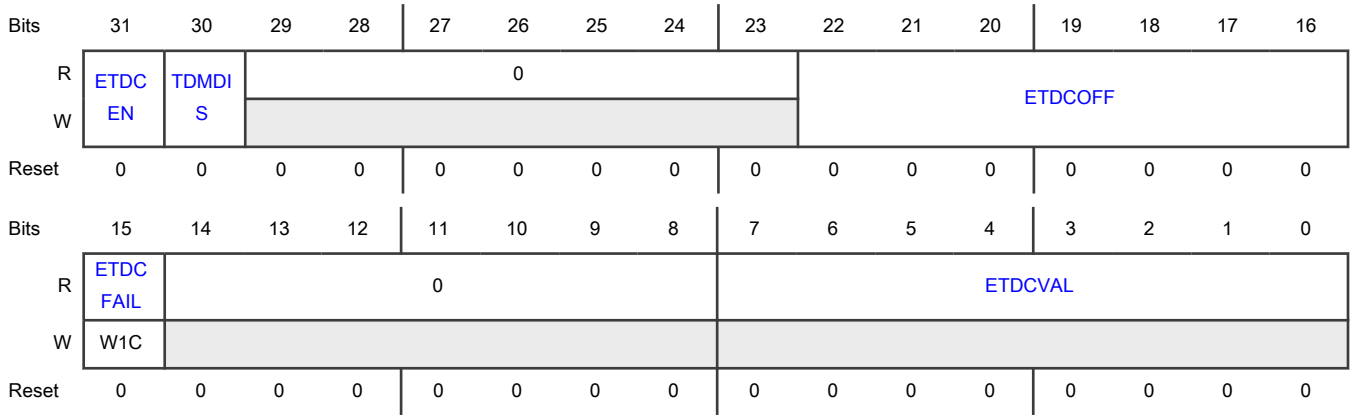
Function

Contains extended versions of FDCTRL[TDCOFF] and FDCTRL[TDCVAL]. This register is used by the module only if CTRL2[BTE] = 1. Otherwise, a write operation has no effect and all fields are read as zero.

NOTE

See Transmitter delay compensation in the CAN Protocol standard ISO 11898-1:2015 for details.

Diagram



Fields

Field	Function
31 ETDCEN	Transceiver Delay Compensation Enable Enables the TDC feature. It can be written in Freeze mode only. **NOTE** See Transmitter delay compensation in the CAN Protocol standard ISO 11898-1:2015 for details. **NOTE** TDC must be disabled when the Loop Back Mode is enabled. See CTRL1[LPB]. 0b - Disable 1b - Enable
30 TDMDIS	Transceiver Delay Measurement Disable Disables the transceiver delay measurement. When the TDC measurement is disabled, only ETDC[ETDCOFF] determines the secondary sample point position. If TCD measurement is enabled, the sum of the transceiver delay measurement plus the enhanced TDC offset determines the secondary sample point position. Soft reset does not affect this field. **NOTE** This bit can be enabled only if CTRL2[BTE] = 1. 0b - Enable 1b - Disable
29-23 —	Reserved
22-16	Enhanced Transceiver Delay Compensation Offset

Table continues on the next page...

Table continued from the previous page...

Field	Function
ETDCOFF	Contains the offset value to be added to the loop delay of the measured transceiver. This value defines the position of the delayed comparison point when bit rate switching is active. See Transceiver delay compensation for details on how the loop delay measurement is performed. This field can be written in Freeze mode only. Its value can be defined in protocol engine (PE) clock periods (CANCLK, see Protocol timing for more details). It must be smaller than the CAN bit duration in the data bit rate for proper operation. Do not write 0 to this field. **NOTE** If CTRL2[BTE] becomes 1 after a chip-level hard reset, this field is read as 1h.
15 ETDCFAIL	Transceiver Delay Compensation Fail Indicates whether the transceiver delay compensation (TDC) mechanism is out of range. In this case, it is unable to compensate the loop delay of the transceiver and successfully compare the delayed received bits to the transmitted ones. (See Transceiver delay compensation.) This field becomes 0 the first time FlexCAN detects the out of range condition. 0b - In range 1b - Out of range
14-8 —	Reserved
7-0 ETDCVAL	Enhanced Transceiver Delay Compensation Value Contains ETDC[ETDCOFF] added to the measured value of the transceiver loop delay in the latest transmitted CAN FD frame, with BRS = 1. The module only updates this field when ETDC[ETDCEN] = 1. Soft reset affects this field. **NOTE** If ETDC[TDMDIS] = 1, this field stores ETDC[ETDCOFF] only.

73.6.2.35 CAN FD Control (FDCTRL)

Offset

Register	Offset
FDCTRL	C00h

Function

Contains control bits for CAN FD operation. It also defines the data size of message buffers allocated in different partitions of RAM (memory blocks) as described in Table 524.

When an 8-byte payload is selected:

- Block R0 allocates MB0–MB31.
- Block R1 allocates MB32–MB63.
- Block R2 allocates MB64–MB95.

When a payload larger than eight bytes is selected, the maximum number of message buffers in a block is limited as described below.

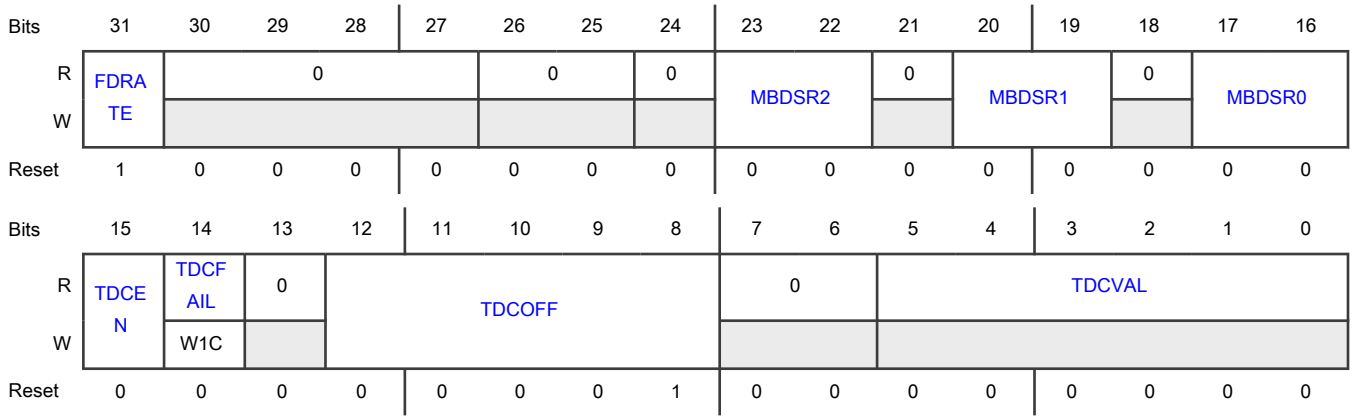
Table 524. Number of message buffers

Payload size	Maximum number of message buffers per RAM block
8 bytes	32
16 bytes	21
32 bytes	12
64 bytes	7

One memory block fits exactly 32 message buffers with an 8-byte payload. For other possible payload sizes, empty memory may exist between the last message buffer in a block and the beginning of the next block. This empty memory corresponds to less than one message buffer, and must not be used.

Soft reset does not affect the contents of this register.

Diagram



Fields

Field	Function
31 FDRATE	Bit Rate Switch Enable Enables the effect of the Bit Rate Switch (BRS bit) during the data phase of TX messages. When 1, if BRS = 1 in the TX message buffer, frames are transmitted with bit rate switching. When 0, frames are transmitted at a nominal rate, and the BRS bit in the TX MB has no effect. The CPU can write to this field at any time. However, its effect becomes active only under one of these conditions: • The CAN bus is in the Wait for Bus Idle state. • The CAN bus is in the Bus Idle state.

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>

The CAN bus is in the Bus Off state.
The current frame under reception or transmission reaches the interframe space.

By writing 0 to FDCTRL[FDRATE], the CPU can force all bits in CAN FD messages to be transmitted at nominal bit rate. This transmission occurs regardless of the value in the BRS bit of the TX message buffers.
0b - Disable
1b - Enable
</td></tr>
<tr><td>30-27
—</td><td>Reserved</td></tr>
<tr><td>26-25
—</td><td>Reserved</td></tr>
<tr><td>24
—</td><td>Reserved</td></tr>
<tr><td>23-22
MBDSR2</td><td>
Message Buffer Data Size for Region 2
Selects the data size per message buffer for region R2 of message buffers allocated in RAM.
This field can be written in Freeze mode only.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>CAN_0</td><td>FDCTRL</td><td>—</td></tr>
<tr><td>CAN_1</td><td>FDCTRL</td><td>—</td></tr>
<tr><td>CAN_2</td><td>FDCTRL</td><td>—</td></tr>
<tr><td>CAN_3</td><td>—</td><td>FDCTRL</td></tr>
<tr><td>CAN_4</td><td>—</td><td>FDCTRL</td></tr>
<tr><td>CAN_5</td><td>—</td><td>FDCTRL</td></tr>
<tr><td>CAN_6</td><td>—</td><td>FDCTRL</td></tr>
<tr><td>CAN_7</td><td>—</td><td>FDCTRL</td></tr>
<tr><td>CAN_8</td><td>—</td><td>FDCTRL</td></tr>
</table>
</td></tr>
</table>

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>CAN_9</td><td>—</td><td>FDCTRL</td></tr>
<tr><td>CAN_10</td><td>—</td><td>FDCTRL</td></tr>
<tr><td>CAN_11</td><td>—</td><td>FDCTRL</td></tr>
</table>
00b - 8 bytes
01b - 16 bytes
10b - 32 bytes
11b - 64 bytes</td></tr>
<tr><td>21
—</td><td>Reserved</td></tr>
<tr><td>20-19
MBDSR1</td><td>Message Buffer Data Size for Region 1
Selects the data size per message buffer for region R1 of message buffers allocated in RAM.
This field can be written in Freeze mode only.
00b - 8 bytes
01b - 16 bytes
10b - 32 bytes
11b - 64 bytes</td></tr>
<tr><td>18
—</td><td>Reserved</td></tr>
<tr><td>17-16
MBDSR0</td><td>Message Buffer Data Size for Region 0
Selects the data size per message buffer for region R0 of message buffers allocated in RAM.
This field can be written in Freeze mode only.
00b - 8 bytes
01b - 16 bytes
10b - 32 bytes
11b - 64 bytes</td></tr>
<tr><td>15
TDCEN</td><td>Transceiver Delay Compensation Enable
Enables the TDC feature. It can be written in Freeze mode only.
See Transmitter delay compensation in the CAN Protocol standard ISO 11898-1:2015 for details.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	TDC must be disabled when Loopback mode is enabled (see CTRL1[LPB]). **NOTE** If CTRL2[BTE] = 1, this field is read as 0 and a write operation has no effect. 0b - Disable 1b - Enable
14 TDCFAIL	Transceiver Delay Compensation Fail Indicates whether the Transceiver Delay Compensation (TDC) mechanism is out of range. In this case, the mechanism cannot compensate for the loop delay of the transceiver and successfully compare the delayed received bits to the transmitted ones (see Transceiver delay compensation). The first time that FlexCAN detects the out-of-range condition, this field becomes 1. **NOTE** If CTRL2[BTE] = 1, this field is read as 0 and a write operation has no effect. 0b - In range 1b - Out of range
13 —	Reserved
12-8 TDCOFF	Transceiver Delay Compensation Offset Contains the offset value to be added to the loop delay of the measured transceiver. This value defines the position of the delayed comparison point when bit rate switching is active. See Transceiver delay compensation for details about loop delay measurement. This field can be written in Freeze mode only. Its value can be defined in Protocol Engine Clock periods (CANCLK, see Protocol timing for more details). The value must be smaller than the CAN bit duration in the data bit rate for proper operation. **NOTE** If CTRL2[BTE] = 1, TDCOFF is read as 0 and a write operation has no effect. Do not write 0 to this field.
7-6 —	Reserved
5-0 TDCVAL	Transceiver Delay Compensation Value Contains the value of the transceiver loop delay measured from the transmitted EDL-to-R0 transition edge to the respective received one added to FDCTRL[TDCOFF]. This value is an integer multiple of the Protocol Engine Clock period (CANCLK). If CTRL2[BTE] = 1, this field is read as 0. See Protocol timing for details on the loop delay measurement.

73.6.2.36 CAN FD Bit Timing (FDCBT)

Offset

Register	Offset
FDCBT	C04h

Function

Stores the CAN bit timing variables used in the data phase of CAN FD messages when the FDCTRL[FDRATE] = 1, compatible with the CAN FD specification. Fields in this register define:

- The time quantum duration
- The number of time quanta per CAN bit
- The sample point position for the data bit rate portion of a CAN FD message with BRS = 1

Soft reset does not affect the contents of this register.

The sum of the Fast Propagation Segment (FPROPSEG) and Fast Phase Segment 1 (FPSEG1) must be at least two time quanta.

Ensure bit time settings and protocol engine tolerance are in compliance with the CAN Protocol standard ISO 11898-1:2015.

NOTE

If CTRL2[BTE] = 1, this register is read as zero and a write operation has no effect.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				FPRESDIV								0	FRJW		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	FPROPSEG					0		FPSEG1			0	FPSEG2			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29-20 FPRESDIV	Fast Prescaler Division Factor Defines the ratio between the PE clock frequency and the serial clock (Sclock) frequency in the data bit rate portion of a CAN FD message with BRS = 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	The Sclock period defines the time quantum of the CAN FD protocol for the data bit rate. This field can be written only in Freeze mode; the module blocks it in other modes. Sclock frequency = PE clock frequency ÷ (FPRESDIV + 1). **NOTE** To minimize errors when processing FD frames, use the same value for this field and for CTRL1[PRESDIV] or CBT[EPRESDIV]. See the first note in CAN FD frames for details.
19 —	Reserved
18-16 FRJW	Fast Resync Jump Width Defines the maximum number of time quanta that one resynchronization can change a bit time in the data bit rate portion of a CAN FD message with BRS = 1. This field can be written only in Freeze mode; the module blocks it in other modes. Resync Jump Width = FSJW + 1. One Time Quantum = one Sclock period.
15 —	Reserved
14-10 FPROPSEG	Fast Propagation Segment Defines the length of the propagation segment in the bit time in the data bit rate portion of a CAN FD message with BRS = 1. This field can be written only in Freeze mode; the module blocks it in other modes. Propagation Segment Time = FPROPSEG × Time Quanta. One Time Quantum = one Sclock period.
9-8 —	Reserved
7-5 FPSEG1	Fast Phase Segment 1 Defines the length of phase segment 1 in the bit time in the data bit rate portion of a CAN FD message with BRS = 1. This field can be written only in Freeze mode; the module blocks it in other modes. Phase Segment 1 = (FPSEG1 + 1) × Time Quanta. One Time Quantum = one Sclock period.
4-3 —	Reserved
2-0 FPSEG2	Fast Phase Segment 2 Defines the length of phase segment 2 in the data bit rate portion of a CAN FD message with BRS = 1. This field can be written only in Freeze mode; the module blocks it in other modes.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Phase Segment 2 = (FPSEG2 + 1) × Time Quanta. One Time Quantum = one Sclock period.

73.6.2.37 CAN FD CRC (FDCRC)

Offset

Register	Offset
FDCRC	C08h

Function

Provides information about the cyclic redundancy check (CRC) of transmitted messages.

FlexCAN uses different CRC polynomials for different frame formats.

- The CRC_15 polynomial is used for all frames in CAN format.
- The CRC_17 polynomial is used for frames in CAN FD format with a DATA FIELD up to 16 bytes.
- The CRC_21 polynomial is used for frames in CAN FD format with a DATA FIELD longer than 16 bytes.

Each polynomial shown below results in a Hamming distance of 6. This register is updated at the same time that the TX Interrupt flag is set.

$$\text{CRC_15} = \text{C599h: } (x^{15} + x^{14} + x^{10} + x^{8} + x^{7} + x^{4} + x^{3} + 1)$$

$$\text{CRC_17} = \text{3685Bh: } (x^{17} + x^{16} + x^{14} + x^{13} + x^{11} + x^{6} + x^{4} + x^{3} + x^{1} + 1)$$

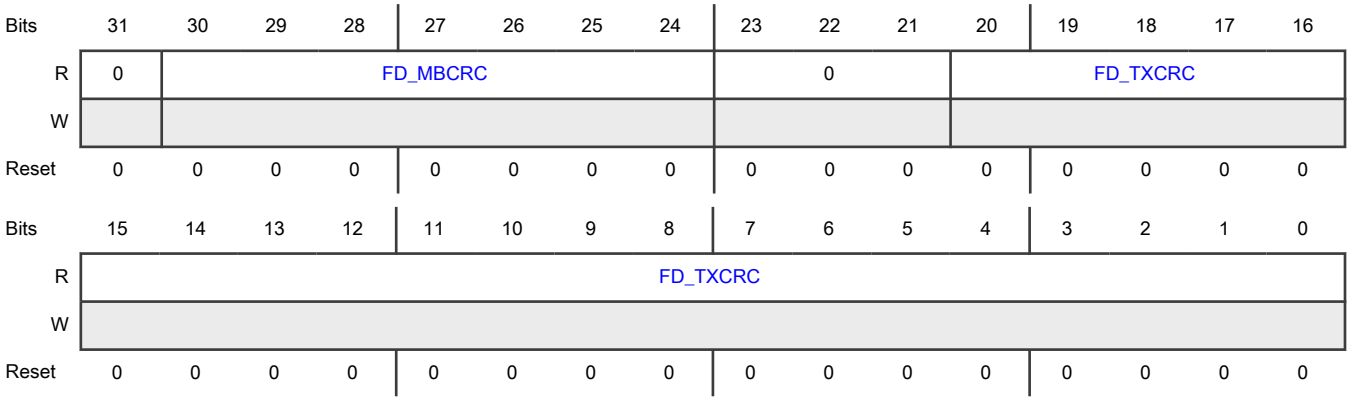
$$\text{CRC_21} = \text{302899h: } (x^{21} + x^{20} + x^{13} + x^{11} + x^{7} + x^{4} + x^{3} + 1)$$

Equation 31. CRC polynomial used on CAN frame

NOTE

See CRC sequence calculation in the CAN Protocol standard ISO 11898-1:2015 for details.

Diagram



Fields

Field	Function
31 —	Reserved
30-24 FD_MBCRC	CRC Message Buffer Number for FD_TXCRC Indicates the number of the message buffer corresponding to the value in FDCRC[FD_TXCRC], for both FD and non-FD frames. It reports the same information as in CRCR[MBCRC].
23-21 —	Reserved
20-0 FD_TXCRC	Extended Transmitted CRC value Contains the CRC value calculated over the most recent transmitted message. Different CRC polynomials are used for different frame formats. For CRC_15 and CRC_17, the six most significant bits and the four most significant bits are reported as zeroes, respectively. For CRC_15, this field has the same content as Cyclic Redundancy Check (CRCR).

73.6.2.38 Enhanced RX FIFO Control (ERFCR)

Offset

Register	Offset
ERFCR	C0Ch

Function

Defines the Enhanced RX FIFO configuration.

This register can be written only in Freeze mode.

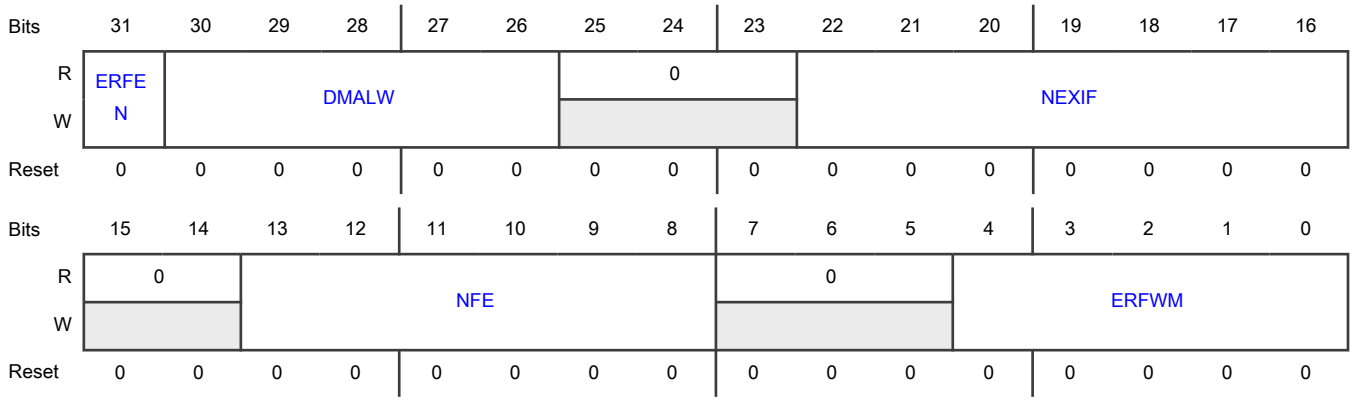
Soft reset does not affect any of the contents of this register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
CAN_0	ERFCR	—
CAN_1	ERFCR	—
CAN_2	ERFCR	—
CAN_3	—	ERFCR
CAN_4	—	ERFCR
CAN_5	—	ERFCR
CAN_6	—	ERFCR
CAN_7	—	ERFCR
CAN_8	—	ERFCR
CAN_9	—	ERFCR
CAN_10	—	ERFCR
CAN_11	—	ERFCR

Diagram



Fields

Field	Function
31	Enhanced RX FIFO enable

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>ERFEN</td><td>Enables the Enhanced RX FIFO.
NOTE
If MCR[RFEN] = 1, do not write 1 to this field.
0b - Disable
1b - Enable</td></tr>
<tr><td>30-26
DMALW</td><td>DMA Last Word
Defines the last DMA address for each Enhanced RX FIFO element.
This table shows the number of elements and the last address for each Enhanced RX FIFO element according to the value of DMALW.
<table>
<tr><th>DMALW</th><th>Number of 32-bit words transferred</th><th>Last FIFO address</th></tr>
<tr><td>0</td><td>1</td><td>2000h</td></tr>
<tr><td>1</td><td>2</td><td>2004h</td></tr>
<tr><td>2</td><td>3</td><td>2008h</td></tr>
<tr><td>3</td><td>4</td><td>200Ch</td></tr>
<tr><td>4</td><td>5</td><td>2010h</td></tr>
<tr><td>5</td><td>6</td><td>2014h</td></tr>
<tr><td>6</td><td>7</td><td>2018h</td></tr>
<tr><td>7</td><td>8</td><td>201Ch</td></tr>
<tr><td>8</td><td>9</td><td>2020h</td></tr>
<tr><td>9</td><td>10</td><td>2024h</td></tr>
<tr><td>10</td><td>11</td><td>2028h</td></tr>
<tr><td>11</td><td>12</td><td>202Ch</td></tr>
<tr><td>12</td><td>13</td><td>2030h</td></tr>
<tr><td>13</td><td>14</td><td>2034h</td></tr>
<tr><td>14</td><td>15</td><td>2038h</td></tr>
<tr><td>15</td><td>16</td><td>203Ch</td></tr>
<tr><td>16</td><td>17</td><td>2040h</td></tr>
<tr><td>17</td><td>18</td><td>2044h</td></tr>
<tr><td>18</td><td>19</td><td>2048h</td></tr>
<tr><td>19</td><td>20</td><td>204Ch</td></tr>
</table>
</td></tr>
</table>

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>DMALW</th><th>Number of 32-bit words transferred</th><th>Last FIFO address</th></tr>
</table>
NOTE
Undefined DMALW values in the table are reserved and must not be used.
</td></tr>
<tr><td>25-23
—</td><td>Reserved</td></tr>
<tr><td>22-16
NEXIF</td><td>Number of Extended ID Filter Elements
Defines the number of extended ID filter elements used during the Enhanced RX FIFO matching process.
The value of this field must be less than or equal to NFE + 1.
The number of standard ID filter elements is 2 × (NFE - NEXIF + 1).
This table shows the number of extended ID filters and standard ID filters available for Enhanced RX FIFO if all filter elements are used.
<table>
<tr><th>NEXIF</th><th>NFE</th><th>Number of Extended ID filter elements</th><th>Number of Standard ID filter elements</th></tr>
<tr><td>0</td><td>63</td><td>0</td><td>128</td></tr>
<tr><td>1</td><td>63</td><td>1</td><td>126</td></tr>
<tr><td>2</td><td>63</td><td>2</td><td>124</td></tr>
<tr><td>3</td><td>63</td><td>3</td><td>122</td></tr>
<tr><td>4</td><td>63</td><td>4</td><td>120</td></tr>
<tr><td>5</td><td>63</td><td>5</td><td>118</td></tr>
<tr><td>6</td><td>63</td><td>6</td><td>116</td></tr>
<tr><td>7</td><td>63</td><td>7</td><td>114</td></tr>
<tr><td>8</td><td>63</td><td>8</td><td>112</td></tr>
<tr><td>9</td><td>63</td><td>9</td><td>110</td></tr>
<tr><td>10</td><td>63</td><td>10</td><td>108</td></tr>
<tr><td>11</td><td>63</td><td>11</td><td>106</td></tr>
<tr><td>12</td><td>63</td><td>12</td><td>104</td></tr>
<tr><td>13</td><td>63</td><td>13</td><td>102</td></tr>
<tr><td>14</td><td>63</td><td>14</td><td>100</td></tr>
<tr><td>15</td><td>63</td><td>15</td><td>98</td></tr>
<tr><td>16</td><td>63</td><td>16</td><td>96</td></tr>
</table>
</td></tr>
</table>

Table continued from the previous page...

Field	Function			
	NEXIF	NFE	Number of Extended ID filter elements	Number of Standard ID filter elements
	17	63	17	94
	18	63	18	92
	19	63	19	90
	20	63	20	88
	21	63	21	86
	22	63	22	84
	23	63	23	82
	24	63	24	80
	25	63	25	78
	26	63	26	76
	27	63	27	74
	28	63	28	72
	29	63	29	70
	30	63	30	68
	31	63	31	66
	32	63	32	64
	33	63	33	62
	34	63	34	60
	35	63	35	58
	36	63	36	56
	37	63	37	54
	38	63	38	52
	39	63	39	50
	40	63	40	48
	41	63	41	46
	42	63	42	44
	43	63	43	42
	44	63	44	40
	45	63	45	38

Table continues on the next page...

Table continued from the previous page...

Field	Function

NEXIF	NFE	Number of Extended ID filter elements	Number of Standard ID filter elements
46	63	46	36
47	63	47	34
48	63	48	32
49	63	49	30
50	63	50	28
51	63	51	26
52	63	52	24
53	63	53	22
54	63	54	20
55	63	55	18
56	63	56	16
57	63	57	14
58	63	58	12
59	63	59	10
60	63	60	8
61	63	61	6
62	63	62	4
63	63	63	2
64	63	64	0

Field	Function
15-14 —	Reserved
13-8 NFE	Number of Enhanced RX FIFO Filter Elements Defines the total number of filter elements used during the enhanced RX FIFO matching process according to the table.

NFE	Maximum number of extended ID filter elements (NEXIF = NFE + 1)	Maximum number of standard ID filter elements (NEXIF = 0)
0	1	2
1	2	4

Table continues on the next page...

Table continued from the previous page...

Field	Function		
	NFE	Maximum number of extended ID filter elements (NEXIF = NFE + 1)	Maximum number of standard ID filter elements (NEXIF = 0)
	2	3	6
	3	4	8
	4	5	10
	5	6	12
	6	7	14
	7	8	16
	8	9	18
	9	10	20
	10	11	22
	11	12	24
	12	13	26
	13	14	28
	14	15	30
	15	16	32
	16	17	34
	17	18	36
	18	19	38
	19	20	40
	20	21	42
	21	22	44
	22	23	46
	23	24	48
	24	25	50
	25	26	52
	26	27	54
	27	28	56
	28	29	58
	29	30	60
	30	31	62

Table continues on the next page...

Table continued from the previous page...

Field	Function		
	NFE	Maximum number of extended ID filter elements (NEXIF = NFE + 1)	Maximum number of standard ID filter elements (NEXIF = 0)
	31	32	64
	32	33	66
	33	34	68
	34	35	70
	35	36	72
	36	37	74
	37	38	76
	38	39	78
	39	40	80
	40	41	82
	41	42	84
	42	43	86
	43	44	88
	44	45	90
	45	46	92
	46	47	94
	47	48	96
	48	49	98
	49	50	100
	50	51	102
	51	52	104
	52	53	106
	53	54	108
	54	55	110
	55	56	112
	56	57	114
	57	58	116
	58	59	118
	59	60	120

Table continues on the next page...

Table continued from the previous page...

Field	Function
	<table><tr><th>NFE</th><th>Maximum number of extended ID filter elements (NEXIF = NFE + 1)</th><th>Maximum number of standard ID filter elements (NEXIF = 0)</th></tr><tr><td>60</td><td>61</td><td>122</td></tr><tr><td>61</td><td>62</td><td>124</td></tr><tr><td>62</td><td>63</td><td>126</td></tr><tr><td>63</td><td>64</td><td>128</td></tr></table>
7-5 —	Reserved
4-0 ERFWM	Enhanced RX FIFO Watermark Defines the minimum number of CAN messages stored in the Enhanced RX FIFO. When that number is reached, ERFSR[ERFWMI] becomes 1. Minimum number of CAN messages = ERFWM + 1. **NOTE** If MCR[DMA] = 1, write 0h to this field.

73.6.2.39 Enhanced RX FIFO Interrupt Enable (ERFIER)

Offset

Register	Offset
ERFIER	C10h

Function

Contains the interrupt enables for the Enhanced RX FIFO.

Soft reset does not affect this register.

NOTE

Each module instance supports a different number of registers.

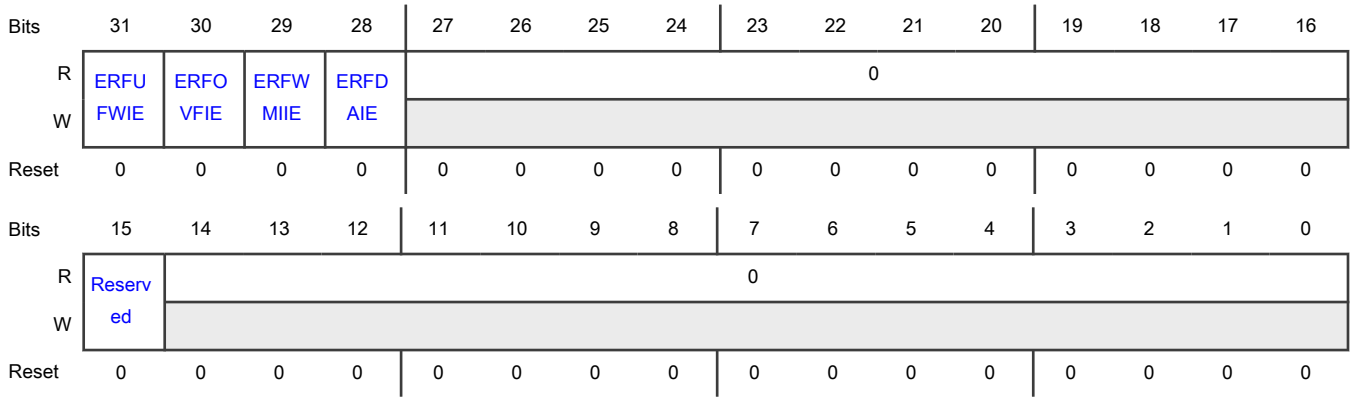
Instance	Register supported	Register not supported
CAN_0	ERFIER	—
CAN_1	ERFIER	—

Table continues on the next page...

Table continued from the previous page...

Instance	Register supported	Register not supported
CAN_2	ERFIER	—
CAN_3	—	ERFIER
CAN_4	—	ERFIER
CAN_5	—	ERFIER
CAN_6	—	ERFIER
CAN_7	—	ERFIER
CAN_8	—	ERFIER
CAN_9	—	ERFIER
CAN_10	—	ERFIER
CAN_11	—	ERFIER

Diagram



Fields

Field	Function
31 ERFUFWIE	Enhanced RX FIFO Underflow Interrupt Enable Enables interrupt for ERFSR[ERFUFW]. 0b - Disable 1b - Enable
30	Enhanced RX FIFO Overflow Interrupt Enable Enables interrupt for ERFSR[ERFOVF].

Table continues on the next page...

Table continued from the previous page...

Field	Function
ERFOVFIE	0b - Disable 1b - Enable
29 ERFWMIIE	Enhanced RX FIFO Watermark Indication Interrupt Enable Enables interrupt for ERFSR[ERFWMI]. 0b - Disable 1b - Enable
28 ERFDAIE	Enhanced RX FIFO Data Available Interrupt Enable Enables interrupt for ERFSR[ERFDA]. 0b - Disable 1b - Enable
27-16 —	Reserved
15 —	Reserved
14-0 —	Reserved

73.6.2.40 Enhanced RX FIFO Status (ERFSR)

Offset

Register	Offset
ERFSR	C14h

Function

Contains the status fields of the Enhanced RX FIFO including error indications and a clear FIFO field.

Soft reset does not affect this register.

NOTE

Each module instance supports a different number of registers.

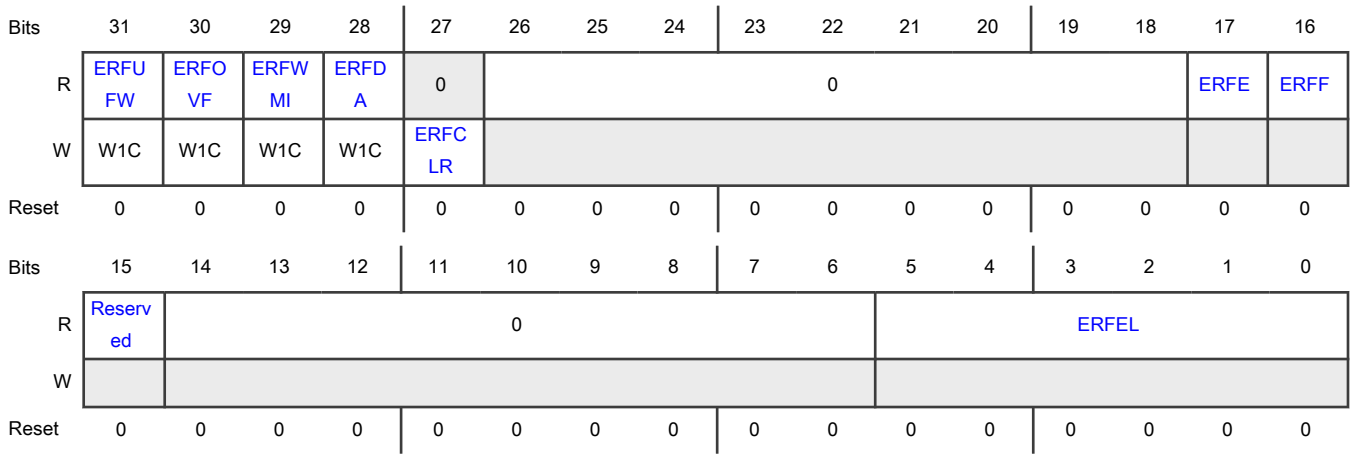
Instance	Register supported	Register not supported
CAN_0	ERFSR	—

Table continues on the next page...

Table continued from the previous page...

Instance	Register supported	Register not supported
CAN_1	ERFSR	—
CAN_2	ERFSR	—
CAN_3	—	ERFSR
CAN_4	—	ERFSR
CAN_5	—	ERFSR
CAN_6	—	ERFSR
CAN_7	—	ERFSR
CAN_8	—	ERFSR
CAN_9	—	ERFSR
CAN_10	—	ERFSR
CAN_11	—	ERFSR

Diagram



Fields

Field	Function
31 ERFUFW	Enhanced RX FIFO Underflow Flag Indicates whether an underflow condition occurred in the enhanced RX FIFO. If ERFIER[ERFUFWIE] = 1, this field generates an interrupt.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No such occurrence 1b - Underflow
30 ERFOVF	Enhanced RX FIFO Overflow Flag Indicates whether an overflow condition occurred in the Enhanced RX FIFO. If ERFIER[ERFOVFIE] = 1, this field generates an interrupt. 0b - No such occurrence 1b - Overflow
29 ERFWMI	Enhanced RX FIFO Watermark Indication Flag Indicates whether the number of messages available in the Enhanced RX FIFO is greater than the watermark defined in ERFCR[ERFWM]. If ERFIER[ERFWMIIE] = 1, this field generates an interrupt. 0b - No such occurrence 1b - Number of messages in FIFO is greater than the watermark
28 ERFDA	Enhanced RX FIFO Data Available Flag Indicates whether there is at least one message stored in the ERX FIFO. If ERFIER[ERFDAIE] = 1, this field generates an interrupt. 0b - No such occurrence 1b - At least one message stored in Enhanced RX FIFO
27 ERFCLR	Enhanced RX FIFO Clear Writing 1 to this field during Freeze mode resets the internal FIFO pointers, but the FIFO content stored in RAM is not changed. Writing to this field outside Freeze mode, or writing 0 to this field, has no effect. 0b - No effect 1b - Clear enhanced RX FIFO content
26-18 —	Reserved
17 ERFE	Enhanced RX FIFO Empty Flag Indicates whether Enhanced RX FIFO is empty. 0b - Not empty 1b - Empty
16 ERFF	Enhanced RX FIFO Full Flag Indicates whether enhanced RX FIFO is full.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Not full 1b - Full
15 —	Reserved
14-6 —	Reserved
5-0 ERFEL	Enhanced RX FIFO Elements Indicates the number of CAN messages stored in the Enhanced RX FIFO.

73.6.2.41 High-Resolution Timestamp (HR_TIME_STAMP0 - HR_TIME_STAMP95)

Offset

For n = 0 to 95:

Register	Offset
HR_TIME_STAMPn	C30h + (n × 4h)

Function

Stores a copy of a 32-bit timer at the start or end of a CAN frame.

HR_TIME_STAMP0 stores the 32-bit timestamp associated with MB0, HR_TIME_STAMP1 stores the 32-bit timestamp associated with MB1, and so on.

Reset does not affect these registers.

NOTE

Do not write to these registers outside Freeze mode.

Table 525. High-Resolution Timestamp register operation

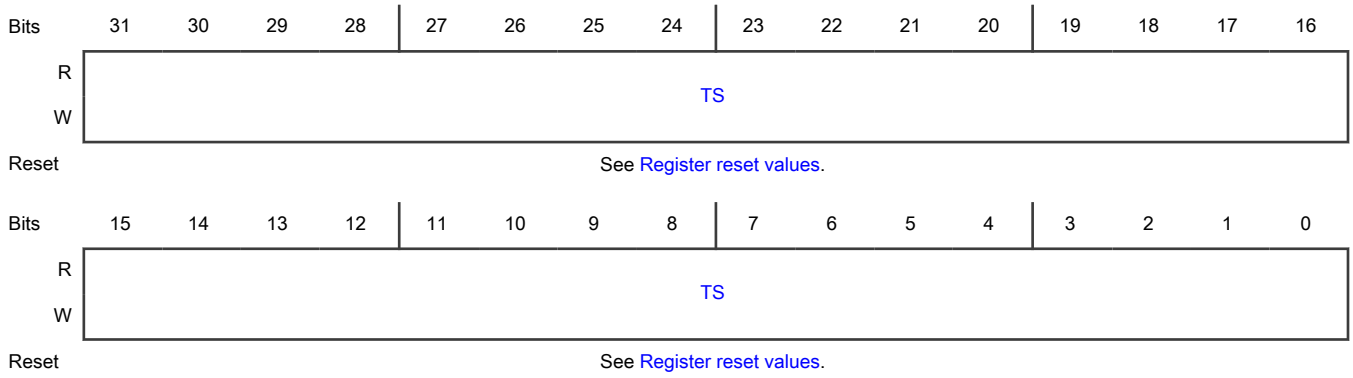
TSTAMPCAP	Captured timebase	Capture point
b00	None	None
b01	32 bits of high-resolution on-chip timer	Seventh bit of the end-of-frame field for transmission and sixth bit of the end-of-frame field for reception.
b10	32 bits of high-resolution on-chip timer	Start of frame
b11	32 bits of high-resolution on-chip timer	Start of frame for classical CAN frame format and res bit for CAN FD frame format

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
CAN_0	HR_TIME_STAMP0–HR_TIME_STAMP95	—
CAN_1	HR_TIME_STAMP0–HR_TIME_STAMP95	—
CAN_2	HR_TIME_STAMP0–HR_TIME_STAMP95	—
CAN_3	HR_TIME_STAMP0–HR_TIME_STAMP63	HR_TIME_STAMP64–HR_TIME_STAMP95
CAN_4	HR_TIME_STAMP0–HR_TIME_STAMP63	HR_TIME_STAMP64–HR_TIME_STAMP95
CAN_5	HR_TIME_STAMP0–HR_TIME_STAMP63	HR_TIME_STAMP64–HR_TIME_STAMP95
CAN_6	HR_TIME_STAMP0–HR_TIME_STAMP63	HR_TIME_STAMP64–HR_TIME_STAMP95
CAN_7	HR_TIME_STAMP0–HR_TIME_STAMP63	HR_TIME_STAMP64–HR_TIME_STAMP95
CAN_8	HR_TIME_STAMP0–HR_TIME_STAMP63	HR_TIME_STAMP64–HR_TIME_STAMP95
CAN_9	HR_TIME_STAMP0–HR_TIME_STAMP63	HR_TIME_STAMP64–HR_TIME_STAMP95
CAN_10	HR_TIME_STAMP0–HR_TIME_STAMP63	HR_TIME_STAMP64–HR_TIME_STAMP95
CAN_11	HR_TIME_STAMP0–HR_TIME_STAMP63	HR_TIME_STAMP64–HR_TIME_STAMP95

Diagram



Register reset values

Register	Reset value
HR_TIME_STAMP0–HR_TIME_STAMP63	CAN_0–CAN_11: undefined
HR_TIME_STAMP64–HR_TIME_STAMP95	CAN_0–CAN_2: undefined CAN_3–CAN_11: Register not supported

Fields

Field	Function
31-0 TS	High-Resolution Timestamp Captures the copy of the timestamp of corresponding message buffer. This field always captures a 32-bit timer value.

73.6.2.42 Enhanced RX FIFO Filter Element (ERFFEL0 - ERFFEL127)

Offset

For n = 0 to 127:

Register	Offset
ERFFELn	3000h + (n × 4h)

Function

Stores the filter elements of the Enhanced RX FIFO.

For standard ID filtering, each ERFFEL register stores one filter element. For extended ID filtering, each pair of ERFFEL registers stores one filter element.

ERFFEL registers can be written only in Freeze mode; otherwise, the module blocks them. Reset does not affect these registers. They are located in RAM and must be explicitly initialized prior to any reception.

NOTE

Each module instance supports a different number of registers.

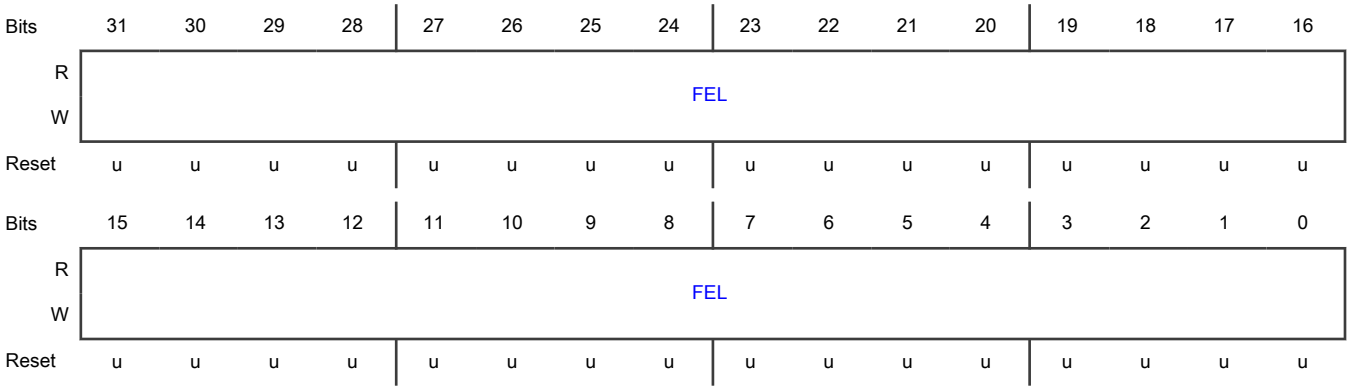
Instance	Register supported	Register not supported
CAN_0	ERFFEL0–ERFFEL127	—
CAN_1	ERFFEL0–ERFFEL127	—
CAN_2	ERFFEL0–ERFFEL127	—
CAN_3	—	ERFFEL0–ERFFEL127
CAN_4	—	ERFFEL0–ERFFEL127
CAN_5	—	ERFFEL0–ERFFEL127
CAN_6	—	ERFFEL0–ERFFEL127
CAN_7	—	ERFFEL0–ERFFEL127
CAN_8	—	ERFFEL0–ERFFEL127

Table continues on the next page...

Table continued from the previous page...

Instance	Register supported	Register not supported
CAN_9	—	ERFFEL0–ERFFEL127
CAN_10	—	ERFFEL0–ERFFEL127
CAN_11	—	ERFFEL0–ERFFEL127

Diagram



Fields

Field	Function
31-0 FEL	Filter Element Bits Stores filter elements. Each filter element is used during the match process. If the matching criteria are met, a message is stored in the Enhanced RX FIFO.

73.6.3 Message buffer structure

The message buffer structure used by FlexCAN is represented in the following figure. Both extended (29-bit identifier) and standard (11-bit identifier) frames used in the CAN specification (Version 2.0 Part B) are represented. Each individual message buffer is 16, 24, 40, or 72 bytes, depending on the quantity of data bytes allocated for the message payload: 8, 16, 32, or 64 data bytes, respectively.

The memory area 80h–67Fh is used by the message buffers. When CAN FD is enabled, the exact address for each message buffer depends on the size of its payload. See FlexCAN memory partition for CAN FD.

Table 526. Message buffer structure example with 64-byte payload

	31	30	29	28	27	24	23	22	21	20	19	18	17	16	15	8	7	0
0h	EDL	BRS	ESI		CODE			SRR	IDE	RTR	DLC				TIMESTAMP			
4h	PRIO				ID (standard/extended)										ID (extended)			
8h	Data byte 0								Data byte 1						Data byte 2		Data byte 3	
Ch	Data byte 4								Data byte 5						Data byte 6		Data byte 7	

Table continues on the next page...

Table 526. Message buffer structure example with 64-byte payload (continued)

10h	Data byte 8	Data byte 9	Data byte 10	Data byte 11
14h	Data byte 12	Data byte 13	Data byte 14	Data byte 15
18h	Data byte 16	Data byte 17	Data byte 18	Data byte 19
1Ch	Data byte 20	Data byte 21	Data byte 22	Data byte 23
20h	Data byte 24	Data byte 25	Data byte 26	Data byte 27
24h	Data byte 28	Data byte 29	Data byte 30	Data byte 31
28h	Data byte 32	Data byte 33	Data byte 34	Data byte 35
2Ch	Data byte 36	Data byte 37	Data byte 38	Data byte 39
30h	Data byte 40	Data byte 41	Data byte 42	Data byte 43
34h	Data byte 44	Data byte 45	Data byte 46	Data byte 47
38h	Data byte 48	Data byte 49	Data byte 50	Data byte 51
3Ch	Data byte 52	Data byte 53	Data byte 54	Data byte 55
40h	Data byte 56	Data byte 57	Data byte 58	Data byte 59
44h	Data byte 60	Data byte 61	Data byte 62	Data byte 63

		= Unimplemented or reserved

Table 527. Field descriptions

Mnemonic	Field	Description
EDL	Extended Data Length	Distinguishes between CAN format and CAN FD format frames. EDL must not be 1 for message buffers configured to RANSWER with code field 1010b (see Table 528).
BRS	Bit Rate Switch	Defines whether the bit rate is switched inside a CAN FD format frame.
ESI	Error State Indicator	Indicates whether the transmitting node is error-active or error-passive.
CODE	Message Buffer Code	Can be accessed (read or write) by the CPU and by the FlexCAN module itself, as part of the message buffer matching and arbitration process. The encoding is shown in Table 528 and Table 529. See Functional description.

Table 528. Message buffer code for RX buffers

CODE description	RX code BEFORE RX new frame	SRV[1]	RX code AFTER successful reception[2]	RRS[3]	Comment
0000b: INACTIVE. Message buffer is not active.	INACTIVE	—	—	—	Message buffer does not participate in the matching process.
0100b: EMPTY. Message buffer is active and empty.	EMPTY	—	FULL	—	When a frame is received successfully (after Move-in), CODE is automatically updated to FULL.

Table continues on the next page...

Table 528. Message buffer code for RX buffers (continued)

CODE description	RX code BEFORE RX new frame	SRV[1]	RX code AFTER successful reception[2]	RRS[3]	Comment
0010b: FULL. Message buffer is full.	FULL	Yes	FULL	—	The act of reading the Control and Status word followed by unlocking the message buffer (SRV) does not make CODE return to EMPTY. It remains FULL. If a new frame is moved to the message buffer after the message buffer is serviced, the code remains FULL. See Matching process for matching details related to FULL code.
		No	OVERRUN	—	If the message buffer is FULL and a new frame is moved to this message buffer before the CPU services it, CODE is automatically updated to OVERRUN. See Matching process for details about overrun behavior.
0110b: OVERRUN. Message buffer is being overwritten into a full buffer.	OVERRUN	Yes	FULL	—	If CODE indicates OVERRUN and the CPU has serviced the message buffer, when a new frame is moved to the message buffer, CODE returns to FULL.
		No	OVERRUN	—	If CODE already indicates OVERRUN, and another new frame must be moved, the message buffer is overwritten again, and CODE remains OVERRUN. See Matching process for details about overrun behavior.
1010b: RANSWER[4]. A frame was configured to recognize a Remote	RANSWER	—	TANSWER (1110b)	0	A Remote Answer was configured to recognize a Remote Request

Table continues on the next page...

Table 528. Message buffer code for RX buffers (continued)

CODE description	RX code BEFORE RX new frame	SRV[1]	RX code AFTER successful reception[2]	RRS[3]	Comment
Request frame and transmit a Response frame in return.[5]					frame received. After that, a message buffer is set to transmit a response frame. CODE is automatically changed to TANSWER (1110b). See Matching process for details. If CTRL2[RRS] = 0, transmit a response frame when a remote request frame with the same ID is received.
		—	—	1	This code is ignored during matching and arbitration process. See Matching process for details.
CODE[0] = 1: BUSY. FlexCAN is updating the contents of the message buffer. The CPU must not access the message buffer.	BUSY[6]	—	FULL	—	Indicates that the message buffer is being updated. It automatically becomes 0 and does not interfere with the next CODE.
		—	OVERRUN	—	

1. SRV: Serviced message buffer. Message buffer was read and unlocked by reading TIMER or another message buffer.
2. A frame is considered a successful reception after the frame is moved to a message buffer (move-in process). See Move-in.
3. Remote Request Stored field. See Control 2 (CTRL2).
4. Code 1010b is not considered TX and a message buffer with this code should not be aborted.
5. Code 1010b must be used in message buffers configured in CAN FD format, with EDL = 1.
6. For TX message buffers, the BUSY bit should be ignored upon read, except when MCR[AEN] = 1. If this field is 1, the corresponding message buffer does not participate in the matching process.

Table 529. Message buffer code for TX buffers

CODE Description	TX Code BEFORE TX frame	MB RTR	TX Code AFTER successful transmission	Comment
1000b: INACTIVE. Message buffer is not active.	INACTIVE	—	—	Message buffer does not participate in arbitration process.
1001b: ABORT. Message buffer is aborted.	ABORT	—	—	Message buffer does not participate in arbitration process.
1100b: DATA. Message buffer is a TX data frame (MB RTR must be 0).	DATA	0	INACTIVE	Transmit data frame unconditionally once. After transmission, the message

Table continues on the next page...

Table 529. Message buffer code for TX buffers (continued)

CODE Description	TX Code BEFORE TX frame	MB RTR	TX Code AFTER successful transmission	Comment
				buffer automatically returns to the INACTIVE state.
1100b: REMOTE. Message buffer is a Transmit Remote Request frame (MB RTR must be 1).	REMOTE	1	EMPTY	Transmit remote request frame unconditionally once. After transmission, the message buffer automatically becomes an RX Empty message buffer with the same ID.
1110b: TANSWER. Message buffer is a Transmit Response frame from an incoming Remote Request frame.	TANSWER	—	RANSWER	This intermediate code is automatically written to the message buffer by the CHI as a result of a match to a Remote Request frame. The Remote Response frame is transmitted unconditionally once, then the code automatically returns to RANSWER (1010b). The CPU can also write this code with the same effect. The Remote Response frame can be a data frame or another remote request frame, depending on the value of RTR. See Matching process and Arbitration process for details.

Table 530. RX and TX message buffer field descriptions

Mnemonic	Field	Description
SRR	Substitute Remote Request	Fixed recessive bit, used only in extended format. Write 1 to SRR for transmission (TX Buffers). SRR is stored with the value received on the CAN bus for RX receiving buffers. It can be received as either recessive or dominant. If FlexCAN receives this bit as dominant, it is interpreted as an arbitration loss. 1: Recessive value is compulsory for transmission in extended format frames. 0: Dominant is not a valid value for transmission in extended format frames.
IDE	ID Extended Bit	Identifies whether the frame format is standard or extended. 1: Frame format is extended 0: Frame format is standard

Table continues on the next page...

Table 530. RX and TX message buffer field descriptions (continued)

Mnemonic	Field	Description
RTR	Remote Transmission Request	Affects the behavior of remote frames and is part of the reception filter. See Table 528, Table 529, and CTRL2[RRS]. If FlexCAN transmits this field as 1 (recessive) and receives it as 0 (dominant), it is interpreted as an arbitration loss. If this field is transmitted as 0 (dominant) and it is received as 1 (recessive), FlexCAN treats it as a bit error. If the value received matches the value transmitted, it is considered a successful bit transmission. 1: If message buffer is TX, indicates that the current message buffer may have a Remote Request frame to be transmitted. If the message buffer is RX, incoming remote request frames may be stored. 0: Indicates that the current message buffer has a Data frame to be transmitted. In an RX message buffer, it may be considered in matching processes. **NOTE** When configuring CAN FD frames, this field must be 0.
DLC	Data Length Code	Indicates the length (in bytes) of the RX or TX data, which is located in offset 8h–Fh of the message buffer space (see Table 526). In reception, this field is written by FlexCAN, copied from the DLC field of the received frame. In transmission, this field is written by the CPU and corresponds to the DLC field value of the frame to be transmitted. When RTR = 1, the frame to be transmitted is a remote frame and does not include the data field, regardless of the DLC field (see Table 532).
TIMESTAMP	Free-Running Counter Timestamp	Provides a copy of the Free-Running Timer, captured for TX and RX frames when the beginning of the Identifier field appears on the CAN bus. See Table 531 for Timestamp operation.
PRIO	Local priority	Used only when MCR[LPRIOEN] = 1, and only makes sense for transmit message buffers. These bits are not transmitted. They are appended to the regular ID to define the transmission priority. See Arbitration process.
ID	Frame Identifier	In standard frame format, only the 11 most significant bits (28 to 18) are used for frame identification in both receive and transmit cases. The 18 least significant bits are ignored. In extended frame format, all bits are used for frame identification in both receive and transmit cases.
DATA BYTE 0–63	Data Field	Up to 64 bytes can be used for a data frame, depending on the size of payload selected for the message buffers. For RX frames, the data is stored as it is received from the CAN bus. DATA BYTE (*n*) is valid only if *n* is less than DLC, as shown in Table 532.

Table 531. Timestamp operation

TSTAMPCAP	MBTSBASE	TIMER_SOURCE	Captured timebase	Capture point
b00	bxx	0	CAN_TIMER incremented by CAN bit clock	Second bit of identifier field
b00	bxx	1	CAN_TIMER incremented by on-chip timer clock	Second bit of identifier field
bxx	b00	0	CAN_TIMER incremented by CAN bit clock	Second bit of identifier field
bxx	b00	1	CAN_TIMER incremented by on-chip timer clock	Second bit of identifier field
b01	b01	x	Lower 16 bits of high-resolution on-chip timer	Seventh bit of the end of frame field for transmission and sixth bit of the end of frame field for reception
b01	b10	x	Upper 16 bits of high-resolution on-chip timer	Seventh bit of the end of frame field for transmission and sixth bit of the end of frame field for reception
b10	b01	x	Lower 16 bits of high-resolution on-chip timer	Start of frame
b10	b10	x	Upper 16 bits of high-resolution on-chip timer	Start of frame
b11	b01	x	Lower 16 bits of high-resolution on-chip timer	Start of frame for classical CAN frame format and res bit for CAN FD frame format
b11	b10	x	Upper 16 bits of high-resolution on-chip timer	Start of frame for classical CAN frame format and res bit for CAN FD frame format

Table 532. DATA BYTE validity

DLC	Valid data bytes
0	None
1	DATA BYTE 0
2	DATA BYTE 0–1
3	DATA BYTE 0–2
4	DATA BYTE 0–3
5	DATA BYTE 0–4
6	DATA BYTE 0–5
7	DATA BYTE 0–6
8	DATA BYTE 0–7

Table continues on the next page...

Table 532. DATA BYTE validity (continued)

DLC	Valid data bytes
9	DATA BYTE 0–11
10	DATA BYTE 0–15
11	DATA BYTE 0–19
12	DATA BYTE 0–23
13	DATA BYTE 0–31
14	DATA BYTE 0–47
15	DATA BYTE 0–63

73.6.4 FlexCAN memory partition for CAN FD

When CAN FD is enabled, FlexCAN RAM can be partitioned into blocks of 512 bytes each. Each block can accommodate a number of message buffers depending on the configuration provided by FDCTRL[MBDSR*n*] as shown in Table 533.

Table 533. RAM partition

RAM block	Number of MBs with 8 bytes (default range)	Size control field in FDCTRL	Number of MBs of different sizes, per block
0	0 to 31	MBDSR0	MBDSR0 = 00, 32 MBs with 8-byte payload MBDSR0 = 01, 21 MBs with 16-byte payload MBDSR0 = 10, 12 MBs with 32-byte payload MBDSR0 = 11, 7 MBs with 64-byte payload
1	32 to 63	MBDSR1	MBDSR1 = 00, 32 MBs with 8-byte payload MBDSR1 = 01, 21 MBs with 16-byte payload MBDSR1 = 10, 12 MBs with 32-byte payload MBDSR1 = 11, 7 MBs with 64-byte payload
2	64 to 95	MBDSR2	MBDSR2 = 00, 32 MBs with 8-byte payload MBDSR2 = 01, 21 MBs with 16-byte payload MBDSR2 = 10, 12 MBs with 32-byte payload MBDSR2 = 11, 7 MBs with 64-byte payload

Payload sizes of 16, 32, or 64 bytes may be configured in some or all of RAM blocks. In those cases, the total number of MBs and their respective number order may differ from the default configuration of 8 bytes. Consider an example where:

- Block0 is configured to an 8-byte payload
- Block1 is configured to a 16-byte payload
- Block2 is configured to 32-byte payload

In this case, Table 534 indicates how the message buffers are arranged in RAM.

Table 534. RAM partition example

RAM block	Payload size	Number of MBs in the RAM block	Message buffer range
0	FDCTRL[MBDSR0] = 00, 8-byte payload	32	0 to 31
1	FDCTRL[MBDSR1] = 01, 16-byte payload	21	32 to 52
2	FDCTRL[MBDSR2] = 10, 32-byte payload	12	53 to 64

73.6.5 FlexCAN message buffer memory map

The FlexCAN memory buffers are allocated in memory according to the tables below.

Table 535. 8-byte message buffers

Address offset (hex)	MBDSR = b00 8-byte payload
0080	MB0
0090	MB1
00A0	MB2
00B0	MB3
00C0	MB4
00D0	MB5
00E0	MB6
00F0	MB7
0100	MB8
0110	MB9
0120	MB10
0130	MB11
0140	MB12
0150	MB13
0160	MB14
0170	MB15
0180	MB16
0190	MB17
01A0	MB18
01B0	MB19
01C0	MB20

Table continues on the next page...

Table 535. 8-byte message buffers (continued)

Address offset (hex)	MBDSR = b00 8-byte payload
01D0	MB21
01E0	MB22
01F0	MB23
0200	MB24
0210	MB25
0220	MB26
0230	MB27
0240	MB28
0250	MB29
0260	MB30
0270	MB31
0280	MB32
0290	MB33
02A0	MB34
02B0	MB35
02C0	MB36
02D0	MB37
02E0	MB38
02F0	MB39
0300	MB40
0310	MB41
0320	MB42
0330	MB43
0340	MB44
0350	MB45
0360	MB46
0370	MB47
0380	MB48
0390	MB49
03A0	MB50
03B0	MB51

Table continues on the next page...

Table 535. 8-byte message buffers (continued)

Address offset (hex)	MBDSR = b00 8-byte payload
03C0	MB52
03D0	MB53
03E0	MB54
03F0	MB55
0400	MB56
0410	MB57
0420	MB58
0430	MB59
0440	MB60
0450	MB61
0460	MB62
0470	MB63
0480	MB64
0490	MB65
04A0	MB66
04B0	MB67
04C0	MB68
04D0	MB69
04E0	MB70
04F0	MB71
0500	MB72
0510	MB73
0520	MB74
0530	MB75
0540	MB76
0550	MB77
0560	MB78
0570	MB79
0580	MB80
0590	MB81
05A0	MB82

Table continues on the next page...

Table 535. 8-byte message buffers (continued)

Address offset (hex)	MBDSR = b00 8-byte payload
05B0	MB83
05C0	MB84
05D0	MB85
05E0	MB86
05F0	MB87
0600	MB88
0610	MB89
0620	MB90
0630	MB91
0640	MB92
0650	MB93
0660	MB94
0670	MB95

Table 536. 16-byte message buffers

Address offset (hex)	MBDSR = b01 16-byte payload
0080	MB0
0098	MB1
00B0	MB2
00C8	MB3
00E0	MB4
00F8	MB5
0110	MB6
0128	MB7
0140	MB8
0158	MB9
0170	MB10
0188	MB11
01A0	MB12
01B8	MB13

Table continues on the next page...

Table 536. 16-byte message buffers (continued)

Address offset (hex)	MBDSR = b01 16-byte payload
01D0	MB14
01E8	MB15
0200	MB16
0218	MB17
0230	MB18
0248	MB19
0260	MB20
0280	MB21
0298	MB22
02B0	MB23
02C8	MB24
02E0	MB25
02F8	MB26
0310	MB27
0328	MB28
0340	MB29
0358	MB30
0370	MB31
0388	MB32
03A0	MB33
03B8	MB34
03D0	MB35
03E8	MB36
0400	MB37
0418	MB38
0430	MB39
0448	MB40
0460	MB41
0480	MB42
0498	MB43
04B0	MB44

Table continues on the next page...

Table 536. 16-byte message buffers (continued)

Address offset (hex)	MBDSR = b01 16-byte payload
04C8	MB45
04E0	MB46
04F8	MB47
0510	MB48
0528	MB49
0540	MB50
0558	MB51
0570	MB52
0588	MB53
05A0	MB54
05B8	MB55
05D0	MB56
05E8	MB57
0600	MB58
0618	MB59
0630	MB60
0648	MB61
0660	MB62

Table 537. 32-byte message buffers

Address offset (hex)	MBDSR = b10 32-byte payload
0080	MB0
00A8	MB1
00D0	MB2
00F8	MB3
0120	MB4
0148	MB5
0170	MB6
0198	MB7
01C0	MB8

Table continues on the next page...

Table 537. 32-byte message buffers (continued)

Address offset (hex)	MBDSR = b10 32-byte payload
01E8	MB9
0210	MB10
0238	MB11
0280	MB12
02A8	MB13
02D0	MB14
02F8	MB15
0320	MB16
0348	MB17
0370	MB18
0398	MB19
03C0	MB20
03E8	MB21
0410	MB22
0438	MB23
0480	MB24
04A8	MB25
04D0	MB26
04F8	MB27
0520	MB28
0548	MB29
0570	MB30
0598	MB31
05C0	MB32
05E8	MB33
0610	MB34
0638	MB35

Table 538. 64-byte message buffers

Address offset (hex)	MBDSR = b11 64-byte payload
0080	MB0
00C8	MB1
0110	MB2
0158	MB3
01A0	MB4
01E8	MB5
0230	MB6
0280	MB7
02C8	MB8
0310	MB9
0358	MB10
03A0	MB11
03E8	MB12
0430	MB13
0480	MB14
04C8	MB15
0510	MB16
0558	MB17
05A0	MB18
05E8	MB19
0630	MB20

73.6.6 Legacy RX FIFO structure

When MCR[RFEN] = 1, the memory area 80h–DCh (which is normally occupied by MBs 0–5) is used by the reception Legacy FIFO engine.

The region 80h–8Ch contains the output of the Legacy RX FIFO, which the CPU must read as a message buffer. This output contains the oldest message that has been received but not yet read. The region 90h–DCh is reserved for internal use of the Legacy RX FIFO engine.

An additional memory area, which starts at E0h and may extend up to 2DCh (normally occupied by MBs 6–37) depending on the value of CTRL2[RFFN], contains the ID filter table (configurable from 8 to 128 table elements) that specifies filtering criteria for accepting frames into the Legacy RX FIFO.

Out of reset, the ID filter table flexible memory area defaults to E0h and extends only to FCh, which corresponds to MBs 6 to 7 for RFFN = 0, for backward compatibility with previous versions of FlexCAN.

The following shows the Legacy RX FIFO data structure.

Table 539. Legacy RX FIFO structure

	31 28	24 23	22	21	20	19 18 17 16	15 8	7 0
80h	IDHIT		SRR	IDE	RTR	DLC	TIMESTAMP	
84h		ID standard				ID extended		
88h	Data byte 0		Data byte 1				Data byte 2	Data byte 3
8Ch	Data byte 4		Data byte 5				Data byte 6	Data byte 7
90h–DCh	Reserved							
E0h	ID filter table element 0							
E4h	ID filter table element 1							
E8h–2D4h	ID filter table elements 2 to 125							
2D8h	ID filter table element 126							
2DCh	ID filter table element 127							
	= Unimplemented or reserved							

Each ID filter table element occupies an entire 32-bit word. One, two, or four Identifier Acceptance Filters (IDAFs) can compound each element, depending on MCR[IDAM]. The following tables show the IDAF indexing.

Table 540 shows the three different formats of the ID table elements. All elements of the table must have the same format. See Legacy RX FIFO for more information.

Table 540. ID table structure

Format	31	30	29 24	23 16	15	14	13 8	7 1	0
A	RTR	IDE	RXIDA (standard = 29–19, extended = 29–1)						
B	RTR	IDE	RXIDB_0 (standard = 29–19, extended = 29–16)		RTR	IDE	RXIDB_1 (standard = 13–3, extended = 13–0)		
C	RXIDC_0 (std and ext = 31–24)		RXIDC_1 (std and ext = 23–16)		RXIDC_2 (std and ext = 15–8)		RXIDC_3 (std and ext = 7–0)		
	= Unimplemented or Reserved								

Table 541. Field descriptions

Mnemonic	Field	Description
RTR	Remote Frame	Specifies whether remote frames are accepted into the Legacy FIFO if they match the target ID. 1: Remote frames can be accepted and data frames are rejected.

Table continues on the next page...

Table 541. Field descriptions (continued)

Mnemonic	Field	Description
		0: Remote frames are rejected and data frames can be accepted.
IDE	Extended Frame	Specifies whether extended or standard frames are accepted into the Legacy FIFO if they match the target ID. 1: Extended frames can be accepted and standard frames are rejected. 0: Extended frames are rejected and standard frames can be accepted.
RXIDA	RX Frame Identifier (Format A)	Specifies an ID to be used as acceptance criteria for the Legacy FIFO. In the standard frame format, only the 11 most significant bits (29 to 19) are used for frame identification. In the extended frame format, all bits are used.
RXIDB_0, RXIDB_1	RX Frame Identifier (Format B)	Specifies an ID to be used as acceptance criteria for the Legacy FIFO. In the standard frame format, the 11 most significant bits (a full standard ID) (29 to 19 and 13 to 3) are used for frame identification. In the extended frame format, all 14 bits of the field are compared to the 14 most significant bits of the received ID.
RXIDC_0, RXIDC_1, RXIDC_2, RXIDC_3	RX Frame Identifier (Format C)	Specifies an ID to be used as acceptance criteria for the Legacy FIFO. In both standard and extended frame formats, all 8 bits of the field are compared to the 8 most significant bits of the received ID.
IDHIT	Identifier Acceptance Filter Hit Indicator	Indicates which identifier acceptance filter the received message in the output of the Legacy RX FIFO hit. See Legacy RX FIFO for more information.

73.6.7 Enhanced RX FIFO structure

When ERFCR[ERFEN] = 1, the Enhanced RX FIFO is enabled. The region 2000h–204Ch contains the output of the Enhanced RX FIFO, which the CPU must read as a message buffer. This output contains the oldest message that has been received but not yet read.

Table 542. Enhanced RX FIFO structure

	31	30	29	28 … 24	23	22	21	20	19	18	17	16	15 … 8	7	6 … 0
2000h	EDL	BRS	ESI	Reserved		SRR	IDE	RTR	DLC				TIMESTAMP LEGACY		
2004h	Reserved			ID (standard/extended)									ID (extended)		
2008h	Data byte 0					Data byte 1							Data byte 2		Data byte 3
200Ch	Data byte 4					Data byte 5							Data byte 6		Data byte 7
2010h	Data byte 8					Data byte 9							Data byte 10		Data byte 11
2014h	Data byte 12					Data byte 13							Data byte 14		Data byte 15
2018h	Data byte 16					Data byte 17							Data byte 18		Data byte 19
201Ch	Data byte 20					Data byte 21							Data byte 22		Data byte 23
2020h	Data byte 24					Data byte 25							Data byte 26		Data byte 27
2024h	Data byte 28					Data byte 29							Data byte 30		Data byte 31
2028h	Data byte 32					Data byte 33							Data byte 34		Data byte 35

Table continues on the next page...

Table 542. Enhanced RX FIFO structure (continued)

202Ch	Data byte 36	Data byte 37	Data byte 38	Data byte 39
2030h	Data byte 40	Data byte 41	Data byte 42	Data byte 43
2034h	Data byte 44	Data byte 45	Data byte 46	Data byte 47
2038h	Data byte 48	Data byte 49	Data byte 50	Data byte 51
203Ch	Data byte 52	Data byte 53	Data byte 54	Data byte 55
2040h	Data byte 56	Data byte 57	Data byte 58	Data byte 59
2044h	Data byte 60	Data byte 61	Data byte 62	Data byte 63
IH_OFF	Reserved			ID HIT
TS_OFF	HR TIMESTAMP			
2050h	19 Enhanced FIFO Elements (Reserved)			
...				
263Ch				

NOTE

ID HIT offset and high-resolution timestamp offset change dynamically according to data length code (DLC) as shown in Table 543.

Table 543. ID HIT offset and high-resolution timestamp offset

Data Length Code (DLC)	ID HIT offset (IH_OFF)	High-resolution timestamp offset (TS_OFF)
0	2008h	200Ch
1–4	200Ch	2010h
5–8	2010h	2014h
9	2014h	2018h
10	2018h	201Ch
11	201Ch	2020h
12	2020h	2024h
13	2028h	202Ch
14	2038h	203Ch
15	2048h	204Ch

Table 544. Field descriptions

Mnemonic	Field	Description
EDL	Extended Data Length	Distinguishes between classical CAN format and CAN FD format frames. 0: Classical CAN frame format 1: CAN FD frame format

Table continues on the next page...

Table 544. Field descriptions (continued)

Mnemonic	Field	Description
BRS	Bit Rate Switch	Defines whether the bit rate is switched inside a CAN FD format frame. 0: Bit rate is not switched in a CAN FD frame. 1: Bit rate is switched in a CAN FD frame.
ESI	Error State Indicator	Indicates whether the transmitting node is error-active or error-passive. This field is meaningful only if EDL = 1. 0: Error-active 1: Error-passive
SRR	Substitute Remote Request	Fixed recessive bit, used only in extended format. Transmitting nodes always send it as recessive and receiving nodes can receive it as either recessive or dominant. If FlexCAN receives this bit as dominant, it is interpreted as an arbitration loss.
IDE	ID Extended Bit	Identifies whether the frame format is standard or extended. 0: Standard 1: Extended
RTR	Remote Frame	Identifies whether the current frame is a data frame or a remote request. 0: Data frame 1: Remote request
DLC	Data Length Code	Defines the number of bytes in the data field of a CAN frame (Data byte 0 to Data byte 63). When RTR = 1, the frame is a remote request and does not include the data field, regardless of the DLC field. See Table 532 for more details.
LEGACY TIMESTAMP	16-bit Timestamp	Provides a copy of the Free-Running Timer, captured during the CAN frame. See Table 531 for details about legacy timestamp operation.
ID	Frame Identifier	In base frame format, only the 11 most significant bits are used for frame identification. The 18 least significant bits are ignored. In extended frame format, all bits are used for frame identification.
DATA BYTE 0–63	Data Field	Up to 64 bytes can be stored in the data field.
IDHIT	Identifier Acceptance Filter Hit Indicator	Indicates which Enhanced RX FIFO Filter Element (ERFFEL0 - ERFFEL127) the received message in the output of the Enhanced RX FIFO hit. For each filter region, standard-ID filter space, and extended-ID filter space, there is an independent index starting from zero. Table 545 shows how FlexCAN writes IDHIT according to each filter element.
HR TIMESTAMP	High-resolution Timestamp	32-bit timebase captured during the CAN frame. When CTRL2[TSTAMPCAP] is not zero, a 32-bit timebase is captured from a dedicated on-chip timer which operates in free-running mode. See CTRL2[TSTAMPCAP] for details about capture point configuration of the high-resolution timestamp.

Table 545. IDHIT for Enhanced RX FIFO

Enhanced RX FIFO filter element - ERFFEL	IDHIT value	Filter element type
ERFFEL0	0	Extended-ID
ERFFEL1	1	Extended-ID
. . .	. . .	Extended-ID
ERFFEL(m-1)	m-1	Extended-ID
ERFFEL(m)	0	Standard-ID
ERFFEL(m+1)	1	Standard-ID
. . .	. . .	Standard-ID
ERFFEL(2n-m+1)	2x(n-m)+1	Standard-ID

NOTE

Where m = NEXIF and n = NFE. If NEXIF = 0, only standard-ID filter elements exist. If NEXIF > NFE, only extended-ID filter elements exist.

73.7 Glossary

Active message buffer — A message buffer is active if it can participate in the current matching or arbitration process.

Bus interface unit — FlexCAN submodule responsible for the interface to the CPU.

Bus off — See CAN Specification.[13]

CAN — Controller Area Network, a serial communication protocol defined in CAN Specification[13] and ISO International Standard.[14]

CHI — Controller-host interface, a FlexCAN submodule responsible for message buffer matching and arbitration algorithms.

CRC — Cyclic redundancy check

Dominant bit — A dominant bit wins the arbitration on the CAN bus. It is transmitted as 0.

Doze mode — A system low-power mode where the CPU bus remains active and a global Doze mode request is sent to all peripherals asking them to enter low power mode.

Error active — See CAN Specification.[13]

Error frame — See CAN Specification.[13]

Error passive — See CAN Specification.[13]

[13] Controller Area Network - CAN Specification Version 2.0 Part A, Part B, Robert Bosch GmbH, 1991.
[14] ISO International Standard - ISO 11898 First Edition 1993 Road Vehicles - Interchange of Digital Information - Controller Area Network (CAN) for high-speed Communication.

Form error	See CAN Specification.[13]
Hard reset	A reset coming from an external pin and/or following power-on. It resets everything.
Idle	See CAN Specification.[13]
Information processing time	See CAN Specification.[13]
Intermission	See CAN Specification.[13]
Matching elements	Data used in the matching process, such as ID, filter, mask, etc.
MB	See Message buffer.
Message buffer	Internal FlexCAN data structure containing bytes received from, or to be transmitted to, the CAN line, as well as information about this data.
Overload frame	See CAN Specification.[13]
Phase buffer segment	See CAN Specification.[13]
Recessive bit	A recessive bit loses the arbitration on the CAN bus. It is transmitted as 1.
Sclock	Serial clock. Obtained by dividing the clock feeding the CAN engine (oscillator or bus clock) by a prescaler factor. The Sclock period defines the time quantum for CAN protocol timing.
Soft reset	Global reset typically used by peripherals to reinitialize some of its registers, but not all of them.
Stop mode	A system low-power mode in which all chip clocks are stopped for maximum power savings.
Stuffing error	See CAN Specification.[13]
Time quantum	This is equal to the Sclock period. It is the minimum time period used to compose the CAN protocol bit timing.

Chapter 74
Synchronous Audio Interface (SAI)

74.1 Chip-specific SAI information

74.1.1 SAI instances and configuration

Table 546. SAI instances

Instance	S32K322/S32K342/S32K341/S32K314/S32K324/S32K344/S32K358/ S32K356/S32K348/S32K338/S32K328/S32K388/S32K389	S32K310/ S32K311/S32K312
SAI_0	Yes	No
SAI_1	Yes	No

Table 547. SAI configuration

Chip	Instances	Synchronous operation between instances	DMA support	Data Rate/ channel	No. of Channels	Frame Synchronization	Master Clock Frequency	FIFO size(TX/RX)/ Channel	No. of words/ frame
S32K322/ S32K342/ S32K341/ S32K314/ S32K324/ S32K344/ S32K358/ S32K356/ S32K348/ S32K338/ S32K328/ S32K388/ S32K389	SAI_0 SAI_1	No	Yes	12.288 Mbps	4 1	I2S, AC97, and CODEC/DSP interfaces	24.576 external audio	8X32 bit	8-16 words

NOTE

The SAI_BCLK (SAI_TX_BCLK, SAI_RX_BCLK), SAI_SYNC (SAI_TX_SYNC, SAI_RX_SYNC), and SAI_DATA (SAI_TX_DATA, SAI_RX_DATA) pins are shared between each module instance's (SAI_0, SAI_1) receiver and transmitter. See the IOMUX file attached to this document for details. At any specific time, either the SAI receiver or the SAI transmitter should be active or they should operate synchronously (only one active at a time). The SAI transmitter and SAI receiver should not be configured to operate simultaneously with inconsistent configurations.

NOTE

Timing specification of SAI master and SAI slave may limit maximum frequency.

74.2 Overview

SAI provides an interface that supports full-duplex serial digital audio interfaces with frame synchronization formats such as I^2S, AC97, TDM, and Codec/DSP interfaces.

74.2.1 Block diagram

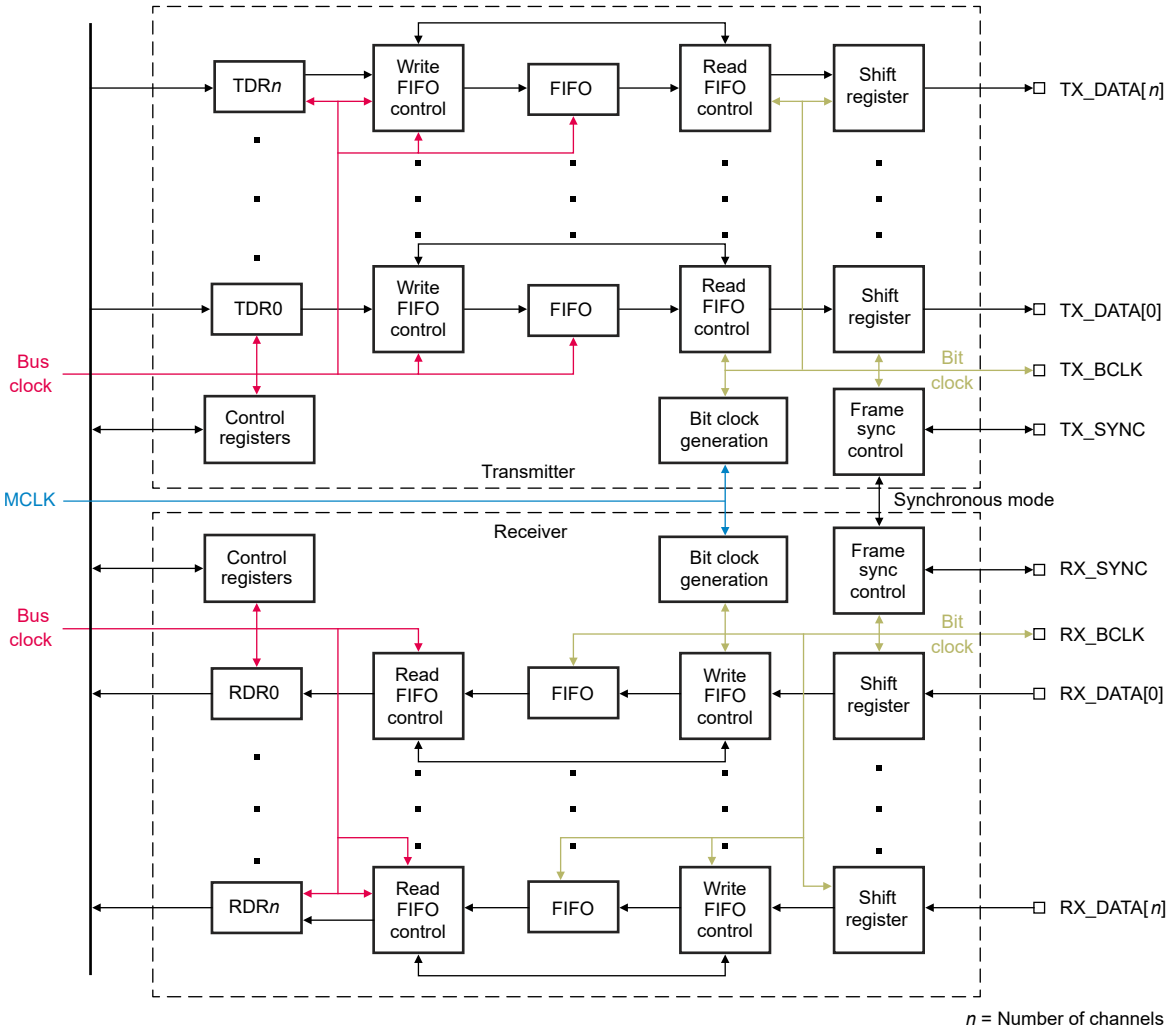


Figure 414. Block diagram

74.2.2 Features

Features can vary among chips and among SAI modules on the same chip. See the chip-specific SAI information at the beginning of this chapter.

- Transmitter with independent bit clock and frame synchronization supporting 4 data lines
- Receiver with independent bit clock and frame synchronization supporting 4 data lines
- Maximum frame size of 16 words per data line
- Word length of 8 to 32 bits
- Word length configured separately for the first word and remaining words in a frame
- Asynchronous 8 × 32-bit FIFO for each transmit and receive data line supports:
  - Graceful restart after FIFO error.
  - Automatic restart after FIFO error without software intervention.
  - 8-bit and 16-bit data packing into each 32-bit FIFO word.

— Multiple-data-line FIFOs combining into single-data-line FIFO.

74.3 Functional description

This section provides a complete functional description of SAI.

74.3.1 Modes of operation

- Run mode
- Debug mode

74.3.1.1 Run mode

In Run mode, the SAI transmitter and receiver operate normally.

74.3.1.2 Debug mode

You can configure either or both the SAI transmitter and receiver to continue operating in Debug mode:

- Write 1 to TCSR[DBGE] to configure the transmitter to run in Debug mode.
- Write 1 to RCSR[DBGE] to configure the receiver to run in Debug mode.

When TCSR[DBGE] and RCSR[DBGE] are 0 and the chip enters Debug mode, SAI is disabled after completing the current transmit or receive frame. Debug mode does not affect the transmitter and receiver bit clocks.

74.3.2 Synchronous modes

The SAI transmitter and receiver can operate synchronously to each other. You can configure the SAI transmitter and receiver to operate with a synchronous bit clock and frame sync (see TCR2[SYNC] and RCR2[SYNC]).

If both the transmitter and receiver use the transmitter bit clock and frame sync:

- Configure the transmitter for asynchronous operation and the receiver for synchronous operation.
- Enable the receiver in Synchronous mode only after configuring both the transmitter and receiver.
- Enable the transmitter last and disable the transmitter first.

If both the transmitter and receiver use the receiver bit clock and frame sync:

- Configure the receiver for asynchronous operation and the transmitter for synchronous operation.
- Enable the transmitter in Synchronous mode only after configuring both the receiver and transmitter.
- Enable the receiver last and disable the receiver first.

When operating in Synchronous mode, the transmitter and receiver share only the bit clock, frame sync, and transmitter and receiver enable. Otherwise, the transmitter and receiver operate independently, although the configuration register settings must be consistent across both the transmitter and receiver.

74.3.3 Frame sync configuration

When enabled, SAI continuously transmits and receives frames of data. Each frame consists of a fixed number of words, with each word consisting of a fixed number of bits. Within each frame, you can mask any word, causing the receiver to ignore that word and the transmitter to 3-state during that word.

The frame sync signal indicates the start of a frame. A valid frame sync requires the detection of a rising edge (if active high) or falling edge (if active low). The transmitter or receiver cannot be busy with a previous frame. SAI ignores a valid frame sync (in Target mode) or does not generate one (in Controller mode) for the first four bit-clock cycles after enabling the transmitter or receiver.

You can configure the transmitter and receiver frame sync independently by using any of the following options:

- Generate externally or internally
- Configure to be active high or active low
- Assert with the first bit in the frame or one bit early
- Assert for a duration between one bit-clock cycle and the first word length
- Configure the frame length from 1 to 16 words
- Configure the word length from 8 to 32 bits, with separate configurations for the length of the first word and that of the remaining words
- Transmit or receive words MSB first or LSB first

You cannot change these configuration options after enabling the SAI transmitter or receiver.

74.3.4 Data FIFO

Each transmit and receive channel includes a 8 × 32-bit FIFO. Use Transmit Data (TDR0 - TDR3) and Receive Data (RDR0 - RDR3) to access FIFO data.

74.3.4.1 Data alignment

Data in the FIFO can be aligned anywhere within the 32-bit-wide register via the First Bit Shifted (FBT) configuration field. This field selects the bit index (between 31 and 0) of the first bit shifted.

Examples of supported data alignment and the required FBT configuration are shown in Figure 415 for LSB-first configurations and Figure 416 for MSB-first configurations.

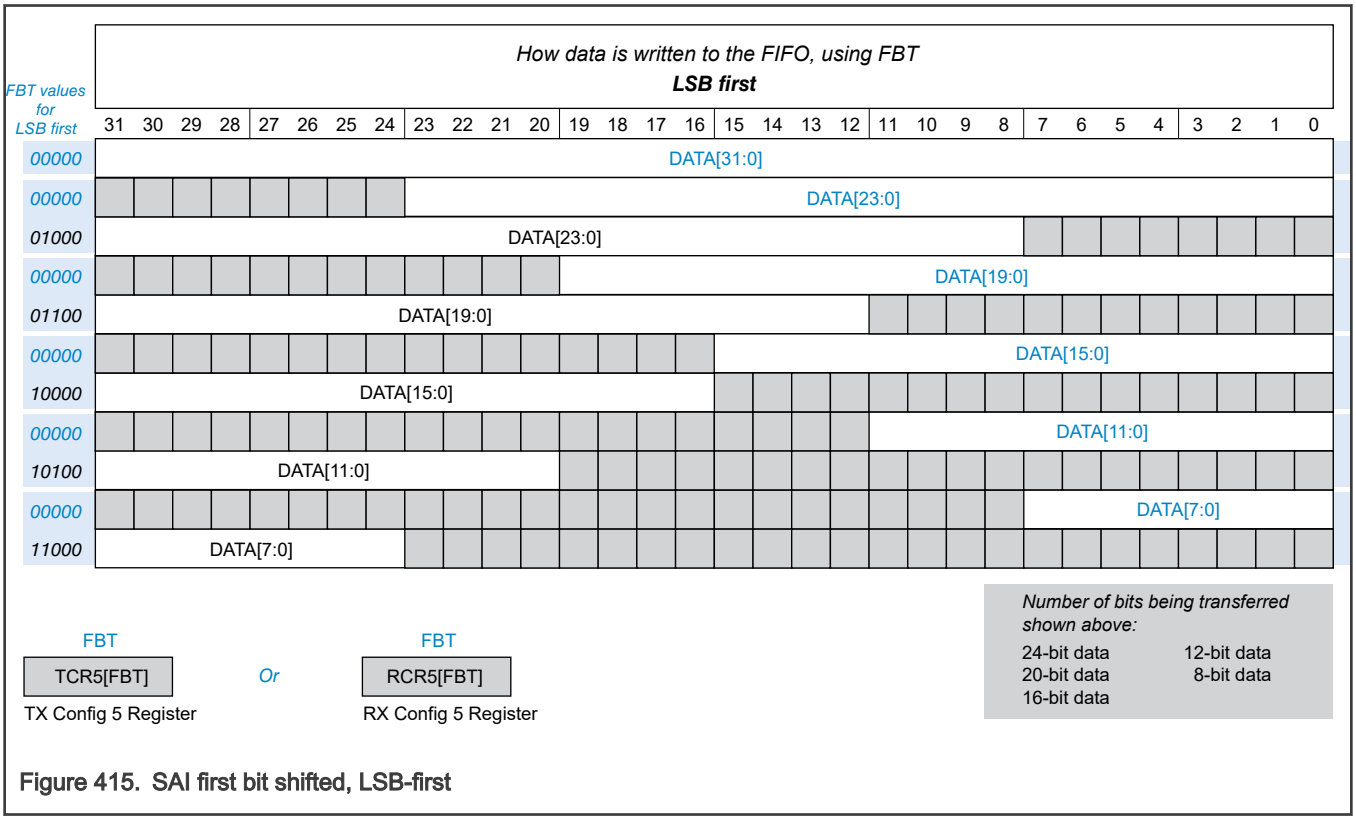


Figure 415. SAI first bit shifted, LSB-first

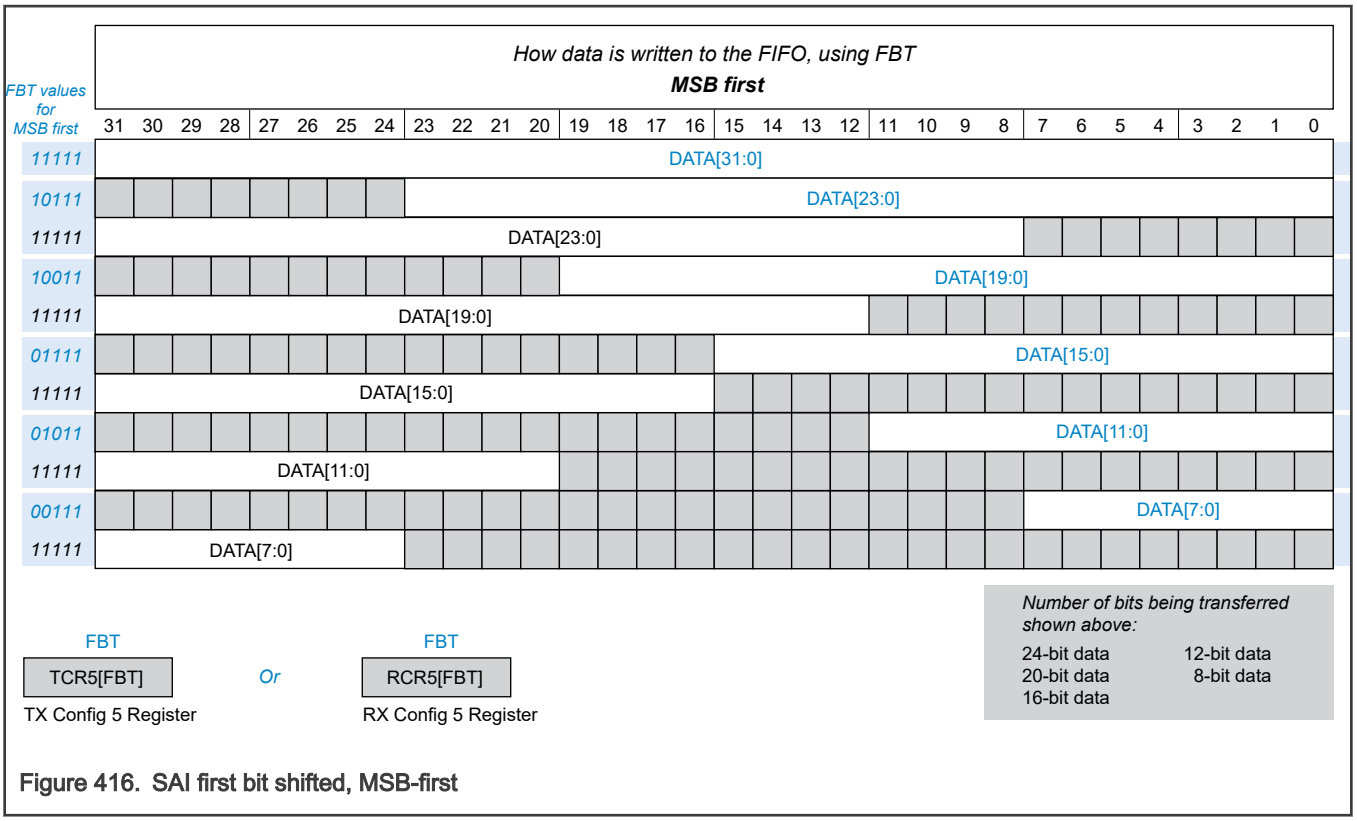


Figure 416. SAI first bit shifted, MSB-first

74.3.4.2 FIFO pointers

When writing to one of the Transmit Data (TDR0 - TDR3) registers, the write FIFO pointer of the corresponding Transmit FIFO register (TFR*n*[WFP]) increments after each valid write. SAI supports 8-bit, 16-bit, and 32-bit writes to TDR*n*, and the FIFO pointer increments after each individual write. Use only 8-bit writes when transmitting up to 8-bit data, and use only 16-bit writes when transmitting up to 16-bit data:

- If the transmit FIFO is full, SAI ignores writes to TDR*n*.
- If the transmit FIFO is empty, write to TDR*n* at least three bit-clock cycles before the start of the next unmasked word. This write avoids a FIFO underrun.
- Before enabling the transmitter, initialize the transmit FIFO with data. The transmitter starts a new frame after enabling the transmitter. If no data is in the FIFO, the transmitter immediately generates an error.

When reading one of the Receive Data (RDR0 - RDR3) registers, the read FIFO pointer of the corresponding Receive FIFO register (RFR*n*[RFP]) increments after each valid read. SAI supports 8-bit, 16-bit, and 32-bit reads from RDR*n*, and the FIFO pointer increments after each individual read. Use only 8-bit reads when receiving up to 8-bit data, and use only 16-bit reads when receiving up to 16-bit data:

- If the receive FIFO is empty, SAI ignores reads to RDR*n*.
- If the receive FIFO is full, read RDR*n* at least three bit-clock cycles before the end of an unmasked word. This read avoids a FIFO overrun.

74.3.4.3 FIFO packing

Using FIFO packing, you can store multiple 8-bit or 16-bit data words in one 32-bit FIFO word for the transmitter or receiver. You could emulate this feature by adjusting the number of bits per word and number of words per frame. FIFO packing, however, does not require even multiples of words per frame, and it fully supports word masking.

When FIFO packing is enabled, the FIFO pointers only increment when the full 32-bit FIFO word has been written (transmit) or read (receive). In this way, FIFO packing supports scenarios where different words within each frame are stored in different areas of memory.

Using 16-bit FIFO packing for transmitting, the transmit shift register loads at the start of each frame and after every second unmasked transmit word. The first transmitted word is taken from the 16-bit word at byte offset 0h. (The first bit is selected by TCR5[FBT], and you must configure it within this 16-bit word.) The second transmitted word is taken from the 16-bit word at byte offset 2h. (The first bit is selected by TCR5[FBT][3:0].) After the 16-bit word has been transmitted, the transmitter transmits logic zeroes until the start of the next word.

Using 16-bit FIFO packing for receiving, the receive shift register is stored after every second unmasked received word. If there is an odd number of unmasked-received words in each frame, the receive register is also stored, at the end of each frame. The first received word is stored in the 16-bit word at byte offset 0h. (The first bit is selected by RCR5[FBT], and you must configure it within this 16-bit word.) The second received word is stored in the 16-bit word at byte offset 2h. (The first bit is selected by RCR5[FBT][3:0].) The receiver ignores received data until the start of the next word after the 16-bit word has been received.

8-bit FIFO packing is similar to 16-bit packing, except four words are loaded or stored into each 32-bit FIFO word. The first word is stored in byte offset 0h, the second word in byte offset 1h, and so on. You must configure TCR5[FBT], RCR5[FBT], or both within byte offset 0h.

74.3.4.4 FIFO Combining mode

This mode enables separate FIFOs for multiple data channels to be used as a single FIFO for software accesses, as a single data channel, or both. The enabled data channels must be contiguous and data channel 0 must be enabled when using FIFO Combining mode.

You can combine FIFOs for software access by writing to transmit FIFO registers and reading from receive FIFO registers. Doing so enables a DMA controller or software to read or write multiple FIFOs without incrementing the address that is accessed. After FIFO Combining mode is enabled (TCR4[FCOMB] > 00b), the first software access to a FIFO register accesses the first enabled channel FIFO. The second access to a FIFO register accesses the second enabled channel FIFO. This process continues until software accesses the last enabled channel FIFO and the pointer resets to the first enabled channel FIFO. To reset the pointer manually, you can reset the FIFOs or disable FIFO combining on software accesses.

Combining FIFOs for transmit data channels enables one data channel to use the FIFOs of all enabled channel FIFOs. In this case, the data output is identical on each enabled data channel. The transmit shift registers for all enabled data channels load at the start of each frame. The registers also load upon every *n*th unmasked word, where *n* is the number of enabled data channels. The first word transmitted loads from the first enabled channel FIFO; the second word transmitted loads from the second enabled channel FIFO. This loading continues until the end of the frame.

Combining FIFOs for receive data channels enables one data channel to use the FIFOs of all enabled channel FIFOs. In this case, the received data from channel 0 is stored in each enabled data channel. The receive shift registers for all enabled data channels are stored after every *n*th unmasked word, where *n* is the number of enabled data channels. The first word received is stored in the first enabled channel FIFO; the second word received is stored in the second enabled channel FIFO. This storage continues until the end of the frame.

NOTE

The first word in each frame is always stored in the first enabled data channel. NXP recommends that the number of unmasked words in each frame be evenly divisible by the number of enabled data channels.

Combining FIFOs for data channels loads or stores each channel FIFO at the same time. As a result, FIFO error conditions are only checked every *n*th word, where *n* is the number of enabled data channels. If any enabled data channel meets the warning flag or request flag conditions, SAI asserts the FIFO warning and request flags.

74.3.5 Word mask register

The SAI transmitter and receiver each have a word mask register (Transmit Mask (TMR) and Receive Mask (RMR)) that you can use to mask any word in the frame. The word mask register is double buffered. You can update it before the end of each frame to mask a particular word in the next frame.

TMR causes the transmit data pin to be 3-stated for the length of each selected word; the transmit FIFO is not read for masked words.

RMR causes the received data for each selected word to be discarded and not written to the receive FIFO.

74.3.6 Clocking

- Audio clock
- Bit clock
- Bus clock

74.3.6.1 Audio clock

The audio clock (MCLK) generates the bit clock when you configure the receiver or transmitter for an internally generated bit clock. The transmitter and receiver can independently select between the bus clock and up to three audio clocks to generate the bit clock.

The audio clock generation and selection is chip-specific. See chip-specific clocking information about how the audio clocks are generated.

74.3.6.2 Bit clock

The SAI transmitter and receiver support asynchronous free-running bit clocks that an audio clock can generate internally or an external source can supply. SAI can also have synchronous bit clock and frame sync operation between the receiver and transmitter, or between multiple SAI peripherals. The transmitter and receiver configuration affects the bit clock and frame sync in the following ways:

- If you configure both the transmitter and receiver for asynchronous operation, each uses its own bit clock and frame sync.
- If you configure the transmitter for asynchronous operation and the receiver for synchronous operation, both use the transmitter bit clock and frame sync.
- If you configure the receiver for asynchronous operation and the transmitter for synchronous operation, both use the receiver bit clock and frame sync.

Your choice of synchronous or asynchronous operation selects the bit clock and frame sync used.

Externally generated bit clocks must be:

- Enabled before the SAI transmitter or receiver is enabled.
- Disabled after the SAI transmitter or receiver is disabled and completes its current frames.

In asynchronous operation, a SAI transmitter or receiver can use a bit clock externally generated by a SAI module instance that is disabled in Stop mode. In this case, disable the transmitter or receiver before entering Stop mode. This issue does not apply when the transmitter or receiver is in synchronous operation because all synchronous SAI modules are enabled and disabled simultaneously.

74.3.6.3 Bus clock

The control and configuration registers use the bus clock to generate synchronous interrupts and DMA requests.

NOTE

Although no minimum bus clock frequency is specified, the frequency must be fast enough (relative to the bit clock frequency) to serve the FIFOs. You must meet this requirement without generating a transmitter FIFO underrun or receiver FIFO overflow condition.

74.3.7 Reset

SAI is asynchronously reset on system reset. SAI has a Software reset and a FIFO reset.

74.3.7.1 Software reset

The SAI transmitter includes a software reset that resets all transmitter internal logic, including the bit clock generation, status flags, and FIFO pointers. The SAI receiver includes a software reset that resets all receiver internal logic, including bit clock generation, status flags, and FIFO pointers.

These software resets do not reset the configuration registers. The software resets remain asserted until you clear them via software.

74.3.7.2 FIFO reset

The SAI transmitter includes a FIFO reset that synchronizes the FIFO write pointer to the value of the FIFO read pointer. This FIFO reset empties the FIFO contents. Use this reset after TCSR[FEF] becomes 1, and before SAI reinitializes the FIFO and TCSR[FEF] becomes 0. SAI asserts the FIFO reset for one cycle only.

The SAI transmitter can also reset the FIFO of individual data channels by writing 1 to the appropriate bit in TCR3[CFR]. Use this reset only when the corresponding bit in TCR3[TCE] is 0.

The SAI receiver includes a FIFO reset that synchronizes the FIFO read pointer to the value of the FIFO write pointer. This FIFO reset empties the FIFO contents. Use this reset after RCSR[FEF] becomes 1 and SAI reads any remaining data from the FIFO, and before RCSR[FEF] becomes 0. SAI asserts the FIFO reset for one cycle only.

The SAI receiver can also reset the FIFO of individual data channels by writing 1 to the appropriate bit in RCR3[RCE]. Use this reset only when the corresponding bit in RCR3[RCE] is 0.

74.3.8 Interrupts and DMA requests

In SAI, the transmitter and receiver generate separate interrupts and DMA requests, but use the same status flags. SAI only generates asynchronous interrupts when the system clock is gated but the corresponding BCLK signal is active.

74.3.8.1 FIFO request flag

SAI sets the transmit FIFO request flag (TCSR[FRF]) when the number of entries in any enabled transmit FIFO is less than or equal to the transmit FIFO watermark configuration (TCR1[TFW]). SAI clears this flag when the number of entries in each enabled transmit FIFO is greater than TCR1[TFW].

SAI sets the receive FIFO request flag (RCSR[FRF]) when the number of entries in any enabled receive FIFO is greater than the receive FIFO watermark configuration (RCR1[RFW]). SAI clears this flag when the number of entries in each enabled receive FIFO is less than or equal to RCR1[RFW].

The FIFO request flag can generate an interrupt when TCSR[FRIE] = 1. It can generate a DMA request when TCSR[FRDE] = 1.

74.3.8.2 FIFO warning flag

SAI sets the transmit FIFO warning flag (TCSR[FWF]) when the number of entries in any of the enabled transmit FIFOs is empty. SAI clears this flag when the number of entries in each enabled transmit FIFO is not empty.

SAI sets the receive FIFO warning flag (RCSR[FWF]) when the number of entries in any of the enabled receive FIFOs is full. SAI clears this flag when the number of entries in each enabled receive FIFO is not full.

The FIFO warning flag can generate an interrupt when TCSR[FWIE] = 1. The flag can generate a DMA request when TCSR[FWDE] = 1.

74.3.8.3 FIFO error flag

SAI sets the transmit FIFO error flag (TCSR[FEF]) when any enabled transmit FIFOs underrun. After the error flag is set, all enabled transmit channels transmit zero data before SAI clears TCSR[FEF].

When you write 1 to TCR4[FCONT], the FIFO continues transmitting data following an underrun without software intervention. To ensure that data transmits in the correct order, the transmitter continues from the same word number in the frame that caused the FIFO to underrun. It continues only after new data is written to the transmit FIFO. In this case, clear TCSR[FEF] without reinitializing the transmit FIFOs.

SAI sets the receive FIFO error flag (RCSR[FEF]) when any enabled receive FIFO overflows. After the flag is set, all enabled receive channels discard their received data until SAI clears RCSR[FEF] and the next receive frame starts. Empty all enabled receive FIFOs before SAI clears RCSR[FEF].

When you write 1 to RCR4[FCONT], the FIFO continues receiving data following an overflow without software intervention. To ensure that data is received in the correct order, the receiver continues from the same word number in the frame that caused the FIFO to overflow. It continues only after data has been read from the receive FIFO. In this case, clear RCSR[FEF] without emptying the receive FIFOs.

TCSR[FEF] and RSCR[FEF] can only generate an interrupt.

74.3.8.4 Sync error flag

SAI sets a sync error flag (TCSR[SEF] or RCSR[SEF]) when both of these conditions are true:

- The frame sync is generated externally.
- The external frame sync asserts when the transmitter or receiver is busy with the previous frame.

SAI ignores the external frame sync assertion and sets the sync error flag. When this flag is set, the transmitter or receiver continues checking for frame sync assertion when idle or at the end of each frame.

TCSR[SEF] and RCSR[SEF] can only generate an interrupt.

74.3.8.5 Word start flag

SAI sets the word start flag (TCSR[WSF]) at the start of the second bit-clock cycle for the selected word. You can select this word via TCR3[WDFL].

TCSR[WSF] can only generate an interrupt.

74.4 External signals

Name	Function	I/O
TX_BCLK	Transmit bit clock: the bit clock is an input when externally generated and an output when internally generated.	I/O
TX_SYNC	Transmit frame sync: the frame sync is an input sampled synchronously by the bit clock when externally generated. It is an output generated synchronously by the bit clock when internally generated.	I/O
TX_DATA[3:0]	Transmit data: the bit clock synchronously generates the transmit data; this signal is 3-stated when not transmitting a word.	O
RX_BCLK	Receive bit clock: the bit clock is an input when externally generated and an output when internally generated.	I/O
RX_SYNC	Receive frame sync: the frame sync is an input sampled synchronously by the bit clock when externally generated. It is an output generated synchronously by the bit clock when internally generated.	I/O

Table continues on the next page...

Table continued from the previous page...

Name	Function	I/O
RX_DATA[3:0]	Receive data: the bit clock synchronously samples the receive data.	I

74.5 Initialization

To initialize SAI:

1. Enable the clock to SAI.
2. Reset the internal transmitter logic and receiver logic by writing 1 to TCSR[SR] and RCSR[SR], respectively.

74.6 Memory map and register definition

A read or write access to an address from offset 100h and above results in a bus error.

74.6.1 SAI register descriptions

SAI provides an interface that supports full-duplex serial digital audio interfaces with frame synchronization formats.

74.6.1.1 SAI memory map

SAI_0 base address: 4036_C000h

SAI_1 base address: 404D_C000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID (VERID)	32	R	0301_0000h
4h	Parameter (PARAM)	32	R	See section
8h	Transmit Control (TCSR)	32	RW	0000_0000h
Ch	Transmit Configuration 1 (TCR1)	32	RW	0000_0000h
10h	Transmit Configuration 2 (TCR2)	32	RW	0000_0000h
14h	Transmit Configuration 3 (TCR3)	32	RW	0000_0000h
18h	Transmit Configuration 4 (TCR4)	32	RW	0000_0000h
1Ch	Transmit Configuration 5 (TCR5)	32	RW	0000_0000h
20h - 2Ch	Transmit Data (TDR0 - TDR3)	32	RW	See section
40h - 4Ch	Transmit FIFO (TFR0 - TFR3)	32	R	See section
60h	Transmit Mask (TMR)	32	RW	0000_0000h
88h	Receive Control (RCSR)	32	RW	0000_0000h
8Ch	Receive Configuration 1 (RCR1)	32	RW	0000_0000h
90h	Receive Configuration 2 (RCR2)	32	RW	0000_0000h
94h	Receive Configuration 3 (RCR3)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
98h	Receive Configuration 4 (RCR4)	32	RW	0000_0000h
9Ch	Receive Configuration 5 (RCR5)	32	RW	0000_0000h
A0h - ACh	Receive Data (RDR0 - RDR3)	32	R	See section
C0h - CCh	Receive FIFO (RFR0 - RFR3)	32	R	See section
E0h	Receive Mask (RMR)	32	RW	0000_0000h

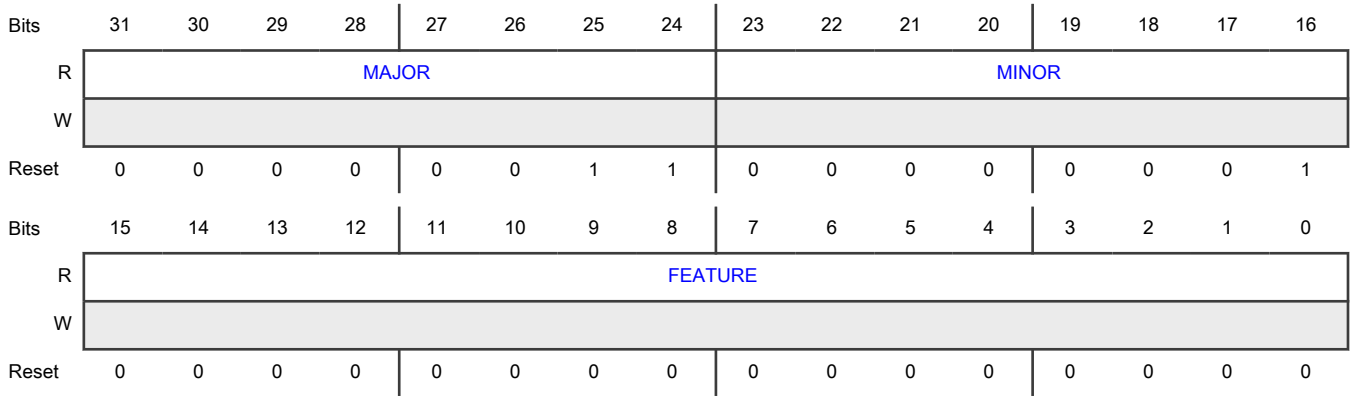
74.6.1.2 Version ID (VERID)

Offset

Register	Offset
VERID	0h

Function
Contains version numbers for the module design and feature set.

Diagram



Fields

Field	Function
31-24 MAJOR	Major Version Number Indicates the major version number for the specification.
23-16 MINOR	Minor Version Number Indicates the minor version number for the specification.

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-0 FEATURE	Feature Specification Number Indicates the feature set number. 0000_0000_0000_0000b - Standard feature set

74.6.1.3 Parameter (PARAM)

Offset

Register	Offset
PARAM	4h

Function

Contains parameter values implemented in the module.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												FRAME			
W																
Reset	See Register reset values.															

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				FIFO				0				DATALINE			
W																
Reset	See Register reset values.															

Register reset values

Register	Reset value
PARAM	SAI_0: 0004_0304h SAI_1: 0004_0301h

Fields

Field	Function
31-20 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
19-16 FRAME	Frame Size Describes the maximum number of words per frame, which is 2^FRAME.
15-12 —	Reserved
11-8 FIFO	FIFO Size Describes the number of words in each FIFO, which is 2^FIFO.
7-4 —	Reserved
3-0 DATALINE	Number of Data Lines Contains the number of data lines implemented.

74.6.1.4 Transmit Control (TCSR)

Offset

Register	Offset
TCSR	8h

Function

Contains transmitter enable fields including resets, error and interrupt enable fields, and error flag fields.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TE	Reserved	DBGE	BCE	0		0	SR	0			WSF	SEF	FEF	FWF	FRF
W							FR					W1C	W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0			WSIE	SEIE	FEIE	FWIE	FRIE	0			0			FWDE	FRDE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 TE	Transmitter Enable Enables the transmitter. When you write 0 to this field, the transmitter remains enabled, and this field remains 1 until the end of the current frame. 0b - Disable 1b - Enable (or transmitter has been disabled and has not yet reached the end of the frame)
30 —	Reserved Only write zero to this field.
29 DBGE	Debug Enable Enables transmitter operation in Debug mode, which does not affect the transmit bit clock. If you write 0 to this field, the transmitter operation is disabled after completing the current frame. 0b - Disable 1b - Enable
28 BCE	Bit Clock Enable Enables the transmit bit clock, separately from TCSR[TE]. This field automatically becomes 1 when TCSR[TE] becomes 1. When you write 0 to this field, the transmit bit clock remains enabled, and the field remains 1 until the end of the current frame. 0b - Disable 1b - Enable
27-26 —	Reserved
25 FR	FIFO Reset Empties the FIFO and sets the FIFO read and write pointers to the same value, which may or may not be zero. **NOTE** The FIFO reset is asserted for one cycle only. Reading this field always returns zero. SAI resets the FIFO pointers when the transmitter is disabled or the FIFO error flag is set. 0b - No effect 1b - FIFO reset
24 SR	Software Reset Resets the internal transmitter logic, including the FIFO read and write pointers. Software-visible registers are not affected, except for the status registers.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE The software reset remains asserted until software writes 0 to the field. 0b - No effect 1b - Software reset
23-21 —	Reserved
20 WSF	Word Start Flag Indicates whether the start of the configured word has been detected. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not detected 1b - Detected When writing 0b - No effect 1b - Clear the flag
19 SEF	Sync Error Flag Indicates whether an error in the externally generated frame sync has been detected. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not detected 1b - Detected When writing 0b - No effect 1b - Clear the flag
18 FEF	FIFO Error Flag Indicates whether an enabled transmit FIFO has underrun. **NOTE** This field behaves differently for register reads and writes. When reading

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Not detected 1b - Detected When writing 0b - No effect 1b - Clear the flag
17 FWF	FIFO Warning Flag Indicates whether an enabled transmit FIFO is empty. 0b - Not empty 1b - Empty
16 FRF	FIFO Request Flag Indicates whether the number of words in an enabled transmit channel FIFO is less than or equal to the transmit FIFO watermark. 0b - Watermark not reached 1b - Watermark reached
15-13 —	Reserved
12 WSIE	Word Start Interrupt Enable Enables word start interrupts. 0b - Disable 1b - Enable
11 SEIE	Sync Error Interrupt Enable Enables sync error interrupts. 0b - Disable 1b - Enable
10 FEIE	FIFO Error Interrupt Enable Enables FIFO error interrupts. 0b - Disable 1b - Enable
9 FWIE	FIFO Warning Interrupt Enable Enables FIFO warning interrupts. 0b - Disable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Enable
8 FRIE	FIFO Request Interrupt Enable Enables FIFO request interrupts. 0b - Disable 1b - Enable
7-5 —	Reserved
4-2 —	Reserved
1 FWDE	FIFO Warning DMA Enable Enables the DMA warning. 0b - Disable 1b - Enable
0 FRDE	FIFO Request DMA Enable Enables DMA requests. 0b - Disable 1b - Enable

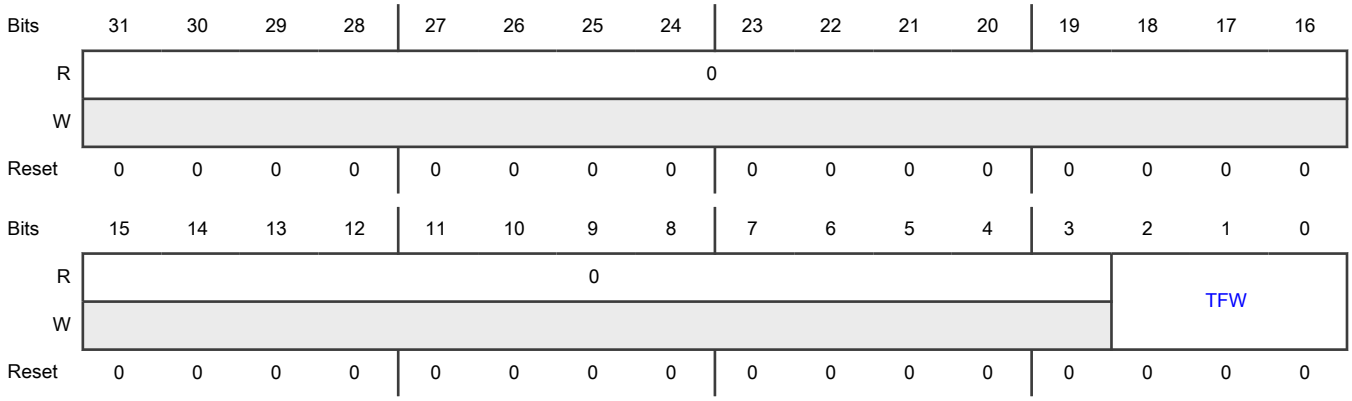
74.6.1.5 Transmit Configuration 1 (TCR1)

Offset

Register	Offset
TCR1	Ch

Function
Configures the watermark level for all enabled transmit channels.

Diagram



Fields

Field	Function
31-3 —	Reserved
2-0 TFW	Transmit FIFO Watermark Indicates the number of 32-bit FIFO words in the watermark level of the transmit FIFO. 000b - 0 001b - 1 010b-110b - (TFW value) 111b - 7

74.6.1.6 Transmit Configuration 2 (TCR2)

Offset

Register	Offset
TCR2	10h

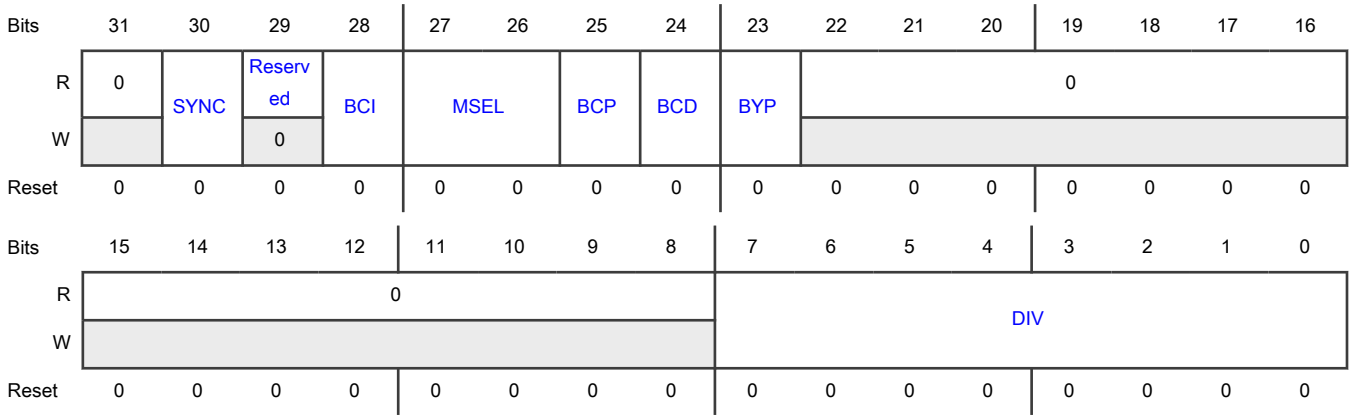
Function

Contains the SYNC mode and clock setting fields.

NOTE

Do not alter this register when TCSR[TE] is 1.

Diagram



Fields

Field	Function
31 —	Reserved
30 SYNC	Synchronous Mode Configures Asynchronous and Synchronous modes of operation. When configured for Synchronous mode, you must configure the receiver for asynchronous operations. 0b - Asynchronous mode 1b - Synchronous with receiver
29 —	Reserved
28 BCI	Bit Clock Input Enables the internal logic to be clocked as if the bit clock is generated externally. When this field is 1 while using an internally generated bit clock in Synchronous or Asynchronous mode, the bit clock used by the transmitter is delayed by the pad output delay. (The pad input clocks the transmitter as if the clock is externally generated.) This setting decreases the data input setup time, but increases the data output valid time. Use the Target mode timing from the data sheet for the transmitter when this field is 1. In Synchronous mode, this field allows the transmitter to use the Target mode timing from the data sheet, while the receiver uses the Controller mode timing. This field has no effect when configured for an externally generated bit clock. **NOTE** When BCI = 1, both the input buffer and output buffer must be enabled for the BCLK pad. 0b - Disable 1b - Enable
27-26	MCLK Select

Table continues on the next page...

Table continued from the previous page...

Field	Function
MSEL	Selects the audio clock option used to generate an internally generated bit clock. This field has no effect when SAI is configured for an externally generated bit clock. **NOTE** Depending on the chip, some audio clock options might not be available. See the chip-specific information for the meaning of each option. 00b - Bus clock 01b - Audio clock (MCLK) option 1 10b - Audio clock (MCLK) option 2 11b - Audio clock (MCLK) option 3
25 BCP	Bit Clock Polarity Configures the polarity of the bit clock. If you write 0 to this field, the bit clock becomes active high with drive outputs on rising edge and sample inputs on falling edge. If you write 1 to this field, the bit clock becomes active low with drive outputs on falling edge and sample inputs on rising edge. 0b - Active high 1b - Active low
24 BCD	Bit Clock Direction Configures the direction of the bit clock. 0b - Generate externally in Target mode 1b - Generate internally in Controller mode
23 BYP	Bit Clock Bypass Bypasses the bit clock divider. When bypassed, the internal bit clock is divide-by-one of the audio clock. When not bypassed, the internal bit clock is generated from the bit clock divider. 0b - Disable 1b - Enable
22-8 —	Reserved
7-0 DIV	Bit Clock Divide Determines the value by which SAI divides down the audio clock to generate the bit clock (when configured for an internal bit clock). The division value is (DIV + 1) × 2.

74.6.1.7 Transmit Configuration 3 (TCR3)

Offset

Register	Offset
TCR3	14h

Function

Contains the transmit channel settings.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				0				0				TCE			
W					CFR											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												WDFL			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved
27-24 CFR	Channel FIFO Reset Resets the FIFO pointers for a specific channel. Reading this field always returns zero. You must reset FIFO pointers only when a channel is disabled or the FIFO error flag is set. The width of this field = the number of transmit channels (call it *n*). For example, if this field is 2 bits wide, bit position 24 refers to the transmit channel 1 FIFO pointer and bit position 25 refers to the transmit channel 2 FIFO pointer. Writing 1 to bit 24 resets the transmit channel 1 FIFO pointer, and writing 1 to bit 25 resets the transmit channel 2 FIFO pointer. Writing 1 to bit *n* resets the transmit channel *n* FIFO pointer. **NOTE** When there is only one channel, you do not need the individual channel FIFO reset (TCR3[CFR]). For a single channel, use the global FIFO reset (TCSR[FR]). • 0b – No effect • 1b – Reset transmit data channel *n* FIFO

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>SAI_0</td><td>TCR3</td><td>—</td></tr>
<tr><td>SAI_1</td><td>—</td><td>TCR3</td></tr>
</table>
</td></tr>
<tr><td>23-20
—</td><td>Reserved</td></tr>
<tr><td>19-16
TCE</td><td>
Transmit Channel Enable
Enables the corresponding data channel for transmit operations. Changing this field takes effect immediately for generating the FIFO request and warning flags. It takes effect at the end of each frame for transmit operations.
The width of this field = the number of transmit channels (call it n). For example, if this field is two bits wide, bit position 16 refers to transmit channel 1 and bit position 17 refers to transmit channel 2. Writing 1 to bit 16 enables transmit channel 1, and setting bit 17 enables transmit channel 2. Writing 1 to bit n enables transmit channel n. For each transmit data channel:

0b – Disable
1b – Enable

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>SAI_0</td><td>TCR3</td><td>—</td></tr>
<tr><td>SAI_1</td><td>TCR3[16]</td><td>TCR3[19–17]</td></tr>
</table>
</td></tr>
<tr><td>15-4
—</td><td>Reserved</td></tr>
<tr><td>3-0
WDFL</td><td>
Word Flag Configuration
Selects the word that sets the word start flag (TCSR[WSF]). The value must be one less than the word number. For example, writing 0 selects the first word in the frame. When you configure this field with a value greater than TCR4[FRSZ], TCSR[WSF] is never set.
</td></tr>
</table>

74.6.1.8 Transmit Configuration 4 (TCR4)

Offset

Register	Offset
TCR4	18h

Function

Contains the transmit fields for FIFO Combine mode, FIFO Packing mode, and frame sync settings.

NOTE

Do not alter this register when TCSR[TE] is 1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0			FCONT	FCOMB		FPACK		0				FRSZ			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0			SYWD					0		CHMOD	MF	FSE	ONDEM	FSP	FSD
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	Reserved
28 FCONT	FIFO Continue on Error Configures when SAI must continue transmitting after a FIFO error is detected. When this field is 0 and a FIFO error occurs, SAI continues from the start of the next frame after the FIFO error flag clears. When this field is 1 and a FIFO error occurs, SAI continues from the same word that caused the FIFO error after the FIFO warning flag clears. 0b - Continue from the start of the next frame 1b - Continue from the same word that caused the FIFO error
27-26 FCOMB	FIFO Combine Mode Enables FIFO Combine mode for specified operations. When FIFO Combine mode is enabled for FIFO writes, software writing to any FIFO data register alternates the write among the enabled data channel FIFOs. For example, if two data channels are enabled:

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>

The first write is performed to the first enabled data channel FIFO.
The second write is performed to the second enabled data channel FIFO.

Resetting the FIFO or disabling FIFO Combine mode for FIFO writes resets the pointer back to the first enabled data channel.

When FIFO Combine mode is enabled for FIFO reads from the transmit shift registers, the transmit data channel output alternates between the enabled data channel FIFOs. For example, if two data channels are enabled:

The first unmasked word is transmitted from the first enabled data channel FIFO.
The second unmasked word is transmitted from the second enabled data channel FIFO.

Because the first word of the frame is always transmitted from the first enabled data channel FIFO, NXP recommends that the number of unmasked words per frame is evenly divisible by the number of enabled data channels.

NOTE
This field is not supported in every instance. The following table includes only supported registers.

<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>SAI_0</td><td>TCR4</td><td>—</td></tr>
<tr><td>SAI_1</td><td>—</td><td>TCR4</td></tr>
</table>
00b - Disable
01b - Enable on FIFO reads (from transmit shift registers)
10b - Enable on FIFO writes (by software)
11b - Enable on FIFO reads (from transmit shift registers) and writes (by software)
</td></tr>
<tr><td>25-24
FPACK</td><td>FIFO Packing Mode

Enables packing of 8-bit data or 16-bit data into each 32-bit FIFO word. If the word size is greater than 8 bits or 16 bits, only the first 8 bits or 16 bits are loaded from the FIFO. The first word in each frame always starts with a new 32-bit FIFO word and the first bit shifted must be configured within the first packed word. When FIFO packing is enabled, the FIFO write pointer only increments when you write the full 32-bit FIFO word.
00b - Disable FIFO packing
01b - Reserved
10b - Enable 8-bit FIFO packing
11b - Enable 16-bit FIFO packing</td></tr>
<tr><td>23-20
—</td><td>Reserved</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
19-16 FRSZ	Frame Size Configures the number of words in each frame. The value must be one less than the number of words in the frame. For example, write 0 for one word per frame. The maximum supported frame size is 16 words.
15-13 —	Reserved
12-8 SYWD	Sync Width Configures the length of the frame sync in number of bit-clock cycles. The value must be one less than the number of bit-clock cycles. For example, write 0 for the frame sync to assert for one bit-clock cycle only. You cannot configure the sync width to be longer than the first word of the frame.
7-6 —	Reserved
5 CHMOD	Channel Mode Specifies the mode of transmit data pins. In TDM mode, transmit data pins are 3-stated when slots are masked or channels are disabled. In Output mode, transmit data pins are never 3-stated, and they output zero when slots are masked or channels are disabled. 0b - TDM mode 1b - Output mode
4 MF	MSB First Configures what is transmitted first: LSB or MSB. 0b - LSB 1b - MSB
3 FSE	Frame Sync Early Determines when frame sync is asserted. 0b - First bit of the frame 1b - One bit before the first bit of the frame
2 ONDEM	On-Demand Mode Determines when the internal frame sync is generated. When this field is 1, and the frame sync is generated internally, a frame sync is only generated after the FIFO warning flag is cleared. 0b - Generated continuously 1b - Generated after the FIFO warning flag is cleared
1 FSP	Frame Sync Polarity Configures the polarity of the frame sync.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Active high 1b - Active low
0 FSD	Frame Sync Direction Configures the direction of the frame sync. 0b - Generated externally in Target mode 1b - Generated internally in Controller mode

74.6.1.9 Transmit Configuration 5 (TCR5)

Offset

Register	Offset
TCR5	1Ch

Function

Contains transmit word width and bit index settings.

NOTE

Do not alter this register when TCSR[TE] is 1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				WNW				0				W0W			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0			FBT					0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
28-24 WNW	Word N Width Configures the number of bits in each word, except for the first one in the frame. The value written must be one less than the number of bits per word. SAI does not support word widths of less than 8 bits. SAI does not support bit settings 0–6 (0b–0_0110b). 0_0111b - 8 0_1000b - 9 0_1001b-1_1110b - (WNW value + 1) 1_1111b - 32
23-21 —	Reserved
20-16 W0W	Word 0 Width Configures the number of bits in the first word of each frame. The value written must be one less than the number of bits in the first word. SAI does not support word widths of less than 8 bits when there is only one word per frame. SAI does not support bit settings 0–6 (0b–0_0110b). 0_0111b - 8 0_1000b - 9 0_1001b-1_1110b - (W0W value + 1) 1_1111b - 32
15-13 —	Reserved
12-8 FBT	First Bit Shifted Configures the bit index for the first bit transmitted for each word in the frame. If configured for MSB-first, the index of the next bit transmitted is one less than the current bit transmitted. If configured for LSB-first, the index of the next bit transmitted is one more than the current bit transmitted. The value written must be greater than or equal to the word width when configured for MSB-first. The value written must be less than or equal to 31-word width when configured for LSB-first. See Data alignment for details. 0_0000b - 0 0_0001b-1_1110b - FBT 1_1111b - 31
7-0 —	Reserved

74.6.1.10 Transmit Data (TDR0 - TDR3)

Offset

Register	Offset
TDR0	20h
TDR1	24h
TDR2	28h
TDR3	2Ch

Function

Contains transmit data.

When the transmit FIFO is not full, writes to this register push the data written into the transmit data FIFO. When the transmit FIFO is full, SAI ignores writes to this register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
SAI_0	TDR0–TDR3	—
SAI_1	TDR0	TDR1–TDR3

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W	TDR															
Reset	See Register reset values.															

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W	TDR															
Reset	See Register reset values.															

Register reset values

Register	Reset value
TDR0	SAI_0,SAI_1: 0000_0000h
TDR1–TDR3	0000_0000h

Fields

Field	Function
31-0 TDR	Transmit Data

74.6.1.11 Transmit FIFO (TFR0 - TFR3)

Offset

Register	Offset
TFR0	40h
TFR1	44h
TFR2	48h
TFR3	4Ch

Function

Contains the transmit FIFO pointers. The MSB of the read and write pointers distinguishes between FIFO full and empty conditions. If the read and write pointers are identical, the FIFO is empty. If the read and write pointers are identical except for the MSB, the FIFO is full.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
SAI_0	TFR0–TFR3	—
SAI_1	TFR0	TFR1–TFR3

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	WCP				0								WFP			
W																
Reset	See Register reset values.															

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												RFP			
W																
Reset	See Register reset values.															

Register reset values

Register	Reset value
TFR0	SAI_0,SAI_1: 0000_0000h
TFR1–TFR3	0000_0000h

Fields

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>31
WCP</td><td>Write Channel Pointer
Indicates whether this data channel is the next FIFO to be written when FIFO Combine mode is enabled for write operations.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>SAI_0</td><td>TFR0–TFR3</td><td>—</td></tr><tr><td>SAI_1</td><td>—</td><td>TFR0</td></tr></table>
0b - No effect
1b - Next FIFO to be written</td></tr>
<tr><td>30-20
—</td><td>Reserved</td></tr>
<tr><td>19-16
WFP</td><td>Write FIFO Pointer
Indicates the FIFO write pointer for the transmit data channel.</td></tr>
<tr><td>15-4
—</td><td>Reserved</td></tr>
<tr><td>3-0
RFP</td><td>Read FIFO Pointer
Indicates the FIFO read pointer for the transmit data channel.</td></tr>
</table>

74.6.1.12 Transmit Mask (TMR)

Offset

Register	Offset
TMR	60h

Function

Contains the mask for the transmit word. This register is double-buffered and updates:

- When TCSR[TE] first becomes 1.
- At the end of each frame.

This setup allows the masked words in each frame to change from frame to frame.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TWM															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-0 TWM	Transmit Word Mask Configures whether the transmit word *n* is masked for the corresponding word in the frame. In this case, being masked means that transmit data pins are 3-stated or driven zero and transmit data is not read from the FIFO. 0000_0000_0000_0000b - Enable 0000_0000_0000_0001b - Mask

74.6.1.13 Receive Control (RCSR)

Offset

Register	Offset
RCSR	88h

Function

Contains receiver enable fields including resets, error and interrupt enable fields, and error flag fields.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RE	Reserved	DBGE	BCE	0		0	SR	0			WSF	SEF	FEF	FWF	FRF
W							FR					W1C	W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0			WSIE	SEIE	FEIE	FWIE	FRIE	0			0			FWDE	FRDE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 RE	Receiver Enable Enables the receiver. When you write 0 to this field, the receiver remains enabled and the field remains 1 until the end of the current frame. 0b - Disable 1b - Enable (or receiver disabled and not yet reached end of frame)
30 —	Reserved Only write zero to this field.
29 DBGE	Debug Enable Enables receiver operation in Debug mode, which does not affect the receive bit clock. 0b - Disable after completing the current frame 1b - Enable
28 BCE	Bit Clock Enable Enables the receive bit clock, separately from RCSR[RE]. This field becomes 1 automatically when RCSR[RE] becomes 1. When you write 0 to this field, the receive bit clock remains enabled, and this field remains 1 until the end of the current frame. 0b - Disable 1b - Enable
27-26 —	Reserved
25 FR	FIFO Reset Empties the FIFO, and sets the FIFO read and write pointers to the same value, which can be zero. Reading this field always returns zero.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE The FIFO reset is asserted for one cycle only. You must reset FIFO pointers only when the receiver is disabled or the FIFO error flag is 1. 0b - No effect 1b - Reset
24 SR	Software Reset Resets the internal receiver logic including the FIFO pointers. Software-visible registers are not affected, except for the status registers. **NOTE** The software reset remains asserted until you clear it. 0b - No effect 1b - Software reset
23-21 —	Reserved
20 WSF	Word Start Flag Indicates whether SAI has detected the start of the configured word. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not detected 1b - Detected When writing 0b - No effect 1b - Clear the flag
19 SEF	Sync Error Flag Indicates whether SAI has detected an error in the externally generated frame sync. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not detected 1b - Detected When writing

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No effect 1b - Clear the flag
18 FEF	FIFO Error Flag Indicates whether an enabled receive FIFO has overflowed. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No error 1b - Receive overflow detected When writing 0b - No effect 1b - Clear the flag
17 FWF	FIFO Warning Flag Indicates whether an enabled receive FIFO is full. 0b - Not full 1b - Full
16 FRF	FIFO Request Flag Indicates whether the number of words in an enabled receive channel FIFO is greater than the receive FIFO watermark. 0b - Watermark not reached 1b - Watermark reached
15-13 —	Reserved
12 WSIE	Word Start Interrupt Enable Enables word start interrupts. 0b - Disable 1b - Enable
11 SEIE	Sync Error Interrupt Enable Enables sync error interrupts. 0b - Disable 1b - Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
10 FEIE	FIFO Error Interrupt Enable Enables FIFO error interrupts. 0b - Disable 1b - Enable
9 FWIE	FIFO Warning Interrupt Enable Enables FIFO warning interrupts. 0b - Disable 1b - Enable
8 FRIE	FIFO Request Interrupt Enable Enables FIFO request interrupts. 0b - Disable 1b - Enable
7-5 —	Reserved
4-2 —	Reserved
1 FWDE	FIFO Warning DMA Enable Enables DMA warnings. 0b - Disable 1b - Enable
0 FRDE	FIFO Request DMA Enable Enables DMA requests. 0b - Disable 1b - Enable

74.6.1.14 Receive Configuration 1 (RCR1)

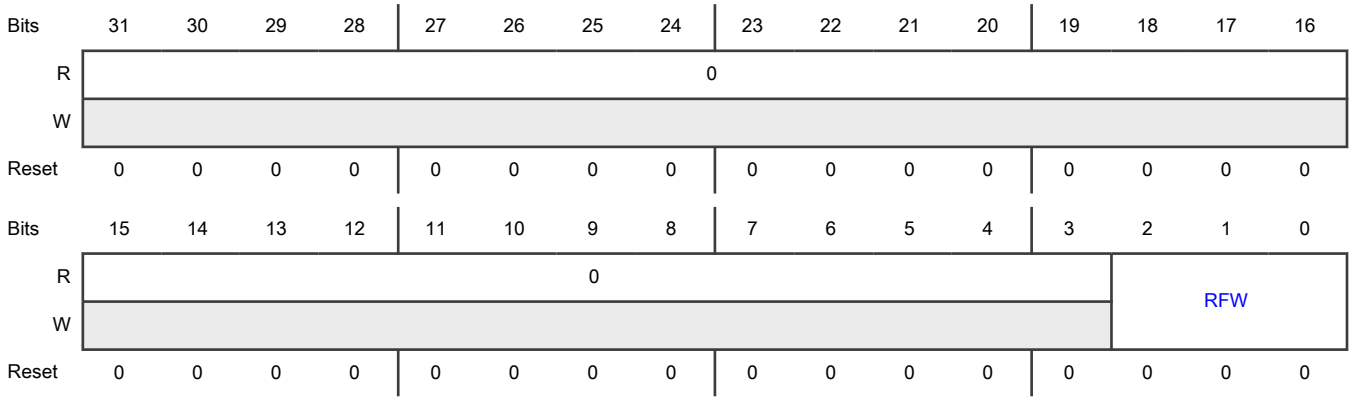
Offset

Register	Offset
RCR1	8Ch

Function

Configures the watermark level for all enabled receiver channels.

Diagram



Fields

Field	Function
31-3 —	Reserved
2-0 RFW	Receive FIFO Watermark Configures the level of the receive FIFO watermark, in 32-bit FIFO words. 000b - 1 001b - 2 010b-110b - (RFW value + 1) 111b - 8

74.6.1.15 Receive Configuration 2 (RCR2)

Offset

Register	Offset
RCR2	90h

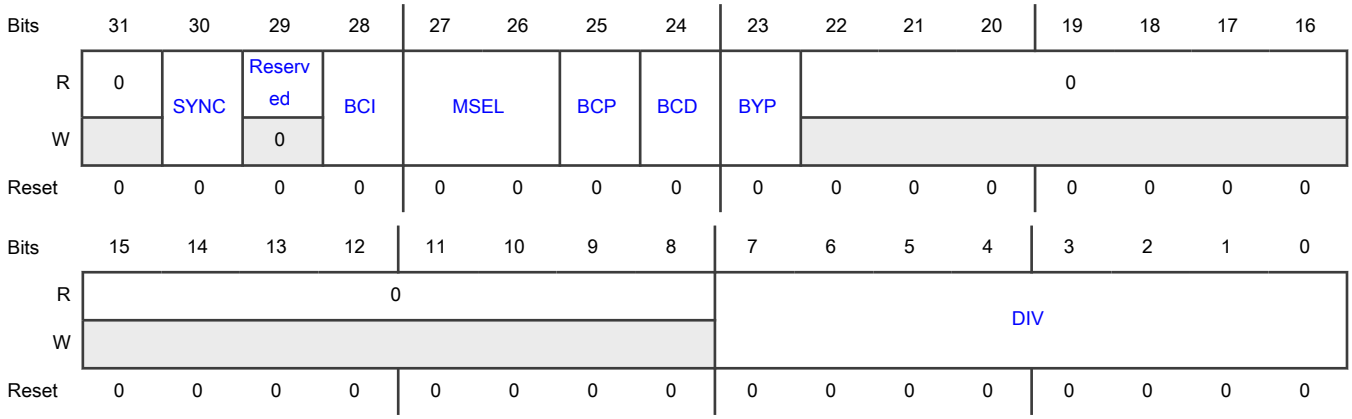
Function

Contains the SYNC mode and clock setting fields.

NOTE

Do not alter this register when RCSR[RE] is 1.

Diagram



Fields

Field	Function
31 —	Reserved
30 SYNC	Synchronous Mode Configures Asynchronous and Synchronous modes of operation. When configured for a Synchronous mode of operation, you must configure the transmitter for asynchronous operation. 0b - Asynchronous mode 1b - Synchronous with transmitter
29 —	Reserved
28 BCI	Bit Clock Input Enables the internal logic to be clocked as if the bit clock is generated externally. When this field is 1 and when using an internally generated bit clock in Synchronous or Asynchronous mode, the bit clock used by the receiver is delayed by the pad output delay. (The pad input clocks the receiver as if the clock is externally generated.) This setting decreases the data input setup time, but increases the data output valid time. Use the Target mode timing from the data sheet for the receiver when this field is 1. In Synchronous mode, this field allows the receiver to use the Target mode timing from the data sheet, while the transmitter uses the Controller mode timing. This field has no effect when configured for an externally generated bit clock. **NOTE** When this field is 1, both the input buffer and output buffer must be enabled for the BCLK pad. 0b - Disable 1b - Enable
27-26	MCLK Select

Table continues on the next page...

Table continued from the previous page...

Field	Function
MSEL	Selects the audio clock option used to generate an internally generated bit clock. This field has no effect when configured for an externally generated bit clock. **NOTE** Some audio clock options are not available for some chips. See the chip-specific information for the availability and chip-specific meaning of each option. 00b - Bus clock 01b - Audio clock (MCLK) option 1 10b - Audio clock (MCLK) option 2 11b - Audio clock (MCLK) option 3
25 BCP	Bit Clock Polarity Configures the polarity of the bit clock. If you write 0 to this field, the bit clock becomes active high with drive outputs on the rising edge and sample inputs on the falling edge. If you write 1 to this field, the bit clock becomes active low with drive outputs on the falling edge and sample inputs on the rising edge. 0b - Active high 1b - Active low
24 BCD	Bit Clock Direction Configures the direction of the bit clock. 0b - Generated externally in Target mode 1b - Generated internally in Controller mode
23 BYP	Bit Clock Bypass Enables the bypass of the bit clock divider. When enabled, the internal bit clock is divide-by-one of the audio clock. When disabled, the internal bit clock is generated from the bit clock divider. 0b - Disable 1b - Enable
22-8 —	Reserved
7-0 DIV	Bit Clock Divide Determines the value by which the audio clock is divided to generate the bit clock, when configured for an internal bit clock. The division value is (DIV + 1) × 2.

74.6.1.16 Receive Configuration 3 (RCR3)

Offset

Register	Offset
RCR3	94h

Function

Contains the receive channel settings.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				0				0				RCE			
W					CFR											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												WDFL			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved
27-24 CFR	Channel FIFO Reset Resets the FIFO pointers for a specific channel. Reading this field always returns zero. Reset FIFO pointers only when a channel is disabled or the FIFO error flag is set. The width of this field = the number of receive channels (call it *n*). For example, if this field is two bits wide, bit position 24 refers to the receive channel 1 FIFO pointer and bit position 25 refers to the receive channel 2 FIFO pointer. Writing 1 to bit 24 resets the receive channel 1 FIFO pointer, and writing 1 to bit 25 enables the receive channel 2 FIFO pointer. Writing 1 to bit *n* resets receive channel *n* FIFO pointer. • 0b – No effect • 1b – Reset receive data channel *n* FIFO **NOTE** This field is not supported in every instance. The following table includes only supported registers.

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>SAI_0</td><td>RCR3</td><td>—</td></tr>
<tr><td>SAI_1</td><td>—</td><td>RCR3</td></tr>
</table>
</td></tr>
<tr><td>23-20
—</td><td>Reserved</td></tr>
<tr><td>19-16
RCE</td><td>Receive Channel Enable
Enables the corresponding data channel for receive operation. Changing this field takes effect immediately for generating the FIFO request and warning flags. It takes effect at the end of each frame for receive operations.
The width of RCE = the number of receive channels (call it n). For example, if RCE is two bits wide, then bit position 16 refers to receive channel 1 and bit position 17 refers to receive channel 2. Writing 1 to bit 16 enables receive channel 1, and writing 1 to bit 17 enables receive channel 2. Writing 1 to bit n enables receive channel n.
NOTE
When there is only a single channel, you do not need an individual channel FIFO reset (RCR3[CFR]). In that case, use the global FIFO reset (RCSR[FR]).
• 0b – Disable receive data channel n
• 1b – Enable receive data channel n
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>SAI_0</td><td>RCR3</td><td>—</td></tr>
<tr><td>SAI_1</td><td>RCR3[16]</td><td>RCR3[19–17]</td></tr>
</table>
</td></tr>
<tr><td>15-4
—</td><td>Reserved</td></tr>
<tr><td>3-0
WDFL</td><td>Word Flag Configuration
Configures which word sets the word start flag (RCSR[WSF]). The value must be one less than the word number (for example, write zero to select the first word in the frame). If you configure this field to a value greater than RCR4[FRSZ], RCSR[WSF] is never set.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0000b - Word 1 0001b - Word 2 0010b-1110b - Word (WDFL value + 1) 1111b - Word 16

74.6.1.17 Receive Configuration 4 (RCR4)

Offset

Register	Offset
RCR4	98h

Function

Contains the receive fields for FIFO Combine mode, FIFO Packing mode, and frame sync settings.

NOTE

Do not alter this register when RCSR[RE] is 1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0			FCONT	FCOMB		FPACK		0				FRSZ			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0			SYWD					0		Reserved	MF	FSE	ONDEM	FSP	FSD
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	Reserved
28 FCONT	FIFO Continue on Error Configures when SAI continues receiving data after a FIFO error is detected.

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>0b - From the start of the next frame after the FIFO error flag is cleared
1b - From the same word that caused the FIFO error to become 1 after the FIFO warning flag is cleared</td></tr>
<tr><td>27-26
FCOMB</td><td>FIFO Combine Mode
Enables FIFO Combine mode for specified operations.
When FIFO Combine mode is enabled for FIFO reads, software reading any FIFO data register alternates the read among the enabled data channel FIFOs. For example, if two data channels are enabled:
• The first read is performed to the first enabled data channel FIFO.
• The second read is performed to the second enabled data channel FIFO.
Resetting the FIFO or disabling FIFO Combine mode for FIFO reads resets the pointer back to the first enabled data channel.
When FIFO Combine mode is enabled for FIFO writes from the receive shift registers, the first enabled data channel input alternates between the enabled data channel FIFOs. For example, if two data channels are enabled:
• The first unmasked received word is stored in the first enabled data channel FIFO.
• The second unmasked received word is stored in the second enabled data channel FIFO.
Because the first word of the frame is always stored in the first enabled data channel FIFO, NXP recommends that the number of unmasked words per frame is evenly divisible by the number of enabled data channels.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>SAI_0</td><td>RCR4</td><td>—</td></tr><tr><td>SAI_1</td><td>—</td><td>RCR4</td></tr></table>
00b - Disable
01b - Enable on FIFO writes (from receive shift registers)
10b - Enable on FIFO reads (by software)
11b - Enable on FIFO writes (from receive shift registers) and reads (by software)</td></tr>
<tr><td>25-24
FPACK</td><td>FIFO Packing Mode
Enables packing of 8-bit data or 16-bit data into each 32-bit FIFO word. If the word size is greater than 8 bits or 16 bits, only the first 8 bits or 16 bits are stored in the FIFO. The first word in each frame always starts with a new 32-bit FIFO word and the first bit shifted must be configured within the first packed word. When FIFO packing is enabled, the FIFO read pointer only increments when software reads the full 32-bit FIFO word.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00b - Disable 01b - Reserved 10b - Enable 8-bit FIFO packing 11b - Enable 16-bit FIFO packing
23-20 —	Reserved
19-16 FRSZ	Frame Size Configures the number of words in each frame. The value must be one less than the number of words in the frame. For example, write 0 for one word per frame. The maximum supported frame size is 16 words. 0000b - 1 0001b - 2 0010b-1110b - (FRSZ value + 1) 1111b - 16
15-13 —	Reserved
12-8 SYWD	Sync Width Configures the length of the frame sync in number of bit-clock cycles. The value must be one less than the number of bit-clock cycles. For example, write 0 for the frame sync to assert for one bit-clock cycle only. You cannot configure this field to be longer than the first word of the frame. 0_0000b - 1 0_0001b - 2 0_0010b-1_1110b - (SYWD value + 1) 1_1111b - 32
7-6 —	Reserved
5 —	Reserved Write only zero to this field.
4 MF	MSB First Configures what is received first: LSB or MSB. 0b - LSB 1b - MSB
3	Frame Sync Early

Table continues on the next page...

Table continued from the previous page...

Field	Function
FSE	Determines when frame sync is asserted. 0b - First bit of the frame 1b - One bit before the first bit of the frame
2 ONDEM	On-Demand Mode Determines when the internal frame sync is generated. When this field is 1, and the frame sync is generated internally, a frame sync is only generated when the FIFO warning flag is 0. 0b - Generated continuously 1b - Generated when the FIFO warning flag is 0
1 FSP	Frame Sync Polarity Configures the polarity of the frame sync. 0b - Active high 1b - Active low
0 FSD	Frame Sync Direction Configures the direction of the frame sync. 0b - Generated externally in Target mode 1b - Generated internally in Controller mode

74.6.1.18 Receive Configuration 5 (RCR5)

Offset

Register	Offset
RCR5	9Ch

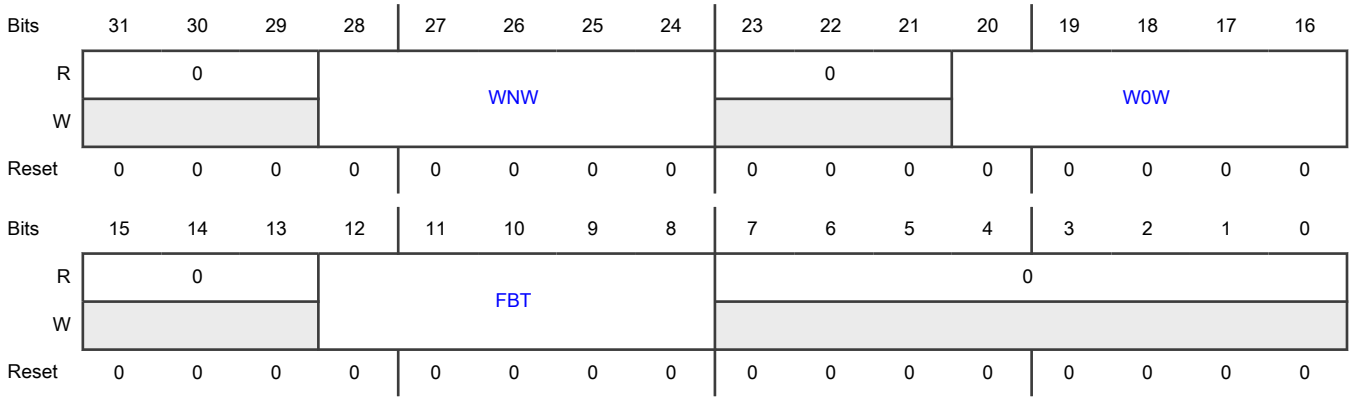
Function

Contains receive word and bit index settings.

NOTE

Do not alter this register when RCSR[RE] is 1.

Diagram



Fields

Field	Function
31-29 —	Reserved
28-24 WNW	Word N Width Configures the number of bits in each word except for the first one in the frame. The value must be one less than the number of bits per word. SAI does not support word widths less than 8 bits. SAI does not support bit settings 0–6 (0b–0_0110b). 0_0111b - 8 0_1000b - 9 0_1001b-1_1110b - (WNW value + 1) 1_1111b - 32
23-21 —	Reserved
20-16 W0W	Word 0 Width Configures the number of bits in the first word of each frame. The value must be one less than the number of bits in the first word. SAI does not support word widths less than 8 bits when there is only one word per frame. 0_0000b - 1 0_0001b - 2 0_0010b-1_1110b - (W0W value + 1) 1_1111b - 32
15-13 —	Reserved
12-8	First Bit Shifted

Table continues on the next page...

Table continued from the previous page...

Field	Function
FBT	Configures the bit index for the first bit received for each word in the frame. If configured for MSB-first, the index of the next bit received is one less than the current bit received. If configured for LSB-first, the index of the next bit received is one more than the current bit received. The value must be greater than or equal to the word width when configured for MSB-first. The value must be less than or equal to 31-word width when configured for LSB-first. See Data alignment for details. 0_0000b - 0 0_0001b-1_1110b - FBT value 1_1111b - 31
7-0 —	Reserved

74.6.1.19 Receive Data (RDR0 - RDR3)

Offset

Register	Offset
RDR0	A0h
RDR1	A4h
RDR2	A8h
RDR3	ACh

Function

Contains receive data.

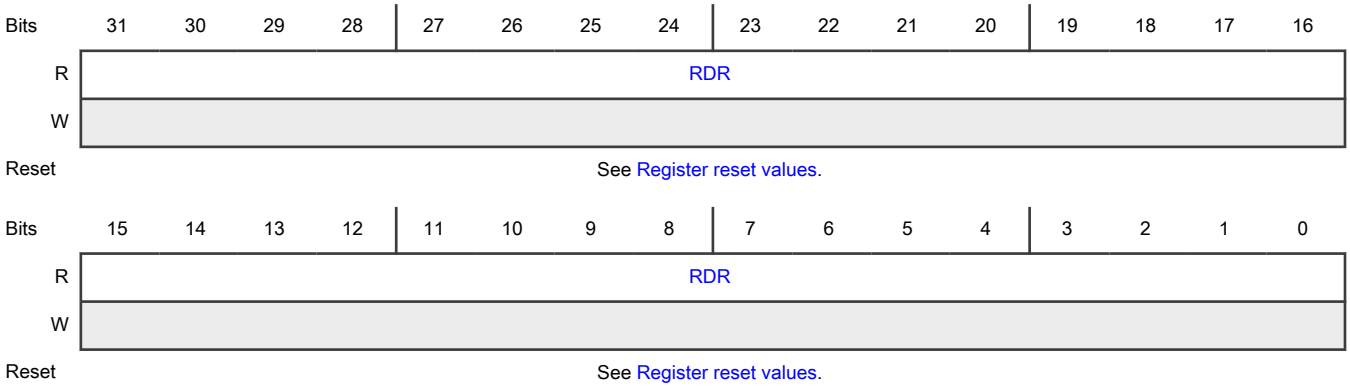
When the receive FIFO is not empty, reads from this register return the data from the top of the receive FIFO. When the receive FIFO is empty, SAI ignores reads from this register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
SAI_0	RDR0–RDR3	—
SAI_1	RDR0	RDR1–RDR3

Diagram



Register reset values

Register	Reset value
RDR0	SAI_0,SAI_1: 0000_0000h
RDR1–RDR3	0000_0000h

Fields

Field	Function
31-0 RDR	Receive Data

74.6.1.20 Receive FIFO (RFR0 - RFR3)

Offset

Register	Offset
RFR0	C0h
RFR1	C4h
RFR2	C8h
RFR3	CCh

Function

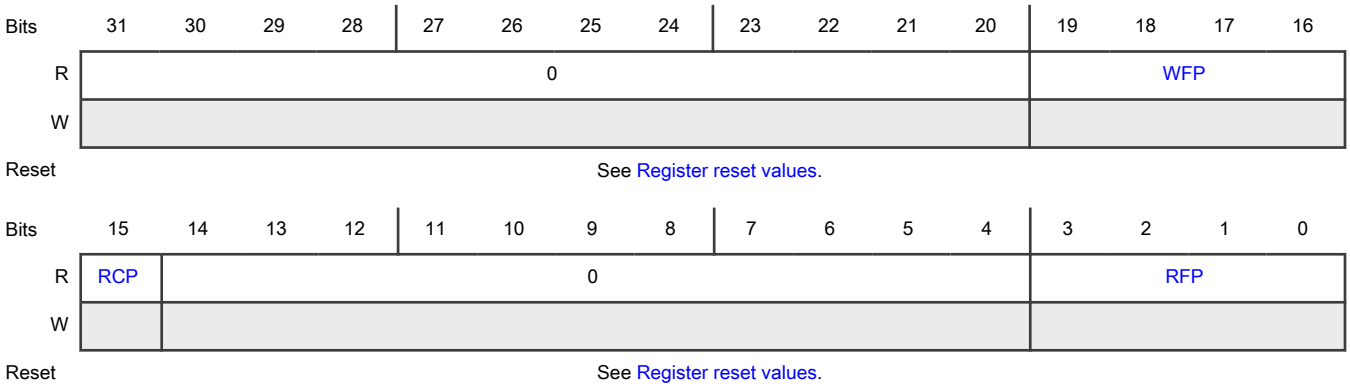
Contains the receive FIFO pointers. The MSB of the read and write pointers distinguishes between FIFO full and empty conditions. If the read and write pointers are identical, the FIFO is empty. If the read and write pointers are identical except for the MSB, the FIFO is full.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
SAI_0	RFR0–RFR3	—
SAI_1	RFR0	RFR1–RFR3

Diagram



Register reset values

Register	Reset value
RFR0	SAI_0,SAI_1: 0000_0000h
RFR1–RFR3	0000_0000h

Fields

Field	Function
31-20 —	Reserved
19-16 WFP	Write FIFO Pointer Indicates the FIFO write pointer for the receive data channel.
15 RCP	Read Channel Pointer Indicates whether this data channel is the next FIFO to be read when FIFO Combine mode is enabled for read operations. **NOTE** This field is not supported in every instance. The following table includes only supported registers.

Table continued from the previous page...

Field	Function
	<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>SAI_0</td><td>RFR0–RFR3</td><td>—</td></tr><tr><td>SAI_1</td><td>—</td><td>RFR0</td></tr></table> 0b - No effect 1b - Next FIFO to be read
14-4 —	Reserved
3-0 RFP	Read FIFO Pointer Indicates the FIFO read pointer for the receive data channel.

74.6.1.21 Receive Mask (RMR)

Offset

Register	Offset
RMR	E0h

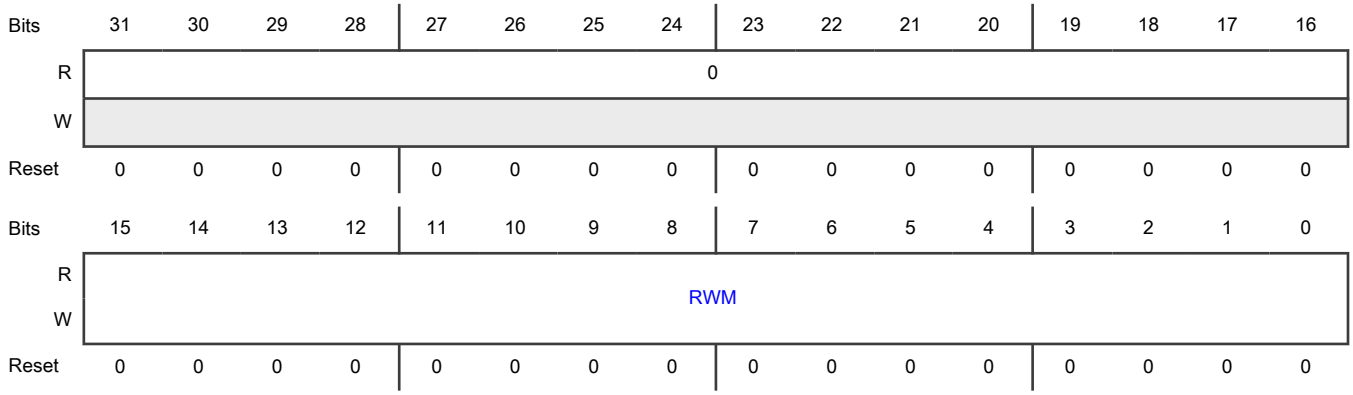
Function

Contains the mask for the receive word. This register is double-buffered and updates:

- When RCSR[RE] first becomes 1.
- At the end of each frame.

This setup allows the masked words in each frame to change from frame to frame.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 RWM	Receive Word Mask Configures whether the receive word is masked (received data ignored and not written to receive FIFO) for the corresponding word in the frame. 0000_0000_0000_0000b - Enable 0000_0000_0000_0001b - Mask

74.7 Glossary

FIFO	First-in first-out
LSB	Least significant bit
MSB	Most significant bit

Chapter 75
Ethernet Media Access Controller (EMAC)

75.1 Chip-specific EMAC information

75.1.1 EMAC instance and configuration

This chip supports up to 1 instance of EMAC IP.

Table 548. EMAC instance

Instance	S32K322/S32K342/S32K341/S32K314/ S32K324/S32K344	S32K310/S32K311/S32K312/S32K358/ S32K356/S32K348/S32K338/S32K328/ S32K388/S32K389
EMAC	Yes	No

NOTE

- DWC_EQOS refer to the Synopsys Ethernet MAC IP.
- EMAC receives only octet aligned preamble.
- The description for some EMAC registers include references to configurations of the Synopsis Ethernet MAC IP. Refer to MAC_HW_Feature registers for the specific features available in this chip.
- EMAC operates only in Clock options A and B, since the module clock becomes lower than the protocol clock (RMII/MII clocks) in other modes.
- It is essential that all PHY input clocks be present and stable to complete the software reset operation and correctly initialize the EMAC IP. The MAC and DMA controller must reset the logic and all internal registers of the DMA, MTL, and MAC layers to initialize the module (see the "Programming" section within this chapter). Therefore, the reset operation is only complete when all resets in all active clock domains are de-asserted (DMA_MODE[SWR] = 0b0). Once the EMAC IP is successfully initialized, the PHY can be reset without affecting the MAC.

Following key features are supported:

- This chip supports only MII/RMII interfaces
- 4 bit MII interface operating at 2.5/25/50Mhz, MII can run at 10/100/200 Mbps.
- 2 bit RMII interface operating at 50Mhz, RMII can run at 10/100Mbps
- 4 bit MII-Lite interface operating at 2.5/25Mhz
- 4 PPS are supported using DMA trigger
- 200Mbps data rate is supported on Ethernet MII interface
- MTL Receive FIFO size 8192bytes
- MTL Transmit FIFO size 8192bytes

NOTE

The DMA referred within EMAC chapter refers to the internal EMAC DMA engine and not the device AXBS master DMA.

NOTE

Register access beyond 8k address space is not supported and will generate a transfer error.

NOTE

Always use store and forward mode if possible. In cut through mode, if multiple masters are active then there is chance (even in round robin arbitration mode) that EMAC DMA may starve for data resulting in insufficient data in TX FIFO leading to Tx underflow. To avoid this, increase MTL_TxQ0_Operation_Mode[TTC] value as needed.

The following diagram depicts EMAC-Memory interfaces

The table below provides the information on the specific configuration utilized for the S32K344/S32K324/S32K314 and S32K342/S32K322/S32K341 devices.

Table 549. Specific configuration information

Application Interface		
	Application interface configuration	EQOS-AHB
	Data width	32
	Endian mode	Little Endian
	Address width	32
	CSR interfcae	APB3 interface
General feature		
	Mode of operation	10/100
	Enable double VLAN processing	Yes
	Enable Que/channel based VLAN tag inertion on Tx	Yes
	Enable SA and VLAN insertion on Tx	Enabled by default
Buffer managemnet		
	MTL Receive FIFO size in bytes	8192
	MTL Receive FIFO size in bytes	8192
	Enable debug memory access	Yes
	Use single port memory	Yes
PHY interface		
	Enable RGMII, RMII only	Enable RMII, MII default
	Enabled single phy interface	No
	Enable MDIO interface	Yes
Filtering		
	Enable Additional 1-31 MAC address registers	2
	Enable Address Filter Hash Table	Yes
	Hash Table Size	64
	Enable VLAN Hash table based filtering	Yes
	Extended Rx VLAN Filter Enable	Yes

Table continues on the next page...

Table 549. Specific configuration information (continued)

	Number of VLAN tag Filters	4
	Enable Layer3 and Layer4 Packet Filter	Yes
	Number of layer3 and Layer 4 packet filters	4
	Enable Flexible Programmable Receive Parser	Yes
	Maximum Packet Header Size for Parsing	128
	Maximum Entries of Parser Look up table	64
	Packet duplication support	Yes
IEEE 1588 timestamp		
	Enable IEEE 1588 Time stamp support	Enabled by default
	IEEE 1588 System Time Support	Both option
	Add IEEE 1588 Higher word register	Yes
	Enable IEEE 1588 sub nanoseconds time stamp support	Yes
	Add IEEE 1588 Auxilary snapshot	No
	Enable Flexible Pulse Per Seconds Output	Yes
	Number of pulse per Second outputs	4
	Enable PTP Timestamp offload feature	No
	Enable One Step timestamp for PTP over UDP/IP feature	No
	Enable One Step timestamp feature	Yes
Multiple queue support		
	Number of transmit Queue	2
	Number of receive Queue	2
	Enable data center Bridging	No
	Enable Audo Video Bridging	Yes
	Enable support for AV in Tx queue 1	No
	Number of DMA Channel on the receive side	2
Time sensitive networking		
	Enable Enhancement to Schedule traffic(EST)	Yes
	Width of Time Interval Field in the Gate Control List	24

Table continues on the next page...

Table 549. Specific configuration information (continued)

	Depth of Gate Control List	256
	Enable Frame Pre-emption support	Yes
Time based scheduling		
	Enable time based Scheduling	Yes
TCP/IP offloading features		
	Enable receive TCP/IP Checksum Check	Enabled by default
	Enable transmit TCP/IP Checksum offload	Enabled by default
	Enable support for Tx COE in Tx queue 0	Yes
	Enable support for Tx COE in Tx queue 1	No
	Enable TCP Segmentation offloading for TCP/IP	No
	Enable Split Header Feature	No
	Enable IPv4 ARP offload	No
RMON counters		
	Enable MAC Management Counters(MMC)	Yes
	Enable the MAC Management Counters for Receiver TCP/IP Checksum offload	No
Automotive safety feature		
	Enable all automotive safety feature	Yes

75.1.2 Impact of MAC flush commands on MAC_PPSx_Target_Time_Seconds register

If the software discards the media clock generation request by configuring the PPSCTRL_PPSCMD (or PPSCMD#i) field of MAC_PPS_Control register to 0, the MAC does not flush the captured presentation time, i.e., the MAC_PPSx_Target_Time_Seconds register contents don't get changed.

75.2 Overview

This section and all its sub-sections are Synopsys Proprietary. Used with permission.

EMAC:

- Supports the 10/100 Mbps application in compliance with IEEE802.3-2015 specifications
- Can support advanced applications such as time-sensitive networking and AVB
- Supports a MAC core that has prominent features such as a flexible receive parser, media clock recovery and generation, and safety

75.2.1 Features

EMAC supports these features in addition to the default feature defined in the IEEE802.3 specification:

- MII (10/100 Mbps), RMII (10/100 Mbps)

- Time-aware shaper (IEEE802.1bv), time synchronization (IEEE802.1AS-rev), and frame preemption (IEEE802.1Qbu) for time-sensitive networking
- Media clock recovery and generation for AVB
- AMBA 2.0 for AHB master and APB3 interface
- RMII specification version 1.2 from RMII consortium
- Separate interface for transmit, receive, and control paths
- Two-buffer ring
- Broadcast and multicast packet duplication
- Full-duplex flow control operations (IEEE 802.3x pause packet and priority flow control)
- Network statistics with MAC management (RMON) counters
- Flexibility to control the pulse per second (PPS) output signal
- MDIO clause 22 and clause 45 interface for configuration and management of the PHY device
- Preamble and SFD insertion and deletion

NOTE

Only byte-aligned preamble is supported.

- Automatic CRC and pad generation/stripping option
- Option to disable CRC checking
- Programmable insert packet gap
- Source address insertion or replacement
- VLAN insertion, replacement, and deletion in transmitted packets with per-packet or static-global control, insertion, replacement, and deletion of up to two VLAN tags, insertion, replacement, and deletion of queue or channel-based VLAN tags
- IEEE802.1Q VLAN tag detection and an option to delete the tags in the receive packets
- Programmable safety watchdog timeout limits
- Flexible address filtering modes that allow:
  - Up to two additional 48-bit perfect DA filters with masks for each byte
  - Up to two 48-bit SA comparison checks with masks for each byte
  - 64-bit hash filter for multicast and unicast DA addresses
  - Option to pass all multi-cast addressed packets
  - Promiscuous mode to pass all packets without any filtering for network monitoring
  - Passing of all incoming packets (according to filter) with a status report
- Additional packet filtering that is based on:
  - Virtual local area network (VLAN) tag—perfect match and hash-based filtering, which is based on either outer or inner VLAN tag, is possible
  - Layer 3 and layer 4—TCP or UDP over IPv4 or IPv6
  - Extended VLAN tag—filtering based on four filter selections

NOTE

The DCB feature discussed in the following section(s) is not supported.

75.2.2 Block diagram

This figure shows the EMAC block diagram. EMAC has four main blocks to perform all the functions.

- The AHB interface is connected to all DMA channels.
- The DMA arbiter helps in the arbitration of all paths (transmit and receive) in DMA channels.
- Each channel has a separate set of control and status registers (CSRs) for managing the transmit and receive functions, descriptor handling, and interrupt handling.

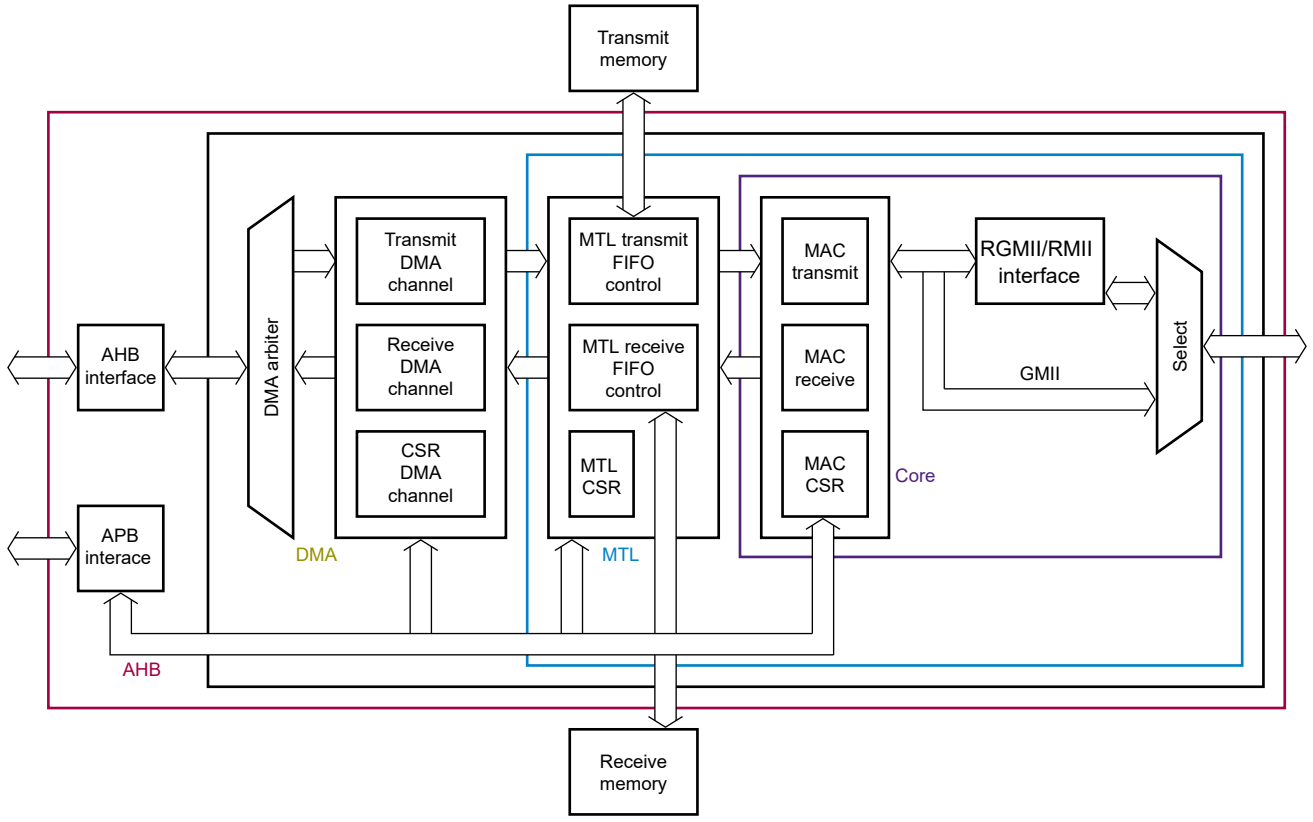


Figure 417. System-level block diagram

75.3 Architecture

This section and all its sub-sections are Synopsys Proprietary. Used with permission.

The section describes EMAC interfaces, protocols, and functionality. These are its main blocks:

- AHB master and APB3 slave interface
- DMA controller
- MTL
- MAC
- Interrupts

75.3.1 AHB master and CSR slave (APB3) interface

The 32-bit APB3 slave interface:

- Provides access to DMA, MTL, and MAC control and status registers (CSRs)

- Supports the 32-bit write and read transfers to these CSRs
- Supports single- and all-burst transfers, and an OKAY response

The DMA block uses the AHB master interface to interact with the external world. The AHB master interface:

- Controls data transfer
- Is AMBA 2.0-compliant with no restrictions
- Supports 32-bit data transfer
- Supports single and INCR transfers
- Supports burst-length modes such as Fixed, Unspecified, and Mixed

75.3.2 DMA inclusions

DMA controller:

- Includes independent transmit (Tx) and receive (Rx) engines, and a CSR space. The transmit engine transfers data from the system memory to the MTL interface, whereas the receive engine transfers data from the MTL interface to the system memory.
- Is designed for packet-oriented data transfers such as packets in Ethernet.
- Communicates with the host through CSR, descriptor lists, and data buffers.

DMA descriptors

DMA supports up to two transmit and two receive descriptor lists (or DMA channels). The base address of each list is written to the respective transmit and receive descriptor list address registers. A descriptor list is forward linked and the next descriptor is always considered at a fixed offset to the current one, and DMA_CH0_Control[DSL] controls the offset. The number of descriptors in the list is programmed using the respective DMA Channel 0 Tx Descriptor Ring Length (DMA_CH0_TxDesc_Ring_Length) and DMA Channel 0 Rx Descriptor Ring Length (DMA_CH0_RxDesc_Ring_Length) registers. After the DMA processes the last descriptor in the list, it automatically jumps to the descriptor in the list address register to create a descriptor ring.

EMAC supports the ring structure for DMA descriptors. For more information, see Descriptors .

The DMA engine uses descriptors to efficiently move data from source to destination with minimal host intervention, and the DMA controller can be programmed to interrupt the host in certain situations. These situations may include packet transmit and receive transfer completion and other normal or error conditions.

The descriptor lists reside in the physical memory address space of the application. Each descriptor can point to a maximum of two buffers in the system memory, enabling two buffers to be used and physically addressed rather than contiguous buffers in the memory.

Data buffers

A data buffer resides in the application physical memory space and consists of an entire packet or part of a packet but cannot exceed a single packet. Buffers contain only data. The buffer status is maintained in the descriptor. Data chaining refers to packets that span multiple data buffers. However, a single descriptor cannot span multiple packets. The DMA skips to the data buffer of the next packet when an EOP is detected.

The next few sections discuss DMA controller.

75.3.2.1 DMA controller bus burst access

DMA engines attempt to transfer data in a burst of the maximum size programmed using DMA_CH0_Tx_Control[TxPBL] and DMA_CH0_Rx_Control[RxPBL] of the respective DMAs, which always access the receive and transmit descriptors in the maximum possible (limited by PBL or 16 * 8/bus width) burst length for 16 bytes to be read. The burst transfers that the DMA initiates can be split into multiple burst transfers according to the AHB requirements and settings specified in DMA System Bus Mode (DMA_SysBus_Mode).

The transmit DMA initiates a data transfer only when sufficient space is available in the MTL transmit queue to accommodate either of these:

- Bytes corresponding to the configured burst (PBL * bus_width/8)
- Remaining bytes in the transmit buffer without EOP
- Number of bytes till EOP

The receive DMA initiates a data transfer in these conditions:

- Sufficient data is available in the MTL receive queue to accommodate the configured burst
- EOP (when it is less than the configured burst length) is detected in the receive queue

DMA indicates the start address and the number of transfers required to the AHB master interface. When the interface is configured for fixed-length burst, it transfers data by using the best combination of the INCR4, INCR8, or INCR16 and SINGLE transactions. If EOP is reached before the fixed burst ends on the AHB interface, DMA performs dummy transfers to complete the fixed burst. Otherwise, DMA_SysBus_Mode[FB] becomes 0. The DMA transfers the data using undefined length (INCR) and SINGLE transactions.

When the AHB interface is configured for address-aligned beats, both DMA engines ensure that the first-burst transfer that AHB initiates is less than or equal to the size of the configured PBL value, and the subsequent beats start at an address aligned to this value. The DMA can only align the address for beats up to size 16 (for PBL > 16) for the AHB interface because it does not support more than INCR16.

75.3.2.2 DMA application data buffer alignment

The transmit and receive data buffers do not have any restrictions on start address alignment. For example, in systems with 32-bit memory, the start address for buffers can be aligned to any of the four bytes. However, the DMA always initiates write transfers with an address aligned to the bus width and dummy data (old data) in the invalid byte lanes. The process of write transfer typically happens during the beginning or end of an Ethernet packet transfer. The software driver must discard the dummy bytes based on the start address of the buffer and size of the packet.

Table 550. Data buffer alignment examples

Examples	
Buffer read	If the transmit buffer address is FF2h (for a 32-bit data bus) and 15 bytes need to be transferred, the DMA reads five full words from address FF0h. However, when DMA transfers data to the MTL transmit queue, the extra bytes (the first two bytes) are dropped or ignored. Similarly, the last 3 bytes of the last transfer are also ignored. The DMA always ensures that it transfers a full 32-bit data to the MTL transmit queue, unless it is the end of the packet.
Buffer write	If the receive buffer address is FF2h (for a 64-bit data bus) and 16 bytes of a received packet need to be transferred, the DMA writes 3 full words from address FF0h. However, the first 2 bytes of the first transfer and the last 6 bytes of the third transfer have dummy data. The DMA considers the offset address only if it is the first receive buffer of the packet. It ignores the offset address and performs full-word writes for the middle and last receive buffers of the packet.

75.3.2.3 DMA buffer size calculations

DMA does not update the size fields in the transmit and receive descriptors. It updates only the status fields (RDES and TDES) of the descriptors. The driver performs the size calculations.

The transmit DMA transfers the exact number of bytes (as the buffer size field of TDES2 indicates) to MAC. If a descriptor is marked as first (TDES3[FD] = 1), the DMA marks the first transfer from the buffer as SOP, and if the descriptor is marked as last (TDES3[LD]), the DMA marks the last transfer from that data buffer as EOP to the MTL.

The receive DMA transfers data to a buffer until the buffer is full or MTL sends the end of packet. When TDES3[FD] = 1, the amount of valid data in a buffer is accurately indicated by the buffer size fields in DMA Channel Rx Control (DMA_CH0_Rx_Control) minus the data buffer pointer offset. The offset is 0 when the data buffer pointer is aligned to the data bus width. If a descriptor is marked as last, the buffer may not be full (as indicated by DMA_CH0_Rx_Control[RBSZ_13_y] and DMA_CH0_Rx_Control[RBSZ_x_0]).

To compute the amount of valid data in this final buffer, the driver must read the packet length (PL fields of RDES3[14:0]) and subtract the sum of the buffer sizes of the preceding buffers in this packet. The receive DMA always transfers the start of the next packet with a new descriptor.

75.3.2.4 DMA arbiter

The arbiter inside the DMA module performs the arbitration between the transmit and receive channel accesses to the AHB master interface. These two types of arbitrations are supported:

- Round-robin: If DMA_Mode[DA] = 0 and both transmit and receive DMAs simultaneously request access, the arbiter allocates the data bus in ratio sets defined in DMA_Mode[PR].
- Fixed-priority: If DMA_Mode[DA] = 1, receive DMA is always prioritized over transmit DMA for data access by default. If DMA_Mode[TXPR] = 1, transmit DMA is prioritized over receive DMA as explained in Table 567.

75.3.3 MTL

MTL provides the FIFO memory interface to buffer and regulate the packets between system memory and MAC. It also enables data to be transferred between the system clock and MAC clock domains. The MTL layer has two data paths: transmit path and receive path. MTL communicates with the host through ATI on the transmit path and through ARI on the receive path.

75.3.3.1 Transmit path

The internal DMA:

- Handles all transactions for the transmit path through ATI.
- Pushes the Ethernet packet reads from system memory to the corresponding queue.

The Ethernet packet, then, pops out and is transferred to MAC when the queue reaches its threshold (Threshold mode) or if the complete packet is in the queue (Store-and-Forward mode). When EOP is transferred, the status of the transmission is taken from MAC and transferred back to the internal DMA.

75.3.3.1.1 Transmit control word

This control information related to packet transmission is provided as part of the control word through the ATI interface:

- Packet length (if DCB is enabled with WFQ scheduling algorithm)
- CRC pad control
- Source address insertion control
- VLAN insertion and replacement control and VLAN tag for outer and inner VLAN tags
- TCP/IP checksum insertion control
- One-step timestamping control correction
- Transmit timestamp enable

75.3.3.1.2 Transmit operation

These two modes of operation trigger data read towards MAC:

- Threshold (or cut-through) mode: This is the default mode. In this mode, as soon as the number of bytes in the queue crosses the configured threshold level (or when the end of packet is written before the threshold is crossed), the data is ready to be popped out and forwarded to MAC. The threshold level is configured by using MTL_TxQ0_Operation_Mode[TTC].
- Store-and-Forward mode: In this mode, MTL pops the packet out to MAC only when one or more of the following conditions are true:
  - A complete packet is stored in the queue.

— The transmit FIFO becomes almost full.

— The ATI watermark becomes low.

The watermark becomes low when the requested queue is not spacious enough to accommodate the requested burst length on ATI. Therefore, MTL, when operating in Store-and-Forward mode, allows packet transmission even if the packet length is bigger than the transmit queue size.

You can flush the complete content of the transmit queue by writing 1 to MTL_TxQ0_Operation_Mode[FTQ] or MTL_TxQ1_Operation_Mode[FTQ], depending on the queue. Doing so initializes the queue pointers to the default state. The FTQ field returns to 0 by itself. If you write 1 to the FTQ field during a packet transfer from MTL to MAC, MTL stops further transfers because MTL considers the queue to be empty. Therefore, an underflow event occurs at the MAC transmitter.

75.3.3.1.3 Initialization flow

After reset, MTL is ready to manage the data flow between DMA and MAC.

- With single-transmit queue configurations, there are no initialization requirements for enabling MTL.
- With multiple-transmit queue configurations, initialize the queue size for each of the queues by programming MTL_TxQ0_Operation_Mode[TQS] corresponding to a transmit queue. Also, initialize the MAC block. Internal DMA controllers must be individually enabled through their respective CSRs.

75.3.3.2 Receive path

MAC sends packets to the MTL receive module and pushes them into the receive queue. MTL indicates the status (fill level) of the queue to the application or DMA when it crosses the configured receive threshold (MTL_RxQ0_Operation_Mode[RTC]), or when the MTL receive module receives the complete packet in Store-and-Forward mode. MTL also indicates the fill level of the queue so that DMA can initiate preconfigured burst transfers to the AHB interface.

75.3.3.2.1 Receive operation

During a receive operation, MTL is MAC's slave. This is the general sequence of events:

1. When MAC receives a packet, it indicates the availability of receive data.
2. MAC indicates the SOP and EOP delimiters.
3. MTL accepts the data and pushes it into the corresponding receive queue.
4. When MAC transfers EOP to the MTL receive module, MAC also drives the status word that MTL pushes into the corresponding receive queue.
5. MTL takes the data out of the queue and sends it to DMA depending on these modes:
   - Threshold mode
   - Store-and-Forward mode

   The sections that follow discuss these modes.

75.3.3.2.1.1 Threshold mode

In Threshold (default) mode, MTL reads the data and indicates its availability to the application or DMA when one of these occurs:

- Data bytes equal to the threshold amount are written to the receive queue (to MTL_RxQ0_Operation_Mode[RTC] and MTL_RxQ1_Operation_Mode[RTC]).
- A full packet of data is received into the queue.

75.3.3.2.1.2 Store-and-Forward mode

In this mode (when MTL_RxQ0_Operation_Mode[RSF] = 1), the initial receive queue locations are reserved for the status words before writing the SOP. A packet is read out only after it is completely written into the receive queue. In this mode, all error

packets are dropped (if configured through MTL_RxQ0_Operation_Mode[FEP]) in such a way that only valid packets are read and forwarded to the application.

75.3.3.2.2 Multi-packet receive operation

In Threshold mode, the packet status is available immediately after the packet data. In Store-and-Forward mode, the packet data is available after the packet status. MTL is capable of storing any number of packets in the queue as long as it is not full.

If MAC receives a packet when the corresponding receive queue is full, MTL ignores that packet and overflow pulse generates on the corresponding receive queue. In addition, MTL increments the overflow counter in MTL Rx Queue Missed Packet Overflow Count (MTL_RxQ0_Missed_Packet_Overflow_Cnt) for the corresponding queue.

75.3.3.2.3 Error handling in receive operation

MTL performs these actions if the MTL receive queue is full before it receives the EOP data from MAC:

1. Declares an overflow
2. Drops the whole packet (including the status word)
3. Increments the overflow counter in DMA (MTL Rx Queue Missed Packet Overflow Count (MTL_RxQ0_Missed_Packet_Overflow_Cnt))

This is true even if MTL_RxQ0_Operation_Mode[FEP] is 1.

If the start address of such a packet is already transferred to the read controller, the rest of the packet is dropped and a dummy EOP is written to the queue along with the status word with overflow status. The status indicates a partial packet because of overflow. In such packets, the Packet Length field is invalid. If the MTL receive queue is configured to operate in the Store-and-Forward mode and the length of the received packet is more than the queue size, overflow occurs and all such packets are dropped.

The MTL receive control logic can filter errors and undersized packets, if enabled by using MTL_RxQ0_Operation_Mode[FEP] and MTL_RxQ0_Operation_Mode[FUP]. If the start address of one such packet is already transferred to the receive queue read controller, the packet is not filtered. The start address of the packet is transferred to the read controller after the packet crosses the receive threshold defined using MTL_RxQ0_Operation_Mode[RTC].

If the MTL receive queue is configured to operate in Store-and-Forward mode, all error packets can be filtered and dropped. For the application or DMA to flush the error packet being read from the queue, it must assert the flush signal. MTL then stops transferring data to the application (DMA). It internally reads the rest of the packet and drops it. MTL then starts transferring the next packet, if it's available.

75.3.4 MAC

MAC can support the MII and RMII PHY interface and it consists of:

- MTI
- MRI
- MCI

75.3.4.1 MAC transmission

The MAC transmission process is as follows:

1. The transmission initiates when MTL pushes in data with the SOP signal asserted.
2. After the SOP signal is detected, MAC accepts the data and begins transmitting to RMII or MII.
3. After the EOP signal is transferred to MAC, it performs one of these steps:
   - Completes normal packet transmission and sends the transmission status to MTL
   - Sends retry requests if a normal collision (in Half-Duplex mode) occurs during transmission, and until one of the following is true:

— Packet is successfully transmitted.

— Maximum retry requests have expired. When this happens, MAC aborts the packet transmission with "excessive collision transmit" status. MAC accepts and drops all further data until it receives the next SOP. The MTL block must retransmit the same packet from SOP on observing a retry request (in the status) from MAC.

- If any one of the following happens, MAC aborts the packet transmission:
  — No carrier (Half-Duplex mode)
  — Loss of carrier (Half-Duplex mode)
  — Excessive deferral (Half-Duplex mode)
  — Late collisions (Half-Duplex mode)
  — Jabber
  — MAC accepts and drops all further data until it receives the next SOP

4. MAC issues an underflow status if MTL is not able to provide data continuously during transmission. Until the next SOP is received, MAC accepts and drops all further data.
5. During the normal transfer of a packet from MTL, if MAC receives an SOP without getting an EOP for the previous packet, it ignores the SOP and considers the new packet as a continuation of the previous packet.

75.3.4.2 MAC reception

The receive operation initiates as follows:

1. MAC detects a state-of-frame data on RMII or MII.
2. MAC strips the preamble and SFD before processing an Ethernet packet.
3. The MAC AFM:
   - Checks the header fields (SA and DA) of an incoming packet for filtering
   - Verifies the CRC contained in the packet's FCS field
4. MAC's AFM checks the header fields (SA and DA) of an incoming packet for filtering.
5. FCS verifies the CRC of the received packet.
6. MAC stores the received packet in a shallow buffer until address filtering is performed.
7. AFM drops the packet if it fails the address filter.

75.3.5 Interrupts

EMAC supports interrupt coalescing, which reduces the number of interrupts generated by the module and reduces the CPU load. MAC Interrupt Status (MAC_Interrupt_Status) captures various interrupt events. If the interrupt enable field of an interrupt is 1, the corresponding interrupt generates based on the event status in the Status registers. The Interrupt mode (INTM) field decides whether the interrupt signal is a level signal or pulse signal. See the following table and interrupt map file attached to this document for details.

Table 551. Interrupt request

Interrupt request	Interrupt Enable / Mask Register	Interrupt Flag Register
Common interrupt	MAC Interrupt Enable (MAC_Interrupt_Enable)	MAC Interrupt Status (MAC_Interrupt_Status)
	MMC Receive Interrupt Mask (MMC_Rx_Interrupt_Mask)	MMC Receive Interrupt (MMC_Rx_Interrupt)

Table continues on the next page...

Table 551. Interrupt request (continued)

Interrupt request	Interrupt Enable / Mask Register	Interrupt Flag Register
	MMC Transmit Interrupt Mask (MMC_Tx_Interrupt_Mask)	MMC Transmit Interrupt (MMC_Tx_Interrupt)
	MMC FPE Transmit Interrupt Mask (MMC_FPE_Tx_Interrupt_Mask)	MMC Transmit FPE Fragment Counter Interrupt Status (MMC_FPE_Tx_Interrupt)
	MMC FPE Receive Interrupt Mask (MMC_FPE_Rx_Interrupt_Mask)	MMC Receive Packet Assembly Error Counter Interrupt Status (MMC_FPE_Rx_Interrupt)
	MTL Debug Control (MTL_DBG_CTL)	MTL Debug Status (MTL_DBG_STS)
	MTL EST Interrupt Enable (MTL_EST_Intr_Enable)	MTL EST Status (MTL_EST_Status)
	MTL Rx Parser Interrupt Control Status (MTL_RXP_Interrupt_Control_Status)	MTL Rx Parser Interrupt Control Status (MTL_RXP_Interrupt_Control_Status)
	MTL Queue 0 Interrupt Control Status (MTL_Q0_Interrupt_Control_Status)	MTL Queue 0 Interrupt Control Status (MTL_Q0_Interrupt_Control_Status)
	MTL Queue 1 Interrupt Control Status (MTL_Q1_Interrupt_Control_Status)	MTL Queue 1 Interrupt Control Status (MTL_Q1_Interrupt_Control_Status)
	DMA Channel 0 Interrupt Enable (DMA_CH0_Interrupt_Enable)(transfer complete interrupts only)	DMA_CH0_Interrupt_Status
	DMA Channel 1 Interrupt Enable (DMA_CH1_Interrupt_Enable)(transfer complete interrupts only)	DMA_CH1_Interrupt_Status
		DMA Interrupt Status (DMA_Interrupt_Status) (First level status. Indicates major block event source: DMA channel, MTL or MAC)
		MTL Interrupt Status (MTL_Interrupt_Status) (Status summary for MTL block interrupts: Rx Parser, EST, FIFO Debug, Queue 1/Queue 0)
Tx interrupt 0/1	DMA Channel Tx Control (DMA_CH0_Tx_Control)	DMA Channel 0 Status (DMA_CH0_Status)
	DMA Channel 0 Interrupt Enable (DMA_CH0_Interrupt_Enable)	
	DMA Channel 1 Tx Control (DMA_CH1_Tx_Control)	DMA Channel 1 Status (DMA_CH1_Status)
	DMA Channel 1 Interrupt Enable (DMA_CH1_Interrupt_Enable)	
Rx interrupt 0/1	DMA Channel Rx Control (DMA_CH0_Rx_Control)	DMA Channel 0 Status (DMA_CH0_Status)

Table continues on the next page...

Table 551. Interrupt request (continued)

Interrupt request	Interrupt Enable / Mask Register	Interrupt Flag Register
	DMA Channel 0 Interrupt Enable (DMA_CH0_Interrupt_Enable)	
	DMA Channel 1 Rx Control (DMA_CH1_Rx_Control)	DMA Channel 1 Status (DMA_CH1_Status)
	DMA Channel 1 Interrupt Enable (DMA_CH1_Interrupt_Enable)	
Safety interrupt	MTL ECC Interrupt Enable (MTL_ECC_Interrupt_Enable)	MTL ECC Interrupt Status (MTL_ECC_Interrupt_Status)
		MTL Safety Interrupt Status (MTL_Safety_Interrupt_Status)
	MTL DPP Control (MTL_DPP_Control)	MAC DPP FSM Interrupt Status (MAC_DPP_FSM_Interrupt_Status)
	MAC FSM Control (MAC_FSM_Control)	

These are the interrupt output signals that are synchronous to the CSR clock.

- Sbd_intr_o – common interrupt
- Sbd_perch_tx_intr_o[max_dma_ch] – interrupt per transmit channel
- Sbd_perch_rx_intr_o [max_dma_ch] – interrupt per receive channel

NOTE

max_dma_ch = number of transmit/receive queues, which is 2

The sbd_intr_o common interrupt is a level signal. When it is asserted, a corresponding interrupt event source can be found in DMA Interrupt Status (DMA_Interrupt_Status), which is a read-only register that contains the event source fields corresponding to each DMA channel (transmit and receive queue pair), MAC transaction layer, and MAC blocks. You must then read the following registers and look for the fields that are 1:

- MAC Interrupt Status (MAC_Interrupt_Status)
- MTL Interrupt Status (MTL_Interrupt_Status)
- DMA Channel 0 Status (DMA_CH0_Status)

The sbd_intr_o interrupt deasserts only when all the enabled interrupt events are clear in their respective status registers, and correspondingly, all the fields in DMA Interrupt Status (DMA_Interrupt_Status) are 0. DMA Channel 0 Status (DMA_CH0_Status) captures all the interrupt events of that transmit DMA and receive DMA channel pair.

DMA Channel 0 Interrupt Enable (DMA_CH0_Interrupt_Enable) contains the corresponding enable fields for each of the interrupt events. These are the two groups of interrupts in the DMA channel:

- Normal—DMA_CH0_Status[NIS] indicates this interrupt, which is used for events that occur during the normal transfer of packets (TI, RI, TBU).
- Abnormal—DMA_CH0_Status[AIS] indicates this interrupt, which is used for error events.

Interrupts are not queued. If the same interrupt event occurs again before the driver responds to the previous one, no additional interrupts generate.

The common sbd_intr_o output signal asserts for the transfer complete interrupts only when the corresponding interrupts are enabled in DMA Channel 0 Interrupt Enable (DMA_CH0_Interrupt_Enable).

EMAC also supports these per-channel transfer-complete interrupt signals:

- sbd_perch_tx_intr_o[max_dma_ch] (transmit per channel interrupts)
- sbd_perch_rx_intr_o[max_dma_ch] (receive per channel interrupts)

The behavior of the RI/TI/sbd_perch_tx_intr_o[]/sbd_perch_rx_intr_o[] changes depends on the configuration specified in DMA_Mode[INTM]. This table explains the transfer complete interrupt behavior.

Table 552. Transfer complete interrupt behavior

Interrupt mode	Behavior of sbd_perch_tx_intr_o[] and sbd_perch_rx_intr_o[]	Behavior of the TI/RI interrupts and sbd_intr_o
INTM = 1	In this mode, these signals indicate the values of the corresponding DMA_CH0_Status[RI] and DMA_CH0_Status[TI] fields when DMA_CH0_Status[RI] and DMA_CH0_Status[TI] are 1, respectively. Therefore, they are level signals that you clear by writing 1 to these fields. The signals do not assert when DMA_CH0_Status[RI] or DMA_CH0_Status[TI] is 0.	DMA_CH0_Status[RI] and DMA_CH0_Status[TI] are configured as explained. For any RI/TI events: • The sbd_intr_o signal does not assert. • DMA_CH0_Status[NIS] remains 0.
INTM = 2	In this mode, RI/TI are queued and indicate the values of the corresponding DMA_CH0_Status[RI] and DMA_CH0_Status[TI] fields when any of these fields is 1. RI and TI are level signals that you clear by writing 1 to these fields. However, the fields become 1 again if another TI/RI event is detected before DMA_CH0_Status[RI] and DMA_CH0_Status[TI] become 1 for the previous event.	DMA_CH0_Status[RI] and DMA_CH0_Status[TI] become 1 whenever the transfer complete event is detected and are reset whenever the software driver changes them to 0 by writing 1. However, if another transfer complete event is detected before the software changes it to 0 (services it), EMAC automatically writes 1 to these fields again. The generation of the sbd_intr_o signal; however, is not based on DMA_CH0_Status[RI] or DMA_CH0_Status[TI].

75.4 External signals

This section and all its sub-sections are Synopsys Proprietary. Used with permission.

75.4.1 Module signals

This table provides port/signal names and their descriptions. See the IOMUX file attached to this document for details.

Table 553. Module signals

Port name (clock/signal)	I/O	Description
MII_RMII_TXCLK (Clock)	I	MII: The external PHY provides this transmission clock, which operates at a frequency of 25 MHz in 100 Mbps mode and at 2.5 MHz in 10 Mbps mode. All transmission signals that MAC generates are synchronous to this clock, which is required for all PHY interfaces.

Table continues on the next page...

Table 553. Module signals (continued)

Port name (clock/signal)	I/O	Description
		RMII: The RMII interface uses this 50 MHz clock. If you select RMII mode, MII_RX_CLK (25 MHz or 2.5 MHz) must be derived from the RMII reference clock.
MII_RX_CLK (Clock)	I	The external PHY provides this receive clock for the MII and RMII interfaces. The clock operates at a frequency of 25 MHz in 100 Mbps mode and at 2.5 MHz in 10 Mbps mode. All MII receive signals that MAC receives are synchronous to MII_RX_CLK. The clock's input is required for all PHY interfaces.
EMAC_PPS[3:0] (Signals)	I/O	This group of signals is used as pulse per second in Output mode and as media clock generation trigger in Input mode. The signals trigger input to DUT to capture presentation time. Based on the presentation control value of MAC PPS Control (MAC_PPS_Control), the signals can be defined as pulse or level signals.
MII_RMII_TX_EN (Signal)	O	MAC drives this signal, which performs multiple functions depending on the selected PHY interface as described in this list: • MII: When high, indicates that valid data is being transmitted to the PHY_TXD_O bus. MII_RMII_TX_EN is synchronous to MII_RMII_TX_CLK. • RMII: When high, indicates that valid data is being transmitted to the PHY_TXD_O bus. MII_RMII_TX_EN is synchronous to MII_RMII_RX_CLK.
MII_RMII_TXD[3:0] (Signals)	O	This is a group of transmit data signals driven by MAC. These signals perform multiple functions depending on the selected PHY interface as described in this list: • MII: Bits[3:0] provide the MII transmit data nibble. The data is valid only when the MII_RMII_TX_EN signal is high. MII_RMII_TXD[3:0] is synchronous to MII_MII_TX_CLK.

Table continues on the next page...

Table 553. Module signals (continued)

Port name (clock/signal)	I/O	Description
		• RMII: Bits[1:0] provide the RMII transmit data. The data is valid only when the MII_RMII_TX_EN signal is high. MII_RMII_TXD[3:0] is synchronous to MII_RMII_TX_CLK.
MII_CRS (Signal)	I	This signal is valid only in MII mode. The PHY drives this signal high when the transmit or receive medium is not idle, and drives the signal low when both these mediums are idle. The signal is not synchronous to any clock.
MII_COL (Signal)	I	This signal is valid only in MII mode. The PHY drives this signal high when a collision is detected on the medium. The signal is not synchronous to any clock.
MII_RMII_RX_DV (Signal)	I	The PHY drives this signal. It performs multiple functions depending on the selected PHY interface as described in this list: • MII: Indicates that the data on the MII_RXD bus is valid. It remains high continuously from the first recovered byte or nibble of the packet through the final recovered byte or nibble of the packet. MII_RMII_RX_DV is synchronous to MII_RX_CLK. • RMII: Contains the CRS and data valid information of the receive interface. MII_RMII_RX_DV is synchronous to MII_RMII_TX_CLK.
MII_RMII_RX_ER (Signal)	I	The PHY drives this signal. It performs multiple functions depending on the selected PHY interface as described in this list: • MII: Indicates an error or carrier extension in the received packet of the MII_RXD[3:0] bus. MII_RMII_RX_ER is synchronous to MII_RX_CLK. • RMII: Is not used.
MII_RMII_RXD[3:0] (Signals)	I	This is a group of data signals received from the PHY. These signals perform multiple functions depending on the

Table continues on the next page...

Table 553. Module signals (continued)

Port name (clock/signal)	I/O	Description
		selected PHY interface as described in this list: • MII: Bits [3:0] provide the MII receive data nibble. The data is valid only when the MII_RMII_RX_DV signals are high. MII_RMII_RXD[3:0] is synchronous to MII_RX_CLK. • RMII: Bits [1:0] provide the RMII receive data. The data is valid only when the MII_RMII_RX_DV signal is high. MII_RMII_RXD[3:0] is synchronous to MII_RMII_TX_CLK.
MII_RMII_MDC	O	MAC provides timing reference for MII_RMII_MDIO or MII through this periodic clock. The application clock generates this clock through a clock divider that MAC_MDIO_Address[CR] controls.
MII_RMII_MDIO (Signal)	I/O	MDIO uses this signal to transfer control and data information to PHY.

75.5 Using PHY interfaces

This section is Synopsys Proprietary. Used with permission.

This module support the following modes:

- RMII 10/100 Mbps interface
- MII 10/100 Mbps interface

Phy_intf_sel input signal decides which mode is selected. Samples the Phy_intf_sel signal at reset. See External signals for more information. MAC_Configuration[PS] and MAC_Configuration[FES] selects the mode's speed.

The module supports the IEEE802.3-2015 specification for MII and RMII specification version 1.2 from RMII consortium.

NOTE

RMII reference clock (50 MHz) can be fed to IP from an external source or internally from PLL on SoC.

The module supports the access of the phy registers through SMA. It is a two wire Station management interface (MIM):

1. MDC
2. MDIO

According to IEEE 802.3 specification, maximum operating frequency of MDC is 2.5 Mhz which system clock derives using a divider. MAC_MDIO_Address[CR] programs to generate different MDC clock frequency.

SMA supports MDIO clause 45 and clause 22 frame structure per the IEEE802.3 specification. Writing 1 to MAC_MDIO_Address[C45E] enables the clause 45 frame structure.

The MII interface reduces the accuracy of the 1588 timestamp if it is overclocked to run at 200 Mbps. This happens because the MAC logic uses the MII interface speed,to adjust or compensate for the timestamps taken at MII, as compared to the time generated in the PTP clock domain as specified in MAC_Configuration[FES].

75.6 VLAN and double VLAN insertion, deletion, replacement and tagging

This section and all its sub-sections are Synopsys proprietary. These are used with permission.

The following sections describe the VLAN and double VLAN features.

75.6.1 Double VLAN processing

In the double VLAN tagging processing, the module supports the processing of two VLAN tags. The two VLAN tags are:

1. Inner VLAN tag (C-VLAN)
2. Outer VLAN tag (S-VLAN)

If there is only one tag in a packet then it is considered as an outer tag.

The module uses MAC_VLAN_Incl and MAC_Inner_VLAN_Incl and MAC_VLAN_Tag registers to support the following functions in a double VLAN processing feature.

- Insertion, replacement, or deletion of up to two VLAN tags in the transmit path.
- Packet filtering and stripping on the basis of any one of the two VLAN tags in the receive path. Stripping and providing up to two VLAN tags in the receive path as a part of the receive status.

75.6.1.1 Transmit path

Table 554. Double VLAN processing features in transmit path

Feature	Description
Support for C-VLAN and S-VLAN tag types	The inner or outer VLAN tag can be of C-VLAN and S-VLAN type. MAC_VLAN_Incl[CSVL] and MAC_Inner_VLAN_Incl[CSVL] specifies the VLAN type. The module supports the processing of any sequence of outer and inner VLAN tags. **NOTE** The module does not support the C-VLAN and S-VLAN sequence. MAC does not examine whether the packet, which the application provides has a valid sequence of the VLAN tag types or the insertion or replacement operation results in an invalid sequence of VLAN tag type. Therefore, the application must provide a correct sequence of VLAN tag types and program the MAC in such a way that it results in correct sequence of VLAN tag types in the transmitted packet. The application must ensure the following: • The inner tag should not be S-VLAN when outer C-VLAN Tag insertion is enabled. • The outer tag should not be C-VLAN when inner S-VLAN Tag insertion is enabled. • The inner tag should not be S-VLAN when C-VLAN replaces the outer tag. • The outer tag should not be C-VLAN when S-VLAN replaces the inner tag.
VLAN tag deletion	MAC_VLAN_Incl[VLC] or MAC_Inner_VLAN_Incl[VLC] respectively enables the VLAN tag deletion for an outer or inner tag. When you enable the VLAN deletion, MAC deletes the tag present at the corresponding position. When a packet has only one tag, it is considered as the outer tag. If you enable the inner tag deletion and if there is only one tag in the packet, then the MAC does not delete the tag.
VLAN tag insertion or replacement	MAC_VLAN_Incl[VLC] or MAC_Inner_VLAN_Incl[VLC] respectively enables the VLAN tag insertion or replacement for an outer or inner tag. When you enable the VLAN tag insertion or replacement, the MAC_VLAN_Incl[VLTI] or MAC_Inner_VLAN_Incl[VLTI] determines whether the VLAN tag must be fetched from the register or from the control word.

75.6.1.2 Receive path

Table 555. Double VLAN processing in receive path

Feature	Description
Outer or inner VLAN tag-based filtering	Mac can filter packets through the ERIVLT bit on the basis of the outer or inner VLAN tag.
C-VLAN or S-VLAN tag-based filtering	Mac can filter packets through the ERSVLM bit on the basis of the C-VLAN or S-VLAN type.
Outer and inner VLAN tag stripping	Mac can strip the outer and inner VLAN tags from received frame on the basis of the EVLS and EIVLS bits.
16-bit outer and inner VLAN tag and type in Rx status	Mac can provide the 16-bit outer and inner VLAN tag and type in the Rx status on the basis of the EVLRXS and EIVLRXS bits, respectively.
Disabling or skipping checking of outer VLAN tag type	Mac can disable or skip checking of an outer VLAN tag type to match C-VLAN or S-VLAN on the basis of the DOVLTC bit.

75.6.1.3 Source address and VLAN insertion, replacement, or deletion

This module supports the SA, and VLAN insertion, replacement and deletion feature. The SA and VLAN fields are Tx packet-related control information that interfaces provide as part of the control word. IP supports the feature to insert or replace the source address on the basis of the information in the MAC address registers, and it also supports the feature to insert, replace, or delete the VLAN fields (VLAN type and VLAN tag) on the basis of the setting of the MAC_VLAN_Incl[VLTI]. SA insertion or replacement feature is enabled for all the transmit packets or selective packets. Similarly, VLAN insertion, replacement, or deletion feature is enabled for all the Tx packets or selective packets.

75.6.1.4 Programming source address insertion or replacement

You can use the SA insertion or replacement feature to instruct MAC to perform the following actions for Tx packets:

- Insert the content of the MAC address registers in the SA field
- Replace the content of the SA field with the content of the MAC address registers

When SA insertion is enabled, the application must ensure that the packets sent to MAC do not have the SA field. The MAC does not check whether the SA field is present in the transmit packet and it inserts the content of MAC address registers in the SA field. Similarly, when the SA replacement is enabled, the application must ensure that the SA field is present in the packets sent to the MAC.

MAC replaces the six bytes following the DA field in the transmit packet with the content of the MAC address registers.

- Configure MAC_Configuration[SARC] to enable the SA insertion or replacement for all Tx packets.
- Enable the SA insertion for selective packet by configuring the SA insertion control field (bits [25:23] of TDES3) in the first transmit descriptor of the packet. When you write 1 to bit 25 of TDES3, the SA insertion control field indicates insertion or replacement by MAC Address1 registers. When bit 25 of TDES3 resets, it indicates insertion or replacement by MAC Address 0 registers.

If you do not enable the MAC Address1 registers, then the MAC Address0 registers are used for insertion or replacement irrespective of the value of the most-significant bit of the SA insertion control field.

75.6.1.5 Programming VLAN insertion, replacement, or deletion

You can use the VLAN insertion, replacement, or deletion feature to instruct MAC to perform the following actions for the Tx packets:

- Delete the VLAN type and VLAN tag fields.
- Insert or replace the VLAN type and VLAN tag fields.

Insertion or replacement is done on the basis of the setting of the MAC_VLAN_Incl[VLTI] as described in theTable 556:

Table 556. VLAN insertion or replacement based on VLTI bit

Condition	Description
VLTI bit=1	MAC inserts or replaces the following: • VLAN type field (C-VLAN or S-VLAN as indicated by the MAC_VLAN_Incl[CSVL]) • VLAN tag field depending on the content of the VT field of transmit context descriptor of the packet
VLTI bit resets	MAC inserts or replaces the following: • VLAN type field (C-VLAN or S-VLAN as indicated by the MAC_VLAN_Incl[CSVL]) • VLAN tag field with the MAC_VLAN_Incl[VLT]

When the VLAN replacement or deletion is enabled, MAC checks whether the VLAN type field (0x8100 or 0x88a8) is present after the DA and SA fields in the transmit packet. The replace or delete operation does not occur if the VLAN type field is not detected in two bytes following the DA and SA fields. However, when the VLAN insertion is enabled, MAC does not check the presence of VLAN type field in the transmit packet and just inserts the VLAN type and VLAN tag fields.

Configuring MAC_VLAN_Incl[VLC] and MAC_VLAN_Incl[VLP] enables the VLAN insertion, replacement, or deletion feature for all Tx packets.

Configuring the VTIR field of TDES2 normal descriptor enables the VLAN insertion, replacement, or deletion for selective packets.

In addition, the VLP (VLAN Priority control) field must reset in MAC_VLAN_Incl (for outer VLAN) and MAC_Inner_VLAN_Incl register (in inner VLAN) for MAC to take the control inputs from the host, depending on the configuration.

75.7 Packet Filtering

This section and all its sub-sections are Synopsys proprietary. Used with permission.

MAC receiver contains various packet filtering scheme. Figure 418 shows the filtering sequence in their precedence order of received packet.

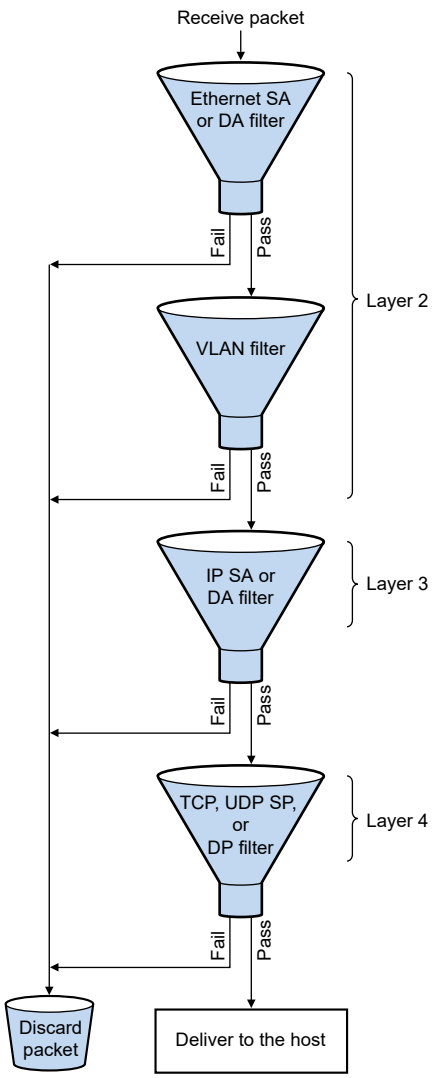


Figure 418. Packet filtering sequence

If you do not enable any of the layer filters, then that filter is bypassed and the subsequent filter is applied. Discard the packet that fails any of the filters. However, you can forward the discarded packet to the host on the basis of the register control.

75.7.1 Source address or destination address filtering

The address filtering module of MAC checks the SA and DA fields of each incoming packet. Programming of different types of address filtering is described in the sections that follows.

75.7.1.1 Unicast destination address filtering

MAC supports up to three MAC addresses for unicast perfect filtering. If perfect filtering is selected (MAC_Packet_Filter[HUC] resets), MAC compares all 48 bits of received unicast address with the programmed MAC address for any match. Default MacAddr0 is always enabled. MAC selects the MacAddr1 to MacAddr2 addresses with an individual enable field. Each byte of MacAddr1 to MacAddr2 can be masked when you compare them with corresponding received DA byte by writing 1 to the corresponding Mask byte control field in the register. This enables group address filtering for the DA.

In hash filtering mode (when you write 1 to HUC bit), MAC performs imperfect filtering by using a 64-bit hash table for unicast addresses . For hash filtering, MAC uses the upper 6 bits CRC of the received destination address to index the content of the hash table. A value of 00000 selects bit 0 of selected register, and a value of 11111 selects bit 63 of hash table register. If the value of

the corresponding bit (indicated by the 6-bit CRC) is 1, the unicast packet is considered to have passed the hash filter, otherwise, the packet is considered to have failed the hash filter.

75.7.1.2 Multicast destination address filtering

Enable MAC_Packet_Filter[PM] to pass all the multicast packets or disable to do multicast addresses filtering on the basis of MAC_Packet_Filter[HMC]. Compare the multicast address with the configured MAC destination address registers (1–2), also supports the group address filtering.

In Hash filtering mode, MAC performs imperfect filtering using a 64-bit hash table. MAC uses the upper 6-bits CRC of received multicast address to index the content of the hash table. A value of 000000 selects bit 0 of selected register and a value of 111111 selects bit 63 of the hash table register. The multicast packet is considered to have passed the hash filter if the value of the corresponding bit is equal to 1. Otherwise, the packet is considered to have failed the hash filter.

75.7.1.3 Hash or perfect address filtering

Configure MAC_Packet_Filter[HPF] to enable or disable the destination address filtering either by hash filter or perfect filter. Also configure MAC_Packet_Filter[HUC] and MAC_Packet_Filter[HMC] to select the hash filtering or perfect filtering for unicast or multicast packet.

75.7.1.4 Broadcast address filtering

MAC does not filter any broadcast packets by default. You must write 1 to MAC_Packet_Filter[DBF] to reject the broadcast packets.

75.7.1.5 Unicast source address filtering

MAC by default compares the source address field with the value configured in the source address registers. Configure bit 30 of MAC address register [1-2] to use it for source address instead of destination address comparison. MAC also supports group filtering with SA. You can filter a group of addresses by masking one or more bytes of the address. MAC drop the packets that fail the SA filter if you write 1 to MAC_Packet_Filter[SAF] field. Otherwise, the result of the SA filter is given as a status bit in the receive status word. When you write 1 to the SAF field, the SA filter and DA filter results ends to decide whether you can forward the packets. This means that the packet is dropped if either filter fails. The packet is forwarded to the application only if the packet passes both filters in-order.

75.7.1.6 Inverse filtering

For DA and SA filtering, you can invert the filter-match result at the final output by writing 1 to the DAIF and SAIF fields of MAC_Packet_Filter register. The DAIF field is applicable for both the unicast and multicast DA packets. The unicast or multicast destination address filter result is inverted in this mode. Similarly, writing 1 to the SAIF field reverses the result of unicast SA filter.

Table 557 and Table 558 summarize the DA and SA filtering on the basis of the type of packets received.

Table 557. Destination address filtering

Packet type	PR	HPF	HUC	DAIF	HMC	PM	DBF	DA filter operation
Broadcast	1	X	X	X	X	X	X	Pass
	0	X	X	X	X	X	0	Pass
	0	X	X	X	X	X	1	Fail
Unicast	1	X	X	X	X	X	X	All packets pass
	0	X	0	0	X	X	X	Pass on perfect or group filter match

Table continues on the next page...

Table 557. Destination address filtering (continued)

	0	X	0	1	X	X	X	Fail on perfect or group filter match
	0	0	1	0	X	X	X	Pass on hash filter match
	0	0	1	1	X	X	X	Fail on hash filter match
	0	1	1	0	X	X	X	Pass on hash or perfect or group filter match
	0	1	1	1	X	X	X	Fail on hash or perfect or group filter match
Multicast	1	X	X	X	X	X	X	Pass all packets
	X	X	X	X	X	1	X	Pass all packets
	0	X	X	0	0	0	X	Pass on perfect or group filter match and drop pause packets if PCF = 0x
	0	0	X	0	1	0	X	Pass on hash filter match and drop pause packets if PCF = 0x
	0	1	X	0	1	0	X	Pass on hash or perfect or group filter match and drop pause packets if PCF = 0x
	0	X	X	1	0	0	X	Fail on perfect or group filter match and drop pause packets if PCF = 0x
	0	0	X	1	1	0	X	Fail on hash filter match and drop pause packets if PCF = 0x
	0	1	X	1	1	0	X	Fail on hash or perfect/group filter match and drop pause packets if PCF = 0x

Table 558. Source address filtering

Packet type	PR	SAIF	SAF	SA filter operation
Unicast	1	X	X	Pass all packets.
	0	0	0	Pass status on perfect or group filter match but do not drop packets that fail
	0	1	0	Fail status on perfect or group filter match but do not drop packet
	0	0	1	Pass status on perfect or group filter match and drop packets that fail
	0	1	1	Fail status on perfect or group filter match and drop packets that fail

NOTE

When you write 1 to the MAC_Packet_Filter[RA] field, all packets are forwarded to the system along with the correct result of the address filtering in the Rx Status.

75.7.2 VLAN filtering

VLAN tag filter can be done either by perfect match or hash table. Depending upon the configuration, MAC can compare either lower 12 bits or all 16 bits of received VLAN tag for perfect match. MAC forward the VLAN tagged packet with match status when it drop all the packets which do not match.

MAC uses 16-bit VLAN hash table for group address filtering on the basis of the VLAN tag. If MAC enables the hash filtering then the most significant 4-bit of CRC-32 of VLAN tag are used to index the content of the MAC_VLAN_Hash_Table. A value of 1 in the VLAN hash table register, corresponding to the index indicates that the VLAN tag of the packet has matched and the packet should be forwarded. A value of 0 indicates that VLAN-tagged packet should be dropped.

MAC supports the inverse matching for VLAN packets. In the Inverse matching mode, the packet must be dropped when the VLAN tag of a packet matches the perfect or hash filter. If MAC enables the VLAN perfect and VLAN hash match, then a packet is considered as matched if either the VLAN hash or the VLAN perfect filter matches. When inverse match is set, MAC forwards a packet only when both the perfect and hash filters indicate mismatch.

Table 559 specifies the different possibilities for VLAN matching and the final VLAN match status. When you write 1 to MAC_Packet_Filter[RA] all packets are received and it indicates the VLAN match status in the VF field of RDES2 normal descriptor (write-back format). When the RA field is not set and when you write 1 to MAC_Packet_Filter[VTFE] the packet drops if the final VLAN match status fails

When VLAN VID programs to 0 in the VL field of MAC_VLAN_Tag register, consider that all the VLAN-tagged packets are perfectly matched but the status of the VLAN hash match depends on the VTHM and VTIM fields in MAC_VLAN_Tag register.

Table 559. VLAN match status

VID	VLAN perfect filter match result	VTHM bit	VLAN hash filter match result	VTIM bit	Final VLAN match status
VID = 0	Pass	0	X	X	Pass
	Pass	1	X	0	Pass
	Pass	1	Fail	1	Pass
	Pass	1	Pass	1	Fail
VID1= 0	Pass	X	X	0	Pass
	Fail	0	X	0	Fail
	Fail	1	Fail	0	Fail
	Fail	1	Pass	0	Pass
	Fail	0	X	1	Pass
	Pass	X	X	1	Fail
	Fail	1	Pass	1	Fail
	Fail	1	Fail	1	Pass

In Table 559, X represents any value.

75.7.3 VLAN filter fail packets queue

When you enable VLAN filtering, route the VLAN filter fail packets to a programmable queue (VFFQ) when the value of MAC_Packet_Filter[RA] = 1 or MAC_Packet_Filter[VTFE] = 0 and write 1 to MAC_RxQ_Ctrl4[VFFQE] for the queue.

Route the packets that passes the VLAN filtering to Rx queues on the basis of the VLAN tag priority field by configuring the PSRQ field in the corresponding MAC_RxQ_Ctrl2 and MAC_RxQ_Ctrl3 registers. Discard the packets that fail the VLAN filter if RA=0 or VTFE=1. However, when RA=1 or VTFE=0, forward the VLAN filter fail packets to the application. In such case, when you write 1 to the VLAN filter fail queue enable (VFFQE) field, the VLAN filter fail packets forward to the Rx queue number programmed in the VFFQ field. If the value of VFFQE=0, the VLAN priority mapping determines the Rx queue number per the PSRQ fields.

Table 560 shows the Rx queue routing table for unicast tagged packets, with DA or SA filter enabled.

Table 560. Rx queue routing table for unicast tagged packets

RA	VFTE	SA or DA filter result	VLAN filter result	VFFQE	Queue routing
X	X	Pass	Pass	X	PSRQ
0	0	Pass	Fail	0	PSRQ
0	0	Pass	Fail	1	VFFQ
0	X	Fail	X	X	Dropped
0	1	Pass	Fail	X	Dropped
1	X	Fail	X	0	UFFQ*/PSRQ
1	X	Fail	X	1	UFFQ*/VFFQ
1	X	Pass	Fail	0	PSRQ
1	X	Pass	Fail	1	VFFQ

X : Don't care condition * : When UFFQE is enabled else PSRQ.

75.7.4 Extended receive VLAN filtering and routing

When the extended Rx VLAN filtering and routing is enabled then both the perfect filtering and hash filtering can be enabled. The overall VLAN filter result is based on the perfect filter result and hash filter result (if enabled). Filter result is passed to application as part of the status bit. Extended routing will take place only if VLAN filter has passed. Routing is only based on perfect filter result. Each perfect filter has a DMA channel enable and DMA channel number filed which must be programmed for routing.

See the extended VLAN based DMA selection in Dynamic (per packet) mapping for more information about routing.

75.7.4.1 Comparison mode

Application has these comparison option for each VLAN tag filter:

- Programs MAC to compare the inner or outer VLAN tag either with 12 bits or 16 bits programmed VID.
- Selects whether the VID comparison is for SVLAN or CVLAN type frames, if type check is enabled for a filter.

75.7.4.2 Filtering

Perfect filtering is done on the basis of the MAC_VLAN_Tag_Filter registers. MAC compares the relevant VLAN tag ID and gives a result for each VLAN tag filter.

The results for each VLAN tag filter are:

- Pass- If any one of the VLAN tag filters gives a match.
- Fail- If the frame mismatches all the filters.

This behavior is applicable only when the inverse filtering is not enabled in MAC_VLAN_Tag_Ctrl.

If inverse filtering is enabled and the frame mis-matches all the relevant filters then it is considered to have passed the VLAN filter. If the frame matches any one of the relevant filters then it is considered to have failed. The frame is bypassed to the application if none of the enabled filters can perform a comparison or if none of the filters are enabled.

The overall filter result and the programming on MAC_Packet_Filter[VTFE] and MAC_Packet_Filter[RA] determines if the frame will drop or it is forwarded to the application. If the value of RA = 1 or VTFE = 0, then the frame is always forwarded whether the filter result is a pass or fail, . If the value of RA = 0 and VTFE = 1, only then, if the VLAN tag filter result is a pass the MAC forwards the frame. If the frame is forwarded to the application, then the relevant filter result is indicated through the status bits.

75.7.4.3 Filter status

The extended receive VLAN filtering and routing feature provides two status fields to show the comparison result of the VLAN tags.

By default, MAC indicates the VLAN filter status through one bit in the status VF field in RDES2. When you enable the extended RX VLAN filtering and routing, two status fields will show the comparison result of the VLAN tags. The outer VLAN tag filter pass and inner VLAN tag filter pass bits are defined in the following positions. The status indicated through these fields depends on the programming as described below.

In RDES2:

- Bit 15 – Outer VLAN tag filter status
- Bit 14 – Inner VLAN tag filter status

In ARI status: MAC filter status:

- Bit 15 – Outer VLAN tag filter status
- Bit 14 – Inner VLAN tag filter status

In MRI status:

- Bit 47 – Outer VLAN tag filter status
- Bit 46 – Inner VLAN tag filter status

Outer VLAN tag filter status (OTS)

- In perfect filtering, without enabling inverse filtering, if you write 1 to this field, it indicates that the frame's outer VLAN tag has matched one of the VLAN tag filters.
- If this field resets, it indicates that the frame's outer VLAN tag has either failed the relevant outer VLAN tag filters or bypassed them.
- This field resets if none of the filters are enabled for outer VLAN tag comparison.
- If inverse filtering is enabled and if you write 1 to this field, then the frame's VLAN tag has passed all the relevant VLAN tag filters. If it is resets, then it has failed at least one filter or bypassed all the filters programmed for outer VLAN tag comparison.
- This bit is valid for both single and double VLAN tagged frames.

Inner VLAN tag filter status (ITS)

- In perfect Filtering, without enabling inverse filtering, if you write 1 to this field, it indicates that the frame's inner VLAN tag has matched one of the VLAN tag filters.
- If this field resets, it indicates that the frame's inner VLAN tag has either failed the relevant inner VLAN tag filters or bypassed them.
- This field resets if none of the filters are enabled for inner VLAN tag comparison.
- If Inverse Filtering is enabled and if you write 1 to this field, then the frame's VLAN tag has passed all the relevant VLAN tag filters. If it resets, then it has failed at least one filter or bypassed all the filters programmed for inner VLAN tag comparison.

- This bit is valid for only double VLAN tagged frames, when double VLAN processing is enabled.

The application must observe the status fields and the program to determine if the frame has passed or failed the VLAN filter.

Table 561 and Table 562 describes the possible filter combinations and the corresponding filter results and they also explain the scenarios when the double VLAN processing and the hash VLAN filter are enabled in the design.

Legend for the table OTS and ITS Bit Values with at least 1 perfect filter enabled

- VTIM: VLAN tag inverse match enable – bit 17 in VLAN_Tag_Ctrl register.
- HFO: Hash filter enabled for outer VLAN tag comparison - bit 25 and bit 27 in VLAN_Tag_Ctrl register.
- HFI: Hash filter enabled for inner VLAN tag comparison – bit 25 and bit 27 in VLAN_Tag_Ctrl register.
- PFO – Perfect filter comparison enabled for outer VLAN tag - Any of the MAC_VLAN_Tag_Filter registers is enabled (write 1 to bit 16) and programmed for outer VLAN tag comparison (write 0 to bit 20).
- PFI – Perfect filter comparison enabled for Inner VLAN Tag - Any of the MAC_VLAN_Tag_Filter registers is enabled (write 1 to bit 16) and programmed for inner VLAN tag comparison (write 1 to bit 20).
- OTS – Outer VLAN tag filter status
- ITS – Inner VLAN tag filter status

Table 561 shows the possible values of status bits (OTS and ITS) when at least one perfect filter is enabled.

Table 561. OTS and ITS bit values with at least 1 perfect filter enabled

VTIM	HFO	HFI	PFO	PFI	OTS	ITS
0	0	0	0	1	0	1/0
0	0	0	1	0	1/0	0
0	0	0	1	1	1/0	1/0
0	1	0	1	1	1/0	1/0
0	1	0	1	0	1/0	0
0	1	0	0	1	1/0	1/0
0	0	1	1	1	1/0	1/0
0	0	1	1	0	1/0	1/0
0	0	1	0	1	0	1/0
1	0	0	0	1	0	1/0
1	0	0	1	0	1/0	0
1	0	0	1	1	1/0	1/0
1	1	0	1	1	1/0	1/0
1	1	0	1	0	1/0	0
1	1	0	0	1	1/0	1/0

Table continues on the next page...

Table 561. OTS and ITS bit values with at least 1 perfect filter enabled (continued)

VTIM	HFO	HFI	PFO	PFI	OTS	ITS
1	0	1	1	1	1/0	1/0
1	0	1	1	0	1/0	1/0
1	0	1	0	1	0	1/0

Table 562 shows the possible values of status bits (OTS and ITS) when no perfect filters are enabled except the VLAN hash filter is enabled.

Table 562. OTS and ITS bit values with only VLAN hash filter enabled

VTIM	HFO	HFI	OTS	ITS
0	0	0	0	0
0	1	0	1/0	0
0	0	1	1/0	1/0
1	0	0	1/0	0
1	1	0	1/0	0
1	0	1	1/0	1/0

With no perfect filters enabled, any VLAN packet is considered to have bypassed the perfect filter. If Hash Filter is enabled for one of the Tags, then the respective Status bit depends on the Filter's result. The Status bits are set to 0 if VLAN Hash Filter is also not enabled.

If the value of ITS/OTS is shown as 1/0; then it indicates that the final result is dependent on the enabled relevant filter's result.

Example 1: The second row of OTS and ITS bit values with only VLAN hash filter enabled indicates that at least one Perfect Filter is enabled for Outer VLAN tag comparison and none of the filters are enabled for Inner VLAN tag comparison. Inverse VLAN Filtering is not enabled. The bit OTS is given as 1/0. If the received frame passes at least one of the enabled Outer VLAN Tag filters then the bit is set to 1. If the frame doesn't pass any of the enabled Outer VLAN Tag filters, then the bit is set to 0.

Example 2: The last row of table "OTS and ITS bit values with only VLAN hash filter enabled" indicates that the inverse filtering is enabled, hash filter and at least one perfect filter is enabled for Inner VLAN Tag comparison, then if the received frame's Inner VLAN tag mismatches with both the Hash Filter and all the enabled Perfect filters, then the frame will have the ITS bit set to 1 else it is set to 0. OTS will be set to 0 as no comparison is performed.

75.7.4.4 Stripping

Each VLAN tags has individual control over stripping. The programming options of always strip, never strip, strip on pass and strip on fail are available. Inner or outer VLAN tag stripping is based on the pass or fail results of the individual tag. If all the relevant filters bypasses a tag, stripping is not applicable for the tag.

- If strip on pass is enabled for the outer VLAN tag, then the stripping occurs only if the outer VLAN tag has passed the relevant filters. The outer VLAN tag filter result field will be 1.
- If strip on fail is enabled for the outer VLAN tag, then the stripping occurs only if the outer VLAN tag has failed relevant filters. The outer VLAN tag filter result field will reset.
- If the outer VLAN tag of the received frame is bypassed by the entire filter (no comparison has been made), then the tag is not stripped, though the status bit is 0.

- As multiple filters are enabled, it is possible that the received VLAN frame could have matched any one or more of the filters. The VLAN Tag's value is not always deterministic from the filter status bits.

If an application strips the VLAN tag on the basis of the filter result, it might lose the VID. You can use it, if stripping is enabled for any of the tags, place the tag in the status. For this the software or application will enable the respective VLAN tag in status bit - 24 or 31 in the MAC VLAN tag control register.

See section Programming guidelines for extended VLAN filtering and routing on receive for more information.

75.7.5 Layer 3 and layer 4 filtering

IP supports the layer 3 based packet filtering which is an IP source or destination address filtering in the IPv4 or IPv6 packets, and layer 4 based packet filtering which is a source or destination port number filtering in TCP or UDP. When you enable layer 3 and layer 4 packet filtering then packets are filtered in the following manner.

- Matched packets: MAC forward the packets that match all enabled fields to the application along with the status. It gives the matched field status only if MAC_Configuration[IPC] = 1 and if one of the following conditions is true:
  - All enabled layer 3 and layer 4 fields match
  - At least one enabled field matches and other fields are bypassed or disabled

When you enable multiple layer 3 and layer 4 filters, any filter match is considered as a match. If more than one filter matches, MAC provides the status of the lowest filter where filter 0 is the lowest filter and filter 3 is the highest filter.

- Unmatched packets: MAC drop the packets that do not match any enabled fields. You can use the inverse match feature to block or drop a packet with specific TCP or UDP over IP fields and forward all other packets.
- Non-TCP or UDP IP packets: By default, all the non-TCP or UDP IP packets bypasses the layer 3 and layer 4 filters. You can optionally program MAC to drop all the non-TCP or UDP packets over IP packets.
- IP has a control register MAC_L3_L4_Control to control the layer L3 and L4 packet filtering along with address registers to program the layer L3 and L4 fields to be matched.

75.7.6 Flexible receive parser

When you enable this function, all incoming packets are parsed per the programmable instruction stored in memory. It perform these functions:

- Packet filter (Accept or reject)
- DMA channel selection

Flexible receive parser block operates over the first 128 bytes data of receive packet on the basis of 64 bytes of instruction on each packet. Figure 419 shows the functional block diagram of flexible receive parser.

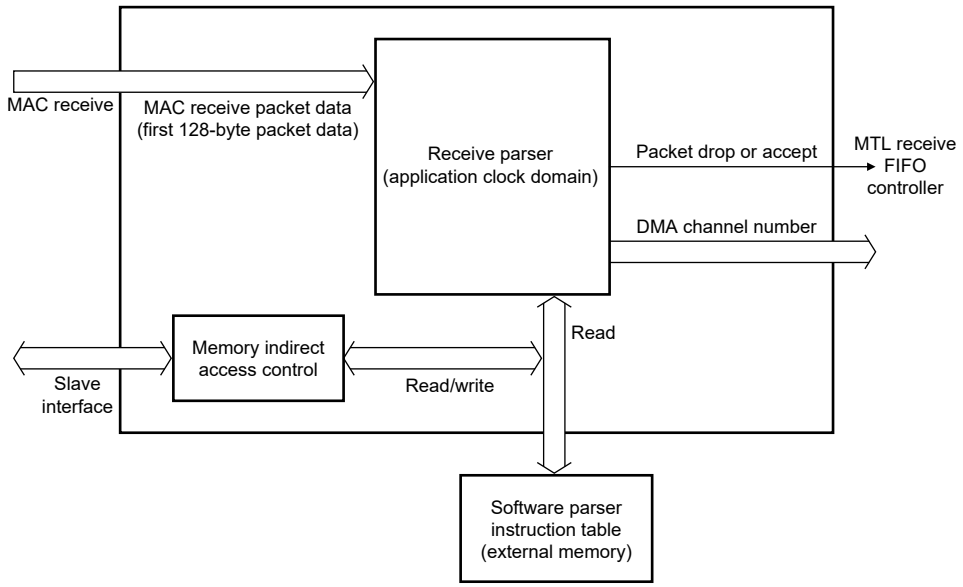


Figure 419. Functional block diagram of flexible receive parser

75.7.6.1 Instruction table format of the flexible receive parser

You must build the 96-bit wide instruction before enabling the receive parser feature and program each instruction as an entry in the instruction memory.

Table 563 shows the format of each entry in the instruction table .

Table 563. Rx parser instruction entry format

Field	Field Name	Description
31:0	MATCH_DATA	This is a 4-byte data. You can use it to compare with incoming packet data which starts at the frame offset as defined in [79:72] of the entry. The comparison is done only on those bits whose corresponding mask bits are 1 (MATCH_EN bits of [63:32]). See the notes (below this table) on the byte order relative to Ethernet arrival data and endianness which the QoS IP supports on the slave interface. The programming of this instruction table must adhere to the same.
62:32	MATCH_EN	When the MATCH_EN = 1, you can use the corresponding packet data bit for comparing. Otherwise, corresponding data bit is don’t care.
64	AF	Accept frame
65	RF	Reject frame
66	IM	Inverse match
67	NC	Next instruction control
71:68	Reserved	-

Table continues on the next page...

Table 563. Rx parser instruction entry format (continued)

Field	Field Name	Description
79:72	Frame offset	[77:72] indicates the frame offset in terms of 4 bytes. (Here [79:78] always 0) Compare the frame offset in the packet data for Match. This is in terms of 4 bytes. The value of actual frame offset in bytes are 0 01 42 8… ..63 252 The max value programmed in this instruction table is 128 B. The 'Actual frame offset' must not cross this limit.
87:80	OK index	If (NIC==0) the memory index you can use next when ENTRY_MATCH==1 and neither AF or RF = 1. If (NIC==1) the memory index you can use next when ENTRY_MATCH==0
95:88	DMA Ch No	Indicates the DMA channel number (1-bit for each). You can use this when ENTRY_MATCH ==1 and AF = 1 (Accept frame). The following bits give the encoding: bit[0]- DMA channel number o, bit[1]- DMA channel number , 1bit[2]- DMA channel number 2… bit[7]- DMA channel number 7. **NOTE** You can encode the DMA channel number using bit wise as QoS IP support multicasting or broadcast across DMA channels. See section 3.3.12 for more information and proper usage.
128:96	Reserved	0 (This field is only for software view and reserved for future enhancements. The memory width remains as 96-bits).

The parser begins parsing from the 0th entry, for each received packet. The subsequent parsing entry location (next entry or OK_INDEX[]) depends on the current entry parser result. Flexible receive parser flow chart shows the detailed flow.

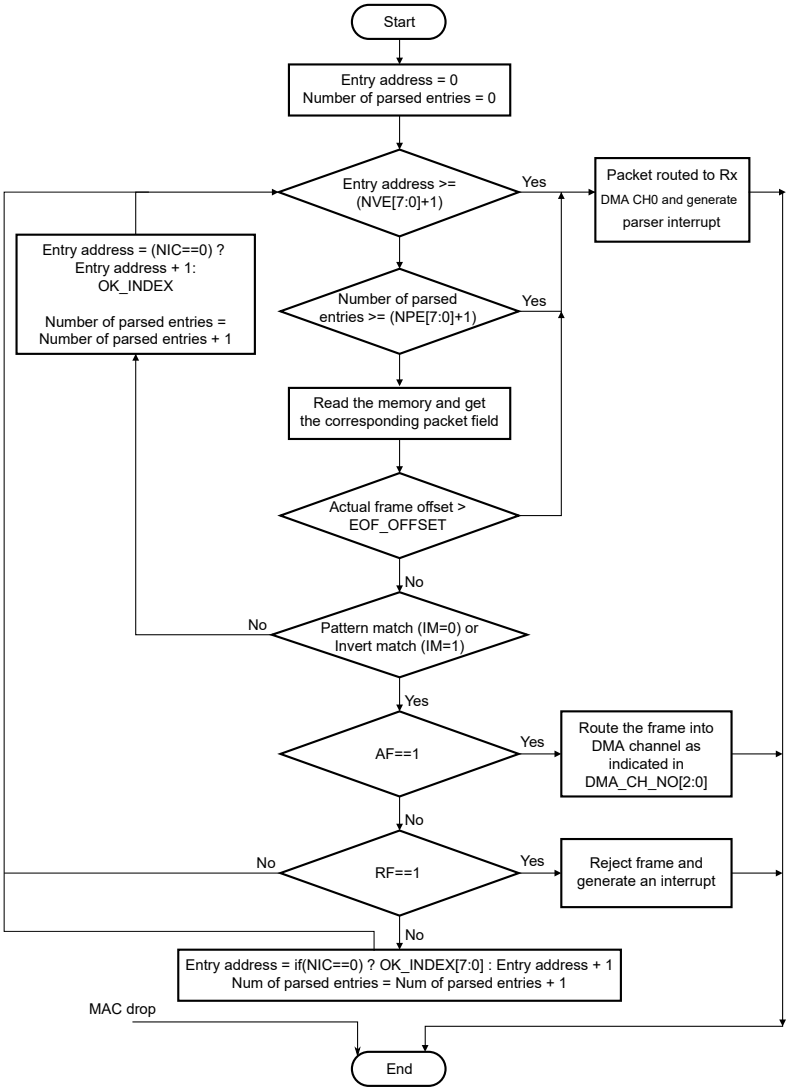


Figure 420. Flexible receive parser flow chart

The EOF_OFFSET shown in the figure is the least among the following:

- Actual packet size
- 128 Byte when frame pre-emption feature is not enabled or configured
- 128 Byte when frame pre-emption feature is enabled and packet is pre-emptive
- 64 Byte when frame pre-emption feature is enabled and packet is express packet

75.7.6.2 Number of valid entries (NVE)

NVE indicates the number of valid entries in the instruction table. The default value is 64. You can change this value by writing into MTL_RXP_Control_Status[NVE].

When parsing, an error is flagged in MTL_RXP_Interrupt_Control_Status[NVEOVIS], if the entry address is greater than MTL_RXP_Control_Status[NVE] + 1. IP generates an interrupt status (MTL_Interrupt_Status[MTLPIS]).

75.7.6.3 Number of parsable entries (NPE)

NPE indicates the number of entries that you can parse on an incoming packet. The default value is 64. Write into MTL_RXP_Control_Status[NPE] to change this value.

When parsing, an error is flagged in MTL_RXP_Interrupt_Control_Status[NPEOVIS], if the number of parsed of parsed entries is more than MTL_RXP_Control_Status[NPE] + 1. IP generates an interrupt status (MTL_Interrupt_Status[MTLPIS]).

NVE[] indicates the number of valid entries in the receive parser memory which you have built and NPE[] indicates the worstcase parsable entries considering the various possible paths that the parser can take. (NVE[] >= NPE[]).

75.7.6.4 Frame offset

It indicates the frame offset, in terms of 4 bytes which you use when comparing against the current table entry.

In an entry, an error is flagged in MTL_RXP_Interrupt_Control_Status[FOOVIS] in any of the following cases:

When frame offset is more than:

- Packet size
- 128B if pre-emption is disabled
- 128B if pre-emption is enabled and packet is pre-empitble
- 64B if pre-emption is enabled and packet is an express packet

IP generates an interrupt status (MTL_Interrupt_Status[MTLPIS]).

NOTE

If the frame ends after the (frame_offset *4) bytes but before ((frame_offset *4)+4) bytes, the receive parser declares a frame offset error if you enable comparison (via MASK bits) for the non-received bytes; from (frame_offset *4) to ((frame_offset *4)+4).

75.7.6.5 Frame reject

You can drop the frame in cases, where an entry matches (considering the inverse match, IM bit in the instruction table) and frame accept bit is not 1, and frame reject bit (RF) = 1.

When a frame drops because RF = 1, IP generates an interrupt status (MTL_Interrupt_Status[MTLPIS]).

75.7.6.6 Out of order processing

The Rx parser flow can be out of order. The decision tree is not based on the order in which packet arrives. This implies that the next entry and next frame offset (OK_INDEX[] and FRAME_OFFSET[]) can be less than the current entry address and current frame offset.

75.7.6.7 DMA channel selection

When you enables the receive parser (MTL_Operation_Mode[FRPE] = 1) it overrides the MAC DMA selection criteria.

When you disables the receive parser (MTL_Operation_Mode[FRPE] = 0) the MAC/MTL determines the DMA channel number.

75.7.6.8 Receive queue selection

The existing MAC functionality determines the receive queue, irrespective of whether the receive parser is enabled or disabled. This is because the receive parser is mainly designed to select the DMA channel and the MAC decides the receive queuing to enable the proper functioning of the pause or PFC feature.

NOTE

When there are multiple queues and multiple DMA Channels are enabled, a particular DMA channel might receive out of order packets because queue selection is based on MAC criteria (VLAN Priority and other criteria) and Rx DMA channel selection is based on receive parser.

75.7.6.9 MAC packet filtering/drop/error handling

Packet filtering is done on the basis of the received packet fields. You must disable the packet filtering features inside the MAC when the receive parser is enabled. Follow these steps to disable the packet filtering features inside the MAC :

1. Write 1 to the Promiscuous mode (MAC_Packet_Filter[PR]).
2. Write 1 to all other fields in MAC_Packet_Filter to its default values.

When MAC decides to drop the packet due to the receive MAC dependent error such as:

- GMII error
- Receive watchdog error
- CRC error
- Giant frame error

the packet drops irrespective of the receive parser decision.

75.7.6.10 Pad strip or CRC strip handling

Software controls the pad strip and CRC strip and it is applicable to all the received packets. Also,software must build the receive parser instructions accordingly, when you enable these features.

NOTE

Enable pad and CRC stripping in real use case.

75.7.6.11 VLAN strip handling

MAC supports stripping the outer VLAN as well as inner VLAN. When you program MAC to strip the VLAN, the receive parser considers those VLAN tags to be part of the incoming packet.

75.7.6.12 Multicast and broadcast support

IP supports multicast and broadcast packet and the packet route to the highest queue number.

When you enables the receive parser, the DMA channel numbers (DMA CH NO, Bits [95:88]) in the instruction table, decide the packet routing.

When you disables the receive parser, DCS (DMA Channel Select) field of the MAC_AddressX_High register determines the packet routing. See the Broadcast/multicast packet duplication.

NOTE

You can select multiple DMA channels for the packet routing because the DCS/DMA CH NO field is per the DMA channel control.

75.7.6.13 Pre-emption support

IP supports the frame preemption feature when the flexible receive parser feature is enabled. It has the following limitations:

- Number of bytes would be 64, so that the receive parser can operate on an express traffic
- Number of bytes would be 128, so that receive parser can operate on a pre-emptive traffic.

75.7.6.14 Software access to the flexible receive parser memory

Software can read and write into the receive parser memory. It uses the following registers via indirect addressing.

- MTL_RXP_Indirect_Acc_Control_Status
- MTL_RXP_Indirect_Acc_Data

For more details, see EMAC register descriptions.

75.7.6.15 Statistical counters

IP supports these statistics counters for flexible receive parser.

1. MTL_RXP_Drop_Cnt
2. DMA_RXP_Error_Cnt

DMA_CH(#i)_Rxp_Accept_cnt (i=0; i<2)

For more details, see EMAC register descriptions.

75.7.6.16 Changing the instruction table by software

Software can update the instruction table in the memory in the following ways:

- Disables the MAC receiver and receive parser by writing 0 to MAC_Configuration[RE] and MTL_Operation_Mode[FRPE]. It waits for receive parser to become inactive (RXPIA, MTL_RXP_Control_Status register bit[31]).
- (Optional) Programs in the entry address 0 to unconditionally (all MATCH_EN bit 0) skip to OK_INDEX[] (to certain location in the memory), where you can program it to reject or accept all the packets.

75.7.6.17 Receive cut-through functionality

In the Cut-Through mode, the receive controller transfer the received packets to DMA just after it receives the cut-through threshold amount of packet data (RTC field in the MTL_RxQx_Operation_Mode register).

However due to Rx parser's result worst case arrival time can be more than the programmed cut-through threshold. So, the receive controller delays the packet transfer to DMA accordingly, and the cut-through functionality is re-defined as follows:

The receive controller transfers packet to DMA after at least RTC number of bytes have been received and receive parser results are available.

75.7.6.18 Receive packet drop indication

In most cases, the packet drops internally in the receive queue. However, back-to-back packets can arrive for the same queue and the first packet's receive parser result is not available when the second packet arrives. In such cases, the packet cannot be flushed internally and forwarded to DMA or the application interface and indicates the status accordingly. See RDES2, bit 16(RXPD) in Receive normal descriptor (write-back format) for more information.

75.7.6.19 Runt packet handling

The receive parser can performs the operation with 64B minimum size packet. If the runt packet is received (MAC must indicate it), the receive parser assumes that it is dropped in the MAC, this is because, the back-to-back runt packets (< 64B) cannot be handled in the receive parser.

75.7.6.20 Uncorrectable ECC error handling

The packet does not drops when the parser detects an uncorrectable ECC error while parsing the parser memory. The packet is sent to the application with an error indication. The RXPI (RX parser incomplete) sets in the packet status along with error summary (ES) bit.

See the Receive normal descriptor (write-back format) for more information.

75.8 IEEE 1588 timestamp support

This section and all its sub-sections are Synopsys proprietary. Used with permission.

IP supports the IEEE 1588 precision time protocol (PTP). It is also able to do precise time stamping and captures incoming and outgoing frame because of the PTP protocol implementation in the IP. It supports all the clock types defined in IEEE 1588-2008. The typical frequency of PTP timestamp clock is 125 MHz.

IP supports these features:

- Provides an option to take snapshot of all packets or only PTP type packets
- Provides an option to take snapshot of only event messages
- Identifies the PTP message type, version, and PTP payload in packets sent directly over Ethernet and sends the status
- Provides an option to measure sub-second time in digital or binary format

75.8.1 Delay request-response mechanism

The system or network is classified into the master and slave nodes for distributing the timing and clock information. Figure 421 shows the process that PTP uses for synchronizing a slave node to a master node by exchanging PTP messages.

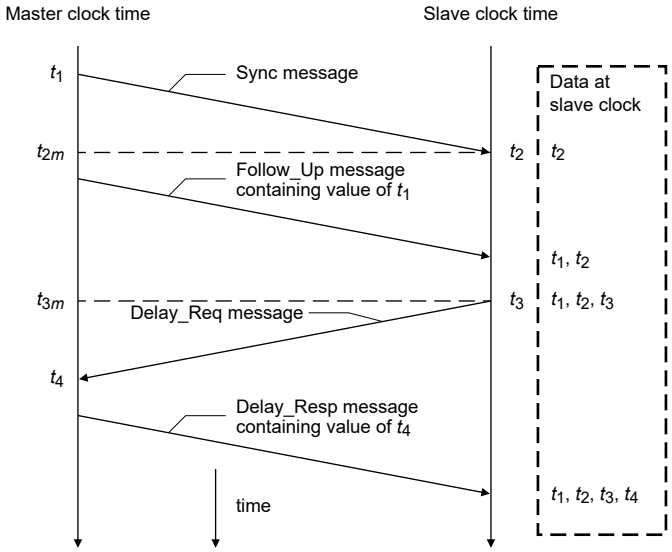


Figure 421. Networked time synchronization

As shown in the figure, PTP uses this process:

1. The master broadcast the PTP sync messages to all its nodes. The sync message contains the reference time information of the master. This message leaves the system of the master at t1. You must capture this time for Ethernet ports at GMII or MII.
2. The slave receives the sync message and it also captures the exact time, t2, using its timing reference.
3. The master sends a Follow_Up message to the slave, which contains t1 information for later use.
4. The slave sends a Delay_Req message to the master and note the exact time, t3, at which this packet leaves the GMII or MII interface.
5. The master receives the message, capturing the exact time t4, at which the message enters its system.
6. The master sends the t4 information to the slave in the Delay_Resp message.
7. The slave use the four values of t1, t2, t3, and t4 to synchronize its local timing reference to the master's timing reference.

Most of the PTP implementation is done in the software above the Ethernet layer. However, the hardware captures the exact time when specific PTP packets enter into or leave the Ethernet port at the MII interface. This timing information must be captured and returned to the software for proper implementation of PTP with high accuracy.

75.8.2 Peer-to-peer PTP transparent clock (P2P TC) message support

The IEEE 1588-2008 supports the peer-to-peer PTP (Pdelay) message in addition to the sync, delay request, follow-up, and delay response messages. Figure 422 shows the method to calculate the propagation delay in clocks supporting the peer-to-peer path correction.

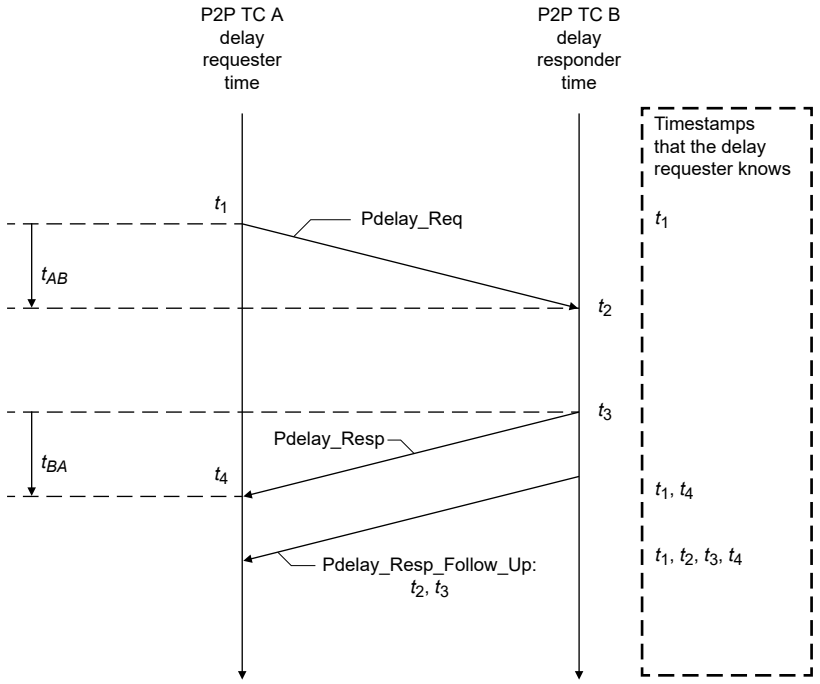


Figure 422. Propagation delay calculation in clocks supporting peer-to-peer path correction

As shown in the above figure, the propagation delay is calculated in the following way:

1. Port 1 issues a Pdelay_Req message and generates a timestamp (t1) for the Pdelay_Req message.
2. Port 2 receives a Pdelay_Req message and generates a timestamp (t2) for this message.
3. Port 2 returns a Pdelay_Resp message and generates a timestamp (t3) for this message. Port 2 returns the Pdelay_Resp message as quickly as possible after the receipt of the Pdelay_Req message, to minimize the errors because of any frequency offset between the two ports. Port 2 returns any one of the following:
   - Difference between the timestamps t2 and t3 in the Pdelay_Resp message
   - Difference between the timestamps t2 and t3 in the Pdelay_Resp_Follow_Up message
   - Timestamps t2 and t3 in the Pdelay_Resp and Pdelay_Resp_Follow_Up messages, respectively
4. Port 1 generates a timestamp (t4) after receiving the Pdelay_Resp message.
5. Port 1 uses all the four timestamps t1, t2, t3, and t4 to compute the mean link delay.

75.8.3 Timestamp correction

According to the IEEE 1588 specification, you must capture a timestamp when the PTP message timestamp point (leading edge of the first bit of the octet immediately following the start frame delimiter octet) crosses the boundary between the node and the network. As MAC takes the timestamp at an internal point far from the actual boundary of the node and network, you must correct or update the captured timestamp for the ingress or egress path latency (including the delay in the PHY layers). Further correction

is done for the inaccuracies or errors introduced because the clock (MII Tx, Rx clock) is different at the capture point as compared to the PTP clock (CLK_PTP_REF_I) that generates the time. The resultant CDC (Clock domain crossing) circuits add an error depending on the clock period of the MII and PTP clocks.

75.8.3.1 Ingress correction

The timestamp captured at the internal snapshot point In the receive side is delayed (later in time) as compared to the time at which the packet's SFD bit is received at the port's boundary. Therefore, the ingress latency and the errors in CDC sampling reduces the captured timestamp. You must determine or calculate the correction value and write into the MAC_Timestamp_Ingress_Corr_* registers.

The correction value consists of these three components:

1. External latency in the PHY layer between boundary point and the input of the core.

   If the PHY is compliant with the IEEE 802.3 Clause 45 MMD registers, it has a register which indicates the maximum and minimum ingress latency. You can read these registers and determines the average ingress latency in the PHY. Alternatively (if the PHY does not support these registers), you must determine the ingress latency from its datasheet or timing characteristics.

2. Internal latency from the core's input to the internal capture point.

   You can read the internal ingress latency from MAC_Timestamp_Ingress_Latency. This is a read-only register and provides the latency in scaled nanoseconds format as defined in IEEE 1588 Clause 5.3.2. The latency differs on the basis of the active PHY interface (MII, RMII, so on) and the operating speed. Therefore, you must read this register after any speed change in MAC, to determine the current internal latency.

3. CDC synchronization

   The CDC synchronization error is almost equal to twice the clock-period of the PTP clock (clk_pt-p_ref_i).

You must add the values that these three components determine and must write into the TSIC and TSICSNS fields of the MAC_Timestamp_Ingress_Corr_* registers.

NOTE

The value written into the register must be negative (two's complement as explained below), because it is subtracted from the captured timestamp. The MAC receiver adds the value in this register to the captured timestamp and then gives the resultant value as the timestamp of the received packet.

When MAC_Timestamp_Control[TSCTRLSSR] is 1, it indicates that the nanoseconds field of the captured timestamp is in decimal format with a granularity of 1ns. So the Bit31 of TSIC must be 1 (for negative value) and bits[30:0] must write with "10^9 - total ingress_correction_value[nanosecond part]" represented in binary. For example, if the required correction value is -5 ns, then the value is 0xBB9A_C9FB.

When MAC_Timestamp_Control[TSCTRLSSR] becomes 0, it indicates that the nanoseconds field of the captured timestamp is in binary format with a granularity of ~0.466ns. Therefore, bits[30:0] must write with "2^31 - total ingress_correction_value" represented in binary with bit[31] = 1.

75.8.3.2 Egress correction

The timestamp captured at the internal snapshot point in the transmit side is earlier (advanced in time) as compared to the time at which the packet's SFD bit is output at the port's boundary. Therefore, the egress latency and the errors in CDC sampling must compensate the captured timestamp. You must determine or calculate this correction value and write into the MAC_Timestamp_Egress_Corr_* registers.

The correction value consists of these three components:

1. External latency in the PHY layer between the core output and the port and the network boundary

   If the PHY is compliant with the IEEE 802.3 Clause 45 MMD registers, it has a register which indicates the maximum and minimum egress latency. You can read these registers and determine the average egress latency in the PHY. Alternatively (if the PHY does not support these registers), you must determine the egress latency from its datasheet or timing characteristics.

2. Internal latency from the internal capture point and the core output of the c

   You can read this internal egress latency from MAC_Timestamp_Egress_Latency. This is a read-only register and gives the latency in scaled nanoseconds format as defined in IEEE 1588 Clause 5.3.2. The latency differs on the basis of the active PHY interface (MII, RMII, so on) and the operating speed. Therefore, you must read this register after any speed change in MAC to determine the current internal latency.

3. CDC synchronization error

   The CDC synchronization error value for one-step timestamping = (1 * period of CLK_PTP_REF_I + 4 * period of CLK_TX_I). Otherwise (Two-step timestamping mode), the value = -(2 * period of CLK_PTP_REF_I).

75.8.3.3 Frequency range of reference timing clock

The timestamp information is transferred across asynchronous clock domains that is from the MAC clock domain to the application/system clock domain. Therefore, a minimum delay is required between the two consecutive timestamp captures. This delay is 4 clock cycles of MII and 3 clock cycles of PTP clocks. If the delay between the two captured timestamp is less than this delay, MAC does not take a timestamp snapshot for the second packet.

75.8.3.4 PTP processing and control

Table 564 shows the common message header for the PTP messages. This format is defined in the IEEE 1588-2008.

Table 564. Message format defined in IEEE 1588-2008

Bits								Octet	Offset
7	6	5	4	3	2	1	0		
transportSpecific				messageType				1	0
Reserved				versionPTP				1	1
messageLength								2	2
domainNumber								1	4
Reserved								1	5
flagField								2	6
correctionField								8	8
Reserved								4	16
sourcePortIdentity								10	20
sequenceId								2	30
controlField (*)								1	32
logMessageInterval								1	33

(*) – control Field is used in version 1. In version 2 message type field is used for detecting different message types.

There are some fields in the Ethernet payload that detects the PTP packet type and controls the snapshot to be taken. These fields are specified in Table 565 for PTP packets over Ethernet.

Following table provides the information about the fields that match to control the snapshots for the PTP packets sent over Ethernet for IEEE 1588 version 1 and version 2. The octet positions are offset by 4 for the tagged packets. This is based on the IEEE 1588-2008, Annex D, and the message format.

Table 565. Ethernet PTP packet fields required for control And status

Field matched	Octet position	Matched value	Description
MAC destination multicast address[1]	0–5	01-1B-19-00-00-00 01-80-C2-00-00-0E	All PTP messages can use any of the following multicast addresses[2]: • 01-1B-19-00-00-00 • 01-80-C2-00-00-0E[3]
MAC packet type	12, 13	0x88F7	PTP Ethernet packet
PTP control field (IEEE 1588 version 1)	46	0x00, 0x01, 0x02, 0x03, or 0x04	• 0x00: SYNC • 0x01: Delay_Req • 0x02: Follow_Up • 0x03: Delay_Resp • 0x04: Management
PTP message type field (IEEE 1588 version 2)	14 (nibble)	0x0, 0x1, 0x2, 0x3, 0x8, 0x9, 0xB, 0xC, or 0xD	• 0x0: SYNC • 0x1: Delay_Req • 0x2: Pdelay_Req • 0x3: Pdelay_Resp • 0x8: Follow_Up • 0x9: Delay_Resp • 0xA: Pdelay_Resp_Follow_Up • 0xB: Announce • 0xC: Signaling • 0xD: Management
PTP version	15 (nibble)	0x1 or 0x2	• 0x1: Supports PTP version 1 • 0x2: Supports PTP version 2

1. The unicast address match of destination addresses (DA), which is programmed in MAC address 0 to 31, is used if MAC_Timestamp_Control[TSENMACADDR] = 1.
2. IEEE 1588-2008, Annex F
3. MAC does not consider the PTP version 1 messages with peer delay multicast address (01-80-C2-00-00-0E) as valid PTP messages.

75.8.4 Transmit path functions

MAC captures a timestamp when the start packet delimiter (SFD) of a packet is sent on the MII interface. You can control the packets, for which the timestamps are captured on a per-packet basis and mark each transmit packet to indicate whether to capture a timestamp for it. MAC does not process the transmitted packets to identify the PTP packets. You can use the control bits in the transmit descriptor to specify the packets. MAC returns the timestamp to the software inside the corresponding transmit descriptor and therefore connects the timestamp automatically to the specific PTP packet. You must write the 64-bit timestamp information to TDES0 and TDES1 fields. The TDES0 field holds the 32 least significant bits of the timestamp.

75.8.5 Receive path functions

You can program MAC to capture the timestamp of all packets received on the MII interface or to process the packets to identify the valid PTP messages.

You can use these options of MAC_Timestamp_Control to control the snapshot of the time sent to the application:

- Enable snapshot for all packets
- Enable snapshot for IEEE 1588 version 1 or version 2 timestamp
- Enable snapshot for PTP packets transmitted directly over Ethernet
- Enable timestamp snapshot for the received packet for IPv4 or IPv6
- Enable timestamp snapshot only for EVENT messages (SYNC, DELAY_REQ, PDELAY_REQ, or PDELAY_RESP)
- Enable the node to be a master or slave and select the snapshot type

Table 566. Timestamp snapshot dependency on register bits

SNAPTYPSEL	TSMSTRENA	TSEVNTENA	PTP messages
00	x	0	SYNC, Follow_Up, Delay_Req, Delay_Resp
00	0	1	SYNC
00	1	1	Delay_Req
01	x	0	SYNC, Follow_Up, Delay_Req, Delay_Resp, Pdelay_Req, Pdelay_Resp, Pdelay_Resp_Follow_Up
01	0	1	SYNC, Pdelay_Req, Pdelay_Resp
01	1	1	Delay_Req, Pdelay_Req, Pdelay_Resp
10	x	x	SYNC, Delay_Req
11	x	x	Pdelay_Req, Pdelay_Resp

DMA returns the timestamp to the software inside the corresponding receive descriptor. The extended status, contains the timestamp message status and the IPC status. You must write the extended status in the normal descriptor RDES1 and the snapshot of the timestamp in RDES0 and RDES1 fields of context descriptor. The RDES0 field holds the 32 least significant bits of the timestamp.

See the System time correction for programming guidelines for IEEE 1588 timestamping (System time correction).

75.8.6 IEEE 1588 system time source

IP supports both the external and internal time source for reference timestamping.

- External time source uses the 64-bit external time reference and clock as an input in IP. The clock input synchronizes the external timing reference into the MAC clock domain. Upper 32-bit indicates the time in seconds and the lower 32-bit indicates the time in nanoseconds.
- Internal time source uses the only clock input to generate timing reference internally for snapshot and capture timestamps. Internal time reference has two fields.
  - UInteger48 seconds field: The seconds field is the integer portion of the timestamp in seconds units. It is 48-bit wide.

    For example, 2.000000001 seconds are represented as seconds field = 0x0000_0000_0002.

— UInteger32 nanoseconds field The nanoseconds field is the fractional portion of the timestamp in nanoseconds units.

For example, 2.000000001 seconds are represented as nanoseconds field = 0x0000_0001. The nanoseconds field supports the following two modes:

- Digital rollover mode: In this mode, the maximum value in the nanoseconds field is 0x3B9A_C9FF, that is, (10e9-1) nanoseconds.
- Binary rollover mode: In this mode, the nanoseconds field rolls over and increments the seconds field after 0x7FFF_FFFF value . Accuracy is ~0.466 ns per bit.

There is a system time register module, it is used when you use an internal time reference. The 80-bit time is maintained in this module and updated using the input reference clock (CLK_PTP_REF_I). This time is the source for taking snapshots (timestamps) of Ethernet packets which is transmitted or received at the MII interface.

Initialize or correct the system time counter using the coarse correction method. In this method, write the initial value or the offset value to the timestamp update register. For initialization, write the system time counter with the value in the timestamp update register. For system time correction, the offset value is added to or subtracted from the system time.

In the fine correction method, correct the frequency offset and/or frequency drift of a slave clock (CLK_PTP_REF_I) with respect to the master clock (as defined in IEEE 1588-2002) over a period of time instead of in one clock, as in coarse correction. This maintains linear time and does not introduce drastic changes (or a large jitter) in the reference time between the PTP sync message intervals.

See Initialization guidelines for system time generation for programming guidelines for IEEE 1588 timestamping (For internal timestamp source configuration).

75.8.7 IEEE 1588 higher word register

You can invoke the higher word register if system time source is internal. MAC timestamp is 64-bit wide. The values of the upper 16-bits of the seconds field are read from the CSR register.

75.8.8 Flexible pulse-per-second output

IP supports the flexibility of programming the start or stop time, generates pulse width and interval on pulse-per-second output. It also support four such output signals. By default, IP is in fixed pulse-per-second output mode with interval of 1 second.

Initially you must program the start time in target time registers "MAC_PPSx_Target_Time_Seconds and MAC_PPSx_Target_Time_Nanoseconds" for all desired or enabled pps output. If you re-programs start or stop time then you must do it after the synchronization of earlier programmed value. Bit 31 of MAC_PPS#_Target_Time_Nanoseconds register indicates that the synchronization is complete. If the application programs a start or stop time that has already elapsed, MAC sets an error status bit which indicates the programming error. If enabled, MAC also sets the target time reached interrupt event. The application can cancel the start or stop request only if the corresponding start or stop time has not elapsed. If the time has elapsed, the cancel command has no effect.

Program the PPS width and interval in terms of granularity of system time, that is, the number of units of sub-second increment value.

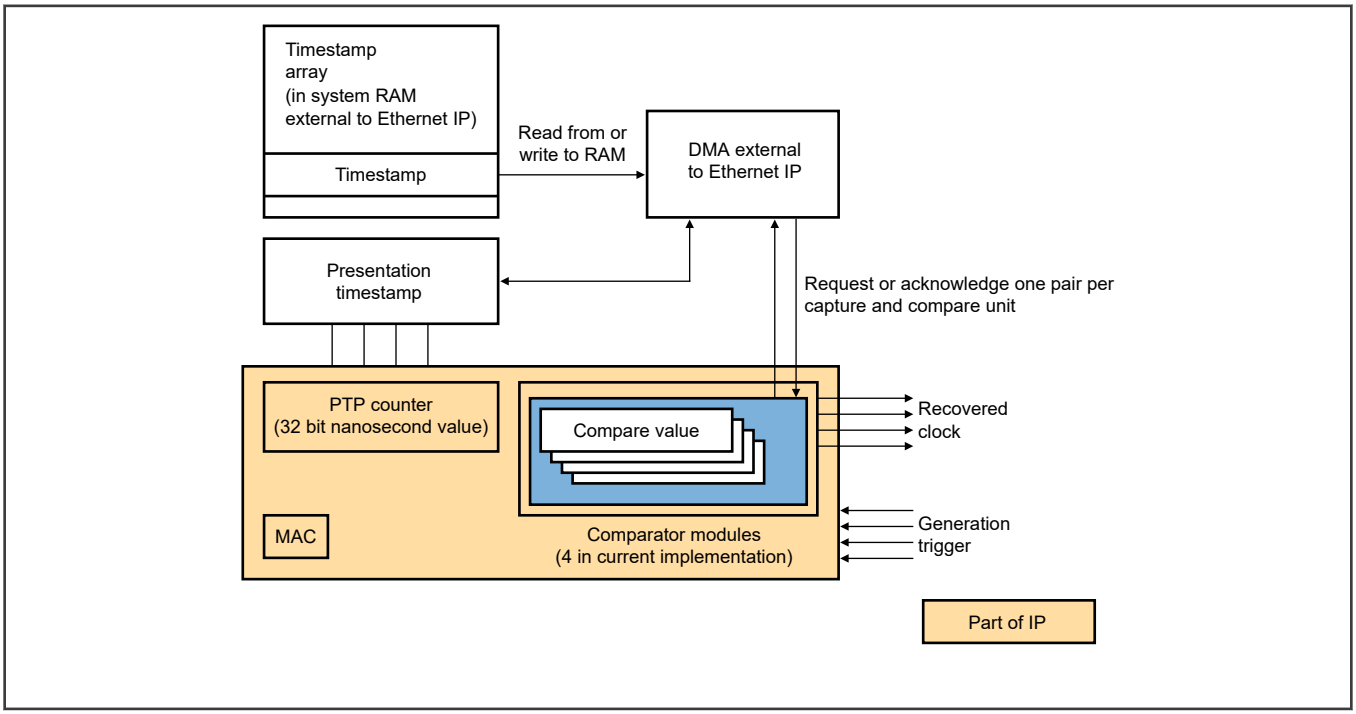
See Programming guidelines for flexible pulse-per-second output for more information.

75.8.9 Media clock generation and recovery

Media clock recovery and generation requires a dedicated hardware and software to complete the operation. IP generates a reference clock with the help of a dedicated hardware which multiplies to get the desired media clock. Software extracts the presentation time from the incoming PTP 1722 frames and program these values into CSR registers so that IP can read them for media clock recovery.

Figure 423 shows that the timestamp array and the DMA module are external to IP. IP supports a 32-bit presentation time counter in nanoseconds that completes at the full 32-bit value to match with 1722 presentation time format and an handshake mechanism with an external application (such as DMA) to program the presentation time into CSR register.

Figure 423. Media clock generation and recovery block diagram



75.8.9.1 Presentation time counter

Presentation time counter provides an another perspective of the PTP system time (1588 timer). For a given PTP system time (PTP[63:32] represents time in seconds Comparator modules and PTP[31:0] represents time in nanoseconds), there exists a corresponding presentation time (32-bit value in nanosecond). The presentation time computed is referred to as current presentation time (CPT).

CPT derives from 64-bit PTP system time. PTPNS[63:0] = (PTP[63:32] * 32'd1,000,000,000) + PTP[31:0] where, PTPNS is the PTP system time converted into a 64-bit nanosecond value.

Current presentation time[31:0] = PTPNS[31:0]

Media clock recovery is possible when the system time is internally generated in IP and compute the CPT in the same way as the internal PTP system time generation. The increment cycle of CPT and system time are same because the values of both the timers are in nanoseconds and they are synchronous. The timer updates at the same instance, but the updated value could be different. You must compute a separate 32-bit update value in nanoseconds for the CPT update. CPT sampled at different edges of a triggering input generates media clock timestamps that are inserted in 1722 based AVBTP packets, to recover the clock at the destination.

75.8.9.2 Comparator modules

Figure 423 shows that the module supports four comparator modules to handle multiple media clocks that it may require. When MAC_Timestamp_Control[PTGE] is 1, it indicates that the comparator modules handshake with the application to program the timestamps into the MAC_PPS(#i)_Target_Time_Seconds register.

Write 1 to presentation time control fields of the particular instance for recovery mode.

The timestamps received from the application are referred as target presentation time (TPT).

When MCGREN#i field of MAC_PPS_Control is 1, it indicates that the IP operates in MCGR mode. The comparator module sends up to two requests to the application for TPT write, when MAC_Timestamp_Control[PTGE] and the presentation time control bits of the particular instance are 1 for MCGR mode. Subsequent requests are generated each time a presentation time match occurs. CPT transitions from a value less than TPT, to a value greater than or equal to CPT. The first request asserts when a specific comparator instance sets to MCGR mode with a non-zero presentation time control and an additional request is made when the comparator receives the first data. This allows the application to write the next TPT value when IP processes the previous TPT value for a match.

TPT read from the MAC_PPS(#i)_Target_Time_Second is considered as a future time. A toggle or pulse is generated in the next cycle when a match is detected. Also,the mcgr_dma_req_o#i request for that comparator asserts (to obtain next TPT) until the corresponding application acknowledgment is set, when a match is detected.

The presentation control fields (PPSCMD#i field of MAC_PPS_Control) determine the shape of the generated waveform. You can program these fields to either toggle or generate a high/low pulse for one PTP clock cycle, when matched.

Media clock recovery: A match occurs, when the free-running CPT value matches the received TPT value. An output signal, EMAC_TMR signal as shown in the IO mux sheet asserts (toggle, low pulse, or high pulse) on the basis of the programmed presentation control value.

Media clock generation: On the basis of the presentation control value programmed in MAC_PPS_Control, the particular comparator captures the presentation time and programs it into MAC_PPS(#i)_Target_Time_Second register. The captured timestamp is read when a request is raised to the application. No new timestamps are captured, until the read operation is complete acknowledgment is received from the application.

75.8.9.3 Media clock generation and recovery flow

Figure 424 shows the media clock generation and recovery flow. Write 1 to MAC_Timestamp_Control[PTGE] for CPT generation and write 1 to MAC_PPS_Control MCGREN#i field to enable the corresponding instance in the MCGR mode, for both media clock generation and recovery.

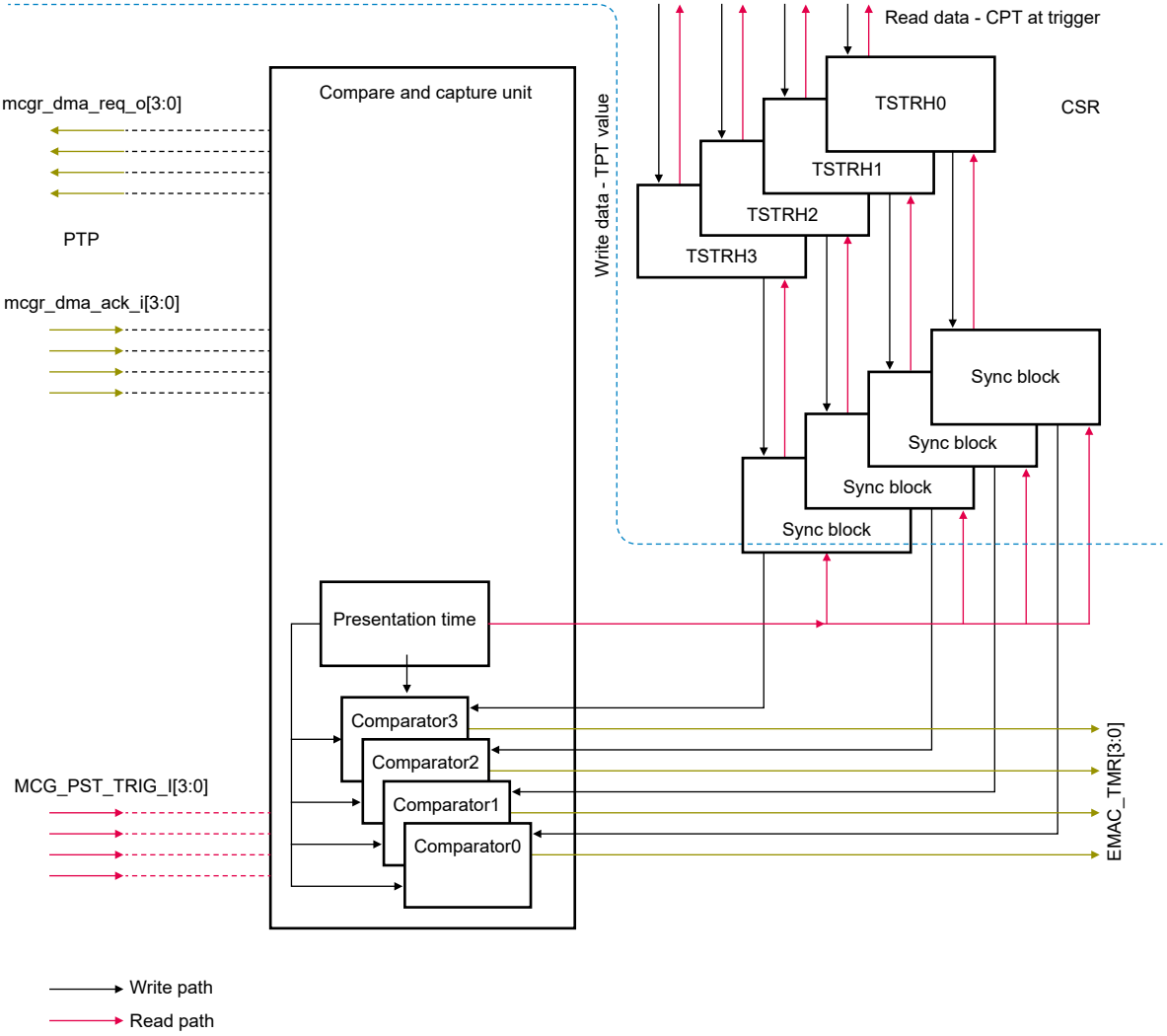


Figure 424. Media clock generation and recovery flow

NOTE

1. The consecutive triggers to sample the presentation time must assert after a few PTP clock cycles so as to allow synchronization delays. (There is no such issue when the input trigger maximum frequency is 8 KHz and the PTP clock runs at least at 1MHz)
2. In the Media Clock Recovery mode, the DMA acknowledgment is both posted as well as non-posted.

See Programming guidelines for media clock generation and recovery for more information.

75.8.10 One step timestamp

IP supports the one step timestamp feature. Writing 1 to bit 20 (OSTC) in the control word, enables the one step timestamp feature for a packet.

See the Programming guidelines for IEEE 1588 timestamping for more information.

75.8.11 IEEE 1588 sub nanoseconds timestamp

IP supports the ingress and egress correction accuracy in sub nanoseconds. It is programmed in MAC_Timestamp_Ingress_Corr_Subnanosec[TSICSNS] and MAC_Timestamp_Egress_Corr_Subnanosec[TSECSNS],

respectively. In this case, the correction has an unit of nanosecond, multiplied by 2^8. You must program the least significant 8 bits of the value in the sub-nanoseconds register.

For example, if the required egress correction is 3.6 nS and MAC_Timestamp_Control[TSCTRLSSR] is 1, then MAC_Timestamp_Egress_Corr_Nanosecond[TSEC] must be 0x3 and MAC_Timestamp_Egress_Corr_Subnanosec[TSECSNS] must be 0x99 (0.6 * 256). Similarly, if the required ingress correction is -3.6 nS and MAC_Timestamp_Control[TSCTRLSSR] is 1, then MAC_Timestamp_Ingress_Corr_Nanosecond[TSIC] must be 0xBB9A_C9FC and MAC_Timestamp_Ingress_Corr_Subnanosec[TSICSNS] must be 0x66 (fractional part of (10^9 - 3.6) * 256).

75.9 Multiple channels and queues support

This section and all its sub-sections are Synopsys Proprietary. Used with permission.

IP support two queues and channel on Tx and Rx paths. Figure 425 shows the architecture of IP with two queues and channel.

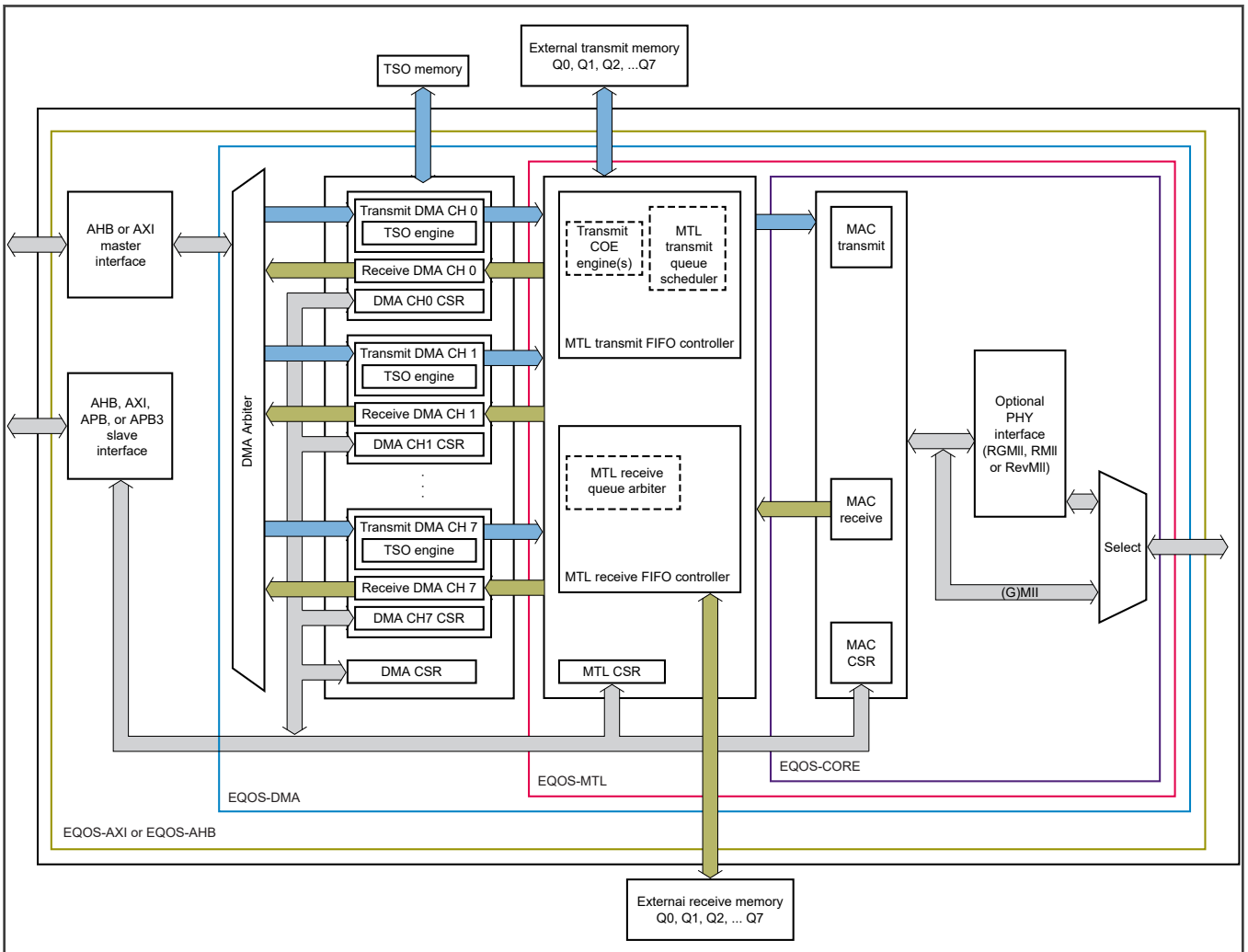


Figure 425. Module with multiple channels and queues block diagram

The above block diagram, support only two queues.

IP DMA arbiter support two types of arbitration scheme, fixed priority and weighted round-robin. The DMA arbiter performs the arbitration between the Tx and Rx paths of DMA channels to access descriptors and data buffers. DMA_Mode[DA] specifies the arbitration scheme (fixed or weighted round-robin) between the channel Tx and Rx DMA.

DMA_Mode[TXPR] sets the priority between the corresponding Tx DMA and Rx DMA. Writing 1 to DMA_Mode[PR] specifies the weighted priority between the Tx DMA and Rx DMA in round robin arbitration scheme.

Table 567 provides an information about the priority scheme between Tx DMA and Rx DMA.

Table 567. Priority scheme for Tx DMA and Rx DMA

Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Priority schemes
x	x	x	0	1	Rx always has priority over Tx
0	0	0	0	0	Tx and Rx have equal priority. Rx gets the access first on simultaneous requests
0	0	1	0	0	Rx has priority over Tx in ratio 2:1
0	1	0	0	0	Rx has priority over Tx in ratio 3:1
0	1	1	0	0	Rx has priority over Tx in ratio 4:1
1	0	0	0	0	Rx has priority over Tx in ratio 5:1
1	0	1	0	0	Rx has priority over Tx in ratio 6:1
1	1	0	0	0	Rx has priority over Tx in ratio 7:1
1	1	1	0	0	Rx has priority over Tx in ratio 8:1
x	x	x	1	1	Tx always has priority over Rx
0	0	0	1	0	Tx and Rx have equal priority. Tx gets the access first on simultaneous requests
0	0	1	1	0	Tx has priority over Rx in ratio 2:1
0	1	0	1	0	Tx has priority over Rx in ratio 3:1
0	1	1	1	0	Tx has priority over Rx in ratio 4:1
1	0	0	1	0	Tx has priority over Rx in ratio 5:1
1	0	1	1	0	Tx has priority over Rx in ratio 6:1
1	1	0	1	0	Tx has priority over Rx in ratio 7:1
1	1	1	1	0	Tx has priority over Rx in ratio 8:1

75.9.1 Multiple queues and channels support in the transmit path

IP support two transmit queues and two channels.

Fixed priority scheme is the default priority scheme for the DMA channels and channel with highest priority always wins the arbitration when it requests for the bus. Channel 0 is always of lowest priority and channel 1 is always of highest priority.

In weighted strict priority (WSP), the weight corresponds to the number of burst transfers given to a channel in one arbitration cycle. Reallocate the unused burst transfers of one or more channels on the basis of the priority. The channel with highest priority receives the unused burst transfer time before it is allocated to a channel with the next highest priority. You can process the next lower priority, when a channel uses the allocated burst transfers. After processing the allocated bandwidth of all the channels that have packets to transmit, allocate any unused burst transfer time to the channel of the highest priority (if required), and then next highest priority (if required), and so on.

In weighted round robin (WRR) priority scheme, program the weight through TCW field of DMA_CH#_Tx_Control register. IN WRR scheme, service all channels in round-robin order according to the weights settings. Table 568 shows that the TCW field of DMA_CH#_Tx_Control register provides the weight for each transmit channel.

Table 568. Weight for DMA channels

TCW Field	Transmit Channel Weight
000	1
001	2
010	3
011	4
100	5
101	6
110	7
111	8

The configured weights correspond to the number of burst transfers given to a channel in one arbitration cycle. The unused or excess burst transfers are distributed equally to all channels.

See Programming guidelines for multi-channel multi-queuing for more information.

75.9.2 Multiple queues and channels support in the receive path

IP support two receive channels and two queues.

In the receive direction, the MTL Rx controller selects the Rx DMA for which it transfers or reads the data from the receive FIFO memory. This scheduling is based on the programming done in the respective MTL_RxQ[x]_Control register.

Each Rx DMA indicates when it is ready to transfer data and the size of the burst-length (number of beats) that it will transfer. The scheduler checks whether sufficient data (of requested burst length) is available to transfer to these DMAs and then selects the receive DMA that is serviced using the programmed priorities.

See Programming guidelines for multi-channel multi-queuing for more information.

75.9.3 Rx queue to DMA mapping

You can program MTL_RxQ_DMA_Map0 (for queues 0 and 1) to route the packets in the MTL receive queues to any one of the multiple DMA channels.

The following types of receive queue to DMA mapping is possible through programming.

75.9.3.1 Static mapping

In this mode, you can connect all the packets of the receive queue to a specific DMA channel. For example, all the packets from the receive queue 0 program MTL_RxQ_DMA_Map0[Q0MDMACH] (bit[3:0]) and MTL_RxQ_DMA_Map0[Q0DDMACH] (bit[7] = 0) to route to a DMA channel.

Similarly, packets from other receive queues program register fields corresponding to each queue to route to any DMA channel.

75.9.3.2 Dynamic (per packet) mapping

In this mode, the destination DMA channel of a packet read from a Rx queue is not constant but it is decided independently for each packet. For example, if you write 1 to MTL_RxQ_DMA_Map0[Q1DDMACH], it indicates that the static mapping disables for receive queue 1 and ignores the value in MTL_RxQ_DMA_Map0[Q1MDMACH]. The MAC receiver decides the destination DMA channel for each packet, depending on the following in decreasing order of priority:

1. **L3-L4 filter based DMA selection**: The TCP/UDP and IP header fields of the received packet are matched against the corresponding values programmed and enabled for comparison in the MAC_L3_L4_Address_Control register. If the match is successful, the DMA channel number programmed in the DMCHN field of the MAC_L3_L4_Address_Control register is selected as the destination DMA channel number, if DMCHEN field of the same register is 1. If none of the L3-L4 registers give a comparison match, then the module proceeds to the next step below.
2. **Extended VLAN based DMA selection**: Extended routing is applicable only if the VLAN filter has passed. Routing is done only on the basis of the perfect filter result. Each perfect filter has a DMA channel enable and a DMA channel number field which you can program. Routing is applicable for a filter, only if the DMA channel enable field is 1. The frame routes to the smallest matching filter's DMA channel, if it is enabled. If the filter's DMA channel number is not enabled, the frame route to channel 0. For example, if a frame's VLAN tag matches filters 7, 3, and 1, then the MAC checks if DMA channel number of filter 1 is enabled through programming. If yes, the frame route to the programmed value; otherwise it route to DMA. When the inverse filter is enabled; is routed to the least mismatched filter's DMA channel number provided it is enabled. If the DMA Channel enable bit is not set, then the frame routes on the basis of the DA based addressing or to channel 0.

   If hash filter is also enabled, it is determines the filter result only. Routing will still depends on the enabled perfect filters. Routing based on the VLAN filter will not occur if none of the perfect filters are enabled or if all of them are bypassed. The frame will route via DA based addressing or to channel 0. If all the perfect filters fails and the hash filter has passed, then the VLAN filter result is a pass but routing will be based on DA based addressing or to channel 0. Similar behavior is seen when you enables inverse filtering as well.
3. **Ethernet DA-based DMA selection**: The DA address of the received packet is compared against the programmed DA values in MAC address registers. If the address matches any of the programmed values, the corresponding DCS field (when enabled) determines the destination DMA channel number.

If none of the above operations make a successful match or decision, then the packet routes to the DMA channel 0 by default.

75.9.3.2.1 Broadcast/multicast packet duplication

This feature provides a mechanism to send the received broadcast or multicast packets to multiple DMA channels.

MAC_Address0_High[DCS] and other optional MAC_Address(#i)_High (i=1 to 2) registers determines the DMA channel number to which the received packet (that matches the MAC address present in that register) must route.

DMA channel routing mechanisms such as extended VLAN, or L3-L4 based routing does not support the packet duplication.

The packet duplication feature is supported only on the highest MTL queue configured. Therefore, you must program MAC_RxQ_Ctrl1[MCBCQ] to the highest RxQ present in the configuration for the packet duplication for broadcast or multicast packets.

75.9.4 Selection of tag priorities assigned to receive queues

Program the PSRQ field in the corresponding MAC_RxQ_Ctrl2 to assign the VLAN tag priorities to receive queues. The bit corresponding to the VLAN tag priority can be set in the PSRQ field for assigning that priority to the receive queue. For example, if you want to assign VLAN tag priority of 3 to receive queue 0, write 1 to bit [3] in PSRQ field of MAC_RxQ_Ctrl2. The VLAN tag priority assigned to particular receive queue must be unique, that is, you cannot assign more than one receive queue to the same VLAN tag priority. However, more than one VLAN tag priorities can be assigned to same receive queue.

The settings in the PSRQ field is used for VLAN tagged receive packet routing to receive queues as well as for PFC based transit flow control. The received VLAN tagged receive packet route to receive queue that has the VLAN tag priority match. In PFC based transit flow control, PSRQ field corresponding to a particular receive queue enables VLAN tag priorities in the PFC packet transmitted when corresponding receive queue reach the threshold level.

75.9.5 Receive side routing from MAC to queues

IP supports receive side routing from MAC to queues. MAC route the receive packets to the receive queues on the basis of the these packet types in that order

- Unicast/Multicast destination address packets that fail the destination address filter
- Multicast/Broadcast destination address packets that pass the destination address filter
- VLAN tag priority field in VLAN tagged AV data packets
- AV control packets
- VLAN tagged IEEE 1588 PTP over Ethernet packets
- Untagged IEEE1588 PTP over Ethernet packets
- VLAN tag priority field in VLAN tagged packets
- Untagged packets

You can route the outer C-VLAN tagged AV data receive packets on the basis of the priorities assigned to receive queues through PSRQ field in the corresponding MAC_RxQ_Ctrl2 and MAC_RxQ_Ctrl3 and the corresponding receive queue is enabled for AV through RXQ#EN field in MAC_RxQ_Ctrl0. These packets may be single VLAN tagged with C-VLAN type or double VLAN tagged with outer VLAN tag of C-VLAN type when double VLAN feature enables (MAC_VLAN_Tag_Ctrl[EDVLP] = 1) with an inner C-VLAN tagged or inner S-VLAN tagged when SVLAN processing is enabled (MAC_VLAN_Tag_Ctrl[ESVL] = 1). This type of Rx packet routing is available when you select the AV feature and multiple receive queues in the configuration.

You can route the AV control receive packets on the basis of the receive queue number specified in MAC_RxQ_Ctrl1[AVCPQ] and the corresponding receive queue is enabled for AV through RXQ#EN field in MAC_RxQ_Ctrl0. These packets may be single VLAN tagged with C-VLAN type or double VLAN tagged with outer VLAN tag of C-VLAN type when double VLAN feature is enabled (MAC_VLAN_Tag_Ctrl[EDVLP] = 1) with inner C-VLAN tagged or inner S-VLAN tagged when SVLAN processing is enabled (MAC_VLAN_Tag_Ctrl[ESVL] = 1). This type of receive packet routing is available when you select the AV feature and multiple receive queues in the configuration.

You can route the VLAN tagged receive packets on the basis of the priorities assigned to receive queues through PSRQ field in corresponding MAC_RxQ_Ctrl2 and MAC_RxQ_Ctrl3 and RXQ#EN field in MAC_RxQ_Ctrl0 enables the corresponding receive queue for DCB/generic. This type of receive packet routing is available when you select the multiple receive queues in the configuration.

You can route the untagged IEEE 1588 PTP over Ethernet receive packets on the basis of the receive queue number specified in MAC_RxQ_Ctrl1[PTPQ] and the corresponding receive queue is enabled through RXQ#EN field in MAC_RxQ_Ctrl0. Also, you can route the VLAN tagged IEEE 1588 PTP over Ethernet receive packets on the basis of the priorities assigned to receive queues through PSRQ field in corresponding MAC_RxQ_Ctrl2 and MAC_RxQ_Ctrl3 or the receive queue number specified in the MAC_RxQ_Ctrl1[PTPQ] and RXQ#EN field in MAC_RxQ_Ctrl0 register enables the corresponding Rx queue. Programming MAC_RxQ_Ctrl1[TPQC] determines this. This type of receive packet routing is available when IEEE 1588 timestamp feature supports and the multiple receive queues are selected in the configuration. The VLAN tagged IEEE 1588 PTP over Ethernet receive packets are detected only when you disables 802.1AS mode (MAC_Timestamp_Control[AV8021ASMEN] = 0), otherwise this type of receive packets are routed as generic VLAN tagged Rx packets.

You can route the multicast or broadcast destination address receive packets that passes the destination address filter on the basis of the receive queue number specified in MAC_RxQ_Ctrl1[MCBCQ]the MCBCQ field of MAC_RxQ_Ctrl1 register when enabled through MAC_RxQ_Ctrl1[MCBCQEN] and the corresponding receive queue is enabled through RXQ#EN field in MAC_RxQ_Ctrl0. This type of Rx packet routing is available when you select the multiple receive queues in the configuration.

You can route the untagged receive packets on the basis of the receive queue number specified in MAC_RxQ_Ctrl1[UPQ] and RXQ#EN field in MAC_RxQ_Ctrl0 enables the corresponding receive queue. This type of receive packet routing is available when you select the multiple receive queues in the configuration.

The unicast destination address receive packets that fail the destination address filter can be routed based on the receive queue number specified in MAC_RxQ_Ctrl4[UFFQ] when enabled through MAC_RxQ_Ctrl4[UFFQE]UFFQE, MAC_Packet_Filter[RA] = 1 and the corresponding receive queue is enabled through RXQ#EN field in MAC_RxQ_Ctrl0. This type of receive packet routing is available when you select the multiple receive queues in the configuration.

You can route the multicast destination address receive packets that fails the destination address filter on the basis of the receive queue number specified in MAC_RxQ_Ctrl4[MFFQ] when MAC_RxQ_Ctrl4[MFFQE] enables it. MAC_Packet_Filter[RA] = 1 and RXQ#EN field in MAC_RxQ_Ctrl0 enables the corresponding receive queue. This type of receive packet routing is available when you select the multiple receive queues in the configuration.

The receive packet will route through the receive queue 0, if it is not classified in any of the defined packet type for routing.

75.9.6 Receive side arbitration between DMA and MTL

IP controller supports receive side arbitration between DMA and MTL.

After the current packet processing completes, the DMA channel controller fetches the next descriptor and after the descriptor fetching completes, the DMA channel controller evaluates the amount of data to transfer to the Rx buffer on the basis of the programmed PBL and receive buffer length. Accordingly, it requests the MTL to transfer the data.

After servicing the current request, the MTL receive queue arbitration scheme selects receive queue on the basis of the arbitration scheme (MTL_Operation_Mode[RAA]) and the weights programmed in queue <n> receive control register. The arbitration is done among queues for which DMA is ready to service. After the receive queue is selected, PBL amount of data is read out from that queue and route to the receive DMA channel on the basis of the receive channel selection criteria.

The arbitration occur after every PBL data transfer completes and descriptors are ready for processing from at least one DMA channel.

75.9.7 Transit side arbitration between DMA and MTL

IP supports transit side arbitration between DMA and MTL. Transit DMA channels and transit queues are always mapped directly because the number of transit DMA channels are always equal to the number of transit queues in MTL. For instance, each transit DMA pushes data into its respective transit queue assigned to it.

The data inside each transit queue is stored in packets. Therefore, if two DMAs are allowed to transfer data into the same queue, when a transit DMA starts a packet transfer, the other DMA cannot transfer data unless the previous packet is completely pushed-in. This means that the second DMA remains idle until the first packet is transferred. Hence, each DMA is always connected directly to its corresponding transit queue.

75.9.8 Audio video bridging

IP supports the audio video bridging in 100 Mbps mode only, which allows the transmission of time sensitive traffic over network. IP implements these protocol for supporting the AVB feature:

- IEEE 802.1Qbv-2015 (Enhancements to scheduling traffic)
- IEEE 802.3br (Interspersing traffic)
- IEEE 802.1Qbu (Frame preemption)

The queue 0 transmit path supports the strict priority algorithm, and it is used for best-effort traffic when transmit paths of additional queues use the credit-based shaper algorithm to support traffic management. For a queue, the credit-based shaper algorithm determines that a queue is available for transmission if these conditions are true:

- The queue contains one or more packets.
- The credit for the queue is positive per the algorithm
- AV traffic is received on all queues. IP can also receive untagged PTP packets in addition to AV traffic on any queue.

75.9.9 Queue modes

Transit and receive queues both handles the AV traffic.

Queueing is based on WRR (Weighted round robin) or WSP (Weighted strict priority) algorithm for generic traffic. It is based on CBS (Credit Base Shaper) and SP (Strict Priority) algorithm for AV traffic.

The receive queue is configured for generic or AV based routing, which depends on the RXQ#EN field of corresponding queue. Queueing is done on the basis of the VLAN tag priority. The VLAN tag priority must match the PSRQ field of MAC_RxQ_Ctrl2. The receive packets can route to a particular DMA channel on the basis of the DCS field of perfectly-matched MAC address register.

By default, the untagged packets in a generic traffic route to the receive queue specified in MAC_RxQ_Ctrl1[UPQ]. Queue 0 is the default value of MAC_RxQ_Ctrl1[UPQ]. You can override the default value with any other value, for the MAC_RxQ_Ctrl1[UPQ].

The AV control packets (tagged or untagged) route on the basis of MAC_RxQ_Ctrl1[AVCPQ]. The PTP over Ethernet packets route on the basis of MAC_RxQ_Ctrl1[PTPQ].

75.9.10 Queue priorities

You can program the priority of an receive queue in the corresponding field of MAC_RxQ_Ctrl2. Also, you must program the AV queue (high priority) as an higher priority than the best-effort queue (low priority).

75.9.11 Enhancements to scheduled traffic (EST)

The IEEE 802.1Qbv-2015 defines the schedule for each queues on every egress port which makes the implementation aware of traffic arrival schedule. This information blocks the lower priority traffic from transmission in this time window or slot. This ensures that the sender forward the scheduled traffic to receiver through all the network nodes with a deterministic delay.

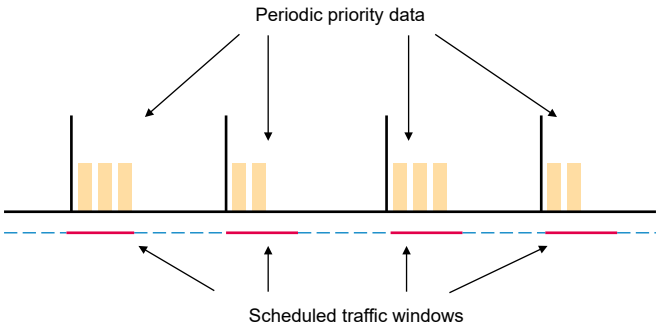


Figure 426. Time aware shaper implementation for traffic scheduling

An important requirement to achieve a low latency is to ensure that there are no interfering frames during the scheduled windows that are reserved for high priority traffic. The use of scheduled traffic imposes limitations when a transmission starts.

As shown in Figure 427 , if an interfering frame begins transmission just before the reserved time period starts, it can extend transmission into the reserved window, and potentially interfere with higher priority traffic. Therefore, a guard band whose size is equal to the largest possible interfering frame is required before the window starts.

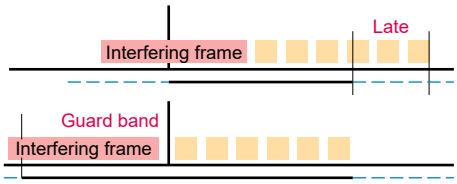


Figure 427. Implementing a guard band to avoid delays due to interfering frames

A larger guard band equates to a less efficient use of network bandwidth. However, this issue is addressed with the implementation of IEEE802.1Qbu (frame preemption). Frame preemption breaks the interfering frame into smaller fragments. Therefore, the guard band must be as large as the largest possible interfering fragment instead of the largest possible interfering frame.

During the guard band, only the frames that can complete the transmission of the entire frame before the next gate close event are permitted. This ensures that the high priority traffic can always start at the beginning of the window reserved for it.

75.9.11.1 Frequently used terms in EST

These are some of the frequently used terms in the EST support and their definitions.

- Gate control list: The list in the hardware memory that hold the gate controls and the associated time intervals.
- Gate controls: For a given schedule (row in gate control list) there is a gate open (O) and gate close (C) state associated with each TC. The set of O or C values, whose width is same as configured TCs is called the gate controls. Example: CCOCOOCO means TC7=C, TC6=C, TC5=O and so on.
- Time interval: Time interval (in nanoseconds) is a 16, 20, or 24-bit configurable field in the gate control list that indicates the time for which the associated gate controls are valid.
- Base time register: Each gate control list is associated with a 64-bit base time register that holds the start time (in PTP format) for the list.

75.9.11.2 Updates to the transmit scheduling to support EST

To support EST, these updates are required to transmit scheduling:

- Implementation of gate control list (GCL)
- Enforcing gate controls in the scheduler
- Utilizing gate open (O) duty cycle in the computation of idleSlope (CBS)

75.9.11.2.1 Implementation of gate control list (GCL)

Figure 428 shows how the gate control list governs the gate close(C) and open(O) events on the basis of the schedule provided for each event. GCL has two parts:

- Time interval: Defines the time in nanoseconds for which the gate controls are valid and must apply before reading the next gate controls from the list. The configurable width of 16, 20 or 24-bit represents a maximum of 64us, 1ms, or 16ms schedule interval respectively. It supports left shift of the time interval up to 7 bits to be able to apply a multiplication factor from 1 to 128ns (in steps of 2(power)n). The maximum value (post shifting) of this field must be 999,999,999 ns.
- Gate control: Defines the open (O) represented by logic 1 or close (C) represented by logic 0 state for the gate of each TC.

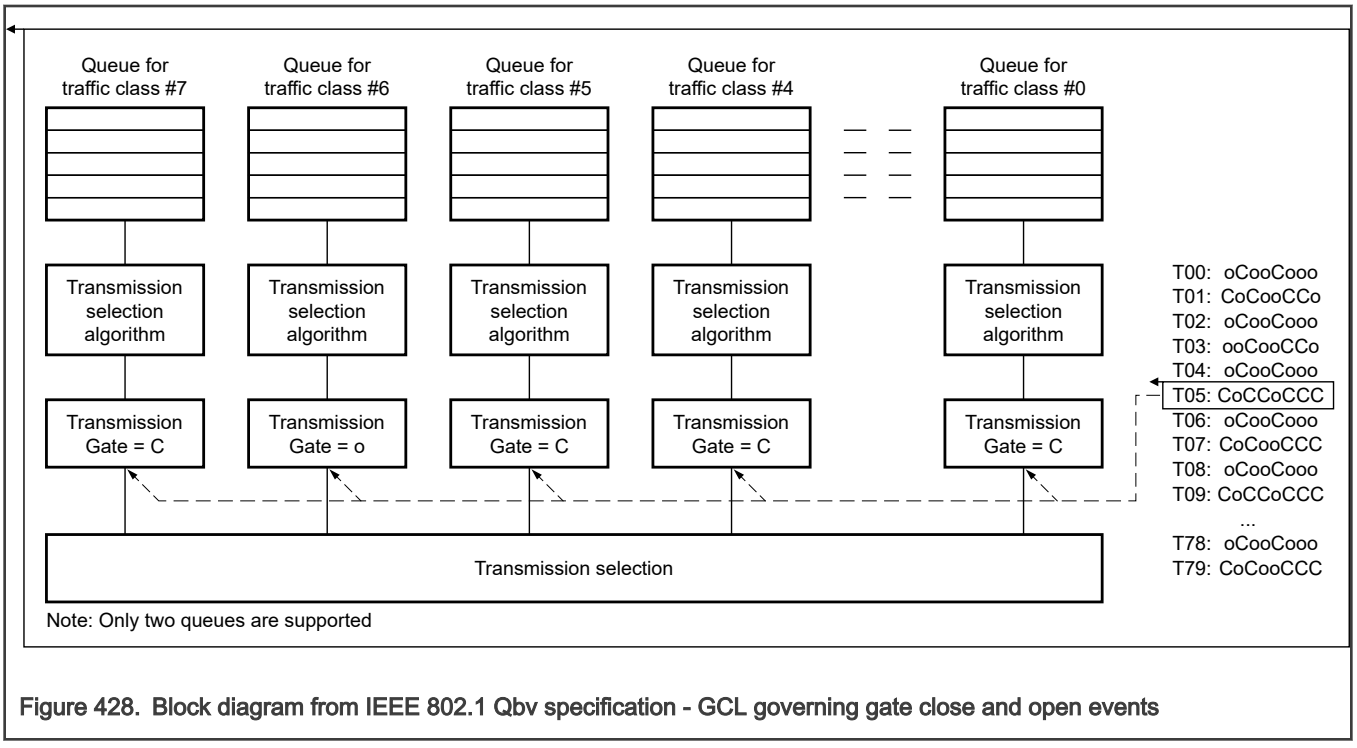


Figure 428. Block diagram from IEEE 802.1 Qbv specification - GCL governing gate close and open events

The implementation of GCL consists of these two gate control lists:

- HWOL - Hardware owned list which is a list for hardware access.
- SWOL - Software owned list which is a list for software access.

The access to these lists is mutually exclusive. Hardware sets the ownership to the list in MTL_EST_Status[SWOL]. Table 569 provides the implementation details of GCL.

Table 569. External memory used for holding the two gate-control lists

Gate control (up to 8 bits)	Time interval (ns) 16, 20, or 24 bits	
OOCCCCCC	T0	HWOL or SWOL
OOOOCCCC	T1	
CCCCOOCC	T2	
CCCCCCOO	T3	
I	I	
I	I	
OCOCOCOO		
OOOOCCCC	T last	
OOOOCCCC	T0	SWOL or HWOL
OOOOCCCC	T1	
OOCCCCCC	T2	
OOOOCCCC	T3	
I	I	
I	I	
OCOCCCCC		
OOCCCCCC	T last	

75.9.11.2.2 Registers related to gate control list

The following are the set of four registers (one for each GCL) related to GCL. These registers are implemented through Indirect addressing using MTL_EST_GCL_Control and MTL_EST_GCL_Data registers.

1. 64-bit Base time register (BTR)

2. 40-bit Cycle time register (CTR)

3. m-bit Time extension register (TER)

4. n-bit [Gate control] List length register (LLR) (n depends of the configured GCL depth)

75.9.11.2.2.1 Base time register (BTR)

This is a 64-bit register that specifies the start time to execute the GCL. The format of the BTR is same as the PTP format (upper 32-bits holds time in seconds and lower 32 bits hold time in nanoseconds). After the execution of a given list starts, the implementation can update the value in BTR to indicate the next list execution start time.

75.9.11.2.2.2 Cycle time register (CTR)

This is a 40-bit register that specifies the time at which the execution of the GCL must repeat. This register consists of an 8-bit value in seconds, and a 32-bit value in nanoseconds (similar to the PTP time format with truncated seconds register). For a given GCL, the start time is "Base Time" + N * "Cycle Time" where N is an integer value that represents the iteration number starting with 0 for first iteration. If the GCL execution takes longer than the cycle time, then the list is truncated at the cycle time and the subsequent loop begins at cycle time.

75.9.11.2.2.3 Time extension register (TER)

This is a m-bit (where m = Configured time interval width + 7) register that specifies the amount of time (in nano seconds) the current GCL can extend before switching to the new GCL. This helps to avoid the execution of the small fragments of the current list before switching to a new list.

75.9.11.2.2.4 List length register (LLR)

This is an n-bit register (when n is 7, 8, 9, 10, or 11 for a GCL configured depth of 64, 128, 256, 512, or 1024 respectively) that specifies the integer value of the length of the GCL (that is the number of valid rows in each GCL). The processing of the GCL stops after the number of rows read equals to the LLR value.

75.9.11.2.3 Transmission gating implementation

A bridge or an end station can enhance to allow transmission from each TC that is yet to be scheduled relative to a known timescale. To achieve this, a transmission gate is associated with each TC; the state of the transmission gate determines if you can select the queued frames for transmission.

For a TC, the transmission gate can be in one of these two states:

1. Open: Select queued frames for transmission, in accordance with the definition of the transmission selection algorithm associated with the TC.
2. Closed: Does not select queued frames for transmission.

A frame on a traffic class queue is not available for transmission if the transmission gate is in the closed state or if there is insufficient time available to transmit the entirety of that frame before the next gate-close event associated with that queue.

The implementation has visibility to the current schedule of gate controls and the immediate next schedule of the gate controls. So the maximum Gate Open period does not exceed the sum of the two Time Intervals. This is because, a frame is selected for transmission only if the gate is currently Open and the duration of gate open interval is greater than the time taken to transfer the entire frame.

The implementation must know the frame size before the transmission, so that you can avoid the transmission overruns and only the frames that can complete are scheduled at all times.

The implementation adequately compensates for the implementation delays in the data transfer from the buffer to the line by offsetting the current time with all the relevant delays (provided by MTL_EST_Control[CTOV]). This ensures that the schedule provided is always accurately implemented at the line.

You must ensure that the GCL slot interval is always greater than the expected packet size and overhead (scheduler delay, inter frame gap (IFG), and preamble, all combined).

75.9.11.3 Idle slope computation updates

When EST is enabled, credit accumulates only when the gate is open therefore, the effective data rate of the idleSlope must increase to reflect the duty cycle for the transmission gate associated with the queue.

The idleSlope is computed on the basis of the gate open time and oper cycle time values. Program the idleSlope registers (implemented one per CBS enabled TC) based on the following equation. The existing MTL register has sufficient field width to accommodate the new values for idleSlope.

idleSlope = (operIdleSlope(N) * OperCycle/GateOpenTime)

75.9.11.3.1 Operational details of GCL

Write 1 to MTL_EST_Control[SSWL], so that the hardware can access the programmed gate control list. The first set of gate controls are applied when the current time is equal to the value in the base time register (BTR) and is held until the programmed "time interval" value.

One additional gate control event is always read ahead from the list, to avoid the transmission overruns. This enables the GCL to determine the next gate close events (if any) for the open TCs.

The scheduling is done on the basis of the gate open state and time interval of only the current and subsequent schedule. An internal accumulator adds the time intervals when gate controls are applied. BTR + Accumulator specifies the time at which the next set of gate controls are applied.

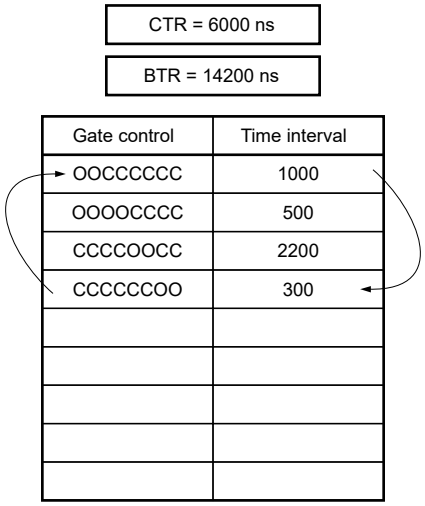


Figure 429. GCL and associated registers - basetime and cycletime

GCL is read progressively from the first row adhering to the schedule. The read operations continue until the list length (from LLR register) is reached and the execution of the list restarts at BTR + CTR time. At this point the value in CTR increments BTR to mark the beginning of a new cycle. In the absence of any gate controls, all the gates are in open state, during the execution of the list.

In cases where the execution time of the list is greater than the cycletime, the list is truncated and the next iteration starts when the current time equals BTR + CTR.

Table 570. GCL and associated registers - BTR and CTR

Current time	Gate control applied	Accumulator value	BTR (with updates)
14200	OOCCCCCC	1000	14200
15200	OOOOCCCC	1500	14200
15700	CCCCOOCC	3700	14200
17900	CCCCCCOO	4000	14200

Table continues on the next page...

Table 570. GCL and associated registers - BTR and CTR (continued)

Current time	Gate control applied	Accumulator value	BTR (with updates)
18200	OOOOOOOO	0	20200
20200	OOCCCCCC	1000	20200
21200	OOOOCCCC	1500	20200

Table 570 describes an example in which the execution starts at 14200 and the first set of gate controls "OOCCCCCC" are applied immediately. The time interval is 1000 ns, so the next set of gate controls are applied at 14200 (BTR) + 1000 (Accumulator) = 15200 ns . The above table shows that there are no gate controls available after the execution of the last gate control and before the next iteration of the loop. The gates are deemed in open state during that time period as depicted at 18200 current time.

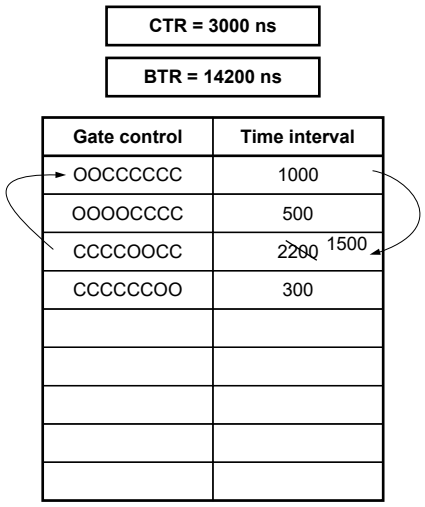


Figure 430. GCL and associated registers - BaseTime and CycleTime list execution time > CycleTime

Table 571 indicates that the list execution takes longer than the allocated CycleTime, so the list is truncated and the list starts from the BTR+CTR.

Table 571. GCL and associated registers - BTR and CTR, execution time > Cycle Time

Current Time	Gate Control Applied	Accumulator Value	BTR (with updates)
14200	OOCCCCCC	1000	14200
15200	OOOOCCCC	1500	14200
15700	CCCCOOCC	3700	14200
17200	OOCCCCCC	1000	17200
18200	OOOOCCCC	1500	17200
18700	CCCCOOCC	3700	17200

When you apply the third set of gate controls, BTR + CycleTime (17200) < BTR + Accumulator (17900), so the list is truncated and execution switches to a new iteration at 17200.

75.9.11.3.2 Installing a new GCL

The switch to the new GCL can happen in one of the following ways when a new software programmed GCL is available and executed at the new BTR value:

- New base time aligned with current schedule
- New base time unaligned with current schedule

75.9.11.3.2.1 New base time aligned with current schedule

If the choice of cycle time for the new gating cycle is unchanged from the cycle time for the current gating cycle, and if the BTR chosen for the new gating cycle (new BTR) is an integer multiple of the current cycle time (+ current BTR), then the new gating cycle exactly lines up with the old gating cycle, that is, the cycle start times for the new gating cycle is same as they would have been for the old configuration. This could be considered to be the ideal case and allows the new gating cycle to be installed and executed with no timing issues. The implementation completes the execution of an iteration of the current list and switches to the new list at the beginning of the BaseTime listed in the new list.

If (New base time > = Current time)

ConfigChangeTime = New BaseTime

Else If (New base time < Current time)

1. Set the BTRError

2. ConfigChangeTime = (New BaseTime + N* New CycleTime)

where N is the smallest integer for which the relation ConfigChangeTime >= CurrentTime and (N =< 8) is true.

When N > 8 the hardware cannot auto recover and the loop count value in BTRError reporting is set to 1111 requires the software to reprogram the new base time.

Figure 431 illustrates the installation of the new GCL along with the timelines.

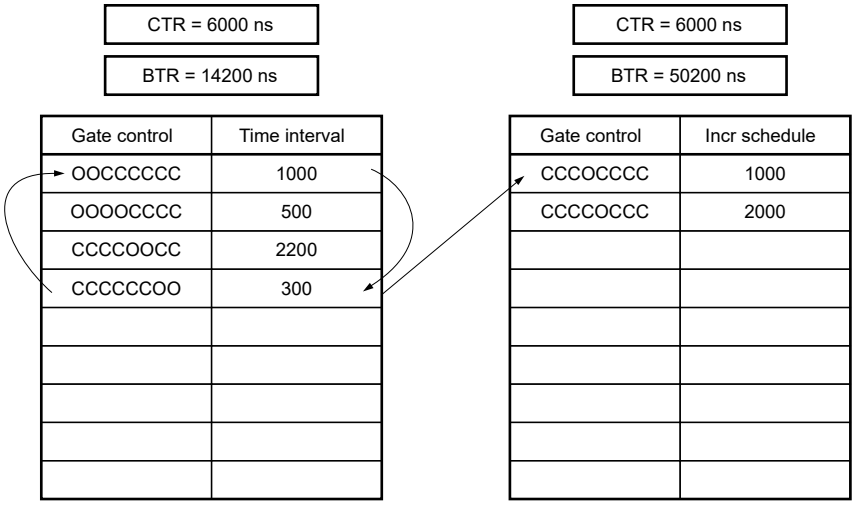


Figure 431. Switching to a new configuration that is aligned with the existing configuration

In the above example, after the sixth iteration of the first GCL, the BTR values of the old and new GCL are equal. At that point the new GCL is processed as a natural extension to the existing GCL.

Table 572. GCL and associated registers - BTR and CTR

Current time	Gate control applied	Accumulator value	BTR (with updates)
44200	OOCCCCCC	1000	44200
45200	OOOOCCCC	1500	44200
45700	CCCCOOCC	3700	44200
47900	CCCCCCOO	4000	44200
48200	OOOOOOOO	0	50200
50200	CCCOCCCC	1000	50200
51200	CCCCOCCC	3000	50200
53200	OOOOOOOO	0	56200

75.9.11.3.2.2 New base time unaligned with current schedule

If the new cycletime differs from the current cycletime or new basetime in the future and is not an integer multiple of current cycletime, then the old and new cycles do not line up. When new basetime is reached (when the new configuration is installed and starts to execute), the last old cycle is normally truncated to start the first new cycle. This could be undesirable if it results in a very short last old cycle; arguably it would be better to simply extend the penultimate old cycle by that small amount, rather than starting a very short cycle. The Cycle Time Extension Register (related to the current config list) allows this extension of the last old cycle to be done in a defined way; if the last complete old cycle ends normally in less than current Cycle Time Extension (TER) ns before the new base time, then the last complete cycle before new BaseTime is reached is extended so that it ends at new BaseTime.

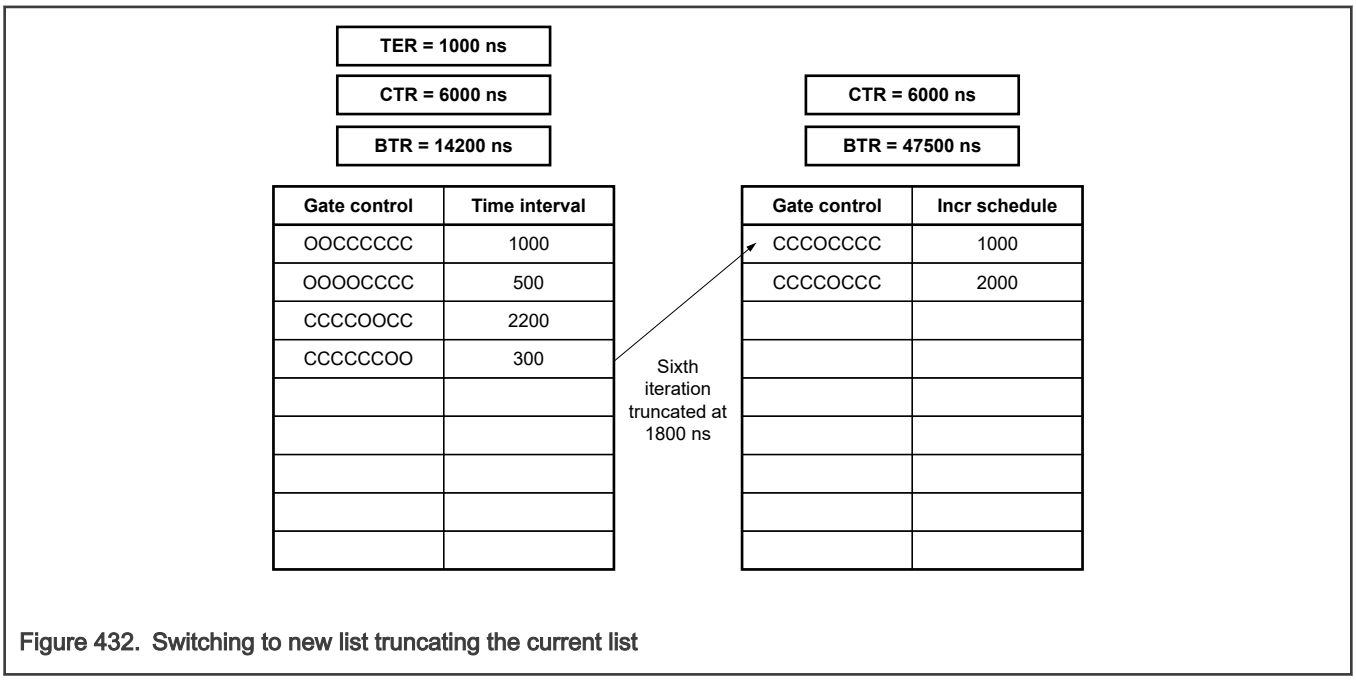


Figure 432. Switching to new list truncating the current list

At the end of the fifth iteration the Current time + Cycle time extension (TER) < New BTR so the sixth iteration of current configuration is started. During the sixth iteration of the current list when the new BTR value is smaller than the next schedule in the current list, it switches to the new list.

Table 573. Extending to new list by truncating the current list

Current time	Gate control applied	Accumulator value	BTR (with updates)
44200	OOCCCCCC	1000	44200
45200	OOOOCCCC	1500	44200
45700	CCCCOOCC	3700	44200
47500	CCCOCCCC	4000	44200
48500	CCCCOCCC	0	50200
50500	OOOOOOOO	1000	50200

Below is an example where the current config list is extended instead of starting a new iteration as the extension time of 800 ns is less than the allowed cycle extension time (TER) of 1000 ns.

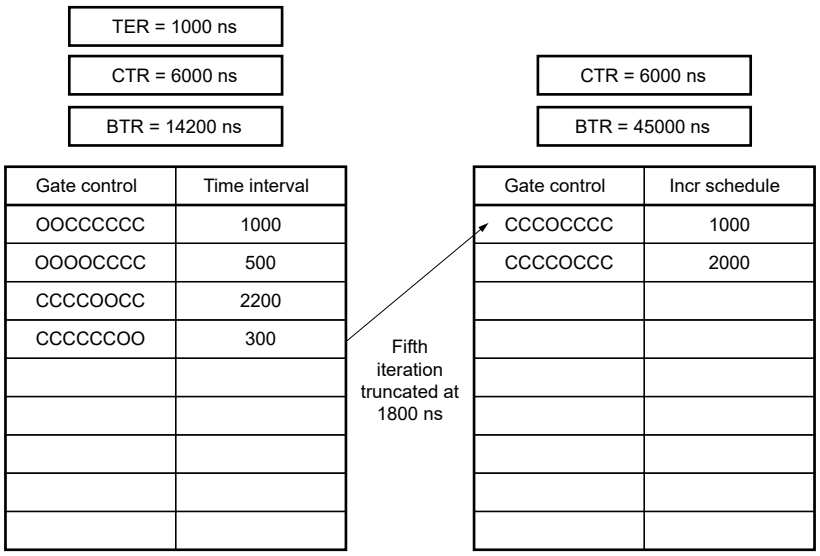


Figure 433. Switching to new list by extending the current list

Table 574. Switching to new list by extending the current list

Current time	Gate control applied	Accumulator value	BTR (with updates)
38200	OOCCCCCC	1000	38200
39200	OOOOCCCC	1500	38200
39700	CCCCOOCC	3700	38200
41900	CCCCCCOO	4000	38200
42200	OOOOOOOO	0	44200

Table continues on the next page...

Table 574. Switching to new list by extending the current list (continued)

Current time	Gate control applied	Accumulator value	BTR (with updates)
45000	CCCOCCCC	1000	45000
46000	CCCCOCCC	3000	45000
48000	OOOOOOOO	0	51000

75.9.11.4 Impact of Transmit Scheduling Algorithms on EST

When EST is used in isolation, the Gate Control List manages the final open/close state of the Queues along with the checks carried out by the Transmission Selection Algorithm in MTL. As the Gate Controls operate on a predefined repetitive schedule, it is recommended to use Strict Priority or Credit Based Shaper (CBS) scheduling algorithms.

Other algorithms such as the Weighted Round Robin (WRR), Deficit Weighted Round Robin (DWRR) and Weighted Fair Queuing (WFQ) implement masking of the queues based on the current winning queue. The algorithm is applied only among the group of queues that open simultaneously. To ensure Queues whose gates are "Open" get priority, these algorithms are modified to treat "gate open" queues and "gate closed" queues as separate groups giving priority to gate open queues.

For example, consider 4 queues (Q3, Q2, Q1, Q0) with weights 4:3:2:1; Q3 and Q2 are in Open state in slot one slot, while Q1 and Q0 are in Open state in another slot. In this case, the scheduler works as follows:

1. In the first slot, the Q3 and Q2 are serviced in the ratio of 4:3 for the duration the slot is open.
2. In the second slot, the queues Q1 and Q0 are serviced in the ratio of 2:1.
3. Fresh arbitration is started every time a slot is opened.

In other words, the traffic does not get distributed in the intended ratio of 4:3:2:1; but as two groups with different ratios and only for the duration of the slot when the gates are open continuously

NOTE

See Programming guidelines for EST

75.9.12 Frame preemption (FPE)

Frame preemption breaks the interfering frame into smaller fragments. Therefore, the guard band needs to be only as large as the largest possible interfering fragment instead of the largest possible interfering frame. During the guard band, only the frames that can complete the transmission of the entire frame before the next gate close event are permitted. This ensures that the high priority traffic can always start at the beginning of the window reserved for it.

Preemption allows one or more higher priority (express) frames to interrupt the transmission of a lower priority (preemptable) frame; the preemptable frame transmission is resumed and completed after the express frame transmission is complete. To support frame preemption, the following two abstractions of the MAC are used:

- A preemptable MAC, called pMAC, which carries the preemptable traffic.
- An express MAC, called eMAC, which carries the express traffic.

In the implementation, only parts of the MAC that holds the state during preemption is replicated and represented as pMAC and eMAC. The MAC uses the following two ways to puts on hold, the transmission of the preemtable traffic, in the presence of express traffic:

- The MTL scheduler interrupts the preemptable traffic that is currently being transmitted. When the preemption capability is active, the MAC interrupts the transmission and reception of preemptable frames. A preempted fragment can be followed by zero or more express frames, before the continuation fragments. The preemptable frame can be fragmented any number of times, however, the minimum final and non-final fragment length criterion must be is met. However, interleaving of more than 1 preemptable packet is not permitted. This implies that if a preemptable packet is fragmented by an express

packet, another preemptable packet cannot be transferred until all the remaining fragments of the first preempted packet are transferred.

- The MTL scheduler prevents starting the transmission of preemptable traffic. When the preemption capability is inactive, the pMAC entity is disabled and only express traffic is transmitted or received. Frame preemption feature can be enabled by setting EFPE field of the MAC_FPE_CTRL_STS register.

If you are unable to schedule fragmented packets after a certain number of attempts because of some reason, there is a possibility that the remaining fragmented packets are dropped. For this, software can program MTL_EST_Control[LCSE] to increase number of iterations or MTL_EST_Control[DFBS] to disable dropping of packet.

NOTE

EST and FPE cannot be used at the same time.

75.9.12.1 GCL Modification to Support FPE

In the EST-only configuration, the GCL entry has up to 24 bits of Time Interval and up to 8 high order bits representing the Gate Open/Close state requirements as shown in the following figure.

Gate control (up to 8 bits)	Time interval (ns) 16, 20 or 24 bits
OOCCCCCC	T0
OOOOCCCC	T1
CCCCOOCC	T2
CCCCCCOO	T3
|	|
|	|
OCOCOCOO	T126
OOOOCCCC	T127

Figure 434. Gate Control List When FPE is Disabled

EST only supports SetGate operation, which implies that the gates are set to either open or close at a given time interval. However, when both Frame Preemption (FPE) and EST are enabled, the GCL also supports Set-and-Hold-MAC and Set-and-Release-MAC operations, in addition to the SetGate operation. To enable the support of hold and release operations, the format of the GCL is slightly changed while configuring and enabling the FPE. The first Queue (Q0) is always programmed to carry preemption traffic and therefore it is always Open. The bit corresponding to Q0 is renamed as Release/Hold MAC control. The TX Queues whose packets are preemptable are indicated by setting the PEC field of the MTL_FPE_CTRL_STS register. The GCL bit of the corresponding Queue are ignored and considered as always "Open". So, even if the software writes a "0" ("C"), it is ignored for such queues.

Gate control (up to 7 bits)	Release or hold indication	Time interval (ns) 16, 20 or 24 bits
CCCOOOO	0	T0
CCCCOOO	0	T1
OCCCOOO	1	T2
COCCOOO	1	T3
CCOCOOO	1	T4
CCCOOOO	0	T5
CCCCOOO	0	T6

Figure 435. Gate Control List When FPE is Enabled

When the Release/Hold bit transitions from a '0' to '1', a Set-and-Hold-MAC operation is performed. This marks the cease of the preemptable traffic. This is achieved by sending a Hold request to MTL "ha" ns in advance (where ha is the time interval mentioned in the Hold Advance (HADV) field of the MTL_FPE_Advance register). When the Release/Hold bit transitions from a '1' to '0' a Set-and-Release-MAC operation is performed. This marks the resuming of the preemptable traffic. This is achieved by sending a Release request to MTL "ra" ns in advance (where ra is the time interval mentioned in the Release Advance (RADV) field of the MTL_FPE_Advance register). The preemptable traffic is blocked for the time duration the Release/Hold bit is set to a '1' in the Gate Control List.

75.9.12.2 Impact of Preemption on CBS

In Credit Base Shaper(CBS), the definition of "Transmit" is as follows:

- TRUE for the duration of frame transmission from the queue.
- FALSE when frame transmission from the queue is complete.
- When CBS algorithm is used along with frame preemption, the value of "Transmit" is TRUE only while the frame is being transmitted by the MAC. If the frame transmission is delayed or interrupted (for instance, a preemptable frame transmission is interrupted to allow the transmission of an express frame from a different queue, or the start of express frame is delayed because a preemptable frame is being transmitted) the value of "Transmit" is FALSE until transmission of the frame commences or is resumed.

Also, the value of "Transmit" is FALSE during the transmission of any overhead that is a consequence of frame preemption. For example, additional frame overhead (mCRC, Fragment Count) that is added to the preemptable frame.

At any given time, if there are no frames in the queue, and the value of Transmit is FALSE, and credit is positive value, the credit value is set to zero if here is no preemptable frame from the queue for which transmission is in progress but has been interrupted.

75.9.12.3 mPacket Format

When the preemption capability is active, MAC sends mPackets to the PHY. An mPacket can be one of the following:

1. A express packet
2. A preemptable packet
3. An initial fragment of a preemptable packet
4. A continuation fragment of a preemptable packet

Start mPacket Delimiter (SMD)

The value of the SMD indicates whether the mPacket contains an express packet, the start of a preemptable packet (initial fragment or complete packet), or any of continuation fragments of a preemptable packet. Following table shows the valid SMD values.

Table 575. Possible SMD Values of mPacket

mPacket Type	Notation	Frame Count	Value
verify packet	SMD-V	-	0x07
respond packet	SMD-R	-	0x19
express packet	SMD-E	-	0xD5
preemptable packet start	SMD-S0	0	0xE6
	SMD-S1	1	0x4C
	SMD-S2	2	0x7F
	SMD-S3	3	0xB3
continuation fragment	SMD-C0	0	0x61
	SMD-C1	1	0x52
	SMD-C2	2	0x9E
	SMD-C3	3	0x2A

frag_count

A frag_count is a modulo-4 counter that increments for each continuation fragment of the preemptable packet. The frag_count protects against mPacket reassembly errors by enabling detection of the loss of up to 3 packet fragments.

The frag_count field is present only in mPackets with SMD-C notation (continuation fragment). The frag_count is zero in the first continuation fragment of each preemptable packet.

Table 576. Possible frag_count Values

frag_count	Value
0	0xE6
1	0x4C
2	0x7F
3	0xB3

mData

The contents of the packet from the MAC, starting with the first byte after the SFD to the last byte before the FCS are sent in the mData fields of one or more mPackets for that frame. The minimum size of the mData field is 60 bytes.

CRC The CRC field contains a cyclic redundancy check (CRC) and has an indication of the final mPacket of a frame. In the final mPacket of a frame, the CRC field contains the last 4 octets of the MAC frame (the FCS field).

For other mPackets, the CRC field contains an mCRC value. The mCRC is calculated on the octets of the packet from the first octet of the frame (the octet following the SFD of preemption frames) to the last octet of the packet transmitted in that mPacket by performing an XOR of the calculated 32 bit CRC value of the fragment and a value of 0x0000FFFF.

Summary of Packet Formats

- Express Packet: 7bytes of PREAMBLE, SMD-E, Data, and CRC
- Complete Preemptable Packet: 7 bytes of PREAMBLE, <Current Preemptable packet SMD>, Data, CRC
- Initial Fragment (non-final) of Preemptable Packet: 7 bytes of PREAMBLE, <Current Preemptable packet SMD>, Data, mCRC
- Continuation fragments (non-final) of Preemptable packet: 6 bytes of PREAMBLE, <Current Preemptable continuation fragment SMD>, <Current Preemptable continuation fragment FC>, Data, mCRC
- Final fragment of Preemptable packet: 6 bytes of PREAMBLE, <Current Preemptable continuation fragment SMD>, <Current Preemptable continuation fragment FC>, Data, CRC

Table 577. Current and Previous SMD Values

Previous Preemptable packet SMD	Current Preemptable packet SMD
SMD-S0	SMD-S1
SMD-S1	SMD-S2
SMD-S2	SMD-S3
SMD-S3	SMD-S0

Table 578. Current and Previous SMD Values

Previous Preemptable fragment SMD	Previous Preemptable fragment FC	Current Preemptable continuation fragment SMD	Current Preemptable continuation fragment FC
SMD-S0	NA	SMD-C0	FC0
SMD-S1	NA	SMD-C1	FC0
SMD-S2	NA	SMD-C2	FC0
SMD-S3	NA	SMD-C3	FC0
SMD-C0	FC0	SMD-C0	FC1
SMD-C0	FC1	SMD-C0	FC2
SMD-C0	FC2	SMD-C0	FC3
SMD-C0	FC3	SMD-C0	FC0
SMD-C1	FC0	SMD-C1	FC1
SMD-C1	FC1	SMD-C1	FC2
SMD-C1	FC2	SMD-C1	FC3
SMD-C1	FC3	SMD-C1	FC0
SMD-C2	FC0	SMD-C2	FC1

Table continues on the next page...

Table 578. Current and Previous SMD Values (continued)

Previous Preemptable fragment SMD	Previous Preemptable fragment FC	Current Preemptable continuation fragment SMD	Current Preemptable continuation fragment FC
SMD-C2	FC1	SMD-C2	FC2
SMD-C2	FC2	SMD-C2	FC3
SMD-C2	FC3	SMD-C2	FC0
SMD-C3	FC0	SMD-C3	FC1
SMD-C3	FC1	SMD-C3	FC2
SMD-C3	FC2	SMD-C3	FC3
SMD-C3	FC3	SMD-C3	FC0

75.9.12.4 Transmit Preemption

When FPE is enabled (setting EFPE=1 in MAC_FPE_CTRL_STS register), the MTL preempts a preemptable frame, when a "hold" request is asserted (EST/Qbv configured and enabled) express frames are available for transmission, that is, frame is present in MTL FIFO and is qualified for arbitration, after ensuring that the minimum mPacket mData field size is met. Therefore, preemption occurs only if at least 60 bytes of the preemptable frame have been transmitted and at least 64 bytes (including the frame CRC) remain to be transmitted.

The earliest starting position of preemption is controlled by the AFSZ field of the MTL_ FPE_CTRL_STS Register. Preemption does not occur until at least 64 * (1+ AFSZ) - 4 bytes of the preemptable frame have been sent.

When preemption occurs, all the preemptable queues are blocked and only the express queues are allowed to arbitrate (if more than one express queue has traffic) and transmit.

Continuation fragment of the preempted frame is the first frame to be transmitted after "release" request is asserted (EST/Qbv configured and enabled) and all the express traffic transmission completes.

NOTE

All the PTP packets should be transmitted as express packets

MTL communicates the following frame-type information to the MAC using a 2-bit preemption control signal on the MTI interface qualified by SoF and EoF.

- Express Frame
- Preemption Frame (Full or Fragment)
- Continuation Fragment (non-Final or Final)

Table 579. Preemption Control Values on MTI Interface for Various Frame Types

Qualifier	Preemption Control Value	Frame Type
SoF	00	Start Express
SoF	01	Start Preemption

Table continues on the next page...

Table 579. Preemption Control Values on MTI Interface for Various Frame Types (continued)

Qualifier	Preemption Control Value	Frame Type
SoF	10	Continuation Fragment
SoF	11	Reserved
EoF	00	End of Frame
EoF	01	End of Fragment

MTL should wait for the previous fragment status to be received before resuming the continuation fragments of a preempted frame. Fragment status is described in detail in the Tx Fragment Status section.

75.9.12.4.1 MAC Tx Preemption

MAC supports preemption by implementing the functionality needed to generate the mPackets as described in “mPacket Format”.

Based on the Preemption Control value received on the MTI interface (qualified with SoF and EoF), MAC determines the frame type (shown in the above table 'Preemption Control Values on MTI Interface for Various Frame Types') and generates mPackets accordingly.

When the preemption capability is active, MAC replaces the SFD of a preemption packet with an SMD-S value. A 2-bit rolling frame count is encoded in the SMD-S value.

The SMD-E value is the same as the SFD value so the SFD of an express packet does not need to be replaced.

75.9.12.4.1.1 Tx Fragment Status

MAC sends Tx fragment status to indicate successful transmission of fragmented mPackets.

In case of a transmission error (underflow, jabber, and so on) the frame status is sent (and not a fragment status) with an error indication along with all other relevant status fields. In case of receiving an error status for a transmitted fragment, the MTL drops the remaining fragments and does not send any more continuation fragments.

75.9.12.5 Receive Preemption

75.9.12.5.1 MAC Receive Preemption

When FPE is enabled, the MAC Receiver passes the incoming packets and differentiates between Express packets and preemptable packets. An SMD containing an SMD-E indicates express packet, and SMD containing an SMD-S indicates the first mPacket of a preemptable packet.

If an mPacket containing an SMD-S is received when MAC has not completed receiving the previous preempted packet, MAC sets a CRC Error status for the previously received partial packet.

When an SMD-S is detected, MAC records the frame count indicated by the SMD and then begins sending data on the MRI interface.

The MAC checks the last four bytes of the mPacket . If the last four bytes of the mPacket do not match CRC, that indicates the end of the packet with or without a CRC error as per the CRC check result. If the last four octets of the mPacket match, that indicates that the packet was preempted.

An SMD containing an SMD-C indicates an mPacket that continues the data for a preempted packet. Upon receiving an SMD value of SMD-C, MAC checks the following:

1. A preempted packet is in progress
2. The frame count indicated by the SMD matches the frame count of the packet in progress
3. The frag_count value indicates the next fragment count.

If any of the above check fails, the mPacket is discarded and MAC sets a CRC Error status for the partially received packet.

If all the checks pass, the next fragment count is incremented modulo 4.

When a packet is preempted, the MAC saves the state of the partially received packet (filter check status, timestamp, length fields etc.) and will be able to process any received Express packets before the continuation fragment is received.

The MAC Receiver sends a "dummy status" for all the mPacket fragments successfully received and sends the Rx status with the final fragment. If an error is detected during any of the fragments the Rx status is sent and the fragment is marked as final fragment. All subsequent continuation fragments received for this packet are dropped in the MAC.

The MAC communicates the following frame type information to the MTL using a 2-bit preemption control signal.

1. Express Frame
2. Preemption Frame (Full or Fragment)
3. Continuation Fragment (non-Final or Final)

Table 580. Preemption Control Values on MRI Interface for Various Frame Types

Qualifier	Preemption Control Value	Frame Type
SoF	00	Start Express
SoF	01	Start Preemption
SoF	10	Continuation Fragment
SoF	11	Reserved
EoF	00	End of Frame
EoF	01	End of Fragment

75.9.12.5.1.1 Data Alignment

When a received frame cannot be fragmented on any byte boundary, MAC retains the unaligned bytes of data in the previous fragment and resends them with the next fragment as shown in the following figure.

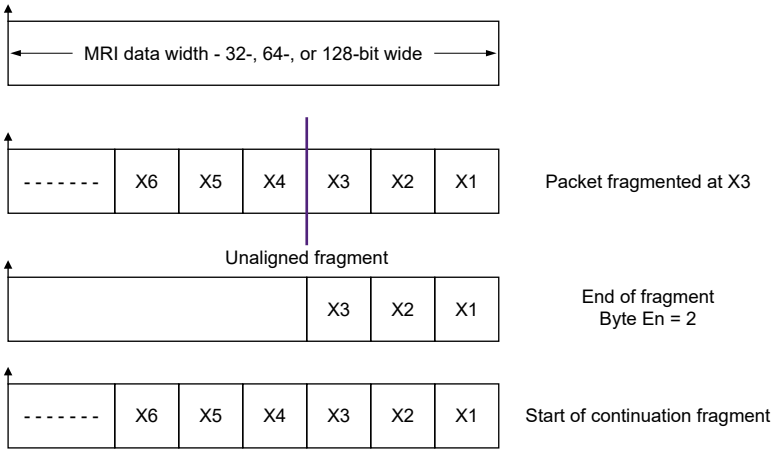


Figure 436. Data Alignment Feature of MAC

75.9.12.5.1.2 MTL Receive Preemption

The MTL Rx must have at least 2 Rx queues to support FPE function as the preemptable packets and the express packets must be routed to separate Rx queues. The destination Rx queue of a received packet is controlled by MAC_RxQ_Ctrl[n] registers. Program these registers such that Preemptable traffic and express traffic are not routed to the same Rx queue. As the queue mapping for tagged packets is based on VLAN user-priority field, this implies that priority of preemptable and express packets are mutually exclusive. In other words, packets of a certain priority (traffic class) are either express or preemptable but cannot be both.

For the preemptable frames, in addition to the PSRQ (Priority Selected in Rx queue) based queue routing, a programmable "Frame Preemption Residue Queue" (FPRQ) is supported to route all other Preemption Packets received (Untagged, SA/ DA or VLAN Filter Fails forwarded due to RA being set or VTFE being reset).

Table 581. Preemption Control Values on MRI Interface for Various Frame Types

Preemption Packet Type	Queue Routing
AV Tagged Packets passing the SA/DA and VLAN filters	PSRQ*
DCB/Generic Tagged Packets passing the SA/DA and VLAN filters	PSRQ**
Tagged Packets failing the SA/DA and VLAN filters without setting the Receive All (RA = 0) or setting VTFE = 1	Dropped
All other packets OR when RA = 1	FPRQ

75.9.12.5.1.3 MTL Receive Arbitration

On the ARI Interface, data is fetched from the MTL Rx FIFO based on the arbitration selected in the MTL_Operation_Mode register. Frame based arbitration can be used only when all the MTL Preemption Queues are operating in Store and Forward mode, else there will be loss of bandwidth on the ARI interface because all the fragments of a preemptable packet is not available. Therefore, any express packets received between the fragments are blocked until all the fragments are received and transferred; this defeats the purpose of express traffic.

When operating in either Threshold (cut through) or Store and forward modes of operation, PBL based arbitration is recommended over Frame based arbitration. In case of PBL based arbitration, the watermark check is always performed and the arbitration/ transfer of data in terms of chunks of "PBL" size of data which is almost similar to the concepts of "fragments". Therefore the express queue packets get blocked for less time as well as the ARI interface transfers the data without loss of efficiency.

75.9.12.6 Verify and Respond mPackets

When FPE function is present, the MAC can receive and detect the Verify and Respond mPackets, even when FPE is not enabled by software. When MAC detects valid Verify/Respond mPackets, it notifies the software by setting the RVER and RRSP fields of MAC_FPE_CTRL_STS register respectively. Optionally an interrupt can be generated. As such packets have empty (all-zero) data payload, they are dropped inside the MAC and not forwarded to the MRI.

Software can set the SVER and SRSP fields of MAC_FPE_CTRL_STS register to request MAC to transmit Verify and Respond mPackets respectively. Upon successful transmission of these frames, MAC clears the SVER/SRSP bits and sets the TVER & TRSP fields of MAC_FPE_CTRL_STS register. Optionally an interrupt can be generated when these events occur.

Software must ensure that frame preemption verify and response packet are not triggered at the same time. Trigger either of the packets and then wait for its completion before triggering another packet.

75.9.12.7 Frame Preemption and MMC Counter and Interrupt Registers

The following MMC counters and associated Interrupt registers are instantiated/present in the MAC when Frame Preemption feature is enabled.

Table 582. MMC Counters and Associated Interrupt Registers

Frame Assembly Error Counter	Description	Associated MMC Counter
Frame Assembly Error Counter	A 32-bit counter that provides the number of MAC frames with reassembly errors on the Receiver, due to mismatch in the Fragment Count value.	MMC_Rx_Packet_Assembly_Err_Cntr
Frame SMD Error Counter	A 32-bit counter that provides the number of received MAC frames rejected due to arriving with an SMD-C when there was no preceding preempted frame.	MMC_Rx_Packet_SMD_Err_Cntr
Frame Assembly OK Counter	A 32-bit counter that provides the number of MAC frames that were successfully reassembled.	MMC_Rx_Packet_Assembly_Ok_Cntr
MAC Rx Fragment Counter	A 32-bit counter that provides the number of additional mPackets received due to preemption.	MMC_Rx_FPE_Fragment_Cntr
MAC Tx Fragment Counter	A 32-bit counter that provides the number of additional mPackets transmitted due to preemption.	MMC_Tx_FPE_Fragment_Cntr
Hold Request Counter	A 32-bit counter that maintains the count of number of times a hold request is given to MAC.	MMC_Tx_Hold_Req_Cntr

75.9.12.7.1 Additional Registers Associated With MMC Interrupts

Following are the additional registers associated with MMC interrupts for the MMC error counters:

- MMC_FPE_Tx_Interrupt
- MMC_FPE_Tx_Interrupt_Mask
- MMC_Rx_Interrupt
- MMC_FPE_Rx_Interrupt_Mask

75.9.13 Time-Based Scheduling

Time-based scheduling feature is suitable for traffic whose periodicity and rate are predictable. To improve the quality-of-service of such traffic,

- The transmit DMA fetches the packet from the host memory for transmission at designated time. This helps the software to setup the Transmit descriptors in advance even before packet is ready/available. It reduces the overhead on the software and avoids constant monitoring of the time and preparing descriptors just in time when the packet is targeted to be transmitted.
- The MAC transmits the packet only at the designated/pre-determined time even if the packets are fetched in advance. This helps in maintaining a constant transmission rate that can be consumed by the receiver station; therefore avoiding congestion and excessive buffering in the network.

The time-based scheduling feature supports fetching and launching an Ethernet packet at (or after) a pre-determined time. The time-based scheduling is supported only in the following modes/configurations:

- Full duplex mode
- Link speed is 100Mbps or higher

The following figure shows the time-based scheduling flow.

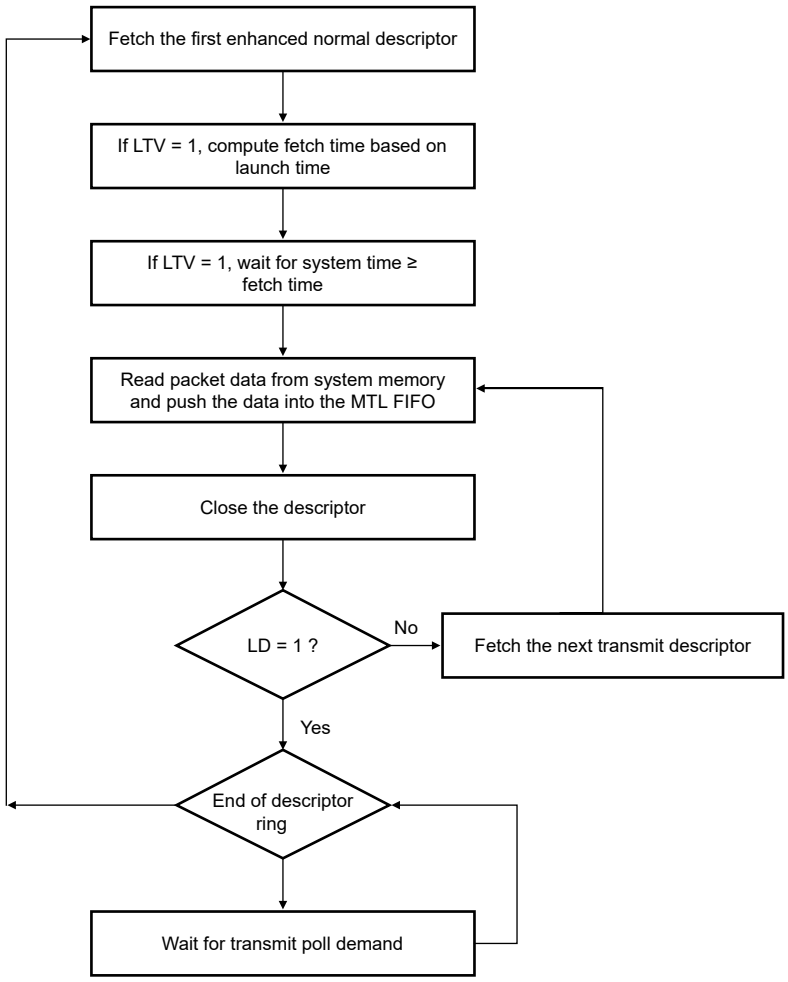


Figure 437. Time-Based Scheduling Flow

Following are the definitions specific to the time-based scheduling feature:

75.9.13.1 Launch time

The time beyond which MTL can schedule the packet for transmission. The launch time resets after the transmission of one frame. These are the two formats of the launch time.

Normal or absolute format:

In this format, the launch time is interpreted in the normal or absolute format if the ESTM bit of the MTL_TBS_CTRL register is set to 0.

The launch time is a 32-bit value, where most-significant 8-bits represent the time in seconds and the rest 24-bits represent the time in 256 ns. The launch time is compared against the IEEE 1588 based system or PTP time (bits[39:8]) and rolls over after 256 seconds.

The maximum value of the lower 32-bits of system time is 999,999,999 decimal (0x3B9AC9FF) and it wraps to 0 when reaching this value (representing a full second). Therefore, the maximum value of the lower 24-bits of the launch time (after multiplying by 256) must be 0x3B9AC9.

As the maximum value of launch time is 256 seconds,

- Launch time is greater than current system time when its value is between system time[39:8] and System time[39:8] + 128 sec.

- Launch time is less than current system time when its value is between System Time[39:8] + 128sec and System Time[39:8] + 256 sec, because this is a modulo 256 computation.

EST or offset format

In this format, the launch time is a offset value relative to the time which the BTR of the GCL list provided in the GCL slot number (GSN) indicates. The value in the BTR is always updated to the start time of the current loop of the GCL.

For each packet, the GSN value and the launch time value are specified in the enhanced transmit descriptor in the DMA configurations, or as control word in the MTL configurations.

The launch time offset is a 32-bit value; the upper 8-bits represent the time in seconds and the rest 24 bits represent the time in 256 ns, which is added to the BTR value corresponding to GCL slot number. The value of the launch time offset must be smaller than the value of the cycle time (specified in the CTR register that is implemented when EST is enabled). If the CTR is greater than or equal to 1s, the maximum value of the lower-24 bits of launch time offset must be 999,999,999 decimal (0x3B9AC9FF).

In this format, the launch time is a 64-bit value, which is interpreted as, Launch time = Launch GSN BTR[63:0] + (Launch time offset[31:0] << 8), which is compared with the system time [63:0].

GSN BTR is the Base Time value at which the Launch GSN loop started execution.

GCL slot number (GSN) A modulo 16 count of the GCL loop count is implemented and it is known as GSN. The count is incremented for every new GCL loop. Installation of new GCL list does not impact the count. Read MTL_EST_Status[CGSN] to obtain the current GCL slot number.

The maximum value of GSN is 15. Therefore the GSN values between MTL_EST_Status[CGSN] and MTL_EST_Status[CGSN]+8 represent current or future slots and all other GSN values are interpreted as elapsed slots or past slots. So, for the correct interpretation of time, GSN value must be between MTL_EST_Status[CGSN] and MTL_EST_Status[CGSN]+8.

75.9.13.2 Launch expiry time

Normal or Absolute Mode

In this mode, when MTL_TBS_CTRL[LEOV] = 1, MTL_TBS_CTRL[LEOS] determines the maximum amount of time a frame is eligible for launch, starting from the time the frame becomes eligible for launch.

The launch expiry offset is a 24-bit value defined in 256 ns units, with a maximum possible value of 999,999,999 ns (0x3B9AC9FF).

Launch expiry time = (Launch time[39:8] + LEOS[32:8]) * 256 ns

The packet with a specific launch time is considered as eligible for transmission when the launch time is less than the system time and (if MTL_TBS_CTRL[LEOV] = 1) the system time is less than the launch expiry time.

When the system time is greater than the launch expiry time, the frame is categorized as expired and it drops from the MTL FIFO.

NOTE

- For correct interpretation and meaningful operation, the fetch, launch, and launch expiry time must never set to a value larger than the Current system time + 128 sec; such a value is interpreted as time that has already elapsed.
- In Full Duplex mode, the frames dropped from the MTL FIFO have error summary (Bit 15) and excessive deferral (bit 3) of TxStatus set.

EST/Offset Mode

In EST mode, when MTL_TBS_CTRL[LEOV] = 1, MTL_TBS_CTRL[LEGOS] and MTL_TBS_CTRL[LEOS] determines the maximum amount of time a frame remains eligible for launch, starting from the time the frame becomes eligible for launch.

Launch Expiry Offset = (LEGOS: LEOS)

LEGOS holds the GSN offset (multiples of CTR time) and LEOS holds maximum value of CTR (sub CTR values) in ns.

Launch expiry offset is a 24-bit value defined in the units of 256 ns, with a maximum possible value of the smaller of 999,999,999 ns or CTR-1 ns.

The Launch expiry GSN is computed as follows:

Launch expiry GSN = (Launch GSN + LEGOS + CCMA) where, CCMA is the CTR carry due to modulo addition. This value is 1 if ((Launch time offset + LEOS) << 8) is equal to or greater than CTR.

When CCMA = 1, Launch expiry offset = (Launch time offset+ LEOS) - CTR

When CCMA = 0,

Launch expiry offset = (Launch time offset + LEOS).

Launch expiry time = Launch expiry GSN BTR [63:0] + Launch expiry time offset.

When MTL_TBS_CTRL[LEOV] is not 1, the launch expiry time is not checked.

When MTL_TBS_CTRL[LEOV] = 1, and

- the system time is greater than the launch expiry time, the frame drops from MTL FIFO. Then the frame is considered as expired.
- the launch time is smaller than the system time and the launch expiry time is greater than the system time, then the frame is considered eligible for launch.

NOTE

- Max value of MTL_TBS_CTRL[LEGOS] is 7. This indicates that when MTL_TBS_CTRL[LEOV] = 1, the frame has a maximum life time of <8 GCL loop iterations after it becomes eligible for launch.
- The slot number of the first GCL list that executes each time after EST is enabled, is zero.

75.9.13.3 Fetch time

This accounts for all possible delays in the DMA fetch operation and ensures that the frame is present in the MTL FIFO before the launch time.

If DMA_TBS_CTRL[FTOV] is 1, it indicates the fetch time for each packet. If DMA_TBS_CTRL[FTOV] is not 1, it indicates that the fetch time offset is not valid and DMA fetches packets without any time constraints.

Normal/Absolute mode

In this mode fetch time is derived or calculated by reducing the time specified in DMA_TBS_CTRL[FTOS] from the given launch time.

In Normal mode, the fetch launch time is computed as:

Fetch Time[39:8] = (Launch Time[39:8] - FTOS[31:8])

The fetch time is 32-bits and is compared against the system time[39:8] to determine whether it is eligible for fetching the frame:

- The fetch time is defined as greater than system time if the fetch time is in the range of system time[39:8]" and system time[39:8] + 128 sec. The frame is considered as not-eligible for fetch.
- The fetch time is defined as smaller than the system time if the fetch time is in the range of system time[39:8] + 128 sec and system time[39:8] + 256 sec. The frame is considered as eligible for fetch.

This is a modulo 256 computation.

EST/Offset mode

In this mode, the fetch GSN offset (DMA_TBS_CTRL[FGOS]) provides the slot number offset to deduct from the launch GSN. In this case the FTOS value must be smaller than 999,999,999 ns or CTR-1 ns.

If (Launch time offset >= FTOS):

Fetch time offset = ((Launch time offset - FTOS) * 256ns)

CBFS(CTR borrow for fetch subtraction) = 0

If (Launch time offset < FTOS) Fetch time offset = CTR + ((Launch time offset - FTOS) * 256ns)

CBFS (CTR borrow for fetch subtraction) = 1

The fetch GSN is computed as follows:

Fetch GSN = Launch GSN - FGOS - CBFS

Fetch time = Fetch GSN BTR[63:0] + Fetch time offset

The frame is eligible for DMA fetch when the fetch time is smaller than the system time.

DMA operations sequence

This is the sequence of operation when FTOV = 1:

1. Fetch the first enhanced normal descriptor (FD = 1).
2. Compute the fetch time on the basis of the launch time and wait for the system time to be greater than fetch time, if LTV = 1 and fetch is enabled.
3. Read the frame (data) from the host memory and transfer to MTL FIFO.
4. Close the normal descriptor.
5. Fetch the next normal descriptor (if the previous descriptor was not the last).
6. Repeat steps 4 to 6, until the last descriptor of the frame (LD = 1).

After the last descriptor of a frame, program the next enhanced normal descriptor with a new launch time and with LTV = 1. Otherwise, process the subsequent frames without any time restrictions.

See Programming the launch time in time-based scheduling for more information.

75.10 TCP/IP Offloading Features

This section and all other sections, along with respective sub-sections are Synopsys Proprietary. Used with permission.

75.10.1 Transmit Checksum Offload Engine

The MAC has a Checksum Offload Engine (COE) to support checksum calculation and insertion in the Transmit path, using which, the software can offload the task of checksum insertion to the hardware. In the transmit path MAC calculates the checksum and inserts it in the Tx packet. This feature helps in reducing the load on the software and can improve the overall throughput of the system. This feature is supported for only Q0 queue.

The checksum offload engine module supports two types of checksum calculation and insertion. The checksum engine can be controlled for each packet by setting the CIC bits (TDES3 Bits[17:16]).

Note: The checksum for TCP, UDP, or ICMP is calculated over a complete packet, and then inserted into its corresponding header field. Because of this requirement, when this function is enabled, the Tx FIFO automatically operates in the store-and-forward mode even if IP is configured for Threshold (cut-through) mode.

75.10.1.1 IP Header Checksum Engine

In IPv4 datagrams, the integrity of the header fields is indicated by the 16-bit Header Checksum field (the eleventh and twelfth bytes of the IPv4 datagram). The COE detects an IPv4 datagram when the Type field of Ethernet packet has the value 0x0800 and the Version field of IP datagram has the value 0x4. The checksum field of the input packet is ignored during calculation and replaced with the calculated value.

NOTE

IPv6 headers do not have a checksum field. Therefore, the COE does not modify the IPv6 header fields.

The result of this IP header checksum calculation is indicated by the IP Header Error status bit in the Transmit status (Bit 0 in Table 19-12 on page 1330). This status bit is set whenever the values of the Ethernet Type field and the Version field of IP header are not consistent, or when the Ethernet packet does not have enough data, as indicated by the IP header Length field. In other words, this bit is set when an IP header error is asserted under the following circumstances:

- For IPv4 datagrams:
  - The received Ethernet type is 0x0800, but the Version field of IP header is not equal to 0x4.
  - The IPv4 Header Length field indicates a value less than 0x5 (20 bytes).
  - The total packet length is less than the value given in the IPv4 Header Length field.
- For IPv6 datagrams:
  - The Ethernet type is 0x86dd but the IP header Version field is not equal to 0x6.
  - The packet ends before the IPv6 header (40 bytes) or extension header (as given in the corresponding Header Length field in an extension header) is completely received.

75.10.1.2 TCP/UDP/ICMP Checksum Engine

The TCP/UDP/ICMP Checksum Engine processes the IPv4 or IPv6 header (including extension headers) and determines whether the encapsulated payload is TCP, UDP, or ICMP. The checksum is calculated for the TCP, UDP, or ICMP payload and inserted into its corresponding field in the header. The Tx COE can work in the following two modes:

- The TCP, UDP, or ICMPv6 pseudo-header is not included in the checksum calculation and is assumed to be present in the Checksum field of the input packet. This engine includes the Checksum field in the checksum calculation, and then replaces the Checksum field with the final calculated checksum.
- The engine ignores the Checksum field, includes the TCP, UDP, or ICMPv6 pseudo-header data into the checksum calculation, and overwrites the checksum field with the final calculated value.

NOTE

For ICMP-over-IPv4 packets, the Checksum field in the ICMP packet must always be 16'h0000 in both modes, because pseudo-headers are not defined for such packets. If it does not equal 16'h0000, an incorrect checksum may be inserted into the packet.

The result of this operation is indicated by the Payload Checksum Error status bit in the Transmit Status vector (Bit 12 in Table 19-3 on page 1324). This engine sets the Payload Checksum Error status bit when it detects that the packet has been forwarded to the MAC Transmitter engine in the store-and-forward mode without the end of packet (EOP) being written to the FIFO, or when the packet ends before the number of bytes indicated by the Payload Length field in the IP Header is received. When the packet is longer than the indicated payload length, the COE ignores them as stuff bytes, and no error is reported. When this engine detects the first type of error, it does not modify the TCP, UDP, or ICMP header. For the second error type, it still inserts the calculated checksum into the corresponding header field. Following table describes the functions supported by Transmit Checksum Offload engine based on the packet type. When the MAC does not insert the checksum, it is indicated as "No" in the table.

NOTE

You should not enable checksum insertion for IPv4 or IPv6 packets that are greater than the frame size constraint specified in Description of Transmit Checksum Offload Engine because it may result in incorrect checksum insertion or unexpected behavior.

Table 583. Transmit Checksum Offload Engine Functions for Different Packet Types

Packet Type	Hardware IP headerchecksum insertion	Hardware TCP/ UDPchecksum insertion
Non-IPv4 or IPv6 packet	No	No
IPv4 packet is greater than 1,522 bytes (2,000 bytes when IEEE 802.3ad Support for 2K Packets is enabled in MAC) but less than or equal to the frame size constraint specified in Transmit Checksum Offload Engine.	Yes	Yes

Table continues on the next page...

Table 583. Transmit Checksum Offload Engine Functions for Different Packet Types (continued)

Packet Type	Hardware IP headerchecksum insertion	Hardware TCP/ UDPchecksum insertion
IPv6 packet is greater than 1,522 bytes (2,000 bytes when IEEE 802.3ad Support for 2K Packets is enabled in MAC) but less than or equal to the frame size constraint specified in Transmit Checksum Offload Engine.	Not Applicable	Yes
IPv4 with TCP, UDP, or ICMP	Yes	Yes
IPv4 packet has IP options (IP header is longer than 20 bytes)	Yes	Yes
Packet is an IPv4 fragment	Yes	No
IPv6 packet with the following next header fields in main or extension headers: • Hop-by-hop options (in IPv6 main header) • Hop-by-hop options (in IPv6 extension header) • Destinations options • Routing (with segment left 0) • Routing (with segment left > 0) • TCP, UDP, or ICMP • Authentication • Any other next header field in main or extension headers	• Not Applicable • Not Applicable • Not Applicable • Not Applicable • Not Applicable • Not Applicable • Not Applicable • Not Applicable	• Yes • No • Yes • No • No • Yes • No • No
IPv4 packet has TCP header with Options fields	Yes	Yes
IPv4 Tunnels: • IPv4 packet in an IPv4 tunnel • IPv6 packet in an IPv4 tunnel	• Yes (IPv4 tunnel header) • Yes (IPv4 tunnel header)	• No • No
IPv6 Tunnels: • IPv4 packet in an IPv6 tunnel • IPv6 packet in an IPv6 tunnel	• Not applicable • Not applicable	• No • No
IPv4 packet has 802.3ac tag (with C-VLAN Tag or S-VLAN Tag when enabled).	Yes	Yes
IPv6 packet has 802.3ac tag (with C-VLAN Tag or S-VLAN Tag when enabled).	Not applicable	Yes
IPv4 frames with security features (such as encapsulated security payload)	Yes	No
IPv6 frames with security features (such as encapsulated security payload)	Not applicable	No

75.10.2 Receive Checksum Offload Engine

IP provides the Checksum Offload Engine that is used to detect any error in an IPv4 or IPv6 packet in the receive path. The MAC verifies the checksum field of the received packet with the internally calculated checksum and provides the status.

The Receive Checksum Offload engine is used to detect errors in IP packets by calculating the header checksum and further matching it with the received header checksum. This engine also identifies a TCP, UDP, or ICMP payload in received IP packets and calculates the checksum of such payloads appropriately.

Here, both IPv4 and IPv6 packet in the received Ethernet packets are detected and processed for data integrity. The MAC receiver identifies IPv4 or IPv6 packets by checking for value 0x0800 or 0x86DD, respectively, in the Type field of the received Ethernet packet. This identification is applicable to single VLAN-tagged packets. It is also applicable to double VLAN-tagged packets when the Enable Double VLAN Processing option is selected and the EDVLP bit of the MAC_VLAN_Tag register is set.

The Rx COE calculates the IPv4 header checksums and checks that they match the received IPv4 header checksums. The result of this operation (pass or fail) is given to the RFC module for insertion into the receive status word. The IP Header Error bit is set for any mismatch between the indicated payload type (Ethernet Type field) and the IP header version, or when the received packet does not have enough bytes, as indicated by the Length field of the IPv4 header (or when fewer than 20 bytes are available in an IPv4 or IPv6 header).

This engine also identifies a TCP, UDP, or ICMP payload in the received IP datagrams (IPv4 or IPv6) and calculates the checksum of such payloads properly, as defined in the TCP, UDP, or ICMP specifications. This engine includes the TCP, UDP, or ICMPv6 pseudo-header bytes for checksum calculation and checks whether the received checksum field matches the calculated value. The result of this operation is given as a Payload Checksum Error bit in the receive status word. This status bit is also set if the length of the TCP, UDP, or ICMP payload does not match the expected payload length given in the IP header.

Following table describes the functions supported by the Rx COE based on the packet type. When the payload of an IP packet is not processed (indicated as "No" in the table), the information (whether the checksum engine is bypassed or not) is given in the receive status.

NOTE

The MAC does not append any payload checksum bytes to the received Ethernet packets.

Table 584. Receive Checksum Offload Engine Functions for Different Packet Types

Packet type	Hardware IP header checksum checking	Hardware TCP/UDP/ICMP checksum checking
Non-IPv4 or IPv6	No	No
IPv4 packet is greater than 1,522 bytes (2,000 bytes when IEEE 802.3ad Support for 2K Packets is enabled in the MAC)	Yes	Yes
IPv6 packet is greater than 1,522 bytes (2,000 bytes when IEEE 802.3ad Support for 2K Packets is enabled in the MAC)	Not Applicable	Yes
IPv4 with TCP, UDP, or ICMP	Yes	Yes
IPv4 header's protocol field contains a protocol other than TCP, UDP, or ICMP	Yes	No

Table continues on the next page...

Table 584. Receive Checksum Offload Engine Functions for Different Packet Types (continued)

Packet type	Hardware IP header checksum checking	Hardware TCP/UDP/ICMP checksum checking
IPv4 packet has IP options (IP header is longer than 20 bytes)	Yes	Yes
Packet is an IPv4 fragment	Yes	No
IPv6 packet with the following next header fields in main or extension headers: • Hop-by-hop options (in IPv6 main header) • Hop-by-hop options (in IPv6 extension header) • Destinations options • Routing (with segment left 0) • Routing (with segment left > 0) • TCP, UDP, or ICMP • Any other next header field in main or extension headers	• Not Applicable • Not Applicable • Not Applicable • Not Applicable • Not Applicable • Not Applicable • Not Applicable	• Yes • No • Yes • Yes • No • Yes • No
IPv4 packet has TCP header with Options fields	Yes	Yes
IPv4 Tunnels: • IPv4 packet in an IPv4 tunnel • IPv6 packet in an IPv4 tunnel	• Yes (IPv4 tunnel header) • Yes (IPv4 tunnel header)	• No • No
IPv4 Tunnels: • IPv4 packet in an IPv6 tunnel • IPv6 packet in an IPv6 tunnel	• Not Applicable • Not Applicable	• No • No
IPv4 packet has 802.3ac tag (with C-VLAN Tag or S-VLAN Tag when enabled)	Yes	Yes
IPv6 packet has 802.3ac tag (with C-VLAN Tag or S-VLAN Tag when enabled)	Not Applicable	Yes
IPv4 frames with security features (such as encapsulated security payload)	Yes	No

Table continues on the next page...

Table 584. Receive Checksum Offload Engine Functions for Different Packet Types (continued)

Packet type	Hardware IP header checksum checking	Hardware TCP/UDP/ICMP checksum checking
IPv6 frames with security features (such as encapsulated security payload)	Not Applicable	No

Set the IPC bit of the MAC_Configuration register for enabling Receive Checksum offload.

75.11 MAC Management Counters

This section is Synopsys Proprietary. Used with permission.

IP supports storing the statistics about the received and transmitted packets in registers that are accessible through the application/software. The MMC module maintains a set of registers for gathering statistics on the received and transmitted packets. The MMC counters are free running.

For MMC register details, see registers "MMC_Control" to "MMC_Rx_FPE_Fragment_Cntr" in EMAC register descriptions.

75.12 Flow Control

This section and its sub-sections are Synopsys Proprietary. Used with permission.

75.12.1 Transmit Flow Control

The Transmit Flow Control involves transmitting Pause packets in full-duplex mode and backpressure in half-duplex mode to control the flow of packets from the remote end.

75.12.1.1 Flow control in Full duplex mode

IP supports the IEEE802.3x Pause Packets.

Pause packet structure is shown in the following table.

Table 585. Pause Packet Fields

Field	Description
DA	Contains the special multicast address
SA	Contains the MAC address 0
Type	Contains 8808
MAC Control opcode	Contains 0001 for IEEE 802.3x Pause Control packets; 0101 for PFC packets
PT	Contains Pause time specified in the PT field of the MAC_Q#_Tx_Flow_Ctrl register

75.12.1.2 Pause Packet Control

When the FCB bit is set, the MAC generates and transmits a single Pause packet. If the FCB bit is set again after the Pause packet transmission is complete, the MAC sends another Pause packet irrespective of whether the pause time is complete or not. To extend the pause or terminate the pause prior to the time specified in the previously-transmitted Pause packet, the application should program the Pause Time register with appropriate value and then again set the FCB bit.

75.12.1.3 Flow Control in Half-Duplex Mode

In half-duplex mode, the MAC uses the deferral mechanism for the flow control (backpressure). When the application requests to stop receiving packets, the MAC sends a JAM pattern of 32 bytes when it senses a packet reception, provided the transmit flow control is enabled. This results in a collision and the remote station backs off. If the application requests a packet to be transmitted, it is scheduled and transmitted even when the backpressure is activated. If the backpressure is kept activated for a long time (and more than 16 consecutive collision events occur), the remote stations abort the transmission because of excessive collisions.

Following table describes the flow control in the Tx path for Queue 0 based on the setting of the following bits:

- EHFC bit of MTL_RxQ0_Operation_Mode register
- TFE bit of MAC_Q0_Tx_Flow_Ctrl register
- DM bit of MAC_Configuration register

Flow control is similar for all queues.

Table 586. Tx MAC Flow Control

EHFC	TFE	DM	Description
x	0	x	The MAC transmitter does not perform the flow control or backpressure operation.
0	1	0	The MAC transmitter performs back-pressure when Bit 0 of MAC_Q0_Tx_Flow_Ctrl register is set or the sideband signal sbd_flowctrl_i is 1.
1	1	0	The MAC transmitter performs back-pressure when Bit 0 of MAC_Q0_Tx_Flow_Ctrl register is set or the sideband signal sbd_flowctrl_i is 1. In addition, the MAC Tx performs back-pressure when Rx Queue level crosses the threshold set by Bits[10:8] of MTL_RxQ0_Operation_Mode register.
0	1	1	The MAC transmitter sends the Pause packet when Bit 0 of MAC_Q0_Tx_Flow_Ctrl register is set or the sideband signal sbd_flowctrl_i is 1.
1	1	1	The MAC transmitter sends the Pause packet when Bit 0 of MAC_Q0_Tx_Flow_Ctrl register is set or the sideband signal sbd_flowctrl_i is 1. In addition, the MAC Tx sends a Pause packet when Rx Queue level crosses the threshold set by Bits[10:8] of MTL_RxQ0_Operation_Mode register.

75.12.1.4 Enabling Transmit Flow Control

To independently enable Transmit Flow Control for each Tx queue, set the TFE bit in the MAC_Q#_Tx_Flow_Ctrl register.

75.12.2 Receive Flow Control

In the Receive path, the Flow Control is functional only in the full-duplex mode. If any Pause packet is received in the half-duplex mode, the packet is considered as a normal control packet.

NOTE

Receive pause packets should have a frame size of 64 bytes.

The Receive Flow Control is implemented by the MAC based on the bit value of the respective register, and the destination address and different fields of the received packet.

Following table describes the flow control in the Rx path based on the setting of the following bits:

- RFE bit of MAC_Rx_Flow_Ctrl register
- DM bit of MAC_Configuration register

Table 587. Rx MAC Flow Control

RFE	DM	Description
0	x	The MAC receiver does not detect the received Pause packets.
1	0	The MAC receiver does not detect the received Pause packets but recognizes such packets as Control packets.
1	1	The MAC receiver detects or processes the Pause packets and responds to such packets by stopping the MAC transmitter.

75.12.2.1 Enabling Receive Flow Control

To enable Pause Flow Control, set the RFE bit in the MAC_Rx_Flow_Ctrl register.

75.13 Loopback Mode

This section is Synopsys Proprietary. Used with permission.

The MAC supports Loopback of transmitted packets to its receiver.

The following are some guidelines for using the loopback mode:

- Enable loopback only with the full-duplex mode. In half-duplex mode, the carrier sense signal (crs) or collision (col) signal inputs get sampled which may result into issues such as packet dropping.
- If the loopback mode is enabled without connecting a PHY chip (for example, in FPGA setup), you should externally generate the Tx and Rx clocks and provide these clocks to the MAC.
- Do not loop back big packets. Big packets (>1522 bytes) may get corrupted in the loopback FIFO.

To enable this feature, program the LM bit of the MAC_Configuration register.

You can enable loopback for all PHY interfaces. The data is always looped back through internal asynchronous FIFO on to the internal Receive MII or GMII interface, irrespective of which PHY interface is selected. The loopback data is also passed through the corresponding transmit PHY interface block, on to the Ethernet line.

75.14 Automotive Safety Features

This section and all its sub-sections are Synopsys Proprietary. Used with permission.

IP supports the automotive safety feature.

75.14.1 Error Correction Code (ECC) Protection for Memories

The Error Correction Code (ECC) block can correct single-bit error and detect double-bit error.

At the write interface, ECC checkbits are generated by computing ECC on the contents of the data bus and respective address is appended with the data that is written to the memory.

At the read interface, ECC checkbits are recomputed on the content of the read data and the respective address is compared with the received checkbits in the memory.

Following figure shows the block diagram of ECC protection for memories

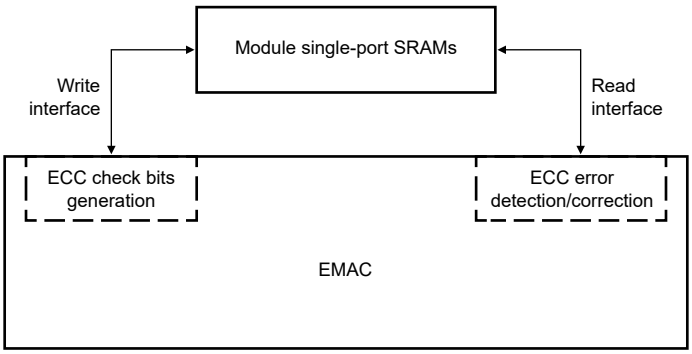


Figure 438. Block Diagram of ECC Protection for Memories

75.14.1.1 Handling the Address Mismatch

The ECC is calculated over the sum of memory data and memory location address. However only data is written in the memory along with the ECC(address is excluded).

On the read interface, the ECC is checked over the sum of memory DATA, memory ECC, and Internally generated address. When an ECC correctable error is detected over the range of the address bit position, it is treated as an uncorrectable error. This is because data might have been read from a different location.

75.14.1.2 Diagnostic Support for the Error Management

Following statistics are provided for monitoring the error behavior on each of the memory blocks.

- Error status provided to management
  - A separate status for correctable, uncorrectable, and address mismatch is specified in the MTL_ECC_Interrupt_Status register.
  - Memory locations at which correctable and uncorrectable errors are detected is specified in the MTL_ECC_Err_Addr_Status register. In addition, a control bit (MEEAO field of MTL_ECC_Control register) decides, whether the first errored memory address is reported or the latest errored memory address is reported.
  - MTL_ECC_Err_Cntr_Status register has separate counters to count the number of correctable and uncorrectable errors
- Interrupts provided to management
  - Separate interrupts are generated for correctable and uncorrectable errors.
  - sbd_sfty_ue_intr_o interrupt is generated when uncorrectable errors are detected and these errors cannot be masked.
  - sbd_sfty_ce_intr_o interrupt is generated when correctable errors are detected. The module sets the respective interrupt enable bits (in the DMA/MTL_ECC_Interrupt_enable register).
  - To find the root cause of the error, read the DMA/MTL_Safety_Interrupt_Status and DMA/MTL_ECC_Interrupt_Status registers.
  - To clear the interrupt, write 1 in the respective interrupt status bit in DMA/MTL_ECC_Interrupt_Status registers.

NOTE

- The status/counters/interrupts are generated per memory.
- MTL Tx and Rx memory is logically divided into multiple queues. But, ECC diagnostics (status/counter/interrupts) are common to all queues.

75.14.1.3 ECC Error Injection Capabilities

IP supports error injection capabilities for each memory as a static configuration. The position where the errors are injected in the data word is random. For each memory, 3 control bits are provided to inject errors. The 3 bits are:

- A single bit to enable error injection
- Two bits to indicate the type of error to be injected ❑ 00: 1 bit error
  - 01: 2 bit error
  - 10: 3 bit error
  - 11: 1 bit error in address field

The control bit descriptions for

- MTL Tx/Rx and DMA TSO memory are specified in the MTL_DBG_CTL register, for and for R.
- MTL EST memory are specified in the MTL_EST_GCL_Control register
- Rx parser memory are specified in the MTL_RXP_Indirect_Acc_Control_Status register

NOTE

While using debug mode for ECC error injection:

- There should be no traffic in the module
- When multiple CSR writes are required for writing single data word into the memory, the application should ensure that all the CSR writes corresponding to one memory write maintains the same value for the error injection control word.
- See Programming guidelines for ECC protection for memories

75.14.2 Finite State Machine(FSM) Parity and Timeout Protection

The FSM protection feature supports FSM parity and time out protection.

75.14.2.1 FSM State Parity Protection

Odd parity is implemented on all the FSM state register bits. Parity is monitored for every clock, after the reset is de-asserted. When a bit flips due to transient errors or permanent faults, the erroneous FSM is set to its default state and an uncorrectable error is indicated.

When the FSM state parity protection is enabled, by setting PRTYEN field of MAC_FSM_Control Register, the FSM state error is indicated by setting the FSMPES field of the MAC_DPP_FSM_Interrupt_Status register. Also, the Safety Interrupt (sbd_sfty_ue_intr_o) is asserted when an FSM Error is detected.

Error Injection mode is also supported for FSM parity error check. Program the Error Injection enable for the respective clock domain denoted by [23:16] bits of MAC_FSM_Control Register.

75.14.2.2 FSM Timeout Protection

The FSM Timeout feature provides a mechanism to ensure that all FSMs in the module can complete a transaction and return to a known completion state (IDLE or any other state that indicates completion of a transaction/transfer) within the programmed timeout value.

Program the TMR field of MAC_FSM_ACT_Timer register with a value that indicates the number of CSR cycles required to generate a 1 microsecond tic. This microsecond tic is internally used to generate the programmed timeout duration. Two timeout tics (normal mode timeout and large mode timeouts) are generated to provide flexibility to choose one per clock domain, by using the bits [31:24] of MAC_FSM_Control Register. Supported values for Timeout duration are 1us, 1.024ms, 16.384ms, 65.5536ms, 262.144ms, 1048.576ms (~1sec), 4194.304ms (~4s), 8.388s, 16.777s, and 33.554s which is based on the programmable bits NTMRMD, LTMRMD fields of MAC_FSM_ACT_Timer register. Interface timeouts are based on the normal mode tic generation; only the FSM timeouts depend on either normal or large tick selection.

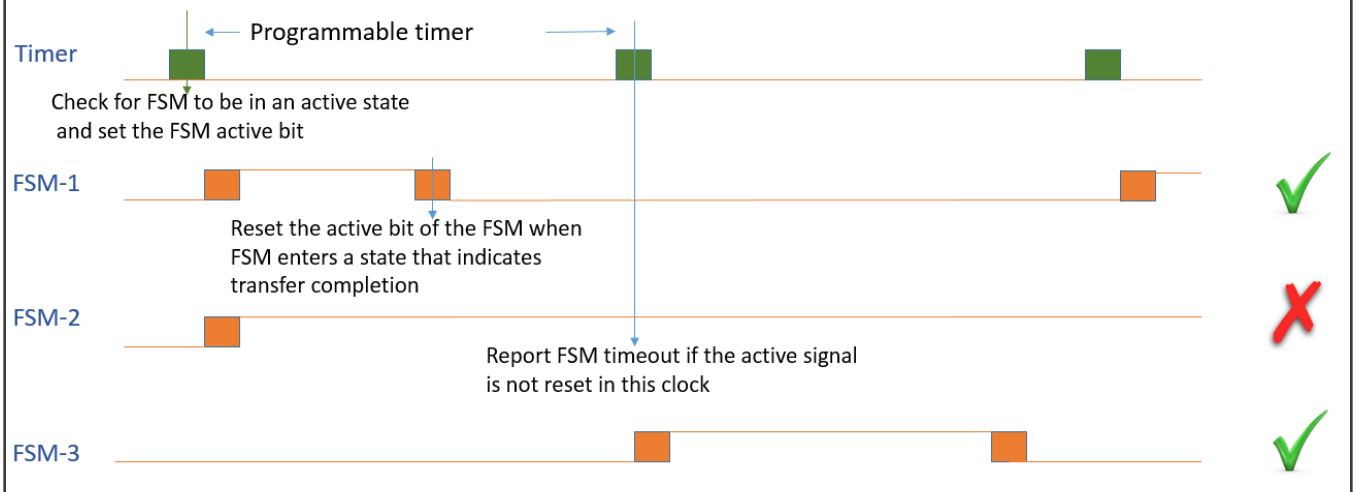


Figure 439. FSM Timeout Timing Diagram

In the above figure, the timer ticks are generated based on the programmable timeout value. The timer ticks are synchronized and made available in all the clock domains that have the FSMs.

At every timer tick, all the FSMs are monitored for being in active state (non-IDLE state, which indicates the FSM is actively processing transactions or hand-shakes) and an FSM active flag bit is set. The active flag bit is implemented for every FSM that is monitored for timeout. When the FSM reaches an IDLE or transaction completion state, the active flag bit of that FSM is reset. At the subsequent timer tick, all the FSMs' active flag bits are monitored and any flag that is set indicates a timeout for that FSM. This process of setting the flag and checking at subsequent timer tick is repeated at every tick.

Timeout Error Injection per clock domain is enabled by programming [15:8] fields of MAC_FSM_Control register. When Timeout error injection enable is set for a clock domain, the FSMs in that clock domain automatically timeout and generate an interrupt even without traffic. As error injection is a debug mode, TMR value need not indicate 1us, but can be programmed to a smaller value at which debug mode ticks are needed. FSM Timeout and Parity Error Injection modes can be used for testing at key-on/key-off.

The FSM timeout status is specified in the [15:8] field of MAC_DPP_FSM_Interrupt_Status register as per the respective clock domains. One FSM timeout status bit is made available per clock domain. The safety interrupt (sbd_sfty_ue_intr_o) is asserted when the timeout error is set for any of the clock domains. IP does not attempt to recover from a FSM timeout condition and relies on the application to take the corrective action (resetting IP).

75.14.2.3 Enabling FSM Parity and Timeout Protection

FSM timeout feature is enabled by setting the TMOUTEN field of MAC_FSM_Control register.

NOTE

See Programming guidelines for FSM parity and timeout

75.14.3 Application/CSR Interface Timeout Protection

This feature provides timeout protection to the application/CSR Interface. All the interfaces which has the handshake mechanism monitored for potential hangs due to the external agent (Master/Slave/Interconnect/Application client) not responding to the requests/transfers initiated by the QoS. After the request is initiated the response arrival interval is monitored. If the response does not arrive within a programmed time (TMR field of MAC_FSM_ACT_Timer) the timeout is triggered, using similar trigger generation as explained in the Section FSM protection.

The timeout is triggered when IP initiates SEQ, NON-SEQ and BUSY transfers on HTRANS[1:0]; and HREADY is not asserted by Slave within the programmed time.

The Timeout status for the AHB master interface is set in the MSTTES field of the MAC_DPP_FSM_Interrupt_Status register. The Safety interrupt (sbd_sfty_ue_intr_o) is asserted when application timeout status is set.

The hardware does not attempt to recover from Application/CSR Interface hangs and relies on software to take appropriate corrective action.

NOTE

- Protection is not needed for APB3 Slave interfaces because potential hangs can be only when IP does not respond to the requests. In such cases the IP internal protection logic (FSM timeout) detects such defects.
- Based on the time at which the transfer is initiated relative to the timer ticks, the timeout period could be any value between the programmed timeout and 2x times the programmed timeout.
- When an uncorrectable safety interrupt is issued and the read of the Interrupt status register returns all zeros, it implies that the CSR read is not functional. As soft reset of IP is not possible without the CSR access, and therefore, a hard reset of IP is recommended.
- See Programming guidelines for FSM parity and timeout .

75.15 Descriptors

This section and all its sub-sections are Synopsys Proprietary. Used with permission.

75.15.1 Overview

The DMA in the Ethernet subsystem transfers data based on a linked list of descriptors. The application creates the descriptors in the system memory. The module supports the following two types of descriptors:

- Normal Descriptor: Normal descriptors are used for packet data and to provide control information applicable to the packets to be transmitted.
- Context Descriptor: Context descriptors are used to provide control information applicable to the packet to be transmitted.

Each normal descriptor contains two buffers and two address pointers. These buffers enable the adapter port to be compatible with various types of memory management schemes.

NOTE

There is no limit for the number of descriptors that can be used for a single packet.

75.15.2 Descriptor Structure

The module supports the ring structure for DMA descriptor as shown in the following figure.

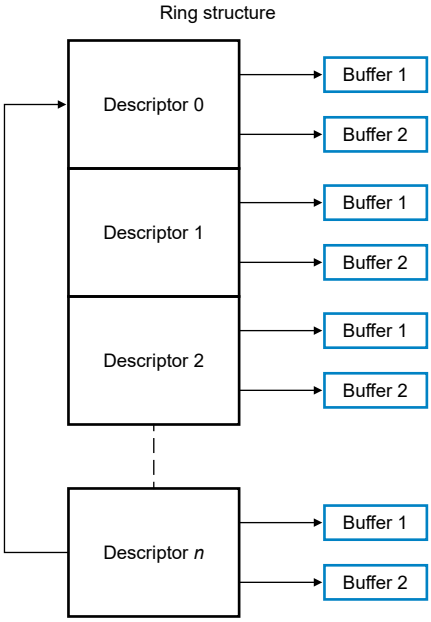


Figure 440. Descriptor Ring Structure

In Ring structure, descriptors are separated by the Word, DWord, or LWord number programmed in the DSL field of the DMA_CH#_Control register. The application needs to program the total ring length, that is, the total number of descriptors in ring span in the following registers of a DMA channel:

- Transmit Descriptor Ring Length Register (DMA_CH#_TxDesc_Ring_Length)
- Receive Descriptor Ring Length Register (DMA_CH#_RxDesc_Ring_Length)

The Descriptor Tail Pointer Register contains the pointer to the descriptor address (N). The base address and the current descriptor pointer decide the address of the current descriptor that the DMA can process. The descriptors up to one location less than the one indicated by the descriptor tail pointer (N – 1) are owned by the DMA. The DMA continues to process the descriptors until the following condition occurs:

Current Descriptor Pointer == Descriptor Tail Pointer;

The DMA goes into the Suspend mode when this condition occurs. The application must perform a write to the Descriptor Tail pointer register and update the tail pointer so that the following condition is true:

Current Descriptor Pointer < Descriptor Tail Pointer;

The DMA automatically wraps around the base address when the end of ring is reached, as shown in the following figure.

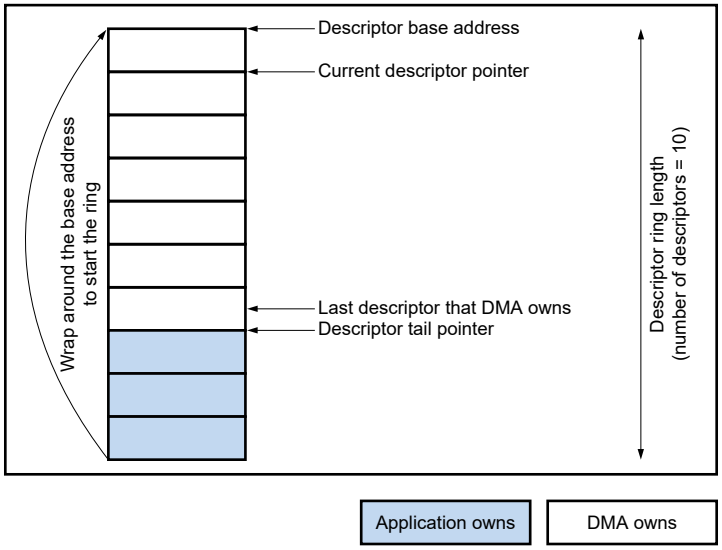


Figure 441. DMA Descriptor Ring

For descriptors owned by the application, the OWN bit of DES3 is reset to 0. For descriptors owned by the DMA, the OWN bit is set to 1. If the application has only one descriptor in the beginning, the application sets the last descriptor address (tail pointer) to Descriptor Base Address + 1. The DMA processes the first descriptor and then waits for the application to advance the tail pointer.

75.15.3 Transmit Descriptor

The DMA in the module requires at least one descriptor for a transmit packet. In addition to two buffers, two byte-count buffers, and two address pointers, the transmit descriptor has control fields which can be used to control the MAC operation on per-transmit packet basis. The Transmit Normal descriptor has two formats: Read format and Write-Back format

NOTE

TSO Split header is not supported.

Software must not write same tail pointer addresses when DMA is in Suspended state before setting new descriptor(s) for further processing.

75.15.3.1 Transmit Normal Descriptor (Read Format)

Following figure shows the Read Format for a Transmit normal descriptor. Table: 'TDES0 Normal Descriptor (Read Format)' through Table: 'TDES3 Normal Descriptor (Read Format)' describe the read format for the Transmit Normal Descriptors: TDES0, TDES1, TDES2, and TDES3.

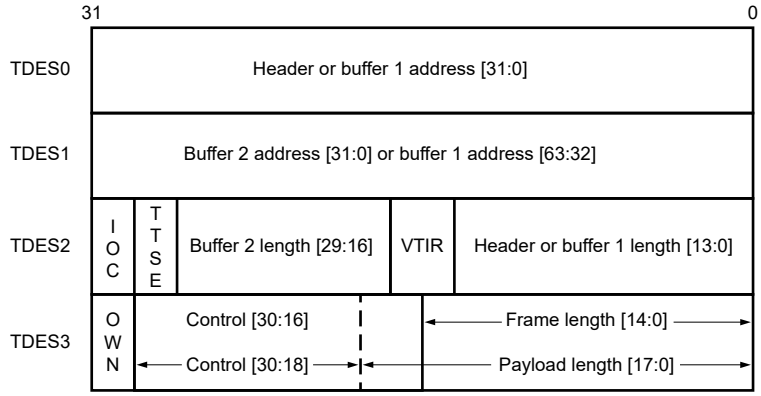


Figure 442. Transmit Descriptor Read Format

75.15.3.1.1 TDES0 Normal Descriptor (Read Format)

Table 588. TDES0 Normal Descriptor (Read Format)

Bit	Name	Description
31:0	BUF1AP	Buffer 1 Address Pointer or TSO Header Address Pointer These bits indicate the physical address of Buffer 1. These bits indicate the TSO Header Address pointer when the following bits are set: • TSE bit of TDES3 • FD bit of TDES3

75.15.3.1.2 TDES1 Normal Descriptor (Read Format)

Table 589. TDES1 Normal Descriptor (Read Format)

Bit	Name	Description
31:0	BUF2AP	Buffer 2 or Buffer 1 Address Pointer This bit indicates the physical address of Buffer 2 when a descriptor ring structure is used. There is no limitation for the buffer address alignment. In 40- or 48-bit addressing mode, these bits indicate the most-significant 8- or 16- bits of the Buffer 1 Address Pointer.

75.15.3.1.3 TDES2 Normal Descriptor (Read Format)

Table 590. TDES2 Normal Descriptor (Read Format)

31	30	29-16	15:14	13:0
IOC	TTSE/ TMWD	B2L	VTIR	HL or B1L

Table 591. TDES2 Normal Descriptor (Read Format)

Bits	Name	Description
31	IOC	Interrupt on Completion This bit controls the setting of TI and ETI status bits in the DMA_CH#_Status register. When ETIC = 1 and TDES2[LD] = 0, this bit sets the ETI bit. When TDES3[LD] = 1, this bit sets the TI status bit.
30	TTSE/ TMWD	Transmit Timestamp Enable or External TSO Memory Write Enable This bit enables the IEEE1588 time stamping for Transmit packet referenced by the descriptor, if TSE bit is not set. If TSE bit is set and external TSO memory is enabled, setting this bit disables external TSO memory writing for this packet.
29:16	B2L	Buffer 2 Length The driver sets this field. When set, this field indicates Buffer 2 length.
15:14	VTIR	VLAN Tag Insertion or Replacement These bits request the MAC to perform VLAN tagging or untagging before transmitting the packets. The application must set the CRC Pad Control bits appropriately when VLAN Tag Insertion, Replacement, or Deletion is enabled for the packet. The following list describes the values of these bits: • 2'b00: Do not add a VLAN tag. • 2'b01: Remove the VLAN tag from the packets before transmission. This option should be used only with the VLAN packets. • 2'b10: Insert a VLAN tag with the tag value programmed in the MAC_VLAN_Incl register or context descriptor. • 2'b11: Replace the VLAN tag in packets with the tag value programmed in the MAC_VLAN_Incl register or context descriptor. This option should be used only with the VLAN packets. These bits are valid when the Enable SA and VLAN Insertion on Tx option is selected while configuring the core.
13:0	HL or B1L	Header Length or Buffer 1 Length For Header length only bits [9:0] are taken. The size 13:0 is applicable only when interpreting buffer 1 length. If the TCP Segmentation Offload feature is enabled through the TSE bit of TDES3, this field is equal to the header length. When the TSE bit is set in TDES3, the header length includes the length in bytes from Ethernet Source address till the end of the TCP header. The maximum header length supported for TSO feature is 1023 bytes. The maximum header length supported for TSO feature is 1023 bytes. If the TCP Segmentation Offload feature is not enabled, this field is equal to Buffer 1 length.

75.15.3.1.4 TDES3 Normal Descriptor (Read Format)

Table 592. TDES2 Normal Descriptor (Read Format)

31	30	29	28	27:26	25:23	22:19	18	17:16	15	14:0
OWN	CTXT	FD	LD	CPC	SAIC	SLOTNUM or THL	TSE	CIC/TPL	TPL	FL/TPL

Table 593. TDES3 Normal Descriptor (Read Format)

Bits	Name	Description
31	OWN	Own Bit When this bit is set, it indicates that the DMA owns the descriptor. When this bit is reset, it indicates that the application owns the descriptor. The DMA clears this bit after it completes the transfer of data given in the associated buffer(s).
30	CTXT	Context Type This bit should be set to 1'b0 for normal descriptor.
29	FD	First Descriptor When this bit is set, it indicates that the buffer contains the first segment of a packet.
28	LD	Last Descriptor When this bit is set, it indicates that the buffer contains the last segment of the packet. When this bit is set, the B1L or B2L field should have a non-zero value.
27:26	CPC	CRC Pad Control This field controls the CRC and Pad Insertion for Tx packet. This field is valid only when the first descriptor bit (TDES3[29]) is set. The following list describes the values of Bits[27:26]: • 2'b00: CRC and Pad Insertion The MAC appends the cyclic redundancy check (CRC) at the end of the transmitted packet of length greater than or equal to 60 bytes. The MAC automatically appends padding and CRC to a packet with length less than 60 bytes. • 2'b01: CRC Insertion (Disable Pad Insertion) The MAC appends the CRC at the end of the transmitted packet but it does not append padding. The application should ensure that the padding bytes are present in the packet being transferred from the Transmit Buffer, that is, the packet being transferred from the Transmit Buffer is of length greater than or equal to 60 bytes. • 2'b10: Disable CRC Insertion The MAC does not append the CRC at the end of the transmitted packet. The application should ensure that the padding and CRC bytes are present in the packet being transferred from the Transmit Buffer. • 2'b11: CRC Replacement The MAC replaces the last four bytes of the transmitted packet with recalculated CRC bytes. The application should ensure that the padding and CRC bytes are present in the packet being transferred from the Transmit Buffer. This field is valid only for the first descriptor. Note: When the TSE bit is set, the MAC ignores this field because the CRC and pad insertion is always done for segmentation.
25:23	SAIC	SA Insertion Control These bits request the MAC to add or replace the Source Address field in the Ethernet packet with the value given in the MAC Address 0 register. The application must set the CRC Pad Control bits appropriately when SA Insertion Control is enabled for the packet.

Table continues on the next page...

Table 593. TDES3 Normal Descriptor (Read Format) (continued)

Bits	Name	Description
		Bit 25 specifies the MAC Address Register (1 or 0) value that is used for Source Address insertion or replacement. The following list describes the values of Bits[24:23]: • 2'b00: Do not include the source address • 2'b01: Include or insert the source address. For reliable transmission, the application must provide frames without source addresses. • 2'b10: Replace the source address. For reliable transmission, the application must provide frames with source addresses. • 2'b11: Reserved These bits are valid in the EQOS-DMA, EQOS-AXI, and EQOS-AHB configurations when the Enable SA and VLAN Insertion on Tx option is selected while configuring the core and when the First Segment control bit (TDES3 [29]) is set. This field is valid only for the first descriptor.
22:19	SLOTNUM or THL	SLOTNUM: Slot Number Control Bits in AV Mode These bits indicate the slot interval in which the data should be fetched from the corresponding buffers addressed by TDES0 or TDES1. When the Transmit descriptor is fetched, the DMA compares the slot number value in this field with the slot interval maintained in the RSN field DMA_CH#_Slot_Function_Control_Status. It fetches the data from the buffers only if a value matches. These bits are valid only for the AV channels. THL: TCP/UDP Header Length If the TSE bit is set, this field contains the length of the TCP/UDP header. The minimum value of this field must be 5 for TCP header. The value must be equal to 2 for UDP header. This field is valid only for the first descriptor.
18	TSE	TSO Split header is not supported. The value of this bit is always zero.
17:16	CIC/TPL	Checksum Insertion Control or TCP Payload Length These bits control the checksum calculation and insertion. The following list describes the bit encoding: • 2'b00: Checksum Insertion Disabled. • 2'b01: Only IP header checksum calculation and insertion are enabled. • 2'b10: IP header checksum and payload checksum calculation and insertion are enabled, but pseudo-header checksum is not calculated in hardware. • 2'b11: IP Header checksum and payload checksum calculation and insertion are enabled, and pseudo-header checksum is calculated in hardware. This field is valid when the Enable Transmit TCP/IP Checksum Offload option is selected and the TSE bit is reset. When the TSE bit is set, this field contains the upper bits [17:16] of the TCP Payload (or IP Payload for UDP fragmentation). This allows the TCP/UDP packet length field to be spanned across TDES3[17:0] to provide 256 KB packet length support.

Table continues on the next page...

Table 593. TDES3 Normal Descriptor (Read Format) (continued)

Bits	Name	Description
		This field is valid only for the first descriptor.
15	TPL	Reserved or TCP Payload Length When the TSE bit is reset, this bit is reserved. When the TSE bit is set, this is Bit 15 of the TCP payload length [17:0]. This field is valid only when the Enable TCP Segmentation Offloading for TCP/IP Packets option is selected while configuring the core.
14:0	FL/TPL	Frame Length or TCP Payload Length This field is equal to the length of the packet to be transmitted in bytes. When the TSE bit is not set, this field is equal to the total length of the packet to be transmitted: Ethernet Header Length + TCP /IP Header Length – Preamble Length – SFD Length + Ethernet Payload Length When the TSE bit is set, this field is equal to the lower 15 bits of the TCP payload length in case of segmentation and IP payload in case of UDP fragmentation. In case of segmentation, this length does not include Ethernet header or TCP/UDP/IP header length. In case of fragmentation, this length does not include Ethernet header and IP header. When DWRR/WFQ algorithm is NOT enabled, value written into this field is not used when TSE = 0.

75.15.3.1.5 Transmit normal descriptor (write-back format)

The transmit descriptor write-back format includes timestamp low, timestamp high, OWN, and Status bits.

The write-back format is applicable only for the last descriptor of the corresponding packet. Write 1 to the LD bit (TDES3[28]) in the descriptor where DMA writes back the status and timestamp information for the corresponding transmit packet.

Figure 443 shows the transmit descriptor write-back format.

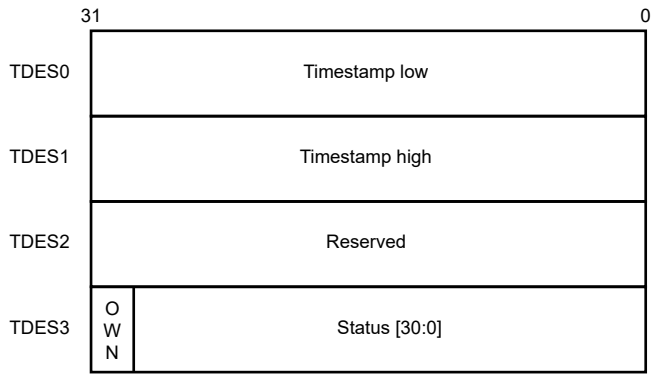


Figure 443. Transmit descriptor write-back format

TDES0 normal descriptor (write-back format)

Table 594 shows that this format is only applicable to the last descriptor of a packet.

Table 594. TDES0 normal descriptor (write-back format)TDES0 normal descriptor (write-back format)

Bit	Name	Description
31:0	TTSL	Transmit Packet Timestamp Low DMA updates this field with least significant 32 bits of the timestamp captured for the corresponding transmit packet. DMA writes the timestamp only if TTSE bit of TDES2 is 1 in the first descriptor of the packet. This field has the timestamp only if you write 1 to the Last Segment bit (LS) in the descriptor and the Timestamp status (TTSS) bit.

TDES1 normal descriptor (write-back format)

Table 595 shows that this format is only applicable to the last descriptor of a packet.

Table 595. TDES1 normal descriptor (write-back format)

Bit	Name	Description
31:0	TTSH	Transmit Packet Timestamp High DMA updates this field with the most significant 32 bits of the timestamp captured for a corresponding transmit packet. DMA writes the timestamp only if the TTSE bit of TDES2 is 1 in the first descriptor of the packet. This field has the timestamp only if you write 1 to the Last Segment bit (LS) in the descriptor and Timestamp status (TTSS) bit.

TDES2 Normal Descriptor (Write-Back Format)

This format is applicable only to the last descriptor of a packet.

Table 596. TDES2 normal descriptor (write-back format)

Bit	Description
31:0	Reserved

TDES3 Normal Descriptor (Write-Back Format)

This format is applicable only to the last descriptor of a packet.

Table 597. TDES3 normal descriptor (write-back format)

31	30	29	28	27:18	17	16	15	14	13	12	11	10	9	8	7:4	3	2	1	0
OWN	CTXT	FD	LD	Rsvd	TTSS	EUE	ES	JT	FF	PCE	LoC	NC	LC	EC	CC	ED	UF	DB	IHE

Table 598. TDES3 normal descriptor (write-back format)

Bit	Name	Description
31	OWN	Own Bit When this field is 1, it indicates that the module DMA owns the descriptor. DMA write 0 to this field when it completes the packet transmission. After the write-back is complete, this field is set to 'b0.
30	CTXT	Context Type

Table continues on the next page...

Table 598. TDES3 normal descriptor (write-back format) (continued)

Bit	Name	Description
		This field must be set to 'b0 for normal descriptor.
29	FD	First Descriptor This field indicates that the buffer contains the first segment of a packet.
28	LD	Last Descriptor This field is set to 'b1 for last descriptor of a packet. DMA writes the status fields only in the last descriptor of the packet.
27:24	Rsvd	Reserved
23	DE	Descriptor Error When this field is 1, it indicates that the descriptor content is incorrect. DMA writes 1 to this field during write-back when the descriptor is closed. Descriptor errors can be: • Incorrect sequence from the context descriptor. For example, a location after the first descriptor for a packet. • All 1s • CTXT is 1 along with LD or FD bits as 1. **NOTE** When a descriptor error occurs due to all ones or CTXT, LD, and FD bits are 1, the transmit DMA closes the transmit descriptor with DE and LD bits set to 1. When you write 1 to IOC bit in TDES2 of corresponding first descriptor, the transmit DMA write 1 to TI bit in the DMA_CH#_Status register **NOTE** Based on CTXT, LD, and FD bits of the transmit descriptor, the subsequent descriptor might consider as the first descriptor (even if FD bit is not 1) and partial packet is sent.
22:18	Rsvd	Reserved
17	TTSS	Tx Timestamp Status This field indicates that a timestamp has been captured for the corresponding transmit packet. When this field is 1, it indicates that TDES0 and TDES1 have timestamp values that were captured for the transmit packet. This field is valid only when the Last Segment control bit (TDES3 [28]) in a descriptor is 1. This field is valid only when IEEE1588 timestamping feature is enabled; otherwise, it is reserved.
16	EUE	ECC Uncorrectable Error Status Indicates the ECC uncorrectable error in the TSO memory. **NOTE** An uncorrectable error in the transmit FIFO memory is reported with (Bit 13) FF = 1. This is because the module flushes all such packets.

Table continues on the next page...

Table 598. TDES3 normal descriptor (write-back format) (continued)

Bit	Name	Description
15	ES	Error Summary This field indicates the logical OR of the following bits: • TDES3[0]: IP header error • TDES3[14]: Jabber timeout • TDES3[13]: Packet flush • TDES3[12]: Payload checksum error • TDES3[11]: Loss of carrier • TDES3[10]: No carrier • TDES3[9]: Late collision • TDES3[8]: Excessive collision • TDES3[3]: Excessive deferral • TDES3[2]: Underflow error This field is 1 when EUE (bit 16) = 1.
14	JT	Jabber Timeout This field indicates that the MAC transmitter experiences a jabber time-out. This field is 1 only when MAC_Configuration[JD] is not 1.
13	FF	Packet Flushed This field indicates that DMA or MTL flushes the packet because the CPU gives a software flush command.
12	PCE	Payload Checksum Error This field indicates that the checksum offload engine had a failure and does not insert any checksum into the encapsulated TCP, UDP, or ICMP payload. This failure can either because of insufficient bytes, as the payload length field of the IP header indicates or the MTL starting to forward the packet to the MAC transmitter in Store-and-forward mode without calculating the checksum. This second error condition only occurs when the transmit FIFO depth is less than the length of the Ethernet packet that is transmitted to avoid deadlock. The MTL starts forwarding the packet when the FIFO is full, even in the Store-and-forward mode. This error can also occur when you detects a bus error during packet transfer. When the full checksum offload engine is not enabled, this bit is reserved.
11	LoC	Loss of Carrier This field indicates that a loss of carrier occurred during packet transmission (that is, the gmii_crs_i signal was inactive for one or more transmit clock periods during packet transmission). This field is valid only for the packets transmitted without collision and when MAC operates in the Half-duplex mode.
10	NC	No Carrier This field indicates that the carrier sense signal from the PHY was not asserted during transmission.

Table continues on the next page...

Table 598. TDES3 normal descriptor (write-back format) (continued)

Bit	Name	Description
9	LC	Late Collision This bit indicates that packet transmission was aborted because a collision occurred after the collision window (64 byte times including Preamble in MII mode and 512 byte times including Preamble and Carrier Extension in GMII mode). This bit is not valid if Underflow Error is set.
8	EC	Excessive Collision This field indicates that the transmission was aborted after 16 successive collisions when you transmits the current packet. If MAC_Configuration[DR] = 1, this field is 1 after first collision and abort the packet transmission.
7:4	CC	Collision Count This 4-bit counter value indicates the number of collisions occurred before transmitting the packet. The count is not valid when the EC bit = 1.
3	ED	Excessive Deferral This field indicates that the transmission ends because of an excessive deferral of over 24,288 bit times (155,680 bits times in 1000 Mbit/s mode or Jumbo packet enabled mode) if MAC_Configuration[DC] = 1. When TBS is enabled in Full duplex mode and this field is 1, it indicates that the frame drops after the expiry time is reached.
2	UF	Underflow Error This field indicates that MAC aborts the packet because the data arrived late from the system memory. The underflow error can occur because of either of the following conditions: • DMA encounters an empty transmit buffer when it transmits the packet • The application filled the MTL Tx FIFO slower than the MAC transmit rate The transmission process enters the Suspended state and sets the underflow bit corresponding to a queue in MTL_Interrupt_Status.
1	DB	Deferred Bit This field indicates that MAC defers before transmitting because of presence of carrier. This field is valid only in the Half-Duplex mode.
0	IHE	IP Header Error When IP Header Error is 1, this field indicates that the checksum offload engine detects an IP header error. This field is valid only when transit checksum offload is enabled. Otherwise, it is reserved. If COE detects an IP header error, it inserts an IPv4 header checksum if the Ethernet type field indicates an IPv4 payload. In Full-Duplex mode, when EST/Qbv is enabled and this field is 1, it indicates the frame drop status due to the frame size error or schedule error.

75.15.3.2 Transmit context descriptor

The transmit context descriptor can provide any time before a packet descriptor. The context is valid for the current packet and subsequent packets. The context descriptor provide the timestamps for one-step timestamp correction and VLAN tag ID for VLAN insertion feature. Write back is done on a context descriptor only to write 0 to OWN bit.

NOTE

DMA internally store the VLAN tag IDs and MSS values which the application provides in a context descriptor with their corresponding valid bits set. DMA always passes the last valid VLAN tag to the MTL, when the outer or inner VLAN tag is provided with the valid bit set. The application cannot invalidate the valid VLAN tag which DMA stores. The VLAN tag is inserted or replaced based on the control inputs provided for the packet.

The inner VLAN tag control input is used only for the next packet that immediately follows the context descriptor. The application must provide a context descriptor before the normal descriptor of each packet for which DMA must use the inner VLAN tag control input.

Figure 444 shows the transmit context descriptor format.

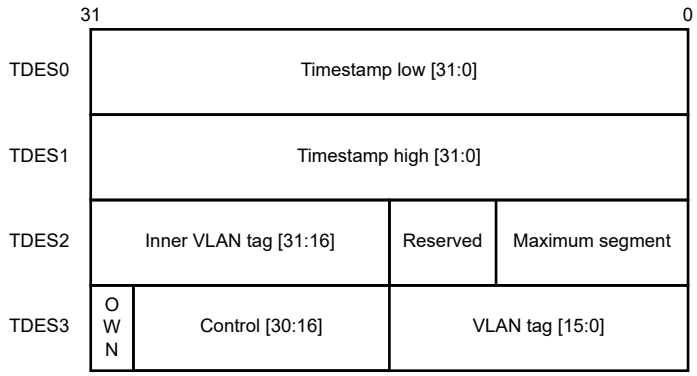


Figure 444. Transmit context descriptor format

75.15.3.2.1 TDES0 context descriptor

Table 599 describes the TDES0 context descriptor format.

Table 599. TDES0 context descriptor

Bit	Name	Description
31:0	TTSL	Transmit Packet Timestamp Low For one-step correction, the driver can provide the lower 32 bits of timestamp in this descriptor word. DMA uses this value as the low word for doing one-step timestamp correction. This field is valid only if the OSTC and TCMSSV bits of TDES3 context descriptor are 1.

75.15.3.2.2 TDES1 context descriptor

Table 600 describes the TDES1 context descriptor format.

Table 600. TDES1 context descriptor

Bit	Name	Description
31:0	TTSH	Transmit Packet Timestamp High For one-step correction, the driver can provide the upper 32 bits of timestamp in this descriptor. DMA uses this value as the high word for doing one-step timestamp correction. This field is valid only if the OSTC and TCMSSV bits of TDES3 context descriptor are 1.

75.15.3.2.3 TDES2 context descriptor

Table 601 shows the TDES2 context descriptor format.

Table 601. TDES2 context descriptor

Bit	Name	Description
31:16	IVT	Inner VLAN Tag When the IVLTV bit of TDES3 context descriptor is 1 and the TCMSSV and OSTC bits of TDES3 context descriptor becomes 0, it indicates that the TDES2[31:16] contains the inner VLAN tag to insert in the subsequent transmit packets.
15:14	Rsvd	Reserved
13:0	MSS	TSO split header is not supported. The value of this bit is always zero.

75.15.3.2.4 TDES3 context descriptor

Table 602. TDES3 context descriptor

31	30	29:28	27	26	25:24	23	22:18	17	16	15:0
OWN	CTXT	Rsvd	OSTC	TCMSSV	Rsvd	CDE	Rsvd	IVLTV	VLTV	VT

Table 603 shows the TDES3 context descriptor format.

Table 603. TDES3 context descriptor

Bit	Name	Description
31	OWN	Own Bit When this field is 1, it indicates that the module DMA owns the descriptor. When this field becomes 0, it indicates that the application owns the descriptor. DMA writes 0 to this field immediately after the read.
30	CTXT	Context Type This field must set to 1'b1 for context descriptor.
29:28	Rsvd	Reserved
27	OSTC	One-Step Timestamp Correction Enable When this field is 1, it indicates that DMA performs a one-step timestamp correction with reference to the timestamp values provided in TDES0 and TDES1.
26	TCMSSV	One-Step Timestamp Correction Input or MSS Valid When this field and the OSTC field are 1, it indicates that the timestamp correction input provided in TDES0 and TDES1 is valid. When the OSTC field becomes 0 and this field and the TSE bit of TDES3 are 1 in subsequent normal descriptor, it indicates that the MSS input in TDES2 is valid.
25:24	Rsvd	Reserved

Table continues on the next page...

Table 603. TDES3 context descriptor (continued)

Bit	Name	Description
23	DE	Descriptor Error When this field is 1, it indicates that the descriptor content is incorrect. DMA writes 1 to this field during write-back when it closes the context descriptor. Descriptor errors can be: • Incorrect sequence from the context descriptor. For example, a location before the first descriptor for a packet. • All ones. • CD, LD, and FD bits = 1. **NOTE** When a descriptor error occurs due to all ones or CTXT, LD, and FD bits are 1, the transmit DMA closes the transmit descriptor with DE and LD bits are 1. When you write 1 to IOC bit in TDES2 of corresponding first descriptor, the transmit DMA writes 1 to TI bit in the DMA_CH#_Status register **NOTE** On the basis of CTXT, LD, and FD bits of the transmit descriptor, the subsequent descriptor might considered as the first descriptor (even if FD bit is not 1) and partial packet is sent. **NOTE** This error is categorized as an abnormal event. Recovery from this event is possible by issuing a software reset only. DMA stopping, reconfiguring, and restarting recovery mechanism is not supported.
22:20	Rsvd	Reserved
19:18	IVTIR	Inner VLAN Tag Insert or Replace When this field is 1, it requests MAC to perform inner VLAN tagging or un-tagging before transmitting the packets. If the packet is modified for VLAN tags, MAC automatically recalculates and replaces the CRC bytes. The following list describes the values of these bits: • 2'b00: Do not add the inner VLAN tag. • 2'b01: Remove the inner VLAN tag from the packets before transmission. You must use this option only with the VLAN frames. • 2'b10: Insert an inner VLAN tag with the tag value programmed in MAC_Inner_VLAN_Incl or context descriptor. • 2'b11: Replace the inner VLAN tag in packets with the tag value programmed in MAC_Inner_VLAN_Incl or context descriptor. You must use this option only with the VLAN frames.. These fields are valid when you select the enable SA and VLAN insertion on transit and enable double VLAN processing options.
17	IVLTV	Inner VLAN Tag Valid When this field is 1, it indicates that IVT field of TDES2 is valid.

Table continues on the next page...

Table 603. TDES3 context descriptor (continued)

Bit	Name	Description
16	VLTV	VLAN Tag Valid When this field is 1, it indicates that VT field of TDES3 is valid.
15:0	VT	VLAN Tag Contains the VLAN tag to insert or replace in the packet. This field is used as VLAN tag only when MAC_VLAN_Incl[VLTI] becomes 0.

75.15.4 Receive descriptor

DMA in the module reads a descriptor only if the tail pointer is different from the base pointer or current pointer. It is recommended to have a descriptor ring with a length that accommodates at least two complete packets which MAC receives. Otherwise, the performance of DMA is impacted because of the unavailability of the descriptors. In such situations, the RxFIFO in MTL becomes full and starts dropping packets.

The following receive descriptors are present:

- Normal descriptors
- Context descriptors

You can prepare all the receive descriptors and give to DMA as normal descriptors with the content as shown in receive normal descriptor (read format). DMA reads this descriptor and the receive DMA closes the descriptor with the corresponding packet status after transferring a received packet (or part of) to the buffers indicated by the descriptor. The receive normal descriptor (write-back format) describes the format of this status.

For some packets, only the normal descriptor bits cannot write the complete status. For such packets, the receive DMA writes the extended status to the next descriptor (without processing or using the buffers or pointers embedded in that descriptor). Receive context descriptor describes the descriptor write back format and content.

75.15.4.1 Receive normal descriptor (read format)

The read format for a receive normal descriptor is made up of the following:

- A header or buffer 1 address
- Reserved field
- Payload or buffer 2 or next descriptor address
- A 30-bit reserved field
- OWN bit
- An interrupt field

Figure 445 shows the read format for a receive normal descriptor.

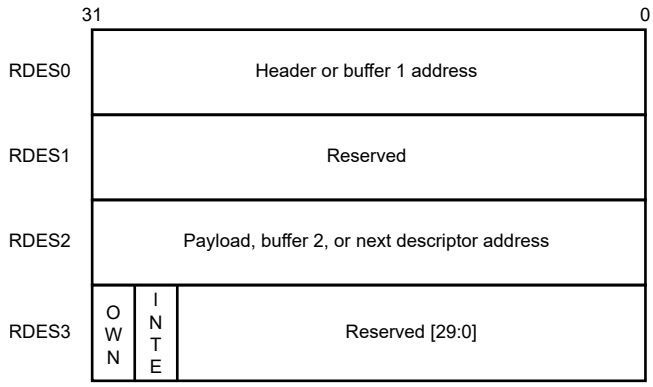


Figure 445. Receive normal descriptor read format

NOTE

In the receive descriptor (read format), if the buffer address field is all zeros, the module does not transfer data to that buffer and skips to the next buffer or next descriptor.

75.15.4.1.1 RDES0 normal descriptor (read format)

Table 604 shows the RDES0 normal descriptor read format.

Table 604. RDES0 normal descriptor (read format)

Bit	Name	Description
31:0	BUF1AP	Header or Buffer 1 Address Pointer When the SPH field of control register of a channel becomes 0, these field indicates the physical address of buffer 1. When the SPH field is 1, these field indicates the physical address of header buffer where the receive DMA writes the L2/L3/L4 header bytes of the received packet. The application can program a byte-aligned address for this buffer which means that the LS bits of this field can be non-zero. However, DMA performs a write operation with RDES0[1:0] (or RDES0[2:0]/[3:0] in case of 64-/128-bit configuration) as zero, when the start of packet is transferred. However, the packet data shifts per the actual offset which the buffer address pointer provides. If the address pointer points to a buffer where the middle or last part of the packet is stored, DMA ignores the offset address and writes to the full location which the data-width indicates .

75.15.4.1.2 RDES1 normal descriptor (read format)

Table 605 shows the RDES1 normal descriptor read format.

Table 605. RDES1 normal descriptor (read format)

Bit	Name	Description
31:0	Reserved or BUF1AP	Contains the most-significant 32 bits of the buffer 1 address pointer in 64-bit Addressing mode. Otherwise, this field is reserved.

75.15.4.1.3 RDES2 normal descriptor (read format)

Table 606 shows the RDES2 normal descriptor read format.

Table 606. RDES2 normal descriptor (read format)

Bit	Name	Description
31:0	BUF2AP	Buffer 2 Address Pointer Indicates the physical address of buffer 2. When the SPH bit of the DMA_CH#_Control register = 1, it indicates that the buffer address pointer must be bus width-aligned, that is, RDES2[3:0, 2:0, or 1:0] = 0 corresponding to 128, 64, or 32 bus width. LSBs are ignored internally. When the SPH bit of the DMA_CH#_Control register becomes 0, it indicates that there is no limitations on the RDES2 value. However, the RxDMA uses the LS fields of the pointer address only when it transfers the start bytes of a packet. If the BUF2AP provides the address of a buffer in which the middle or last part of a packet is stored, DMA ignores BUF2AP[3:0 or 2:0 or 1:0] (corresponding to 128- or 64- or 32-bit data-bus) and writes to the complete location.

75.15.4.1.4 RDES3 normal descriptor (read format)

Following table describes the RDES3 normal descriptor read format.

31	30	19-26	25	24	23:0
OWN	IOC	Rsvd	BUF2V	BUF1V	Rsvd

Table 607. RDES3 normal descriptor (read format)

Bit	Name	Description
31	OWN	Own Bit When this field is 1, it indicates that the module DMA owns the descriptor. When this field becomes 0, it indicates that the application owns the descriptor. DMA write 0 this field when either of the following conditions is true: • DMA completes the packet reception. • The buffers associated with the descriptor are full.
30	IOC	Interrupt Enabled on Completion When this field is 1, it indicates that an interrupt is issued to the application when DMA closes this descriptor.
29:26	Rsvd	Reserved
25	BUF2V	Buffer 2 Address Valid When this field is 1, it indicates to the DMA that the buffer 2 address specified in RDES2 is valid. The application must write 1 to this field so that DMA can use the address, to which the buffer 2 address in RDES2 points to write received packet data.
24	BUF1V	Buffer 1 Address Valid When this field is 1, it indicates to the DMA that the buffer 1 address specified in RDES1 is valid.

Table continues on the next page...

Table 607. RDES3 normal descriptor (read format) (continued)

Bit	Name	Description
		The application must write 1 to this field, if DMA uses the address to which the buffer 1 address in RDES1 points to write the received packet data.
23:0	Rsvd	Reserved

75.15.4.1.5 Receive normal descriptor (write-back format)

Following figure shows the write-back format for a receive normal descriptor.

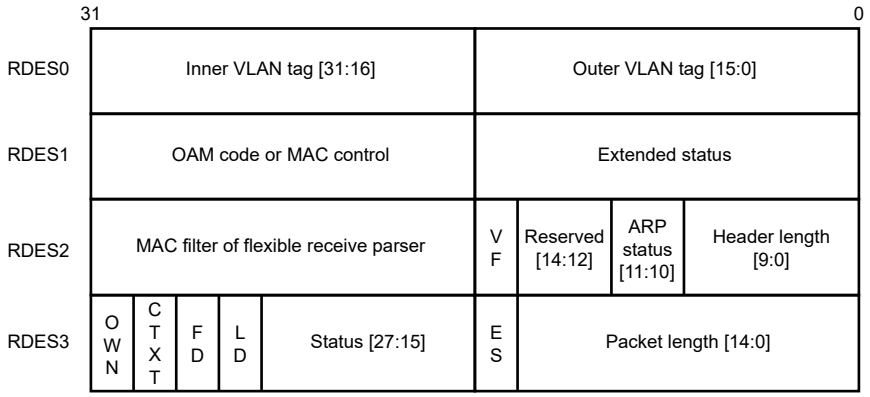


Figure 446. Receive normal descriptor (write-back format)

NOTE

When you enables the flexible receive parser, RDES2[31:16] indicates the parser status, and not the MAC filter status. The MAC filter status is not available when flexible receive parser is enabled.

75.15.4.1.5.1 RDES0 normal descriptor (write-back format)

Table 608 shows the write-back format for the RDES0 normal descriptor.

Table 608. RDES0 normal descriptor (write-back format)

Bit	Name	Description
31:16	IVT	Inner VLAN Tag Contains the inner VLAN tag of the received packet if the RS0V bit of RDES3 = 1. This field is valid only when you enables the double VLAN tag processing and VLAN tag stripping.
15:0	OVT	Outer VLAN Tag Contains the outer VLAN tag of the received packet if the RS0V bit of RDES3 = 1.

75.15.4.1.5.2 RDES1 normal descriptor (write-back format)

The status fields in the write-back format are valid only for the last descriptor (RDES3[28] = 1). Following table provide the details of the write-back format for RDES1 normal descriptor.

31:16	15	14	13	12	11:8	7	6	5	4	3	2:0
OPC	TD	TSA	PV	PFT	PMT	IPCE	IPCB	IPV6	IPV4	IPHE	PT

Table 609. RDES1 normal descriptor (write-back format)

Bit	Name	Description
31:16	OPC	Indicates any one of the following: • OAM sub-type code: If bits [18:16] of RDES3 as described in RDES3 normal descriptor (read format) are 111b, this field contains the OAM sub-type and code fields. • MAC control packet opcode: Bits 15:8 of RDES3 as described in RDES3 normal descriptor (read format) contain the subtype and bits 7:0 contain the code.
15	TD	Timestamp Dropped Indicates that the timestamp was captured for this packet but it dropped in the MTL Rx FIFO because of overflow. This field is available only when you select the timestamp feature. Otherwise, this field is reserved.
14	TSA	Timestamp Available Indicates that the timestamp value is available in a context descriptor word 2 (RDES2) and word 1(RDES1) when timestamp is present. This is valid only when the last descriptor field (RDES3 [28]) = 1 You can write the context descriptor in the next descriptor just after the last normal descriptor for a packet.
13	PV	PTP Version Indicates that the received PTP message has the IEEE 1588 version 2 format. When this field becomes 0, it indicates the IEEE 1588 version 1 format. This field is available only when you select the timestamp feature. Otherwise, this field is reserved.
12	PFT	PTP Packet Type Indicates that the PTP message is sent directly over Ethernet. This field is available only when you select the timestamp feature. Otherwise, this field is reserved.
11:8	PMT	PTP Message Type Encodes to give the type of the message received: • 0000: No PTP message received • 0001: SYNC (all clock types) • 0010: Follow_Up (all clock types) • 0011: Delay_Req (all clock types) • 0100: Delay_Resp (all clock types) • 0101: Pdelay_Req (in peer-to-peer transparent clock) • 0110: Pdelay_Resp (in peer-to-peer transparent clock) • 0111: Pdelay_Resp_Follow_Up (in peer-to-peer transparent clock)

Table continues on the next page...

Table 609. RDES1 normal descriptor (write-back format) (continued)

Bit	Name	Description
		• 1000: Announce • 1001: Management • 1010: Signaling • 1011–1110: Reserved • 1111: PTP packet with reserved message type These fields are available only when you select the timestamp feature.
7	IPCE	IP Payload Error When this field is 1, it indicates either of the following: • The 16-bit IP payload checksum (that is, the TCP, UDP, or ICMP checksum) which MAC calculates does not match the corresponding checksum field in the received segment. • The TCP, UDP, or ICMP segment length does not match the payload length value in the IP header field. • The TCP, UDP, or ICMP segment length is less than minimum allowed segment length for TCP, UDP, or ICMP. Bit 15 (ES) of RDES3 is not set when this field is 1.
6	IPCB	IP Checksum Bypassed Indicates that the checksum offload engine is bypassed. This field is available when you select the enable receive TCP/IP checksum check feature.
5	IPV6	IPv6 header Present Indicates that an IPV6 header is detected. When you select the enable split header feature option and the SPH bit of control register of a channel is 1, the IPV6 header is available in the header buffer area to which RDES0 points.
4	IPV4	IPV4 Header Present Indicates that an IPV4 header is detected. When the SPH bit of RDES3 is 1, the IPV4 header is available in the header buffer area to which RDES0 points.
3	IPHE	IP Header Error When this field is 1, it indicates either of the following: • The 16-bit IPv4 header checksum which MAC calculates does not match the received checksum bytes. • The IP datagram version is not consistent with the Ethernet type value. • Ethernet packet does not have the expected number of IP header bytes. This field is valid when either bit 5 or bit 4 is 1. This field is available when you select the enable receive TCP/IP checksum check feature.
2:0	PT	Payload Type Indicates the type of payload encapsulated in the IP datagram that the receive checksum offload engine (COE) process.

Table continues on the next page...

Table 609. RDES1 normal descriptor (write-back format) (continued)

Bit	Name	Description
		• 3'b000: Unknown type or IP/AV payload not processed • 3'b001: UDP • 3'b010: TCP • 3'b011: ICMP • 3'b110: AV tagged data packet • 3'b111: AV tagged control packet • 3'b101: AV untagged control packet • 3'b100: IGMP, if IPV4 header present field = 1, else DCB (LLDP) control packet If the COE does not process the payload of an IP datagram because there is an IP header error or fragmented IP, it sets these fields to 3'b000.

75.15.4.1.5.3 RDES2 normal descriptor (Write-back format)

31:29	28	27	26:19	18	17	16	15	14	13:11	10	9:0
L3L4FM	L4FM	L3FM	MADRM	HF	DAF	SAF	OTS	ITS	Rsvd	ARPNR	HL

Table 610. RDES2 normal descriptor (Write-back format)

Bit	Name	Description
31:29	L3L4FM	Layer 3 and Layer 4 Filter Number Matched Indicates the number of the layer 3 and layer 4 filter that matches the received packet: • 000: Filter 0 • 001: Filter 1 • 010: Filter 2 • 011: Filter 3 • 100: Filter 4 • 101: Filter 5 • 110: Filter 6 • 111: Filter 7 This field is valid only when bit 28 or bit 27 = 1. When more than one filter matches, these fields provide the number of lowest filter. **NOTE** This status is not available when Flexible RX Parser is enabled.
28	L4FM	Layer 4 Filter Match

Table continues on the next page...

Table 610. RDES2 normal descriptor (Write-back format) (continued)

Bit	Name	Description
		When this field is 1, it indicates that the received packet matches one of the enabled layer 4 port number fields. This status is given only when one of these conditions is true: • Layer 3 fields are not enabled and all enabled layer 4 fields match • All enabled layer 3 and layer 4 filter fields match When more than one filter matches, this bit gives the layer 4 filter status of filter indicated by Bits[31:29]. **NOTE** This status is not available when the flexible receive parser is enabled.
27	L3FM	Layer 3 Filter Match When this field is 1, it indicates that the received packet matches one of the enabled layer 3 IP address fields. This status is given only when one of these conditions is true: • All enabled layer 3 fields match and bypasses all enabled layer 4 fields. • All enabled filter fields match. When more than one filter matches, this field gives the layer 3 filter status of filter which bits[31:29] indicate. **NOTE** This status is not available when the flexible receive parser is enabled.
26:19	MADRM	MAC Address Match or Hash Value When HF becomes 0, it indicates that this field contains the MAC address register number that matches the received packet destination address. This field is valid only if the DAF field becomes 0. When HF = 1, it indicates that this field contains the hash value that MAC computes. A packet passes the hash filter when the field corresponding to the hash value is 1 in the hash filter register. **NOTE** This status is not available when the flexible receive parser is enabled.
18	HF	Hash Filter Status When this field is 1, it indicates that the packet passes the MAC address hash filter. Bits[26:19] indicate the hash value. **NOTE** This status is not available when the flexible receive parser is enabled.
17	DAF/RXPI	Destination Address Filter Fail When flexible receive parser is disabled, and this field is 1, it indicates that the packet fails the DA filter in MAC. When flexible receive parser is enabled, and this field is 1, it indicates that the packet parsing is incomplete (RXPI) due to ECC error.

Table continues on the next page...

Table 610. RDES2 normal descriptor (Write-back format) (continued)

Bit	Name	Description
		NOTE When this field is 1, RDES3 ES field is also 1.
16	SAF/RXPD	SA Address Filter Fail When flexible receive parser is disabled, and this field is 1, it indicates that the packet fails the SA filter in MAC. When flexible receive parser is enabled and this field is 1, it indicates that the parser drops the RXPD packet. **NOTE** When this field is 1, RDES3 ES field is also 1.
15	OTS	VLAN Filter Status When this field is 1, it indicates that the received packet VLAN tag passes the VLAN filter. This field redefines as an outer VLAN tag filter status (OTS). This field is valid for both single and double VLAN tagged frames.
14	ITS	Inner VLAN Tag Filter Status (ITS) This field is valid only for double VLAN tagged frames, when double VLAN processing is enabled. See Filter status for more information.
13:11	Rsvd	Reserved
10	ARPNR	ARP Reply Not Generated When this field is 1, it indicates that MAC did not generate the ARP reply for the ARP request packet it receives.. This field is 1 when MAC is busy transmitting ARP reply to earlier ARP request (only one ARP request is processed at a time). This field is reserved when you do not select the enable IPv4 ARP offload option.
9:0	HL	L3/L4 Header Length Contains the length of the packet header which MAC splits at L3 or L4 header boundary which the MAC receiver identifies. This field is valid only when the first descriptor bit = 1 (FD = 1). You can write the header data to the buffer 1 address of corresponding descriptor. If the header length is zero, this field is not valid. It implies that MAC does not identify and split the header. This field is valid when you select the enable split header feature option.

75.15.4.1.5.4 RDES3 normal descriptor (Write-back format)

Following table describes the write-back format for the RDES3 normal descriptor.

31	30	29	28	27	26	25	24	23	22	21	20	19	18:16	15	14:0
OWN	CTXT	FD	LD	RS2V	RS1V	RS0V	CE	GP	RWT	OE	RE	DE	LT	ES	PL

Table 611. RDES3 normal descriptor (Write-back format)

Bit	Name	Description
31	OWN	Own Bit When this field is 1, it indicates that the module DMA owns the descriptor. If this field becomes 0, it indicates that the application owns the descriptor. DMA writes 0 to this field when either of these conditions is true: • DMA completes the packet reception. • The buffers associated with the descriptor are full.
30	CTXT	Receive Context Descriptor When this field is 1, it indicates that the current descriptor is a context type descriptor. DMA writes 1'b0 to this field for a normal receive descriptor. When CTXT and FD bits are used together, {CTXT, FD} • 2’b00: Intermediate descriptor • 2’b01: First descriptor • 2’b10: Reserved • 2’b11: Descriptor error (due to all 1s) **NOTE** When descriptor error occurs, the receive DMA closes the receive descriptor indicating a descriptor error. This receive descriptor is skipped and the buffer addresses are not used to write the packet data. Receive DMA will write 1 to the CDE bit in DMA_CH#_Status register but not to the RI bit even when IOC = 1, as this is not marked as the last receive descriptor for the packet. The subsequent valid receive descriptor writes the packet data.
29	FD	First Descriptor When this field is 1, it indicates that this descriptor contains the first buffer of the packet. If the size of the first buffer is 0, the second buffer contains the beginning of the packet and if the size of the second buffer is also 0, the next descriptor will contain the beginning of the packet. See the CTXT bit description for more information on how to use the CTXT bit and FD bit together.
28	LD	Last Descriptor When this field is 1, it indicates that this descriptor buffers points to the last buffers of the packet.
27	RS2V	Receive Status RDES2 Valid When this field is 1, it indicates that the status in RDES2 is valid and DMA writes it. This field is valid only when the LD bit of RDES3 = 1.
26	RS1V	Receive Status RDES1 Valid When this field is 1, it indicates that the status in RDES1 is valid and DMA writes it. This field is valid only when the LD bit of RDES3 = 1.
25	RS0V	Receive Status RDES0 Valid When this field is 1, it indicates that the status in RDES0 is valid and DMA writes it. This field is valid only when the LD bit of RDES3 = 1.

Table continues on the next page...

Table 611. RDES3 normal descriptor (Write-back format) (continued)

Bit	Name	Description
24	CE	CRC Error When this field is 1, it indicates that a cyclic redundancy check (CRC) error occurs on the received packet. This field is valid only when the LD bit of RDES3 = 1.
23	GP	Giant Packet When this field is 1, it indicates that the packet length exceeds the specified maximum Ethernet size of 1518, 1522, or 2000 bytes (9018 or 9022 bytes if jumbo packet enable = 1). **NOTE** Giant packet indicates only the packet length. It does not cause any packet truncation.
22	RWT	Receive Watchdog Timeout When this field is 1, it indicates that the receive watchdog timer expires when it receives the current packet. The current packet truncates after watchdog timeout.
21	OE	Overflow Error When this field is 1, it indicates that the received packet is damaged because of buffer overflow in receive FIFO. **NOTE** This field is 1 only when DMA transfers a partial packet to the application. This happens only when the receive FIFO operates in the Threshold mode. In the Store-and-Forward mode, all the partial packets drops completely in the receive FIFO.
20	RE	Receive Error When this field is 1, it indicates that the gmii_rxer_i signal asserts when the gmii_rxdv_i signal asserts during the packet reception. This error also includes carrier extension error in the GMII and Half-duplex mode. Error can be of less or no extension, or error (rxd!= 0f) during extension.
19	DE	Dribble Bit Error When this field is 1, it indicates that the received packet has a non-integer multiple of bytes (odd nibbles). This field is valid only in the MII mode.
18:16	LT	Length/Type Field Indicates if the packet received is a length packet or a type packet. The encoding of the 3 bits is as follows: • 3'b000: The packet is a length packet. • 3'b001: The packet is a type packet. • 3'b011: The packet is a ARP request packet type • 3'b100: The packet is a type packet with VLAN tag • 3'b101: The packet is a type packet with double VLAN tag • 3'b110: The packet is a MAC control packet type • 3'b111: The packet is a OAM packet type

Table continues on the next page...

Table 611. RDES3 normal descriptor (Write-back format) (continued)

Bit	Name	Description
		• 3'b010: Reserved
15	ES	Error Summary When this field is 1, it indicates the logical OR of these bits, : • RDES3[24]: CRC error • RDES3[19]: Dribble error • RDES3[20]: Receive error • RDES3[22]: Watchdog timeout • RDES3[21]: Overflow error • RDES3[23]: Giant packet • RDES2[17]: Destination address filter fail, when the flexible receive parser is enabled • RDES2[16]: SA address filter fail, when flexible receive parser is enabled This field is valid only when the LD bit of RDES3 = 1.
14:0	PL	Packet Length Indicates the byte length of the received packet that was transferred to system memory (including CRC). This field is valid when the LD bit of RDES3 = 1 and overflow error bits becomes 0. The packet length also includes the two bytes that appends to the Ethernet packet when IP checksum calculation is enabled and the received packet is not a MAC control packet. This field is valid when the LD bit of RDES3 = 1. When the last descriptor and error summary bits are not 1, this field indicates the accumulated number of bytes that are transferred for the current packet.

75.15.4.1.6 Receive context descriptor

This descriptor is read-only for the application. Only DMA can write to this descriptor. The context descriptor provides information about the extended status related to the last received packet. Bit 30 of RDES3 indicates the context type descriptor.

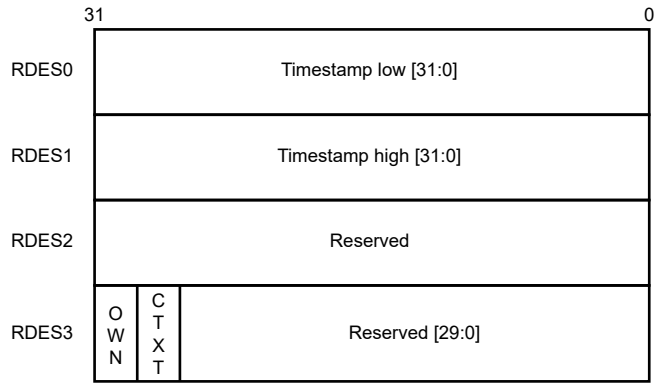


Figure 447. Receive context descriptor format

75.15.4.1.6.1 RDES0 context descriptor

Table 612. RDES0 context descriptor

Bit	Name	Description
31:0	RTSL	Receive Packet Timestamp Low DMA updates this field with least significant 32-bits of the timestamp captured for the corresponding receive packet. When this field and the RTSH field of RDES1 show all-ones value, you must consider the timestamp as corrupt.

75.15.4.1.6.2 RDES1 context descriptor

Table 613. RDES1 context descriptor

Bit	Field	Description
31:0	RTSH	Receive Packet Timestamp High DMA updates this field with the most significant 32-bits of the timestamp captured for the corresponding receive packet. When this field and the RTSL field of RDES0 show all-ones value, you must consider the timestamp as corrupt.

75.15.4.1.6.3 RDES2 context descriptor

Table 614. RDES2 context descriptor

Bit	Description
31:0	Reserved

75.15.4.1.6.4 RDES3 context descriptor

31	30	29	28:0
OWN	CTXT	DE	Rsvd

Table 615. RDES3 context descriptor

Bit	Name	Description
31	OWN	Own Bit When this field is 1, it indicates that DMA owns the descriptor. When this field becomes 0, it indicates that the application owns the descriptor. DMA clears this field when either of these conditions is true: • DMA completes the packet reception. • The buffers associated with the descriptor are full.
30	CTXT	Receive Context Descriptor

Table continues on the next page...

Table 615. RDES3 context descriptor (continued)

Bit	Name	Description
		When this field is 1, it indicates that the current descriptor is a context descriptor. DMA writes 1'b1 to this field for context descriptor. DMA writes 2'b11 to indicate a descriptor error due to all = 1. When CTXT and DE bits are used together {CTXT, DE} • 2'b00: Reserved • 2'b01: Reserved • 2'b10: Context descriptor • 2'b11: Descriptor error Note: When descriptor error occurs, the receive DMA closes the receive descriptor indicating descriptor error. This receive descriptor is skipped and the buffer addresses does not write the packet data. The receive DMA writes 1 to the CDE bit in DMA_CH#_Status register but not to the RI bit even when IOC = 1, because this is not marked as last receive descriptor for the packet. The subsequent valid receive descriptor writes the packet data.
29	DE	Descriptor Error See the CTXT bit description for more information about how to use the DE bit along with CTXT bit.
28:0	Rsvd	Reserved

75.15.5 Enhanced descriptor for time-based scheduling

This feature needs 32 bytes enhanced descriptors to enable on all the DMA channels that uses this feature (write 1 to EDSE bit of DMA_CH(#i)_TX_Control register).

The structure of 32-byte descriptor for the context and the normal descriptor in read and write formats are described in the upcoming sections.

75.15.5.1 Enhanced normal descriptor - Read (32-bit mode)

These fields are present in the first 16 bytes of the enhanced descriptor format of normal descriptor:

- LTV- Indicates that the launch time (LT) and GSN fields present in the descriptor are valid. The LTV must be = 1 only if the FD bit of the descriptor is not.
- GSN- Indicates the GCL slot number associated with the packet.
- LT- Indicates the launch time associated with the packet.

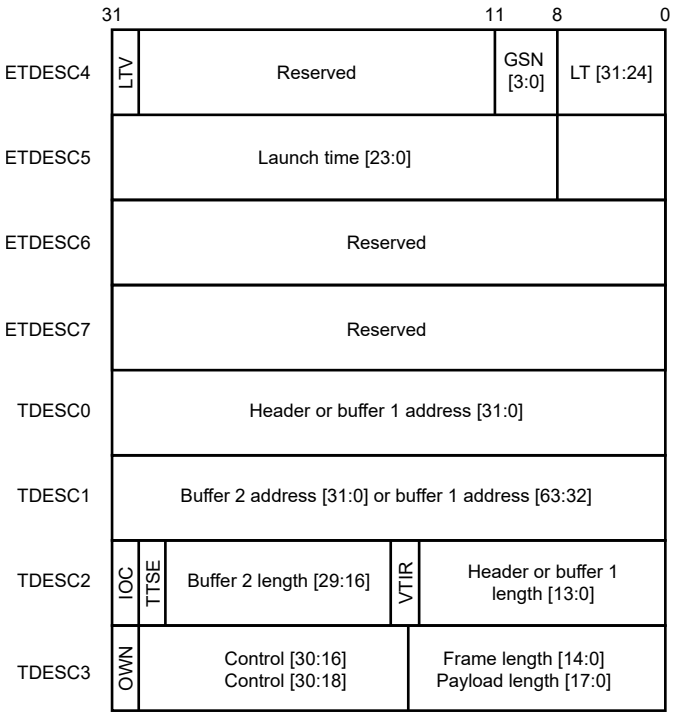


Figure 448. Enhanced normal descriptor - Read (32-bit mode)

75.15.5.2 Enhanced normal descriptor (Write, 32-bit mode)

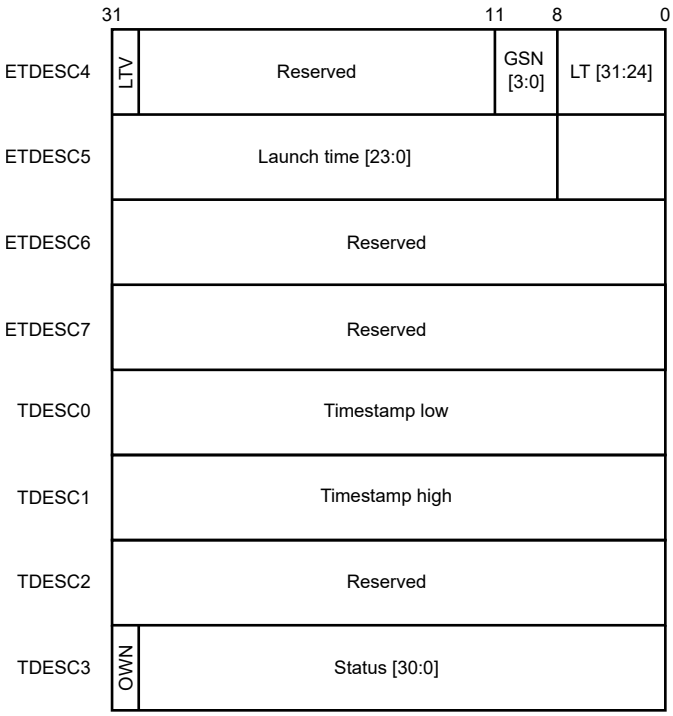


Figure 449. Enhanced normal descriptor write (32-bit mode)

NOTE

- All the enhanced descriptors, example, assumes a 32-bit data width configurations. However, the same definitions are valid in 64-bit and 128-bit configurations.
- In both the write back formats, you cannot modify the extended 16 bytes. The remaining 16 bytes (TDESC0 to TDESC3) are written back per the previous 16 bytes descriptor format.
- When you enable the fetch time it overrides the AV slot function
- When an unaligned new GCL list (with a different CTR) is installed it is recommended not to have traffic during the installation of the new list. Any traffic during the switching of the lists might have unpredictable behavior regarding fetch, launch, and launch expiry because the CTR and BTR values get updated when the frame is processed

75.15.5.3 Enhanced context descriptor (Read, 32-bit mode)

The first 16 bytes of the enhanced context descriptor (ETDESC4 to ETDESC7) are reserved and must be = 0. The fields present in the last bytes of the descriptor (TDESC0 to TDESC3) are same as the context descriptor (TDESC0 to TDESC3) in 16 byte format.

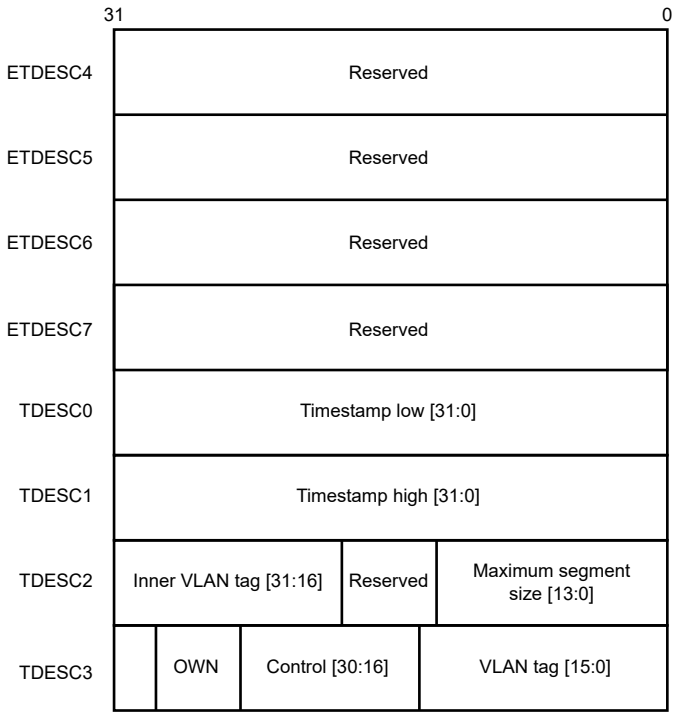


Figure 450. Enhanced context descriptor - Read

75.15.5.4 Enhanced context descriptor (Write, 32-bit mode)

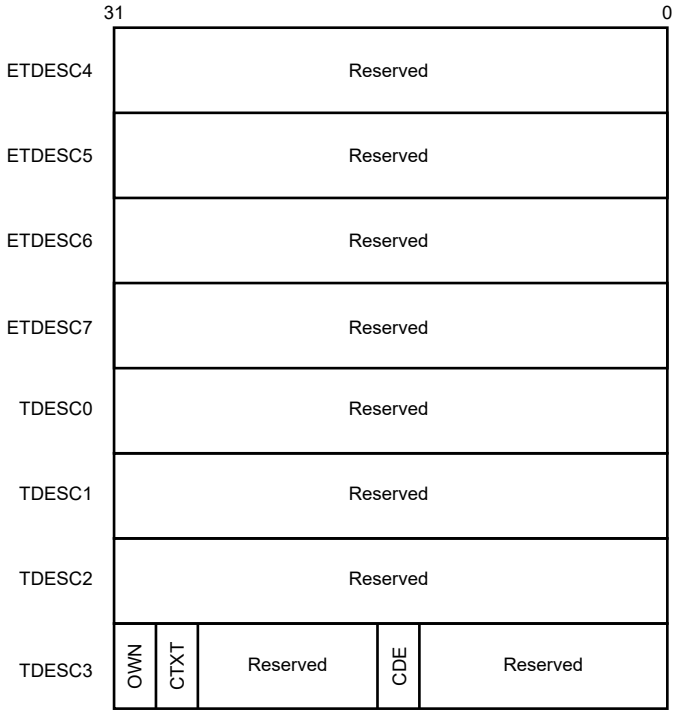


Figure 451. Enhanced context descriptor - Write

75.16 Programming

This section and all its sub-sections are Synopsys Proprietary. Used with permission.

This section provides the instructions for initializing the DMA or MAC registers in the proper sequence.

75.16.1 Initializing DMA

Perform these steps to initialize DMA:

1. Provide a software reset. This resets all the MAC internal registers and logic (bit-0 of DMA_Mode).
2. Wait for the reset process (poll bit 0 of DMA_Mode which is only clears after the reset operation completes) to complete.
3. Program these fields to initialize DMA_SysBus_Mode:
   a. DMA_SysBus_Mode[AAL]
   b. Fixed burst or undefined burst
   c. Burst mode values in case of AHB bus interface, OSR_LMT in case of AXI bus interface.
   d. Select the maximum burst length possible on the AXI bus (bits [7:1]), if fixed length value is enabled.
4. Create a descriptor list for a transmit and receive. Also, ensure that DMA (write 1 to bit 31 of descriptor TDES3/RDES3) owns the descriptors. See Descriptors for more information about descriptors.
5. Program the transmit and receive ring length registers (DMA_CH(#i)_TxDesc_Ring_Length (for i = 0; i <= 1) and DMA_CH(#i)_RxDesc_Ring_Length (for i = 0; i <= 1)). The programmed ring length must be at least 4.

NOTE

The descriptor address from the start to the end of the ring must not cross the 4 GB boundary.

6. Initialize the receive and transmit descriptor list address with the base address of the transmit and receive descriptor (DMA_CH(#i)_TxDesc_List_Address (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1), DMA_CH(#i)_RxDesc_List_Address (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1)). Also, program the transmit and receive tail pointer registers that indicates DMA about the available descriptors (DMA_CH(#i)_TxDesc_Tail_Pointer (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1) and DMA_CH(#i)_RxDesc_Tail_Pointer (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1)).

   NOTE

   For 40-bit or 48-bit addressing mode, program the higher address List registers (DMA_CH[n]_TxDesc_List_HAddress, DMA_CH[n]_RxDesc_List_HAddress).

   The tailpointer registers must advance to the location immediately after the descriptors that are set so that DMA is aware that the additional descriptors are available.

7. Program the settings of these registers for the parameters like maximum burst-length (PBL) which DMA initiates, descriptor skip lengths, OSP in case of TxDMA, RBSZ in case of RxDMA, and so on:
   - DMA_CH(#i)_Control (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1)
   - DMA_CH(#i)_TX_Control (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1)
   - DMA_CH(#i)_RX_Control (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1)
8. Program the DMA_CH(#i)_Interrupt_Enable (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1) register to enable the interrupts.
9. Write 1 to SR (bit 0) of the DMA_CH(#i)_RX_Control (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1) and ST (bit 0) of the DMA_CH(#i)_TX_Control (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1) register to start the receive and transmit DMAs.
10. Repeat steps 4 to 9 for all the transit DMA and receive DMA channels selected in the hardware.

75.16.2 Initializing MTL registers

The transaction layer (MTL) registers must initialize to establish the transmit and receive operating modes and commands.

Perform these steps to initialize the MTL registers:

1. Program MTL_Operation_Mode[SCHALG] and MTL_Operation_Mode[RAA] to initialize the MTL operation in case of multiple transit and receive queues.
2. Program the receive queue to DMA mapping in MTL_RxQ_DMA_Map0.
3. Program these fields to initialize the mode of operation in MTL_TxQ0_Operation_Mode.
   a. MTL_TxQ0_Operation_Mode[TSF] or MTL_TxQ0_Operation_Mode[TTC] in case of Threshold mode.
   b. MTL_TxQ0_Operation_Mode[TXQEN] to value 2'b10 to enable transmit queue0.
   c. MTL_TxQ0_Operation_Mode[TQS]
4. Program these fields to initialize the mode of operation in MTL_RxQ0_Operation_Mode:
   a. MTL_RxQ0_Operation_Mode[RSF] or MTL_RxQ0_Operation_Mode[RTC] in case of Threshold mode.
   b. Flow control activation and de-activation thresholds for MTL receive FIFO (MTL_RxQ0_Operation_Mode[RFA] and MTL_RxQ0_Operation_Mode[RFD]).
   c. Error Packet and undersized good Packet forwarding enable (MTL_RxQ0_Operation_Mode[FEP] and MTL_RxQ0_Operation_Mode[FUP]).
   d. MTL_RxQ0_Operation_Mode[RQS]
5. Repeat previous two steps for all MTL transit and receive queues selected in the configuration.

75.16.3 Initializing MAC

The MAC configuration registers establish the operating mode of MAC. These registers must initialize before initializing DMA.

You can perform these MAC initialization operations after DMA initialization. If the MAC initialization completes before DMA is configured, enable the MAC receiver (last step in the following sequence) only after the DMA is active. Otherwise, the received frames fill the receive FIFO and overflow.

1. Provide the MAC address registers: MAC_Address0_High and MAC_Address0_Low. If more than one MAC address is enabled, , program the MAC addresses appropriately.

2. Program these fields to set the appropriate filters for the incoming frames in MAC_Packet_Filter:

a. Receive all

b. Promiscuous mode

c. Hash or perfect filter

d. Unicast, multicast, broadcast, and control frames filter settings

3. Program these fields for proper flow control in MAC_Q0_Tx_Flow_Ctrl:

a. Pause time and other Pause frame control bits

b. Transmit flow control bits

c. Flow control busy

4. Program MAC_Interrupt_Enable as required, and if applicable, for your configuration.

5. Program the appropriate fields in MAC_Configuration. For example Inter-packet gap when transmission and jabber disable.

6. Write 1 to MAC_Configuration[RE] and MAC_Configuration[TE] to start the MAC transmitter and receiver.

75.16.4 Performing normal receive and transmit operation

During normal operation of the module, read the normal and transmit interrupts, poll the descriptors, suspend the DMA (if it does not own descriptors), and read the values of the current host transmitter or receiver descriptor pointers for debugging.

For normal operation, perform these steps:

1. Read the interrupt status for normal transmit and receive interrupts. Then poll the descriptors, read the status of the descriptor which the host owns (either transmit or receive).

2. Set appropriate values for the descriptors, ensure that the DMA owns the transmit and receive descriptors to resume the data transmission and reception.

3. If DMA does not own the descriptors (or no descriptor is available), DMA enters into SUSPEND state. The transmission or reception can resume by freeing the descriptors and writing the descriptor tail pointer to Tx/Rx tail pointer register (DMA_CH[n]_TxDesc_Tail_Pointer and DMA_CH[n]_RxDesc_Tail_Pointer).

4. Read the current host transmitter or receiver descriptor address pointer values for the debug process (DMA_CH[n]_Current_App_TxDesc and DMA_CH[n]_Current_App_RxDesc register).

5. Read the current host transmit buffer address pointer and receive buffer address pointer values for the debug process (Register DMA_CH[n]_Current_App_TxBuffer and DMA_CH[n]_Cur-rent_App_RxBuffer).

75.16.5 Stopping and starting transmission

Perform these steps to pause the transmission for some time. The steps are provided for channel 0.

1. Write 0 to bit 0 (ST) of DMA_CH(#i)_TX_Control (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1) register to disable the transmit DMA (if applicable) .

2. Wait for any previous frame transmissions to complete. Read the appropriate fields of MTL_TxQ0_Debug (MTL_TxQ0_Debug[TRCSTS] is not 01 and MTL_TxQ0_Debug[TXQSTS] = 0) to check this.

3. Write 0 to MAC_Configuration[RE] and MAC_Configuration[TE] to disable the MAC transmitter and MAC receiver.

4. Disable the receive DMA (if applicable), after you ensure that the data in the receive FIFO transfers to the system memory (read the appropriate fields MTL_RxQ0_Debug[PRXQ] = 0 and MTL_RxQ0_Debug[RXQSTS] = 00).

5. Ensure that both the transit queue and receive queue are empty (MTL_TxQ0_Debug[TXQSTS] = 0 and MTL_RxQ0_Debug[RXQSTS] = 0).

6. Restart the operation by first starting the DMAs, and then enable the MAC transmitter and receiver.

NOTE

- Do not change the configuration (such as duplex mode, speed, port, or loop back) when the MAC is actively transmitting or receiving. You can change these parameters only when the MAC transmitter and receiver are not active.
- Similarly, do not change the DMA related configuration when transmit and receive DMA are active.

75.16.6 Programming guidelines for switching to new descriptor list in RxDMA

Switching to a new descriptor list is different in the receive DMA as compared to the transit DMA. It is permitted when the RxDMA is in SUSPEND state as described below:

- RxDMA prepares the descriptors in advance.
- If the RxDMA goes to SUSPEND state because the descriptors are not available, a major failure occurs (you are not able to free the filled-up descriptors or buffers). If this issue does not rectify immediately, frames will be lost because of an RxFIFO overflow. Therefore, you can create a new descriptor list and program the RxDMA to start using it immediately, without going into STOP state.

75.16.7 Programming guidelines for multi-channel multi-queuing

75.16.7.1 Transmit

1. Program the transmit queue size in TQS field of MTL_TxQ[n]_Operation_Mode register. Based on the value programmed in TQS field, you can determine the queue size.

In the transmit operation, the number of channels is equal to the number of the queues, because of this reason, the channel-to-queue mapping is fixed.

2. Enable the queue in TXQEN in the corresponding MTL_TxQ[n]_Operation_Mode register to use the queue. In DMA configurations, enable the ST field of DMA_CH[n]_Tx_Con-trol register and corresponding TXQEN in MTL_TxQ[n]_Operation Mode register.

3. Program the scheduling method in MTL_Operation_Mode[SCHALG].

4. Program MTL_TxQ[n]_Quantum_Weight register for DCB queue per the selected algorithm. In case of CBS algorithm in AVB queues program the MTL_TxQ[n]_ETS_Control, MTL_TxQ[n]_SendSlopeCredit, MTL_TxQ[n]_HiCredit, and MTL_TxQ[n]_LoCredit registers as required.

5. Program the MAC_TxQ_Prty_Map0 register to assign a fixed priority to the queue, if DCB is enabled and PFC function is required. You can use this assigned priority to determine if the corresponding queue must stop transmitting packet on the basis of the received PFC packet.

75.16.7.2 Receive

1. Program the receive queue size in RQS field of MTL_RxQ[n]_Operation_Mode register. Determine the size of the queue, based on the value programmed in RQS field.
2. Enable the receive queues 0 to 7 in the fields RXQ0EN to RXQ7EN in MAC_RxQ_Ctrl0 for AV or DCB. In DMA configurations, SR bit of statically or dynamically map DMA_CH[n]_Rx_Control register and the corresponding RXQ[n]_EN in MAC_RxQ_Ctrl0 is enabled.

3. Based on these packet types MAC routes the receive packets to the receive queues :
    a. AV PTP packets: Based on the programming of MAC_RxQ_Ctrl1[PTPQ].
    b. AV untagged control packets: Based on the programming of MAC_RxQ_Ctrl1[AVCPQ].
    c. Data center bridging (DCB) related link layer discovery protocol (LLDP) packets. Program DCBCPQ in MAC_RxQ_Ctrl1 to indicate MAC which queue should get the DCB packets.
    d. VLAN tag priority field in VLAN tagged packets: Program PSRQ7-0 of the MAC_RxQ_Ctrl2 and MAC_RxQ_Ctrl3 register for routing the tagged packets based on the received packets USP (user priority) field to the receive queues 0 to 7.
    e. Route the AV tagged control and data packets based on PSRQ field of MAC_RxQ_Ctrl2 and MAC_RxQ_Ctrl3 registers.

NOTE

The priorities set in PSRQ7-0 must be unique.

4. If multiple receive DMA channels are enabled, you must perform the following programming for proper arbitration and mapping:
    a. Program MTL_Operation_Mode[RAA] to select the arbitration algorithm to decide which RxQ is read out from the RxFIFO memory.
    b. Program the MTL_RxQ[n]_Control register to decide the weights and the packet arbitration for each RxQ.
    c. If you program the static mapping in MTL_RxQ_DMA_Map[n] register (RXQ[n]DADMACH becomes 0), fields RXQx2DMA and others are programmed to select the channel to map each queue.
    d. Write 1 to RXQ[n]DADMACH field in MTL_RxQ_DMA_Map0 to select the dynamic mapping of packets in each receive queue.
    e. In dynamic channel mapping, the value of DCS field in the lowest MAC address register the decides the routing of a packet to a specific RxDMA channel.

75.16.7.2.1 Programming guidelines for recovering from DMA channel failure

When the DMA channel issues a bus error, follow these steps to recover from the failure.

Recovering from the receive DMA channel failure.

Perform these steps if you get bus error in the receive DMA channel:

1. Write 1 to DMA_CH0_Rx_Control[RPF]. This flushes all the packets one after the other.

This step is optional. However, writing 1 to this field prevents head-of-line blocking in the receive queues when packets sent to the RXDMA are stopped due to the bus error.

2. Re-program the specific registers of the DMA channel.

3. Start the DMA channel.

Recovering from the transmit DMA channel failure.

1. Stop the specific DMA channel, even if it is in active state.

2. Flush the corresponding MTL queue.

3. Re-program the specific registers of the DMA channel.

4. Start the DMA channel.

NOTE

Due to the known limitations in the design, reprogramming the DMA channel registers might not be always successful in recovering from a bus error. If the module is not fully functional after reprogramming the DMA, you can issue a soft reset to recover from the bus error.

75.16.8 Programming guidelines for GMII link state transitions

75.16.8.1 Transmit and receive clocks running when link down

Perform these steps when the link is down but the transmit and receive clocks are running:

NOTE

The steps are provided for channel 0.

1. Write 0 to bit 0 (ST) of DMA_CH(#i)_TX_Control (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1) register to disable the transmit DMA (if applicable).

2. Write 0 to MAC_Configuration[RE] to disable the MAC receiver.

3. Wait for any previous frame transmissions to complete. You can check this by reading the appropriate fields of MTL_TxQ0_Debug (MTL_TxQ0_Debug[TRCSTS] is not 01) or, flush the Tx FIFO for faster empty operation.

4. Write 0 to MAC_Configuration[TE] to disable the MAC transmitter.

5. Make sure that both the transit queue and receive queue are empty (MTL_TxQ0_Debug[TXQSTS] and MTL_RxQ0_Debug[RXQSTS] are 0).

6. Read the PHY registers to know the latest configuration and accordingly program the MAC registers, after the link is up.

7. Start the transit DMA to restart the operation, and then enable the MAC transmitter and receiver.

You do not disable the receive DMA because the receiver is disabled and the FIFO does not get any data in the receive FIFO.

75.16.8.2 Transmit and receive clocks stopped when link down

Perform these steps when the link is down and the transmit and receive clocks are stopped:

NOTE

The steps are provided for channel 0.

1. Write 0 to MAC_Configuration[RE] and MAC_Configuration[TE] to disable the MAC transmitter and receiver. This does not take effect immediately as the clocks are absent.

2. Wait until the link is up and the clocks are restored.

3. Wait for the transfer of any partial frame, if any to complete at time of stopping of the transmit or receive clock. To check this, read MAC_Debug (should be all-zero). Some old packets may still remain in the TXFIFO because the MAC transmitter is stopped.

4. Read the PHY registers to know the latest operating mode and accordingly program the MAC registers.

5. Write 1 to MAC_Configuration[RE] and MAC_Configuration[TE] to restart the MAC transmitter and receiver.

75.16.9 Programming guidelines for IEEE 1588 timestamping

75.16.9.1 Initialization guidelines for system time generation

Write 1 MAC_Timestamp_Control[TSENA] to enable the timestamp feature. However, it is essential to initialize the timestamp counter after MAC_Timestamp_Control[TSENA] = 1. Perform these steps during the module initialization:

1. Write 0 to the bit 16 of MAC_Interrupt_Enable to mask the timestamp trigger interrupt.
2. Write 1 to MAC_Timestamp_Control[TSENA] to enable timestamping.
3. Program MAC_Sub_Second_Increment based on the PTP clock frequency.
4. Program MAC_Timestamp_Addend and write 1 to MAC_Timestamp_Control[TSADDREG], if you are using the fine correction approach.
5. Poll MAC_Timestamp_Control until MAC_Timestamp_Control[TSADDREG] = 0.
6. Program MAC_Timestamp_Control[TSCFUPDT] to select the fine update method (if required).

7. Program MAC_System_Time_Seconds_Update and MAC_System_Time_Nanoseconds_Update with the appropriate time value.
8. Write 1 to MAC_Timestamp_Control[TSINIT].
9. Initialize the timestamp counter with the value written in the timestamp update registers so that the timestamp counter can start operation. If one-step timestamping is enabled
    a. Program Bit 27 of the TDES3 context descriptor, to enable one-step timestamping.
    b. Program MAC_Timestamp_Ingress_Asym_Corr and MAC_Timestamp_Egress_Asym_Corr to update the correction field in PDelay_Req PTP messages.
10. Enable the MAC receiver and transmitter for proper timestamping.

NOTE

If the timestamp operation is disabled by writing 0 to MAC_Timestamp_Control[TSENA], repeat all these steps to restart the timestamp operation.

75.16.9.2 System time correction

75.16.9.2.1 Coarse correction method

Perform these steps to synchronize or update the system time in one process (coarse correction method):

1. Set the offset (positive or negative) in the timestamp registers that is MAC_System_Time_Seconds_Update and MAC_System_Time_Nanoseconds_Update.

2. Write 1 to MAC_Timestamp_Control[TSUPDT].

The value in the timestamp update registers is added to or subtracted from the system time when MAC_Timestamp_Control[TSUPDT] = 0.

75.16.9.2.2 Fine correction method

Perform these steps to synchronize or update the system time to reduce system-time jitter (fine correction method):

1. Calculate the rate by which you want to increment the system time slower or faster, with the help of the algorithm.

2. Update MAC_Timestamp_Addend with the new value and write 1 to MAC_Timestamp_Control[TSADDREG].

3. Wait for the time for which you want the new value of the addend register to be active. You can do this by enabling the timestamp trigger interrupt after the system time reaches the target value.

4. Program the required target time in MAC_PPS[n]_Target_Time_Seconds register and MAC_PPS[n]_Target_Time_Nanoseconds register.

5. Enable the timestamp interrupt in MAC_Interrupt_Enable[TSIE].

6. Write 1 to bit 4 in MAC_Timestamp_Control.

7. Read MAC_Interrupt_Status when this trigger causes an interrupt.

8. Reprogram MAC_Timestamp_Addend with the old value and write 1 to MAC_Timestamp_Control[TSADDREG] again.

75.16.10 Programming guidelines for AV feature

After you enable the AV feature in the module controller, follow these programming tasks:

- Initializing the DMA for QOS-AHB configurations only
- Enabling slot number checking
- Enabling average bits per slot reporting
- Disabling flow control for AV-enabled queues (transmit and receive flow control)

75.16.10.1 Initializing the DMA in audio video feature

The first step to program the AV feature in a QOS-AHB configuration is to initialize the DMA.

Use this initialization sequence for QOS-AHB/QOS-AXI/QOS-AXI4 configurations with AV feature.

1. Provide a software reset to reset all the QOS internal registers and logic (DMA_Mode[SWR]).

2. Wait for the reset process to complete. Poll DMA_Mode[SWR], which clears only after the reset operation completes.

3. Set the values in DMA_Mode to program the fields to initialize the DMA register.

4. Create a descriptor list for transmit and receive. In addition, ensure that the DMA owns the transmit and receive descriptors. When you use OSF mode, at least two transmit descriptors are required.

See Descriptors for more information about descriptors.

5. Ensure that you create three or more different transmit or receive descriptors in the list before reusing any descriptors.

6. Program the transmit and receive ring length registers (DMA_CH(#i)_TxDesc_Ring_Length (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1) and DMA_CH(#i)_RxDesc_Ring_Length (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1)). The ring length programmed must be at least 4.

7. Initialize receive and transmit descriptor list address with the base address of the transmit and receive descriptor (DMA_CH(#i)_TxDesc_List_Address (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1), DMA_CH(#i)_TxDesc_List_Address (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1)). In addition, you must program the transmit and receive tail pointer registers to indicate the DMA about the available descriptors (DMA_CH(#i)_TxDesc_Tail_Pointer (for i = 0; i <= DWC_EQOS_NUM_D-MA_TX_CH-1) and DMA_CH(#i)_RxDesc_Tail_Pointer (for i = 0; i <= DWC_EQOS_NUM_D-MA_RX_CH-1)).

8. Program these fields to initialize the mode of operation in MTL_TxQ0_Operation_Mode:

a. MTL_TxQ0_Operation_Mode[TSF]

b. MTL_TxQ0_Operation_Mode[TTC]

c. MTL_TxQ0_Operation_Mode[TXQEN] to value 2'b10 to enable transmit queue0

d. MTL_TxQ0_Operation_Mode[TQS]

9. Enable the interrupts by programming DMA_CH(#i)_Interrupt_Enable (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1) register.

10. Repeat steps 4 through 9 for all the additional channels of AV feature.

11. Program the AV queues CBS control register, idleSlope, sendSlope, hiCredit, and loCredit registers.

12. Write 1 to bit 0 of DMA_CH(#i)_TX_Control (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1) and DMA_CH(#i)_RX_Control (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1) register to start the receive and transmit DMA.

75.16.10.2 Enabling slot number checking

You can enable slot number checking feature if you specifies the intervals at which the DMA channels map to AV queues fetch the frames from the AHB/AXI system bus.

You must complete these steps after step 11 and before step 12 of Initializing the DMA in audio video feature .

You can use the slot number check feature to specify the intervals at which the DMA channels map to AV queues fetch the frames from the AHB/AXI system bus. This feature is useful for an uniform and periodic transfer of the AV traffic from the host memory and it is available only when you enable timestamping and programs MAC_Sub_Second_Increment. Perform these steps to enable the slot number checking:

1. Follow the steps described in Initialization guidelines for system time generation to enable timestamping.

2. Ensure that the SLOTNUM field (bits 22:19) of TDES3 normal descriptor (Read format) contains a valid slot number. You can read the current reference slot number from the DMA_CH(#i)_Slot_Func-tion_Control_Status (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1) register.

3. Write 1 to bit 0 (ESC) of the slot function control and status register of a channel to enable the slot number checking.

75.16.10.3 Enabling average bits per slot reporting

You can enable the reporting of average bits transmitted in a slot.

The CBS status register of the additional AV channels provide information about the average bits that are transmitted in a slot. You can asynchronously read this register to retrieve information about the average bits transmitted per slot. Perform these steps to enable average bits per slot reporting:

1. Follow the steps as described in the Initialization guidelines for system time generation to enable timestamping.

2. Program SLC bits [6:4], of the MTL_TxQ[n]_ETS_Control register of a channel with the number of slots over which the average transmitted bits per slot are computed.

3. Enable bit 9(ABPSSIE) of the MTL_Q[n]_Interrupt_Control_Status register of a channel to generate the average bits per slot interrupt.

NOTE

- The frequency of this interrupt depends on the value programmed in step 2. For example, when you program value 0 in the SLC field, the interrupt generates at every 125 microsecond.
- You can disable this interrupt to stop the interrupt flooding, when it is not required.

4. Read ABS bits [16:0], of the MTL_TxQ[n]_ETS_Status register of a channel on each interrupt.

NOTE

You can read the ABS fields in the polling mode even if ABPSIE bit is not enabled. When bit 1 (ABPSIS) of the MTL_TxQ[n]_ETS_Status register is 1, it indicates that a new value is updated in the ABS field.

75.16.10.4 Disabling flow control for AV enabled queues

75.16.10.4.1 Transmit flow

Program the EHFC (Enable Hardware Flow Control) field of the corresponding receive queue's MTL_RxQ[n]_Operation_Mode register to 0.

75.16.10.4.2 Receive flow control

Program PSTQ[n] field, corresponding to AV enabled transit queue in MAC_TxQ_Prty_Map0/1 register to 0.

75.16.10.5 Programming guidelines for flexible pulse-per-second output

After you enable the flexible pulse-per-second output feature in the module controller, you can perform these tasks:

- Generating single pulse on PPS
- Generating next pulse on PPS
- Generating a pulse train on PPS
- Generating an interrupt without affecting the PPS

75.16.10.5.1 Generating single pulse on PPS

Perform these steps to generate single pulse on PPS:

1. Program 11 or 10 (for interrupt) in TRGTMODSEL bits [6:5], of MAC_PPS_Control. This instructs MAC to use the target time registers (MAC_PPS0_Target_Time_Seconds and MAC_PPS0_Target_Time_Nanoseconds) for a start time of the PPS signal output.
2. Program the start time value in the target time registers (MAC_PPS0_Target_Time_Seconds and MAC_PPS0_Target_Time_Nanoseconds).
3. Program the PPS signal output width in MAC_PPS(#i)_Width (for i = 0; i <= 3) register.

4. Program PPSCMD bits [3:0], of MAC_PPS_Control to 0001. This instructs MAC to generate single pulse on the PPS signal output at the time programmed in the target time registers.

75.16.10.5.2 Generating next pulse on PPS

When the PPSCMD is executed (PPSCMD bits = 0), you can give the cancel start command (PPSCMD=0011) before the programmed start time elapses to cancel the pulse generation. You can also program the behavior of the next pulse in advance.

Follow these steps to program the next pulse:

1. Program the start time for the next pulse in the target time registers. This time must be more than the time at which the falling edge occurs for the previous pulse.

2. Program the next PPS signal output width in MAC_PPS(#i)_Width (for i = 0; i <= DWC_EQOS_PPS_OUT_NUM-1) register.

3. Program PPSCMD bits [3:0], of MAC_PPS_Control to generate a single pulse after the time at which the previous pulse is de-asserted. This instructs MAC to generate single pulse on the PPS signal output, at the time programmed in target time registers.

If you give this command before the previous pulse becomes low, then the new command overwrites the previous command and the QOS may generate only 1 extended pulse.

75.16.10.5.3 Generating a pulse train on PPS

Perform these steps to generate a pulse train on PPS:

1. Program 11 or 10 (for interrupt) in TRGTMODSEL bits [6:5] , of MAC_PPS_Control. This instructs MAC to use the target time registers for the start time of the PPS signal output.

2. Program the start time value in the target time registers.

3. Program the interval value between the train of pulses on the PPS signal output in MAC_PPS(#i)_Interval (for i = 0; i <= DWC_EQOS_PPS_OUT_NUM-1) register.

4. Program the PPS signal output width in MAC_PPS(#i)_Width (for i = 0; i <= DWC_EQOS_PPS_OUT_NUM-1) register.

5. Program PPSCMD bits [3:0], of MAC_PPS_Control to 0010. This instructs MAC to generate train of pulses on the PPS signal output with start time programmed in target time registers.

By default, the PPS pulse train is free-running unless the STOP pulse train at time or STOP pulse train immediately commands stop it.

6. Program the stop value in the target time registers. Ensure that bit 31 (TSTRBUSY) of MAC_PPS(#i)_Target_Time_Nanoseconds (for i = 0; i <= DWC_EQOS_PPS_OUT_NUM-1) register becomes 0 before programming the target time registers again.

7. Program PPSCMD field [bit 3:0] of MAC_PPS_Control to 0100. This stops the train of pulses on the PPS signal output after the programmed stop time specified in step 6 elapses.

You can program 0101 in PPSCMD field [bit 3:0] of MAC_PPS_Control to stop the pulse train at any time. Similarly, you can program 0110 in PPSCMD field [bit 3:0] of MAC_PPS_Control before the time (programmed in step 6) elapses to cancel the stop pulse train command (given in Step 7). Also, program 0011 in PPSCMD field [bit 3:0] of MAC_PPS_Control before the programmed start time (in Step 2) elapses to cancel the pulse train generation.

75.16.10.5.4 Generating an interrupt without affecting the PPS

MAC_PPS_Control TRGTMODSEL bits [6:5] enables you to program the target time registers to perform any one of these actions:

- Generate only interrupts.
- Generate interrupts and the PPS start and stop time.
- Generate only PPS start and stop time.

Perform these steps to program the target time registers to generate only interrupt event:

1. Program 00 (for interrupt) in MAC_PPS_Control TRGTMODSEL bits [6:5]. This instructs the MAC to use the target time registers for target time interrupt.

2. Program a target time value in the target time registers. This instructs the MAC to generate an interrupt when the target time elapses.

If TRGTMODSEL bits [6:5], are changed (for example, to control the PPS) then over-write the interrupt generation with the new mode and new programmed target time register value.

75.16.10.6 Programming guidelines for VLAN filtering on receive

Perform these steps to program VLAN filtering on receive:

1. Program MAC_VLAN_Tag register for the following fields to select the filtering method:

a. ETV: Enables 12-Bit VLAN tag comparison or 16-bit VLAN tag comparison.

b. VTHM: VLAN tag hash table match enable.

c. ERIVLT: Enables inner VLAN tag or outer VLAN Tag (to enable the inner or outer VLAN tag filtering, writing 1 to MAC_VLAN_Tag_Ctrl[EDVLP] enables the double VLAN processing).

d. ERSVLM: Enables receive S-VLAN match or C-VLAN match (write 1 to MAC_VLAN_Tag_Ctrl[ESVL] to enable S-VLAN processing).

e. DOVLTC: Ignores VLAN type for tag match

f. VTIM: Enables VLAN tag inverse match instead of the normal VLAN tag matching

2. Program VL of MAC_VLAN_Tag register for the 12-bit or 16-bit VLAN tag.

3. Program MAC_VLAN_Hash_Table, if hash filtering of VLAN tag is enabled. When MAC_VLAN_Tag_Ctrl[ETV] becomes 0, it indicates that the upper 4 bits of the calculated CRC-32 of VLAN tag are inverted and index the content of MAC_VLAN_Hash_Table. When MAC_VLAN_Tag_Ctrl[ETV] is 1, it indicates that the upper 4 bits of calculated CRC-32 of VLAN Tag index the content of MAC_VLAN_Hash_Table. For example, when MAC_VLAN_Tag_Ctrl[ETV] is 1, a hash value of 4b'1000 selects bit 8 of MAC_VLAN_Hash_Table. When MAC_VLAN_Tag_Ctrl[ETV] becomes 0 a hash value of 4'b1000 selects bit 7 of MAC_VLAN_Hash_Table.

75.16.10.7 Programming guidelines for extended VLAN filtering and routing on receive

Perform these steps, for the indirect access of the per VLAN tag registers:

- Write
  - Write the required data into MAC_VLAN_Tag_Data.
  - Program MAC_VLAN_Tag_Ctrl[OFS] with the required filter register's offset and command type to MAC_VLAN_Tag_Ctrl[CT]. For a write command, set this field to 0.
  - Write 1 to MAC_VLAN_Tag_Ctrl[OB] and wait till it becomes 0 to do the next write. This will guarantee that you have programmed the appropriate VLAN Tag Filter register.
- Read
  - Program MAC_VLAN_Tag_Ctrl[OFS] with the required register's offset and command type to MAC_VLAN_Tag_Ctrl[CT]. For a read command, set this field to 1.
  - Write 1 to MAC_VLAN_Tag_Ctrl[OB] and wait till it becomes 0. The appropriate value of the VLAN Tag Filter register is available in MAC_VLAN_Tag_Data.

75.16.10.8 Programming sequence for queue/channel based VLAN inclusion register

Perform these steps to program the queue/channel-based VLAN inclusion register:

NOTE

When MAC_VLAN_Incl[CBTI] = 1 you cannot access the indirect VLAN include registers.

1. Write 1 to MAC_VLAN_Incl[CBTI], to enable queue or channel based VLAN tag insertion on all the transmitted packets. This field must be 1 before any indirect access to the queue or channel specific MAC_VLAN_Incl(#i) register.

2. Program the VLAN tag and VLAN type to insert in packets from a particular queue or channel in MAC_VLAN_Incl[VLT] and MAC_VLAN_Incl[CSVL], corresponding offset address in ADDR field (0 for queue/channel 0, 1 for queue/channel 1, and so on) must be set. Write 0 to MAC_VLAN_Incl[RDWR] to indicate write access. The write to byte 0 (byte 3 in Big Endian mode) of MAC_VLAN_Incl initiates access to indirect access MAC_VLAN_Incl(#i) register.

3. Write 1 to MAC_VLAN_Incl[BUSY] via the module to indicate the progress of access to indirect access MAC_VLAN_Incl(#i) register. After the access completes, MAC_VLAN_Incl[BUSY] = 0. The application must not attempt subsequent access to MAC_VLAN_Incl(#i) register when the MAC_VLAN_Incl[BUSY] is 1.

4. Repeat step 2 and step 3 to program VLAN tag and VLAN type to insert in packets from the remaining queues or channels. The application must ensure that the required VLAN tag and VLAN type for all the queues or channels are programmed, otherwise an unintended VLAN tag and VLAN type might be inserted.

75.16.11 Programming guidelines for EST

Program the gate control values and time intervals in MTL_EST_Status[SWOL] along with the other EST related registers that are described in EMAC register descriptions to appropriate values. The upcoming sub-sections provide step by step details for programming the GCL and the other EST related registers.

75.16.11.1 Programming the GCL and GCL linked registers

Perform these steps to program the GCL and the four other registers implemented per GCL:

1. Access the GCL and the four other GCL-linked registers via indirect addressing using MTL_EST_GCL_Control and MTL_EST_GCL_Data. MTL_EST_Status[SWOL] indicates whether the software owns GCL0 or GCL1.
2. Write the 32-bit write data to MTL_EST_GCL_Data, to program the GCL. Then program MTL_EST_GCL_Control to write the write address and other control information.
3. In MTL_EST_GCL_Data, write data consists of up to 8 bits (configurable) of gate controls and up to 24 bits (configurable) of time interval. Programming a 0 indicates gate close and programming a 1 indicates gate open. For a 4-TC and 20-bit time interval configuration, the data width is 24-bits and the remaining 8-bits are reserved/read-only. You must write the data in the following format.

   {8'h0, TC3, TC2, TC1, TC0, 20-bit Time Interval} where TCx = 0 or 1.
4. Program MTL_EST_GCL_Control[SRWO] to 1 (to start a Write Op) and program the address and R/W fields appropriately.
5. Poll and check that the hardware clears MTL_EST_GCL_Control[SRWO] to indicate that the previous operation completes before initiating a new read-write operation via the same indirect addressing mode.
6. Repeat steps 3, 4, 5 until the programming of the GCL completes.
7. Program the BTR, CTR, TER and LLR registers, using the same indirect addressing method as described above. Write 1 to MTL_EST_GCL_Control[GCRR] appropriately. MTL_EST_GCL_Control[GCRR] interprets the address field as belonging to these registers (instead of the GCL).

After the GCL and the related registers are programmed, program MTL_EST_GCL_Control to allow hardware to own and process the GCL. When the list length (as indicated in LLR) is 1, the associated time interval must be smaller than the cycle time register value. Otherwise, an error is reported (as described in the Error handling section) as a single set of gate controls add no value in the TSN context.

NOTE

The time unit in all the GCL related registers is seconds and nanoseconds. In cases where internally generated PTP system time is used, you must program the nanoseconds field to use the Digital Rollover mode. (The value of MAC_Timestamp_Control[TSCTRLSSR]must be 1)

75.16.11.2 Programming the EST registers

After the steps mentioned in "Programming the GCL and GCL Linked Registers" completes, program MTL_EST_Control

1. Write 1 to MTL_EST_Control[CTOV] and MTL_EST_Control[TILS]. Also, write 1 to MTL_EST_Control[EEST] and MTL_EST_Control[SWOL].

   NOTE

   The CTOV recommendations for GMII for SPRAM configurations are:

   - 96 * Tx clock period, for 32 and 64 bit data width configurations.
   - 128 * Tx clock period for 128 bit data width configurations.

   The CTOV recommendations for GMII for non-SPRAM configurations are:

   - 30 * Tx clock period, when SA/VLAN insertion is enabled.
   - 22 * Tx clock period, when SA/VLAN insertion is not enabled.

2. Enables the hardware to own and process the new GCL and make a switch to the new GCL at the BTR value. The hardware provides an interrupt (if enabled) when the switch to the new list happens.
3. Performs an appropriate action to address any other interrupts (explained in the Interrupts section) received during the hardware execution of the GCL.

Write 1 to MTL_EST_Control[SSWL] to handoff to hardware. Software is not allowed to write to the GCL and GCL linked registers when MTL_EST_Control[SSWL] = 1, because the hardware might be using the new GCL.

The hardware resets or clears MTL_EST_Control[SSWL] when it successfully switches to the new list. The hardware also flips MTL_EST_Status[SWOL] to indicate the new GCL that the software owns.

Program the GCL that software owns (MTL_EST_Status[SWOL] indicates) as described in "Programming the GCL and GCL Linked Registers" and then program MTL_EST_Control as described above, to install a new GCL. Ensure that the new BTR is set to an appropriate value to avoid BTR Error that may require software intervention in some cases.

The packet length (frame size) information must be available at all times, to avoid transmission overruns. Therefore, in the DMA configurations, program the packet length in the first descriptor of every transmit frame. Similarly, in the MTL configuration, provide the packet length in the control word.

NOTE

The packet length provided in the transmit descriptor must account for the SA and VLAN insertion, to avoid transmission overruns, if applicable, that is, packet length must represent the actual packet length on the Ethernet line.

75.16.12 Programming the launch time in time-based scheduling

Configure the launch time in the enhanced normal transmit descriptors in DMA configurations and is driven as a control word in MTL configurations as follows:

The OSTC and launch time features are mutually exclusive and must not be used together. In case of a conflict (if OSTC = LTV = 1 in MTL configuration), give priority to OSTC and ignore the launch time.

In DMA configuration, if a context descriptor is received with a valid OSTC values immediately before receiving a first normal descriptor with LTV = 1, then the LTV is ignored.

75.16.13 Programming guidelines for media clock generation and recovery

75.16.13.1 Programming guidelines for media clock generation

These are the guidelines for the media clock generation:

1. Program the appropriate presentation time control (supported generation modes "1001-1011") to PPSCTRL_PPSCMD (for 0th instance)/PPSCMD#i (for 1,2,3 instances) of MAC_PPS_Control, to define the PPS instance in Media clock generation mode.

2. Based on the selected PPS instance, application must drive the appropriate trigger signal to the corresponding MCG_PST_TRIG_I[#i].

3. Based on the programmed mode, DUT captures the timestamp and programs it in the MAC_PPS(#i)_Target_Time_Seconds register. Then, mcgr_dma_req_o[#i] is set.

4. For every request which DUT generates, the module reads the corresponding MAC_PPS(#i)_Target_Time_Seconds register and asserts the corresponding mcgr_d-ma_ack_i[#i] to acknowledge the read request.

75.16.13.2 Programming guidelines for media clock recovery

These are the guidelines for the media clock recovery:

- Writing 1 to MAC_Timestamp_Control[PTGE] enables the CPT counter. In addition to configuring the initialization values for the system time, update MAC_Presn_Time_Updt with the equivalent presentation time initialization value, then MAC_Timestamp_Control[TSINIT] = 1.
- Use the increment values for the system time and also for the CPT because the increment value is in sub-seconds and sub-nanoseconds.
- Update MAC_Presn_Time_Updt with the equivalent presentation time update value (32 bit ns), when an update is required, and also configure the updated values for the system time. Finally, MAC_Timestamp_Control[TSUPDT] must be 1 for the update.
- Program the MCGREN#i field of MAC_PPS_Control to enable the PPS instance to operate in MCGR mode.
- Program the appropriate presentation time control (supported recovery modes "0001 - 0011") to PPSCTRL_PPSCMD (for 0th instance) or PPSCMD#i (for 1,2,3 instances) of MAC_PPS_Control register, to set the PPS instance in Media clock recovery mode.
- After the PPS instance is set in Recovery mode, DUT sets the corresponding mcgr_dma_req_o[#i]. For every request which DUT generates, program the target presentation time into the MAC_PPS(#i)_Target_Time_Seconds register and assert the corresponding mcgr_dma_ack_i[#i] to acknowledge the request. Acknowledgment can be given before/after programming the Target presentation time.
- Observe the recovered media clock on the corresponding PTP_PPS_O[#i].

75.16.14 Programming guidelines for ECC protection for memories

These are the guidelines for the ECC protection for memories:

- Write 1 to the appropriate field of MTL_ECC_Control to enable the ECC features for the respective memory.
- Generate a correctable interrupt (sbd_sfty_ce_intr_o) or an uncorrectable interrupt (sbd_sfty_ue_intr_o) to indicate the application, if any correctable, uncorrectable, or address errors are detected. DMA_Interrupt_Status or MTL_ECC_Interrupt_Status indicates an appropriate status.
- Provide debug mode for each memory to specify errors.

NOTE

Enable the ECC feature before the traffic is online (or after the reset), otherwise the false interrupts may trigger.

75.16.14.1 Programming guidelines for ECC hardware error injection (Debug mode)

Follow these steps for ECC hardware error injection:

- Write 1 to the appropriate field in MTL_DBG_CTL to enable the ECC error injection feature for MTL TX/RX and DMA TSO.
- Write 1 to the appropriate field in MTL_EST_GCL_Control to enable the ECC error injection feature for the EST memory.
- Write 1 to the appropriate field in MTL_RXP_Indirect_Acc_Control_Status to enable the ECC error injection for the receive parser memory. To access the receive parser memory in Debug mode, write 0 to MTL_RXP_Indirect_Acc_Control_Status[FRPE] to disable the receive parser feature.
- Ensure that all the CSR writes corresponding to one memory write must have the same value for the error injection control word, where multiple CSR writes are required for writing single data word into the memory.

75.16.15 Programming guidelines for on-chip datapath parity protection

Follow these steps for on-chip datapath parity protection:

- Writing 1 to MTL_DPP_Control[EDPP] enables the parity generation and parity detection mismatch on datapath.
- Generates an uncorrectable interrupt (sbd_sfty_ue_intr_o) to indicate the application if any parity mismatch is detected. DMA_Safety_Interrupt_Status or MTL_Safety_Interrupt_Status and MAC_DPP_FSM_Interrupt_Status indicates the appropriate status.
- Supports Debug mode for each parity generator to insert an error.

NOTE

Enable the datapath parity protection before the receive or transit traffic starts, to avoid the false safety interrupts.

75.16.16 Programming guidelines for FSM parity and timeout

Follow these steps to configure the FSM parity and timeout:

- Write 1 to MAC_FSM_Control[PRTYEN] and MAC_FSM_Control[TMOUTEN] to enable the FSM parity and timeout feature respectively.
- Force the FSMs to enter into a state with even number of 1s and wait for the safety uncorrectable interrupt to define with MAC_DPP_FSM_Interrupt_Status[FSMPES] for a parity error detection.
- Write 1 to the parity error injection using the [23:16] bits of MAC_FSM_Control to select the appropriate clock domain, for Error injection mode in FSM parity. Wait for the safety uncorrectable interrupt to be defined with MAC_DPP_FSM_Interrupt_Status[FSMPES].
- Configure the large and normal mode values which MAC_FSM_ACT_Timer[LTMRMD] and MAC_FSM_ACT_Timer[NTMRMD] indicates , for FSM timeouts.
- Use the [31:24] bits of MAC_FSM_Control to select the large or normal mode tic generation per clock domain.
- Write 1 to MAC_FSM_ACT_Timer[TMR] with the appropriate number of CSR clock cycles to generate 1us tic.
- Force the FSM in a particular clock domain to remain in an active state for timeout duration (for example, T-2T). You can check the safety interrupt and the status field for the corresponding clock domain that sets in the [15:8] bits of the MAC_DPP_FSM_Interrupt_Status .
- Select the appropriate clock domain for which timeout error injection must be set using the [15:8] bits of MAC_FSM_Control, for Error injection mode in FSM timeout. Wait for the safety uncorrectable interrupt to define with the corresponding error status in the MAC_DPP_FSM_Interrupt_Status .
- Use only normal mode tic generation, for application or CSR interface timeout. On the basis of the time-outs of configured interface, appropriate fields (MSTTES, SLVTES) of MAC_DPP_FSM_Interrupt_Status is defined with the safety interrupt.

75.17 EMAC register descriptions

This section and all its sub-sections are Synopsys Proprietary. Used with permission.

NOTE

This chip has 16 KB allocated per Ethernet controller for configuration space. EMAC uses only 8 KB of this memory and the remaining 8 KB is reserved for future use. Writing to the reserved memory may lead to unintended behavior.

75.17.1 EMAC memory map

EMAC base address: 4048_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	MAC Configuration (MAC_Configuration)	32	RW	0000_8000h
4h	MAC Extended Configuration (MAC_Ext_Configuration)	32	RW	0000_0000h
8h	MAC Packet Filter (MAC_Packet_Filter)	32	RW	0000_0000h
Ch	MAC Watchdog Timeout (MAC_Watchdog_Timeout)	32	RW	0000_0000h
10h	MAC Hash Table First 32 Bits (MAC_Hash_Table_Reg0)	32	RW	0000_0000h
14h	MAC Hash Table Second 32 Bits (MAC_Hash_Table_Reg1)	32	RW	0000_0000h
50h	MAC VLAN Tag (MAC_VLAN_Tag)	32	RW	0000_0000h
50h	MAC VLAN Tag Control (MAC_VLAN_Tag_Ctrl)	32	RW	0000_0000h
54h	MAC VLAN Tag Data (MAC_VLAN_Tag_Data)	32	RW	0000_0000h
54h	MAC VLAN Tag Filter 0 (MAC_VLAN_Tag_Filter0)	32	RW	0000_0000h
54h	MAC VLAN Tag Filter 1 (MAC_VLAN_Tag_Filter1)	32	RW	0000_0000h
54h	MAC VLAN Tag Filter 2 (MAC_VLAN_Tag_Filter2)	32	RW	0000_0000h
54h	MAC VLAN Tag Filter 3 (MAC_VLAN_Tag_Filter3)	32	RW	0000_0000h
58h	MAC VLAN Hash Table (MAC_VLAN_Hash_Table)	32	RW	0000_0000h
60h	MAC VLAN Inclusion Or Replacement (MAC_VLAN_Incl)	32	RW	0000_0000h
60h	MAC VLAN Inclusion 0 (MAC_VLAN_Incl0)	32	RW	0000_0000h
60h	MAC VLAN Inclusion 1 (MAC_VLAN_Incl1)	32	RW	0000_0000h
60h	MAC VLAN Inclusion 2 (MAC_VLAN_Incl2)	32	RW	0000_0000h
60h	MAC VLAN Inclusion 3 (MAC_VLAN_Incl3)	32	RW	0000_0000h
60h	MAC VLAN Inclusion 4 (MAC_VLAN_Incl4)	32	RW	0000_0000h
60h	MAC VLAN Inclusion 5 (MAC_VLAN_Incl5)	32	RW	0000_0000h
60h	MAC VLAN Inclusion 6 (MAC_VLAN_Incl6)	32	RW	0000_0000h
60h	MAC VLAN Inclusion 7 (MAC_VLAN_Incl7)	32	RW	0000_0000h
64h	Inner VLAN Tag Inclusion Or Replacement (MAC_Inner_VLAN_Incl)	32	RW	0000_0000h
70h	MAC Q0 Tx Flow Control (MAC_Q0_Tx_Flow_Ctrl)	32	RW	0000_0000h
90h	MAC Receive Flow Control (MAC_Rx_Flow_Ctrl)	32	RW	0000_0000h
94h	MAC RxQ Control 4 (MAC_RxQ_Ctrl4)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
A0h	MAC RxQ Control 0 (MAC_RxQ_Ctrl0)	32	RW	0000_0000h
A4h	Receive Queue Control 1 (MAC_RxQ_Ctrl1)	32	RW	0000_0000h
A8h	MAC RxQ Control 2 (MAC_RxQ_Ctrl2)	32	RW	0000_0000h
B0h	MAC Interrupt Status (MAC_Interrupt_Status)	32	R	0000_0000h
B4h	MAC Interrupt Enable (MAC_Interrupt_Enable)	32	RW	0000_0000h
B8h	MAC Rx Transmit Status (MAC_Rx_Tx_Status)	32	R	0000_0000h
110h	MAC Version (MAC_Version)	32	R	0000_1051h
114h	MAC Debug (MAC_Debug)	32	R	0000_0000h
11Ch	MAC Hardware Feature 0 (MAC_HW_Feature0)	32	R	0E09_5135h
120h	MAC Hardware Feature 1 (MAC_HW_Feature1)	32	R	2118_29A6h
124h	MAC Hardware Feature 2 (MAC_HW_Feature2)	32	R	0404_1041h
128h	MAC Hardware Feature 3 (MAC_HW_Feature3)	32	R	2C37_0E31h
140h	MAC DPP FSM Interrupt Status (MAC_DPP_FSM_Interrupt_Status)	32	R	0000_0000h
148h	MAC FSM Control (MAC_FSM_Control)	32	RW	0000_0000h
14Ch	MAC FSM ACT Timer (MAC_FSM_ACT_Timer)	32	RW	0000_0000h
150h	SCS_REG 1 (SCS_REG1)	32	RW	0000_0000h
200h	MAC MDIO Address (MAC_MDIO_Address)	32	RW	0000_0000h
204h	MAC MDIO Data (MAC_MDIO_Data)	32	RW	0000_0000h
230h	MAC CSR Software Control (MAC_CSR_SW_Ctrl)	32	RW	0000_0000h
234h	MAC FPE Control STS (MAC_FPE_CTRL_STS)	32	RW	0000_0000h
240h	MAC Presentation Time (MAC_Presn_Time_ns)	32	R	0000_0000h
244h	MAC Presentation Time Update (MAC_Presn_Time_Updt)	32	RW	0000_0000h
300h	MAC Address 0 High (MAC_Address0_High)	32	RW	8000_FFFFh
304h	MAC Address 0 Low (MAC_Address0_Low)	32	RW	FFFF_FFFFh
308h	MAC Address 1 High (MAC_Address1_High)	32	RW	0000_FFFFh
30Ch	MAC Address 1 Low (MAC_Address1_Low)	32	RW	FFFF_FFFFh
310h	MAC Address 2 High (MAC_Address2_High)	32	RW	0000_FFFFh
314h	MAC Address 2 Low (MAC_Address2_Low)	32	RW	FFFF_FFFFh
700h	MMC Control (MMC_Control)	32	RW	0000_0000h
704h	MMC Receive Interrupt (MMC_Rx_Interrupt)	32	R	0000_0000h
708h	MMC Transmit Interrupt (MMC_Tx_Interrupt)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
70Ch	MMC Receive Interrupt Mask (MMC_Rx_Interrupt_Mask)	32	RW	0000_0000h
710h	MMC Transmit Interrupt Mask (MMC_Tx_Interrupt_Mask)	32	RW	0000_0000h
714h	Transmit Octet Count Good Bad (Tx_Octet_Count_Good_Bad)	32	R	0000_0000h
718h	Transmit Packet Count Good Bad (Tx_Packet_Count_Good_Bad)	32	R	0000_0000h
71Ch	Transmit Broadcast Packets Good (Tx_Broadcast_Packets_Good)	32	R	0000_0000h
720h	Transmit Multicast Packets Good (Tx_Multicast_Packets_Good)	32	R	0000_0000h
724h	Transmit 64-Octet Packets Good Bad (Tx_64Octets_Packets_Good_Bad)	32	R	0000_0000h
728h	Transmit 65 To 127 Octet Packets Good Bad (Tx_65To127Octets_Packets_Good_Bad)	32	R	0000_0000h
72Ch	Transmit 128 To 255 Octet Packets Good Bad (Tx_128To255Octets_Packets_Good_Bad)	32	R	0000_0000h
730h	Transmit 256 To 511 Octet Packets Good Bad (Tx_256To511Octets_Packets_Good_Bad)	32	R	0000_0000h
734h	Transmit 512 To 1023 Octet Packets Good Bad (Tx_512To1023Octets_Packets_Good_Bad)	32	R	0000_0000h
738h	Transmit 1024 To Max Octet Packets Good Bad (Tx_1024ToMaxOctets_Packets_Good_Bad)	32	R	0000_0000h
73Ch	Transmit Unicast Packets Good Bad (Tx_Unicast_Packets_Good_Bad)	32	R	0000_0000h
740h	Transmit Multicast Packets Good Bad (Tx_Multicast_Packets_Good_Bad)	32	R	0000_0000h
744h	Transmit Broadcast Packets Good Bad (Tx_Broadcast_Packets_Good_Bad)	32	R	0000_0000h
748h	Transmit Underflow Error Packets (Tx_Underflow_Error_Packets)	32	R	0000_0000h
74Ch	Transmit Single Collision Good Packets (Tx_Single_Collision_Good_Packets)	32	R	0000_0000h
750h	Transmit Multiple Collision Good Packets (Tx_Multiple_Collision_Good_Packets)	32	R	0000_0000h
754h	Transmit Deferred Packets (Tx_Deferred_Packets)	32	R	0000_0000h
758h	Transmit Late Collision Packets (Tx_Late_Collision_Packets)	32	R	0000_0000h
75Ch	Transmit Excessive Collision Packets (Tx_Excessive_Collision_Packets)	32	R	0000_0000h
760h	Transmit Carrier Error Packets (Tx_Carrier_Error_Packets)	32	R	0000_0000h
764h	Transmit Octet Count Good (Tx_Octet_Count_Good)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
768h	Transmit Packet Count Good (Tx_Packet_Count_Good)	32	R	0000_0000h
76Ch	Transmit Excessive Deferral Error (Tx_Excessive_Deferral_Error)	32	R	0000_0000h
770h	Transmit Pause Packets (Tx_Pause_Packets)	32	R	0000_0000h
774h	Transmit VLAN Packets Good (Tx_VLAN_Packets_Good)	32	R	0000_0000h
778h	Transmit O Size Packets Good (Tx_OSize_Packets_Good)	32	R	0000_0000h
780h	Receive Packets Count Good Bad (Rx_Packets_Count_Good_Bad)	32	R	0000_0000h
784h	Receive Octet Count Good Bad (Rx_Octet_Count_Good_Bad)	32	R	0000_0000h
788h	Receive Octet Count Good (Rx_Octet_Count_Good)	32	R	0000_0000h
78Ch	Receive Broadcast Packets Good (Rx_Broadcast_Packets_Good)	32	R	0000_0000h
790h	Receive Multicast Packets Good (Rx_Multicast_Packets_Good)	32	R	0000_0000h
794h	Receive CRC Error Packets (Rx_CRC_Error_Packets)	32	R	0000_0000h
798h	Receive Alignment Error Packets (Rx_Alignment_Error_Packets)	32	R	0000_0000h
79Ch	Receive Runt Error Packets (Rx_Runt_Error_Packets)	32	R	0000_0000h
7A0h	Receive Jabber Error Packets (Rx_Jabber_Error_Packets)	32	R	0000_0000h
7A4h	Receive Undersize Packets Good (Rx_Undersize_Packets_Good)	32	R	0000_0000h
7A8h	Receive Oversize Packets Good (Rx_Oversize_Packets_Good)	32	R	0000_0000h
7ACh	Receive 64 Octets Packets Good Bad (Rx_64Octets_Packets_Good_Bad)	32	R	0000_0000h
7B0h	Receive 65-127 Octets Packets Good Bad (Rx_65To127Octets_Packets_Good_Bad)	32	R	0000_0000h
7B4h	Receive 128-255 Octets Packets Good Bad (Rx_128To255Octets_Packets_Good_Bad)	32	R	0000_0000h
7B8h	Receive 256-511 Octets Packets Good Bad (Rx_256To511Octets_Packets_Good_Bad)	32	R	0000_0000h
7BCh	Receive 512-1023 Octets Packets Good Bad (Rx_512To1023Octets_Packets_Good_Bad)	32	R	0000_0000h
7C0h	Receive 1024 To Max Octets Good Bad (Rx_1024ToMaxOctets_Packets_Good_Bad)	32	R	0000_0000h
7C4h	Receive Unicast Packets Good (Rx_Unicast_Packets_Good)	32	R	0000_0000h
7C8h	Receive Length Error Packets (Rx_Length_Error_Packets)	32	R	0000_0000h
7CCh	Receive Out of Range Type Packet (Rx_Out_Of_Range_Type_Packets)	32	R	0000_0000h
7D0h	Receive Pause Packets (Rx_Pause_Packets)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
7D4h	Receive FIFO Overflow Packets (Rx_FIFO_Overflow_Packets)	32	R	0000_0000h
7D8h	Receive VLAN Packets Good Bad (Rx_VLAN_Packets_Good_Bad)	32	R	0000_0000h
7DCh	Receive Watchdog Error Packets (Rx_Watchdog_Error_Packets)	32	R	0000_0000h
7E0h	Receive Error Packets (Rx_Receive_Error_Packets)	32	R	0000_0000h
7E4h	Receive Control Packets Good (Rx_Control_Packets_Good)	32	R	0000_0000h
8A0h	MMC Transmit FPE Fragment Counter Interrupt Status (MMC_FPE_Tx_Interrupt)	32	R	0000_0000h
8A4h	MMC FPE Transmit Interrupt Mask (MMC_FPE_Tx_Interrupt_Mask)	32	RW	0000_0000h
8A8h	Transmit FPE Fragment Counter (MMC_Tx_FPE_Fragment_Cntr)	32	R	0000_0000h
8ACh	Transmit Hold Request Counter (MMC_Tx_Hold_Req_Cntr)	32	R	0000_0000h
8C0h	MMC Receive Packet Assembly Error Counter Interrupt Status (MMC_FPE_Rx_Interrupt)	32	R	0000_0000h
8C4h	MMC FPE Receive Interrupt Mask (MMC_FPE_Rx_Interrupt_Mask)	32	RW	0000_0000h
8C8h	MMC Receive Packet Assembly Error Counter (MMC_Rx_Packet_Assembly_Err_Cntr)	32	R	0000_0000h
8CCh	MMC Receive Packet SMD Error Counter (MMC_Rx_Packet_SMD_Err_Cntr)	32	R	0000_0000h
8D0h	MMC Receive Packet Assembly OK Counter (MMC_Rx_Packet_Assembly_OK_Cntr)	32	R	0000_0000h
8D4h	MMC Receive FPE Fragment Counter (MMC_Rx_FPE_Fragment_Cntr)	32	R	0000_0000h
900h	MAC Layer 3 Layer 4 Control 0 (MAC_L3_L4_Control0)	32	RW	0000_0000h
904h	MAC Layer 4 Address 0 (MAC_Layer4_Address0)	32	RW	0000_0000h
910h	MAC Layer 3 Address 0 Reg 0 (MAC_Layer3_Addr0_Reg0)	32	RW	0000_0000h
914h	MAC Layer 3 Address 1 Reg 0 (MAC_Layer3_Addr1_Reg0)	32	RW	0000_0000h
918h	MAC Layer 3 Address 2 Reg 0 (MAC_Layer3_Addr2_Reg0)	32	RW	0000_0000h
91Ch	MAC Layer 3 Address 3 Reg 0 (MAC_Layer3_Addr3_Reg0)	32	RW	0000_0000h
930h	MAC L3 L4 Control 1 (MAC_L3_L4_Control1)	32	RW	0000_0000h
934h	MAC Layer 4 Address 1 (MAC_Layer4_Address1)	32	RW	0000_0000h
940h	MAC Layer 3 Address 0 Reg 1 (MAC_Layer3_Addr0_Reg1)	32	RW	0000_0000h
944h	MAC Layer 3 Address 1 Reg 1 (MAC_Layer3_Addr1_Reg1)	32	RW	0000_0000h
948h	MAC Layer 3 Address 2 Reg 1 (MAC_Layer3_Addr2_Reg1)	32	RW	0000_0000h
94Ch	MAC Layer 3 Address 3 Reg 1 (MAC_Layer3_Addr3_Reg1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
960h	MAC L3 L4 Control 2 (MAC_L3_L4_Control2)	32	RW	0000_0000h
964h	MAC Layer 4 Address 2 (MAC_Layer4_Address2)	32	RW	0000_0000h
970h	MAC Layer 3 Address 0 Reg 2 (MAC_Layer3_Addr0_Reg2)	32	RW	0000_0000h
974h	MAC Layer 3 Address 1 Reg 2 (MAC_Layer3_Addr1_Reg2)	32	RW	0000_0000h
978h	MAC Layer 3 Address 2 Reg 2 (MAC_Layer3_Addr2_Reg2)	32	RW	0000_0000h
97Ch	MAC Layer 3 Address 3 Reg 2 (MAC_Layer3_Addr3_Reg2)	32	RW	0000_0000h
990h	MAC L3 L4 Control 3 (MAC_L3_L4_Control3)	32	RW	0000_0000h
994h	MAC Layer 4 Address 3 (MAC_Layer4_Address3)	32	RW	0000_0000h
9A0h	MAC Layer 3 Address 0 Reg 3 (MAC_Layer3_Addr0_Reg3)	32	RW	0000_0000h
9A4h	MAC Layer 3 Address 1 Reg 3 (MAC_Layer3_Addr1_Reg3)	32	RW	0000_0000h
9A8h	MAC Layer 3 Address 2 Reg 3 (MAC_Layer3_Addr2_Reg3)	32	RW	0000_0000h
9ACh	MAC Layer 3 Address 3 Reg 3 (MAC_Layer3_Addr3_Reg3)	32	RW	0000_0000h
B00h	MAC Timestamp Control (MAC_Timestamp_Control)	32	RW	0000_2000h
B04h	MAC Sub Second Increment (MAC_Sub_Second_Increment)	32	RW	0000_0000h
B08h	MAC System Time In Seconds (MAC_System_Time_Seconds)	32	R	0000_0000h
B0Ch	MAC System Time In Nanoseconds (MAC_System_Time_Nanoseconds)	32	R	0000_0000h
B10h	MAC System Time Seconds Update (MAC_System_Time_Seconds_Update)	32	RW	0000_0000h
B14h	MAC System Time Nanoseconds Update (MAC_System_Time_Nanoseconds_Update)	32	RW	0000_0000h
B18h	MAC Timestamp Addend (MAC_Timestamp_Addend)	32	RW	0000_0000h
B1Ch	MAC System Time Higher Word In Seconds (MAC_System_Time_Higher_Word_Seconds)	32	RW	0000_0000h
B20h	MAC Timestamp Status (MAC_Timestamp_Status)	32	R	0000_0000h
B30h	MAC Transmit Timestamp Status In Nanoseconds (MAC_Tx_Timestamp_Status_Nanoseconds)	32	R	0000_0000h
B34h	MAC Transmit Timestamp Status In Seconds (MAC_Tx_Timestamp_Status_Seconds)	32	R	0000_0000h
B50h	MAC Timestamp Ingress Asymmetry Correction (MAC_Timestamp_Ingress_Asym_Corr)	32	RW	0000_0000h
B54h	MAC Timestamp Egress Asymmetry Correction (MAC_Timestamp_Egress_Asym_Corr)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
B58h	MAC Timestamp Ingress Correction In Nanoseconds (MAC_Timestamp_Ingress_Corr_Nanosecond)	32	RW	0000_0000h
B5Ch	MAC Timestamp Egress Correction In Nanoseconds (MAC_Timestamp_Egress_Corr_Nanosecond)	32	RW	0000_0000h
B60h	MAC Timestamp Ingress Correction In Subnanoseconds (MAC_Timestamp_Ingress_Corr_Subnanosec)	32	RW	0000_0000h
B64h	MAC Timestamp Engress Correction In Subnanoseconds (MAC_Timestamp_Egress_Corr_Subnanosec)	32	RW	0000_0000h
B68h	MAC Timestamp Ingress Latency (MAC_Timestamp_Ingress_Latency)	32	R	0000_0000h
B6Ch	MAC Timestamp Egress Latency (MAC_Timestamp_Egress_Latency)	32	R	0000_0000h
B70h	MAC PPS Control (MAC_PPS_Control)	32	RW	0000_0000h
B80h	MAC PPS0 Target Time In Seconds (MAC_PPS0_Target_Time_Seconds)	32	RW	0000_0000h
B84h	MAC PPS0 Target Time In Nanoseconds (MAC_PPS0_Target_Time_Nanoseconds)	32	RW	0000_0000h
B88h	MAC PPS0 Interval (MAC_PPS0_Interval)	32	RW	0000_0000h
B8Ch	MAC PPS0 Width (MAC_PPS0_Width)	32	RW	0000_0000h
B90h	MAC PPS1 Target Time In Seconds (MAC_PPS1_Target_Time_Seconds)	32	RW	0000_0000h
B94h	MAC PPS1 Target Time In Nanoseconds (MAC_PPS1_Target_Time_Nanoseconds)	32	RW	0000_0000h
B98h	MAC PPS1 Interval (MAC_PPS1_Interval)	32	RW	0000_0000h
B9Ch	MAC PPS1 Width (MAC_PPS1_Width)	32	RW	0000_0000h
BA0h	MAC PPS2 Target Time In Seconds (MAC_PPS2_Target_Time_Seconds)	32	RW	0000_0000h
BA4h	MAC PPS2 Target Time In Nanoseconds (MAC_PPS2_Target_Time_Nanoseconds)	32	RW	0000_0000h
BA8h	MAC PPS2 Interval (MAC_PPS2_Interval)	32	RW	0000_0000h
BACh	MAC PPS2 Width (MAC_PPS2_Width)	32	RW	0000_0000h
BB0h	MAC PPS3 Target Time In Seconds (MAC_PPS3_Target_Time_Seconds)	32	RW	0000_0000h
BB4h	MAC PPS3 Target Time In Nanoseconds (MAC_PPS3_Target_Time_Nanoseconds)	32	RW	0000_0000h
BB8h	MAC PPS3 Interval (MAC_PPS3_Interval)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
BBCh	MAC PPS3 Width (MAC_PPS3_Width)	32	RW	0000_0000h
C00h	MTL Operation Mode (MTL_Operation_Mode)	32	RW	0000_0000h
C08h	MTL Debug Control (MTL_DBG_CTL)	32	RW	0000_0000h
C0Ch	MTL Debug Status (MTL_DBG_STS)	32	RW	0000_0018h
C10h	MTL FIFO Debug Data (MTL_FIFO_Debug_Data)	32	RW	0000_0000h
C20h	MTL Interrupt Status (MTL_Interrupt_Status)	32	R	0000_0000h
C30h	MTL Receive Queue DMA Map 0 (MTL_RxQ_DMA_Map0)	32	RW	0000_0000h
C40h	MTL TBS Control (MTL_TBS_CTRL)	32	RW	0000_0000h
C50h	MTL EST Control (MTL_EST_Control)	32	RW	0000_0000h
C58h	MTL EST Status (MTL_EST_Status)	32	RW	0000_0000h
C60h	MTL EST Scheduling Error (MTL_EST_Sch_Error)	32	RW	0000_0000h
C64h	MTL EST Frame Size Error (MTL_EST_Frm_Size_Error)	32	RW	0000_0000h
C68h	MTL EST Frame Size Capture (MTL_EST_Frm_Size_Capture)	32	R	0000_0000h
C70h	MTL EST Interrupt Enable (MTL_EST_Intr_Enable)	32	RW	0000_0000h
C80h	MTL EST GCL Control (MTL_EST_GCL_Control)	32	RW	0000_0000h
C84h	MTL EST GCL Data (MTL_EST_GCL_Data)	32	RW	0000_0000h
C90h	MTL FPE Control Status (MTL_FPE_CTRL_STS)	32	RW	0000_0000h
C94h	MTL FPE Advance (MTL_FPE_Advance)	32	RW	0000_0000h
CA0h	MTL Rx Parser Control Status (MTL_RXP_Control_Status)	32	RW	803F_003Fh
CA4h	MTL Rx Parser Interrupt Control Status (MTL_RXP_Interrupt_Control_Status)	32	RW	0000_0000h
CA8h	MTL Rx Parser Drop Count (MTL_RXP_Drop_Cnt)	32	R	0000_0000h
CACh	MTL Rx Parser Error Count (MTL_RXP_Error_Cnt)	32	R	0000_0000h
CB0h	MTL Rx Parser Indirect Access Control Status (MTL_RXP_Indirect_Acc_Control_Status)	32	RW	0000_0000h
CB4h	MTL Rx Parser Indirect Access Data (MTL_RXP_Indirect_Acc_Data)	32	R	0000_0000h
CC0h	MTL ECC Control (MTL_ECC_Control)	32	RW	0000_000Fh
CC4h	MTL Safety Interrupt Status (MTL_Safety_Interrupt_Status)	32	R	0000_0000h
CC8h	MTL ECC Interrupt Enable (MTL_ECC_Interrupt_Enable)	32	RW	0000_1111h
CCCh	MTL ECC Interrupt Status (MTL_ECC_Interrupt_Status)	32	RW	0000_0000h
CD0h	MTL ECC Error Status (MTL_ECC_Err_Sts_Rctl)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
CD4h	MTL ECC Error Address Status (MTL_ECC_Err_Addr_Status)	32	R	0000_0000h
CD8h	MTL ECC Error Control Status (MTL_ECC_Err_Cntr_Status)	32	R	0000_0000h
CE0h	MTL DPP Control (MTL_DPP_Control)	32	RW	0000_0003h
D00h	MTL Tx Queue 0 Operation Mode (MTL_TxQ0_Operation_Mode)	32	RW	0000_0000h
D04h	MTL Tx Queue 0 Underflow (MTL_TxQ0_Underflow)	32	R	0000_0000h
D08h	MTL Tx Queue 0 Debug (MTL_TxQ0_Debug)	32	R	0000_0000h
D14h	MTL Tx Queue 0 ETS Status (MTL_TxQ0_ETS_Status)	32	R	0000_0000h
D18h	MTL Tx Queue Quantum Weight (MTL_TxQ0_Quantum_Weight)	32	RW	0000_0000h
D2Ch	MTL Queue 0 Interrupt Control Status (MTL_Q0_Interrupt_Control_Status)	32	RW	0000_0000h
D30h	MTL Rx Queue 0 Operation Mode (MTL_RxQ0_Operation_Mode)	32	RW	0000_0000h
D34h	MTL Rx Queue Missed Packet Overflow Count (MTL_RxQ0_Missed_Packet_Overflow_Cnt)	32	R	0000_0000h
D38h	MTL Rx Queue 0 Debug (MTL_RxQ0_Debug)	32	R	0000_0000h
D3Ch	MTL Rx Queue 0 Control 0 (MTL_RxQ0_Control)	32	RW	0000_0000h
D40h	MTL Tx Queue 1 Operation Mode (MTL_TxQ1_Operation_Mode)	32	RW	0000_0000h
D44h	MTL Tx Queue 1 Underflow (MTL_TxQ1_Underflow)	32	R	0000_0000h
D48h	MTL Tx Queue 1 Debug (MTL_TxQ1_Debug)	32	R	0000_0000h
D50h	MTL Tx Queue 1 ETS Control (MTL_TxQ1_ETS_Control)	32	RW	0000_0000h
D54h	MTL Tx Queue 1 ETS Status (MTL_TxQ1_ETS_Status)	32	R	0000_0000h
D58h	MTL Tx Queue 1 Quantum Weight (MTL_TxQ1_Quantum_Weight)	32	RW	0000_0000h
D5Ch	MTL Tx Queue 1 Sendslope Credit (MTL_TxQ1_SendSlopeCredit)	32	RW	0000_0000h
D60h	MTL Tx Queue 1 HiCredit (MTL_TxQ1_HiCredit)	32	RW	0000_0000h
D64h	MTL Tx Queue 1 LoCredit (MTL_TxQ1_LoCredit)	32	RW	0000_0000h
D6Ch	MTL Queue 1 Interrupt Control Status (MTL_Q1_Interrupt_Control_Status)	32	RW	0000_0000h
D70h	MTL Rx Queue 1 Operation Mode (MTL_RxQ1_Operation_Mode)	32	RW	0000_0000h
D74h	MTL Rx Queue 1 Missed Packet Overflow Counter (MTL_RxQ1_Missed_Packet_Overflow_Cnt)	32	R	0000_0000h
D78h	MTL Rx Queue 1 Debug (MTL_RxQ1_Debug)	32	R	0000_0000h
D7Ch	MTL Rx Queue 1 Control (MTL_RxQ1_Control)	32	RW	0000_0000h
1000h	DMA Mode (DMA_Mode)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1004h	DMA System Bus Mode (DMA_SysBus_Mode)	32	RW	0000_0000h
1008h	DMA Interrupt Status (DMA_Interrupt_Status)	32	R	0000_0000h
100Ch	DMA Debug Status 0 (DMA_Debug_Status0)	32	R	0000_0000h
1050h	DMA TBS Control (DMA_TBS_CTRL)	32	RW	0000_0000h
1080h	DMA Safety Interrupt Status (DMA_Safety_Interrupt_Status)	32	R	0000_0000h
1100h	DMA Channel 0 Control (DMA_CH0_Control)	32	RW	0000_0000h
1104h	DMA Channel Tx Control (DMA_CH0_Tx_Control)	32	RW	0000_0000h
1108h	DMA Channel Rx Control (DMA_CH0_Rx_Control)	32	RW	0000_0000h
1114h	DMA Channel 0 Tx Descriptor List Address (DMA_CH0_TxDesc_List_Address)	32	RW	0000_0000h
111Ch	DMA Channel 0 Rx Descriptor List Address (DMA_CH0_RxDesc_List_Address)	32	RW	0000_0000h
1120h	DMA Channel 0 Tx Descriptor Tail Pointer (DMA_CH0_TxDesc_Tail_Pointer)	32	RW	0000_0000h
1128h	DMA Channel 0 Rx Descriptor List Pointer (DMA_CH0_RxDesc_Tail_Pointer)	32	RW	0000_0000h
112Ch	DMA Channel 0 Tx Descriptor Ring Length (DMA_CH0_TxDesc_Ring_Length)	32	RW	0000_0000h
1130h	DMA Channel 0 Rx Descriptor Ring Length (DMA_CH0_RxDesc_Ring_Length)	32	RW	0000_0000h
1134h	DMA Channel 0 Interrupt Enable (DMA_CH0_Interrupt_Enable)	32	RW	0000_0000h
1138h	DMA Channel 0 Rx Interrupt Watchdog Timer (DMA_CH0_Rx_Interrupt_Watchdog_Timer)	32	RW	0000_0000h
113Ch	DMA Channel 0 Slot Function Control Status (DMA_CH0_Slot_Function_Control_Status)	32	RW	0000_07C0h
1144h	DMA Channel 0 Current Application Transmit Descriptor (DMA_CH0_Current_App_TxDesc)	32	R	0000_0000h
114Ch	DMA Channel 0 Current Application Receive Descriptor (DMA_CH0_Current_App_RxDesc)	32	R	0000_0000h
1154h	DMA Channel 0 Current Application Transmit Descriptor (DMA_CH0_Current_App_TxBuffer)	32	R	0000_0000h
115Ch	DMA Channel 0 Current Application Receive Buffer (DMA_CH0_Current_App_RxBuffer)	32	R	0000_0000h
1160h	DMA Channel 0 Status (DMA_CH0_Status)	32	RW	0000_0000h
1164h	DMA Channel 0 Miss Frame Counter (DMA_CH0_Miss_Frame_Cnt)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1168h	DMA Channel 0 Rx Parser Accept Count (DMA_CH0_RXP_Accept_Cnt)	32	R	0000_0000h
116Ch	DMA Channel 0 Rx ERI Count (DMA_CH0_RX_ERI_Cnt)	32	R	0000_0000h
1180h	DMA Channel 1 Control (DMA_CH1_Control)	32	RW	0000_0000h
1184h	DMA Channel 1 Tx Control (DMA_CH1_Tx_Control)	32	RW	0000_0000h
1188h	DMA Channel 1 Rx Control (DMA_CH1_Rx_Control)	32	RW	0000_0000h
1194h	DMA Channel 1 Tx Descriptor List Address (DMA_CH1_TxDesc_List_Address)	32	RW	0000_0000h
119Ch	DMA Channel 1 Rx Descriptor List Address (DMA_CH1_RxDesc_List_Address)	32	RW	0000_0000h
11A0h	DMA Channel 1 Tx Descriptor Tail Pointer (DMA_CH1_TxDesc_Tail_Pointer)	32	RW	0000_0000h
11A8h	DMA Channel 1 Rx Descriptor Tail Pointer (DMA_CH1_RxDesc_Tail_Pointer)	32	RW	0000_0000h
11ACh	DMA Channel 1 Tx Descriptor Ring Length (DMA_CH1_TxDesc_Ring_Length)	32	RW	0000_0000h
11B0h	DMA Channel 1 Rx Descriptor Ring Length (DMA_CH1_RxDesc_Ring_Length)	32	RW	0000_0000h
11B4h	DMA Channel 1 Interrupt Enable (DMA_CH1_Interrupt_Enable)	32	RW	0000_0000h
11B8h	DMA Channel 1 Rx Interrupt Watchdog Timer (DMA_CH1_Rx_Interrupt_Watchdog_Timer)	32	RW	0000_0000h
11BCh	DMA Channel 1 Slot Function Control Status (DMA_CH1_Slot_Function_Control_Status)	32	RW	0000_07C0h
11C4h	DMA Channel 1 Current Application Transmit Descriptor (DMA_CH1_Current_App_TxDesc)	32	R	0000_0000h
11CCh	DMA Channel 1 Current Application Receive Descriptor (DMA_CH1_Current_App_RxDesc)	32	R	0000_0000h
11D4h	DMA Channel 1 Current Application Transmit Buffer (DMA_CH1_Current_App_TxBuffer)	32	R	0000_0000h
11DCh	DMA Channel 1 Current Application Receive Buffer (DMA_CH1_Current_App_RxBuffer)	32	R	0000_0000h
11E0h	DMA Channel 1 Status (DMA_CH1_Status)	32	RW	0000_0000h
11E4h	DMA Channel 1 Miss Frame Counter (DMA_CH1_Miss_Frame_Cnt)	32	R	0000_0000h
11E8h	DMA Channel 1 Rx Parser Accept Count (DMA_CH1_RXP_Accept_Cnt)	32	R	0000_0000h
11ECh	DMA Channel 1 Rx ERI Count (DMA_CH1_RX_ERI_Cnt)	32	R	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value

75.17.2 MAC Configuration (MAC_Configuration)

Offset

Register	Offset
MAC_Configuration	0h

Function

Establishes MAC's operating mode.

Table 616. Packet length based on CST and ACS bits

Received packet length	CST	ACS	FCS stripping done
<1536	—	0	No
	—	1	Yes (for Ethernet packets)
>=1536	0	—	No
	1	—	Yes (for Type packets)
<1536	—	0	No
	—	1	Yes (for Ethernet packets)

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved	SARC			IPC	IPG			GPSLCE	S2KP	CST	ACS	WD	Reserved	JD	JE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PS	FES	DM	LM	ECRSFD	DO	DCRS	DR	Reserved	BL		DC	PRELEN		TE	RE
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30-28 SARC	Source Address Insertion Or Replacement Control Controls the source address insertion or replacement for all the transmitted packets. Bit 30 of this field specifies which MAC Address register (0 or 1) is used for source address insertion or replacement based on the values of bits [29:28]: • 2'b0x: The mti_sa_ctrl_i and ati_sa_ctrl_i input signals control the SA field generation. • 2'b10: If bit 30 = 0, MAC inserts the content of the MAC Address 0 registers in the SA field of all the transmitted packets. If bit 30 = 1 and the Enable MAC Address Register 1 option is selected while configuring the core, MAC inserts the content of the MAC Address 1 registers in the SA field of all the transmitted packets. • 2'b11: If bit 30 = 0, MAC replaces the content of the MAC Address 0 registers in the SA field of all the transmitted packets. If bit 30 = 1 and the MAC Address Register 1 is enabled, MAC replaces the content of the MAC Address 1 registers in the SA field of all the transmitted packets. **NOTE** Changes to this field take effect only on the start of a packet. If you write to the field when a packet is being transmitted, only the subsequent packet can use the updated value. This means that the current packet does not use the updated value. 000b - mti_sa_ctrl_i and ati_sa_ctrl_i input signals control the SA field generation 010b - Contents of are inserted in the SA field 011b - Contents of replace the SA field 110b - Contents of are inserted in the SA field 111b - Contents of replace the SA field
27 IPC	Checksum Offload Indicates the status of IP header or payload checksum checking. If this field is 1, the field enables IPv4 header checksum checking and IPv4 or IPv6 TCP, UDP, or ICMP payload checksum checking. When the field becomes 0, the COE function in the receiver is disabled. The Layer 3 and Layer 4 packet filter and enable split header features automatically select the IPC full checksum offload engine on the receive side. When any of these features are enabled, you must write 1 to this field. 0b - Disabled 1b - Enabled
26-24 IPG	Inter-Packet Gap Controls the minimum IPG between packets during transmission. The range of minimum IPG is valid in Full-Duplex mode, and in this mode, the minimum IPG can be configured only for 64-bit times (IPG = 100). Lower values are not considered.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When a JAM pattern is being transmitted because of backpressure activation, MAC does not consider the minimum IPG. This function (IPG less than 96-bit times) is valid only when MAC_Ext_Configuration[EIPGEN] is 0. If this field is 1, the minimum IPG (greater than 96-bit times) is controlled according to the field's description. 000b - 96-bit times IPG 001b - 88-bit times IPG 010b - 80-bit times IPG 011b - 72-bit times IPG 100b - 64-bit times IPG 101b - 56-bit times IPG 110b - 48-bit times IPG 111b - 40-bit times IPG
23 GPSLCE	Giant Packet Size Limit Control Enable Enables and disables giant packet size limit control. If this field = 1, MAC considers the value of MAC_Ext_Configuration[GPSL] to declare a received packet as a giant packet. This field must be programmed to have a size of more than 1,518 bytes. Otherwise, MAC considers 1,518 bytes as the giant packet limit. When this field becomes 0, MAC considers the received packet as a giant packet if its size is greater than 1,518 bytes (1522 bytes for a tagged packet). The WD, JE, and S2KP fields have a higher precedence over this field, which means MAC considers a received packet as a giant packet when its size is greater than 9,018 bytes (9,022 bytes for a tagged packet) with JE = 1 and the size of the jumbo packet greater than 2,000 bytes and S2KP = 1. If the WD field is 1, the field terminates the received packet if it reaches the watchdog limit. Therefore, the programmed giant packet limit must be less than the watchdog limit to get the giant packet status. 0b - Disabled 1b - Enabled
22 S2KP	IEEE 802.3 Support For 2K Packets Indicates the status of IEEE 802.3 support for 2K packets. If this field is 1, MAC considers all the packets with up to 2,000-byte length as normal. When the JE field is not 1, MAC considers all the received packets of size more than 2K bytes as giant packets. When this field becomes 0 and the JE field is not 1, MAC considers all the received packets of size more than 1,518 bytes (1,522 bytes for tagged) as giant packets. For more information about how the setting of this field and the JE field impacts the giant packet status, see this status based on the S2KP and JE fields. If the JE field is 1, writing 1 to the S2KP field has no effect on the giant packet status. 0b - Disabled 1b - Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
21 CST	CRC Stripping For Type Packets Indicates the status of CRC stripping for Type packets. If this field is 1, the last four bytes (FCS) of all the Ether-type packets are stripped and dropped before forwarding the packet to the application. This field is valid only when the receive packet size is more than or equal to 1536 bytes. For information about how the settings of this field and the ACS field impact the packet length, see MAC Configuration (MAC_Configuration). 0b - Disabled 1b - Enabled
20 ACS	Automatic Pad Or CRC Stripping Indicates the status of automatic pad or CRC stripping. If this field is 1, MAC strips the pad or the FCS field on the incoming packets only if the size of the packets is less than 1,536 bytes. All the received packets with a size greater than or equal to 1,536 bytes are passed to the application without stripping the pad or the FCS field. When this field becomes 0, MAC passes all the incoming packets to the application, without any modification. For information about how the settings of this field and the CST field impact the packet length, see MAC Configuration (MAC_Configuration). 0b - Disabled 1b - Enabled
19 WD	Watchdog Disable Indicates the status of watchdog. If this field is 1, MAC disables the watchdog timer on the receiver. MAC can receive packets of up to 16,383 bytes. When this field becomes 0, MAC does not allow more than 2,048 bytes (10,240 if the JE field = 1) of the packet being received. MAC cuts off bytes received after 2,048. 0b - Enabled 1b - Disabled
18 —	Reserved
17 JD	Jabber Disable Indicates the status of jabber. If this field is 1, MAC disables the jabber timer on the transmitter. It can transfer packets of up to 16,383 bytes.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	If this field is 0 and the application sends more than 2,048 bytes of data (10,240 if the JE field = 1) during transmission, MAC does not send the rest of the bytes in that packet. 0b - Enabled 1b - Disabled
16 JE	Jumbo Packet Enable Indicates the status of jumbo packets. If this field is 1, MAC allows jumbo packets of 9,018 bytes (9,022 bytes for VLAN-tagged packets) without reporting a giant packet error in the receive packet status. 0b - Disabled 1b - Enabled
15 PS	Port Select Selects the Ethernet line speed. This field, along with the FES field, selects the exact line speed. In the 10/100 Mbit/s-only (always 1) or 1000 Mbit/s-only (always 0) configurations, this field is read-only (RO) with an appropriate value. By default, with 10/100/1000 Mbit/s configurations, this field is read-write (R/W). The mac_speed_o[1] signal reflects the value of this field. 0b - For 1000 or 2500 Mbit/s operations 1b - For 10 or 100 Mbit/s operations
14 FES	Speed Indicates the speed. This field selects the speed mode. The mac_speed_o[0] signal indicates the value of this field. 0b - 10 Mbit/s if PS = 1 and 1 Gbps if PS = 0 1b - 100 Mbit/s if PS = 1 and 2.5 Gbps if PS = 0
13 DM	Duplex Mode Indicates the mode in which MAC operates. If this field is 1, MAC operates in Full-Duplex mode in which it can transmit and receive simultaneously. The field can only be read, with 1 as its default value, in full-duplex-only configurations. 0b - Half-duplex mode 1b - Full-duplex mode
12 LM	Loopback Mode Indicates the status of Loopback mode If this field is 1, MAC operates in Loopback mode at GMII or MII. This mode requires the (G)MII Rx clock input (CLK_RX_I) signal to function properly because the transmit clock is not internally looped back.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disabled 1b - Enabled
11 ECRSFD	Enable Carrier Sense In Full-Duplex Mode Indicates whether ECRSFD is enabled or disabled. If this field is 1, the MAC transmitter checks the CRS signal before packet transmission in Full-Duplex mode. MAC starts transmitting only when the CRS signal is low. If this field becomes 0, the MAC transmitter ignores the status of the CRS signal. 0b - Disabled 1b - Enabled
10 DO	Disable Receive Own Enables or disables receive own. If this field is 1, MAC disables the reception of packets when the gmii_txen_o signal asserts in Half-Duplex mode. When the field is 0, MAC receives all the packets that PHY sent. This field is not applicable in Full-Duplex mode. 0b - Enabled 1b - Disabled
9 DCRS	Disable Carrier Sense During Transmission Enables or disables carrier sense during transmission. If this field is 1, the MAC transmitter ignores the (G)MII CRS signal during packet transmission in Half-Duplex mode. As a result, no errors generate because of loss of carrier or no carrier during transmission. When this field becomes 0, the MAC transmitter generates errors because of carrier sense. MAC can even abort the transmission. 0b - Enabled 1b - Disabled
8 DR	Disable Retry Enables or disables retry attempts. If this field is 1, MAC attempts only one transmission. When a collision occurs on GMII or MII, MAC ignores the current packet transmission and reports a packet abort with excessive collision errors in the transmit packet status. If this field becomes 0, MAC retries based on the settings of the BL field of this register. The settings of this field apply only in Half-Duplex mode. 0b - Enabled 1b - Disabled
7	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
6-5 BL	Back-Off Limit Determines the random integer number (r) of slot time delays (4,096 bit times for 1000/2500 Mbit/s; 512 bit times for 10/100 Mbit/s) for which MAC waits before rescheduling a transmission attempt during retry attempts after a collision. • *n* = retransmission attempt • *r* = random integer that takes the value in the range 0 <= r < 2^k This field is applicable only in Half-Duplex mode. 00b - k = min(n,10) 01b - k = min(n,8) 10b - k = min(n,4) 11b - k = min(n,1)
4 DC	Deferral Check Indicates the status of the deferral check function. If this field is 1, the deferral check function is enabled in MAC, which issues a packet abort status, with Tx_Excessive_Deferral_Error[TXEXSDEF] = 1 in the transmit packet status, when the transmit state machine is deferred for more than 24,288 bit times in 10 Mbit/s or 100 Mbit/s mode. If MAC is configured for a 1000/2500 Mbit/s operation, the threshold for deferral is 155,680 bit times. Deferral begins when the transmitter is ready to transmit, but it is prevented because of an active carrier sense signal (CRS) on GMII or MII. The defer time is not cumulative. For example, if the transmitter defers for 10,000 bit times because the CRS signal is active, and then the CRS signal becomes inactive, the transmitter transmits and collision happens. Because of this collision, the transmitter needs to back off and then defer again after back off completion. In such a scenario, the deferral timer resets to 0 and restarts. When this field becomes 0, the deferral check function disables and MAC defers until the CRS signal becomes inactive. This field is applicable only in Half-Duplex mode. 0b - Disabled 1b - Enabled
3-2 PRELEN	Preamble Length for Transmit Packets Controls the number of preamble bytes that are added to the beginning of every transmit packet. Preamble reduction occurs only when MAC operates in Full-Duplex mode. 00b - 7 bytes 01b - 5 bytes 10b - 3 bytes 11b - Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 TE	Transmitter Enable Indicates the status of the transmitter. • If this field is 1, the MAC transmit state machine is enabled for transmission on GMII or MII. • When the field becomes 0, the MAC transmit state machine is disabled after it completes the transmission of the current packet. The transmit state machine does not transmit any more packets. 0b - Disabled 1b - Enabled
0 RE	Receiver Enable Indicates the status of the receiver. If this field is 1, MAC's receive state machine is enabled for receiving packets from GMII or MII. When this field becomes 0, the MAC receive state machine is disabled after it completes the reception of the current packet. The receive state machine does not receive any more packets from GMII or MII. 0b - Disabled 1b - Enabled

75.17.3 MAC Extended Configuration (MAC_Ext_Configuration)

Offset

Register	Offset
MAC_Ext_Configuration	4h

Function

Establishes MAC's operating mode.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved	Reserved	EIPG					EIPGEN	Reserved	Reserved			PDC	USP	SPEN	DCRCC
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		GPSL													
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30 —	Reserved
29-25 EIPG	Extended Inter-Packet Gap Indicates the value of the extended inter-packet gap. The value of this field is applicable when EIPGEN = 1. The most-significant bits of this field, along with those of MAC_Configuration[IPG], render the minimum IPG greater than 96-bit times in steps of 8-bit times. 8'h00: 104-bit times 8'h01: 112-bit times 8'h02: 120-bit times ---------- 8'hFF: 2144-bit times
24 EIPGEN	Extended Inter-Packet Gap Enable Indicates the status of the extended inter-packet gap. • If this field is 1, MAC interprets the EIPG field of this register and the IPG field of MAC Configuration (MAC_Configuration) together as minimum IPG greater than 96-bit times in steps of 8-bit times. • If this field is 0, MAC ignores the EIPG field of this register and interprets the IPG field of MAC Configuration (MAC_Configuration) as minimum IPG less than or equal to 96-bit times in steps of 8-bit times. **NOTE** You must enable the extended inter-packet gap feature when operating in Full-Duplex mode only. There could be undesirable effects on the back-pressure function and frame transmission if the feature is enabled in Half-Duplex mode. 0b - Disabled 1b - Enabled
23 —	Reserved
22-20 —	Reserved
19 PDC	Packet Duplication Control Indicates the packet duplication control status.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• If this field is 1, the received packet with multicast or broadcast destination address is routed to multiple receive DMA channels. MAC_Address0_High[DCS] identifies the receive DMA channels (corresponding to the MAC Address register) that match the multicast or broadcast destination addresses in the received packet. The DCS field is interpreted to be a one-hot value, each bit corresponding to the receive DMA channel. • If this field is 0, the received packet routes to a single receive DMA channel. MAC_Address0_High[DCS] identifies the receive DMA channel corresponding to the MAC Address register that matches the destination address in the received packet. The DCS field is interpreted as a binary value. 0b - Disabled 1b - Enabled
18 USP	Unicast Slow Protocol Packet Detect Indicates the status of unicast slow-protocol packet detection. • If this field is 1, MAC detects the slow-protocol packets with unicast address of the station specified in MAC Address 0 High (MAC_Address0_High) and MAC Address 0 Low (MAC_Address0_Low). MAC also detects the slow-protocol packets with the slow-protocol multicast address (01-80-C2-00-00-02). • If this field is 0, MAC detects only slow-protocol packets with the slow protocol multicast address specified in IEEE 802.3-2015, section 5. 0b - Disabled 1b - Enabled
17 SPEN	Slow Protocol Detection Enable Enables or disables slow protocol detection. • If this field is 0, EMAC forwards all error-free, slow-protocol packets to the application. EMAC considers such packets as normal-type packets. • If this field is 1, EMAC processes the slow-protocol packets (Ether type 8809h) and provides the slow protocol sub-type and code fields in receive status. 0b - Disabled 1b - Enabled
16 DCRCC	Disable CRC Checking For Received Packets Indicates the status of CRC checking. If this field is 1, MAC receiver does not check the CRC field in the received packets. When this field becomes 0, the MAC receiver always checks the CRC field in the received packets. 0b - Enabled 1b - Disabled
15-14	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
13-0 GPSL	Giant Packet Size Limit Declares the status of a packet based on its size. If the received packet size (in bytes) is greater than the value programmed in this field, MAC declares the received packet as a giant packet. The value programmed in this field must be greater than or equal to 1,518 bytes. Also, any other programmed value is considered as 1,518 bytes. For VLAN-tagged packets, MAC adds 4 bytes to the programmed value. If MAC_VLAN_Tag_Ctrl[EDVLP] = 1, MAC adds 8 bytes to the programmed value for double VLAN-tagged packets. The value of GPSL is valid if MAC_Configuration[GPSLCE] = 1.

75.17.4 MAC Packet Filter (MAC_Packet_Filter)

Offset

Register	Offset
MAC_Packet_Filter	8h

Function

Contains the filter controls for receiving packets.

Some of the controls from this register go to MAC's address check block that performs the first level of address filtering. The second level of filtering is performed on the incoming packet based on other controls such as pass bad packets and pass control packets.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	RA	Reserved									DNTU	IPFE	Reserved			VTFE
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved				HPF	SAF	SAIF	PCF			DBF	PM	DAIF	HMC	HUC	PR
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 RA	Receive All Indicates whether the received packets are enabled or disabled. • If this field is 1, the MAC Receiver module passes all the received packets to the application, irrespective of their passing the address filter. The result (pass or fail) of the SA or DA filtering is updated in the corresponding field of the receive status word. • If this field is 0, the module passes only those packets to the application that pass the SA or DA address filter. 0b - Receive All is disabled 1b - Receive All is enabled
30-22 —	Reserved
21 DNTU	Drop Non-TCP/UDP Over IP Packets Indicates whether MAC drops or forwards non-TCP/UDP protocols over IP packets. • If this field is 1, MAC drops these protocols over IP packets. MAC forwards only those packets that the layer 4 filter processes. • If this field is 0, MAC forwards all these protocols over IP packets. 0b - Forwards 1b - Drops
20 IPFE	Layer 3 and Layer 4 Filter Enable Indicates the status of layer 3 and layer 4 filters. • If this field is 1, MAC drops packets that do not match the enabled layer 3 and layer 4 filters. If these two filters are not enabled for matching, this field does not have any effect. • If this field is 0, MAC forwards all the packets irrespective of the match status of the layer 3 and layer 4 fields. 0b - Disabled 1b - Enabled
19-17 —	Reserved
16 VTFE	VLAN Tag Filter Enable Indicates the status of VLAN tag filter. • If this field is 1, MAC drops the VLAN-tagged packets that do not match the VLAN tag. • If this field is 0, MAC forwards all the packets irrespective of the match status of the VLAN tag. 0b - Disabled 1b - Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-11 —	Reserved
10 HPF	Hash Or Perfect Filter Indicates the status of hash or perfect filter. • If this field is 1, the address filter passes a packet if it matches either perfect filtering or hash filtering as defined by the HMC or HUC fields of this register. • If this field is 0 and HUC = 1 or HMC = 1, the packet passes only if it matches the hash filter. 0b - Disabled 1b - Enabled
9 SAF	Source Address Filter Enable Indicates the status of SA filtering. • If this field is 1, MAC compares the SA field of the received packets with the values programmed in the enabled SA registers. If the comparison fails, MAC drops the packet. • When this field becomes 0, MAC forwards the received packet to the application with an updated field value, depending on the SA address comparison. **NOTE** According to the IEEE specification, bit 47 of the SA field is reserved. However, MAC compares all the 48 bits. You must consider this when programming the MAC address registers for SA. 0b - Disabled 1b - Enabled
8 SAIF	SA Inverse Filtering Indicates the status of SA inverse filtering. • If this field is 1, the address check block operates in the inverse filtering mode for SA address comparison. If the SA of a packet matches the values programmed in the SA registers, it is marked as failing the SA address filter. • If this field becomes 0 and the SA of a packet does not match the values programmed in the SA registers, the SA is marked as failing the SA address filter. 0b - Disabled 1b - Enabled
7-6 PCF	Pass Control Packets Controls the forwarding of all the control packets (including unicast and multicast pause packets). 00b - MAC filters all the control packets from reaching the application 01b - MAC forwards all the control packets, except pause packets, to the application even if they fail the address filter

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10b - MAC forwards all the control packets to the application even if they fail the address filter 11b - MAC forwards all the control packets that pass the address filter
5 DBF	Disable Broadcast Packets Enables or disables broadcast packets. • If this field is 1, AFM blocks all the incoming broadcast packets. In addition, it overrides all the other filter settings. • If this field is 0, AFM passes all the received broadcast packets. 0b - Enabled 1b - Disabled
4 PM	Pass All Multicast Enables or disables the passing of received packets. • If this field is 1, it indicates that all the received packets with a multicast destination address (first bit in the destination address field is 1) are passed. • If this field is 0, filtering of multicast packets depends on the settings of the HMC field. 0b - Disabled 1b - Enabled
3 DAIF	DA Inverse Filtering Indicates the status of DA inverse filtering. • If this field is 1, the Address Check block operates in Inverse Filtering mode for the DA address comparison of both unicast and multicast packets. • If this field is 0, normal filtering of packets is performed. 0b - Disabled 1b - Enabled
2 HMC	Hash Multicast Indicates the status of hast multicast. • If this field is 1, MAC performs the destination address filtering of received multicast packets according to the hash table. • If this field is 0, MAC performs the perfect destination address filtering for multicast packets, that is, it compares the DA field with the values programmed in DA registers. 0b - Disabled 1b - Enabled
1 HUC	Hash Unicast Indicates the status of hash unicast.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• If this field is 1, MAC performs the destination address filtering of unicast packets according to the hash table. • When this field becomes 0, MAC performs a perfect destination address filtering for unicast packets, that is, it compares the DA field with the values you program in the MAC Address registers. 0b - Disabled 1b - Enabled
0 PR	Promiscuous Mode Indicates the status of Promiscuous mode. If this field is 1, the address filtering module passes all the incoming packets irrespective of the destination or source address. MAC clears the SA or DA filter fail status fields of the receive status word when PR = 1. 0b - Disabled 1b - Enabled

75.17.5 MAC Watchdog Timeout (MAC_Watchdog_Timeout)

Offset

Register	Offset
MAC_Watchdog_Timeout	Ch

Function

Controls the watchdog timeout for received packets.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							PWE	Reserved				WTO			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-9 —	Reserved
8 PWE	Programmable Watchdog Enable Indicates the status of programmable watchdog. • If this field is 1 and MAC_Configuration[WD] is 0, the WTO field of this register is used as watchdog timeout for a received packet. • If this field is 0, MAC_Configuration[WD] and MAC_Configuration[JE] control the watchdog timeout for a received packet. 0b - Disabled 1b - Enabled
7-4 —	Reserved
3-0 WTO	Watchdog Timeout Indicates the size of watchdog timer. If PWE = 1 and MAC_Configuration[WD] = 0, this field is used as watchdog timeout for a received packet. If the length of a received packet exceeds the value of this field, the packet terminates as an error packet. **NOTE** If PWE = 1, the value in this field must exceed 1,522 (05F2h). Otherwise, the IEEE 802.3-specified valid tagged packets are declared as error packets and dropped. 0000b - 2 KB 0001b - 3 KB 0010b - 4 KB 0011b - 5 KB 0100b - 6 KB 0101b - 7 KB 0110b - 8 KB 0111b - 9 KB 1000b - 10 KB 1001b - 11 KB 1010b - 12 KB 1011b - 13 KB 1100b - 14 KB 1101b - 15 KB

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1110b - 16383 bytes 1111b - Reserved

75.17.6 MAC Hash Table First 32 Bits (MAC_Hash_Table_Reg0)

Offset

Register	Offset
MAC_Hash_Table_Reg0	10h

Function

Contains the first 32 bits of the hash table, when the total width of the hash table is 128 bits or 256 bits.

NOTE

In this chip, upper 6 bits should be used for 64-bit Hash Table.

The hash table is used for group address filtering, for which the content of the destination address in the incoming packet is passed through the CRC logic and the bits of Receive CRC Error Packets (Rx_CRC_Error_Packets) are used to index the contents of the hash table as follows:

- 6 bits when you have a 64-bit hash
- 7 bits when you have a 128-bit hash
- 8 bits when you have a 256-bit hash

The most-significant bits determine the Hash Table register to be used, and the least-significant 5 bits determine the bit within the register. For example, a hash value of 10_0000b (in 64-bit hash) selects bit 0 of MAC Hash Table Second 32 Bits (MAC_Hash_Table_Reg1).

Perform these steps to calculate the hash value of the destination address:

1. Calculate the 32-bit CRC for DA (see section 3.2.8 in IEEE 802.3 for the steps to calculate CRC32).
2. Perform bit-wise reversal for the value obtained in step 1.
3. Use the upper 6, 7, or 8 bits from the value obtained in step 2.

If this field is 1, the packet is accepted. Otherwise, it is rejected. If MAC_Packet_Filter[PM] = 1, all multicast packets are accepted regardless of the multicast hash values.

If this register is configured to be double-synchronized with the (G)MII clock domain, the synchronization is triggered only when bits [31:24] (in Little-Endian mode) or bits [7:0] (in Big-Endian mode) of the Hash Table registers are written to.

If double-synchronization is enabled, consecutive writes to this register must be performed after at least four clock cycles in the destination clock domain.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	HT31T0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	HT31T0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 HT31T0	MAC Hash Table First 32 Bits Contains the first 32 bits [31:0] of the hash table.

75.17.7 MAC Hash Table Second 32 Bits (MAC_Hash_Table_Reg1)

Offset

Register	Offset
MAC_Hash_Table_Reg1	14h

Function

Contains the second 32 bits of the hash table, when the width of the hash table is 128 or 256 bits.

NOTE

In this chip, upper 6 bits should be used for 64-bit Hash Table.

The hash table is used for group address filtering, for which the content of the destination address in the incoming packet is passed through the CRC logic and the bits of Receive CRC Error Packets (Rx_CRC_Error_Packets) are used to index the contents of the hash table as follows:

- 6 bits when you have a 64-bit hash
- 7 bits when you have a 128-bit hash
- 8 bits when you have a 256-bit hash

The most-significant bits determine the Hash Table register to be used, and the least-significant 5 bits determine the bit within the register. For example, a hash value of 10_0000b (in 64-bit hash) selects bit 0 of MAC Hash Table Second 32 Bits (MAC_Hash_Table_Reg1).

Perform these steps to calculate the hash value of the destination address:

1. Calculate the 32-bit CRC for DA (see section 3.2.8 in IEEE 802.3 for the steps to calculate CRC32).
2. Perform bit-wise reversal for the value obtained in step 1.
3. Take the upper 6, 7, or 8 bits from the value obtained in step 2.

If the corresponding bit value of this register is 1'b1, the packet is accepted. Otherwise, it is rejected. If MAC_Packet_Filter[PM] = 1, all multicast packets are accepted regardless of the multicast hash values.

If this register is configured to be double-synchronized with the (G)MII clock domain, the synchronization is triggered only when bits [31:24] (in Little-Endian mode) or bits [7:0] (in Big-Endian mode) of the Hash Table registers are written to.

If double-synchronization is enabled, consecutive writes to this register must be performed after at least four clock cycles in the destination clock domain.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	HT63T32															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	HT63T32															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 HT63T32	MAC Hash Table Second 32 Bits Contains the second 32 bits [63:32] of the hash table.

75.17.8 MAC VLAN Tag (MAC_VLAN_Tag)

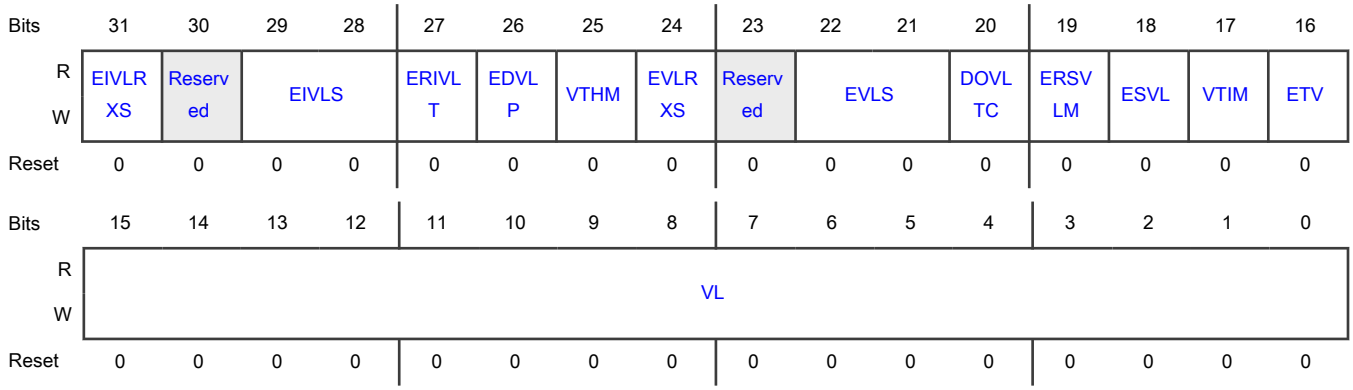
Offset

Register	Offset
MAC_VLAN_Tag	50h

Function

Identifies the IEEE 802.1Q VLAN type packets.

Diagram



Fields

Field	Function
31 EIVLRXS	Enable Inner VLAN Tag In Receive Status Indicates whether the inner VLAN tag in receive status is enabled or disabled. • If this field is 1, MAC provides the inner VLAN tag in the receive status. • If this field is 0, MAC does not provide the inner VLAN tag in the receive status. 0b - Disabled 1b - Enabled
30 —	Reserved
29-28 EIVLS	Enable Inner VLAN Tag Stripping Indicates the stripping operation on the inner VLAN tag in a received packet. The field enables or disables inner VLAN tag stripping on receive. 00b - Do not strip 01b - Strip if VLAN filter passes 10b - Strip if VLAN filter fails 11b - Always strip
27 ERIVLT	Enable Inner VLAN Tag Comparison Enables or disables the inner VLAN tag. • If ERIVLT = VTHM = EDVLP = 1, the EMAC receiver enables the VLAN hash filtering operation on the inner VLAN tag (if present). • If ERIVLT = 0 and VTHM = 1, the EMAC receiver enables the VLAN hash filtering operation on the outer VLAN tag (if present). ERSVLM and DOVLTC determine which VLAN type is enabled for filtering. 0b - Disabled 1b - Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
26 EDVLP	Enable Double VLAN Processing Enables or disables double VLAN processing. • If this field is 1, MAC enables processing of up to two VLAN tags on transmit and receive (if present). • If this field is 0, MAC enables processing of up to one VLAN tag on transmit and receive (if present). 0b - Disabled 1b - Enabled
25 VTHM	VLAN Tag Hash Table Match Indicates the status of VLAN tag hash table match. • If this field is 1, the most-significant four bits of CRC-32 of VLAN tag are used to index the content of MAC VLAN Hash Table (MAC_VLAN_Hash_Table). See VLAN filtering for details. • If MAC_VLAN_Hash_Table[VLHT] = 1, corresponding to the index, it indicates that the Ethernet packet matches the VLAN hash table. See VLAN filtering for details. • If ETV = 1, the CRC of the 12-bit VLAN identifier (VID) is used for comparison. When the ETV field becomes 0, the CRC of the 16-bit VLAN tag is used for comparison. • If VTHM = 1, the VLAN hash match operation is not performed. 0b - Disabled 1b - Enabled
24 EVLRXS	Enable VLAN Tag In Receive Status Indicates whether the VLAN tag in receive status is enabled. • If this field is 1, MAC provides the outer VLAN tag in the receive status. • If this field is 0, MAC does not provide the outer VLAN tag in receive status. 0b - Disabled 1b - Enabled
23 —	Reserved
22-21 EVLS	Enable VLAN Tag Stripping Indicates the stripping operation on the outer VLAN tag in received packets. The field enables or disables VLAN tag stripping on receive. 00b - Do not strip 01b - Strip if VLAN filter passes 10b - Strip if VLAN filter fails 11b - Always strip
20	Disable VLAN Type Check

Table continues on the next page...

Table continued from the previous page...

Field	Function
DOVLTC	Enables or disables VLAN type check for VLAN hash filtering. • If this field is 1, the MAC VLAN hash filter matches or filters the VLAN tag that the ERIVLT field specifies only when the VLAN tag type is similar to the one that the ERSVLM field specifies. • If this field is 0, the MAC VLAN hash filter does not check whether the VLAN tag that the ERIVLT field specifies is of type S-VLAN or C-VLAN. • The VLAN filter is bypassed when VLAN type of received packet do not match the programmed VLAN type in the VLAN filter. 0b - Enabled 1b - Disabled
19 ERSVLM	Enable Receive S-VLAN Match Enables or disables receive S-VLAN match for VLAN hash filtering. • If this field is 1, the MAC receiver enables VLAN hash filtering or S-VLAN matching (type = 88A8h) packets. • If this field is 0, the MAC receiver enables VLAN hash filtering or matching for C-VLAN (type = 8100h) packets. The ERIVLT field determines the VLAN tag position considered for VLAN hash filtering or matching. 0b - Disabled 1b - Enabled
18 ESVL	Enable S-VLAN Enables or disables S-VLAN. If this field is 1, the MAC transmitter and receiver consider the S-VLAN packets (type = 88A8h) as valid VLAN-tagged packets. 0b - Disabled 1b - Enabled
17 VTIM	VLAN Tag Inverse Match Enable Enables or disables VLAN tag inverse matching. • If this field is 1, it enables the VLAN tag inverse matching. The packets without matching VLAN tag are marked as matched. • If this field is 0, it enables the VLAN tag perfect matching. The packets with matched VLAN tag are marked as matched. 0b - Disabled 1b - Enabled
16 ETV	Enable Tag For VLAN Enables or disables 12-bit VLAN tag comparison for VLAN hash filtering.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• If this field is 1, a 12-bit VLAN identifier is used for VLAN hash filtering instead of the complete 16-bit VLAN tag. Bits [11:0] of the VLAN tag in the received VLAN-tagged packet are used for hash-based VLAN filtering. • When this field becomes 0, all 16 bits of the 15th and 16th bytes of the received VLAN packet are used for VLAN hash filtering. 0b - Disabled 1b - Enabled
15-0 VL	VLAN Tag Identifier for Receive Packets Contains the 802.1Q VLAN tag to identify the VLAN packets. This VLAN tag identifier is compared to the 15th and 16th bytes of the packets being received for VLAN packets. The following list describes the bits of this field: • 15:13 — User priority • 12 — Canonical Format Indicator (CFI) or Drop Eligible Indicator (DEI) • 11:0 — VID field of VLAN tag When ETV = 1, only the VID is used for comparison. When VL = 0, and ETV = 1, MAC does not check the 15th and 16th bytes for VLAN tag comparison and declares all packets with Type field value of 8100h or 88a8h as VLAN packets.

75.17.9 MAC VLAN Tag Control (MAC_VLAN_Tag_Ctrl)

Offset

Register	Offset
MAC_VLAN_Tag_Ctrl	50h

Function

Is the redefined version of the MAC_VLAN_Tag register.

This register provides the control and addressing fields required for indirect access to the MAC_VLAN_Tag_Filter registers. It also contains the address offset, command type, and busy bit for CSR access of the MAC VLAN Hash Filter registers.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	EIVLRXS	Reserved	EIVLS		ERIVLT	EDVLP	VTHM	EVLRXS	Reserved	EVLS		DOVLTC	ERSVLM	ESVL	VTIM	ETV
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												OFS		CT	OB
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 EIVLRXS	Enable Inner VLAN Tag In Receive Status Indicates whether the inner VLAN tag in receive status is enabled or disabled. • If this field is 1, MAC provides the inner VLAN tag in the receive status. • If this field is 0, MAC does not provide the inner VLAN tag in the receive status. 0b - Disabled 1b - Enabled
30 —	Reserved
29-28 EIVLS	Enable Inner VLAN Tag Stripping Indicates the stripping operation on the inner VLAN tag in a received packet. The field enables or disables inner VLAN tag stripping on receive. 00b - Do not strip 01b - Strip if VLAN filter passes 10b - Strip if VLAN filter fails 11b - Always strip
27 ERIVLT	Enable Inner VLAN Tag Comparison Enables or disables the inner VLAN tag. • If ERIVLT = VTHM = EDVLP = 1, the EMAC receiver enables the VLAN hash filtering operation on the inner VLAN tag (if present). • If ERIVLT = 0 and VTHM = 1, the EMAC receiver enables the VLAN hash filtering operation on the outer VLAN tag (if present). ERSVLM and DOVLTC determine which VLAN type is enabled for filtering. 0b - Disabled 1b - Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
26 EDVLP	Enable Double VLAN Processing Enables or disables double VLAN processing. • If this field is 1, MAC enables processing of up to two VLAN tags on transmit and receive (if present). • If this field is 0, MAC enables processing of up to one VLAN tag on transmit and receive (if present). 0b - Disabled 1b - Enabled
25 VTHM	VLAN Tag Hash Table Match Indicates the status of VLAN tag hash table match. • If this field is 1, the most-significant four bits of CRC-32 of VLAN tag are used to index the content of MAC VLAN Hash Table (MAC_VLAN_Hash_Table). See VLAN filtering for details. • If MAC_VLAN_Hash_Table[VLHT] = 1, corresponding to the index, it indicates that the Ethernet packet matches the VLAN hash table. See VLAN filtering for details. • If ETV = 1, the CRC of the 12-bit VLAN identifier (VID) is used for comparison. When the ETV field becomes 0, the CRC of the 16-bit VLAN tag is used for comparison. • If VTHM = 1, the VLAN hash match operation is not performed. 0b - Disabled 1b - Enabled
24 EVLRXS	Enable VLAN Tag In Receive Status Indicates whether the VLAN tag in receive status is enabled. • If this field is 1, MAC provides the outer VLAN tag in the receive status. • If this field is 0, MAC does not provide the outer VLAN tag in receive status. 0b - Disabled 1b - Enabled
23 —	Reserved
22-21 EVLS	Enable VLAN Tag Stripping Indicates the stripping operation on the outer VLAN tag in received packets. The field enables or disables VLAN tag stripping on receive. 00b - Do not strip 01b - Strip if VLAN filter passes 10b - Strip if VLAN filter fails 11b - Always strip
20	Disable VLAN Type Check

Table continues on the next page...

Table continued from the previous page...

Field	Function
DOVLTC	Enables or disables VLAN type check for VLAN hash filtering. • If this field is 1, the MAC VLAN hash filter matches or filters the VLAN tag that the ERIVLT field specifies only when the VLAN tag type is similar to the one that the ERSVLM field specifies. • If this field is 0, the MAC VLAN hash filter does not check whether the VLAN tag that the ERIVLT field specifies is of type S-VLAN or C-VLAN. 0b - Enabled 1b - Disabled
19 ERSVLM	Enable Receive S-VLAN Match Enables or disables receive S-VLAN match for VLAN hash filtering. • If this field is 1, the MAC receiver enables VLAN hash filtering or S-VLAN matching (type = 88A8h) packets. • If this field is 0, the MAC receiver enables VLAN hash filtering or matching for C-VLAN (type = 8100h) packets. The ERIVLT field determines the VLAN tag position considered for VLAN hash filtering or matching. 0b - Disabled 1b - Enabled
18 ESVL	Enable S-VLAN Enables or disables S-VLAN. If this field is 1, the MAC transmitter and receiver consider the S-VLAN packets (type = 88A8h) as valid VLAN-tagged packets. 0b - Disabled 1b - Enabled
17 VTIM	VLAN Tag Inverse Match Enable Enables or disables VLAN tag inverse matching. • If this field is 1, it enables the VLAN tag inverse matching. The packets without matching VLAN tag are marked as matched. • If this field is 0, it enables the VLAN tag perfect matching. The packets with matched VLAN tag are marked as matched. 0b - Disabled 1b - Enabled
16 ETV	Enable Tag For VLAN Enables or disables 12-bit VLAN tag comparison for VLAN hash filtering. • If this field is 1, a 12-bit VLAN identifier is used for VLAN hash filtering instead of the complete 16-bit VLAN tag. Bits [11:0] of the VLAN tag in the received VLAN-tagged packet are used for hash-based VLAN filtering.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• When this field becomes 0, all 16 bits of the 15th and 16th bytes of the received VLAN packet are used for VLAN hash filtering. 0b - Disabled 1b - Enabled
15-4 —	Reserved
3-2 OFS	Offset Holds the address offset of the MAC VLAN Tag Filter register that the application tries to access. The width of this field depends on the number of enabled MAC VLAN Tag registers.
1 CT	Command Type Specifies whether the current register access indicates a read or a write operation. It indicates a read operation if this field is 1 and a write operation when the field is 0. 0b - Write operation 1b - Read operation
0 OB	Operation Busy Indicates the operation busy status. • This field becomes 1 to initiate a read or write command for an indirect access to the Per VLAN Tag Filter register. • The field becomes 0 when the read or write command to Per VLAN Tag Filter indirect access register completes. The next indirect register access can be initiated only after the field resets. During a write operation, the field becomes 0 only after the data is written into a MAC_VLAN_Tag_Filter register. During a read operation, the data must be read from MAC VLAN Tag Data (MAC_VLAN_Tag_Data) only after this field becomes 0. 0b - Disabled 1b - Enabled

75.17.10 MAC VLAN Tag Data (MAC_VLAN_Tag_Data)

Offset

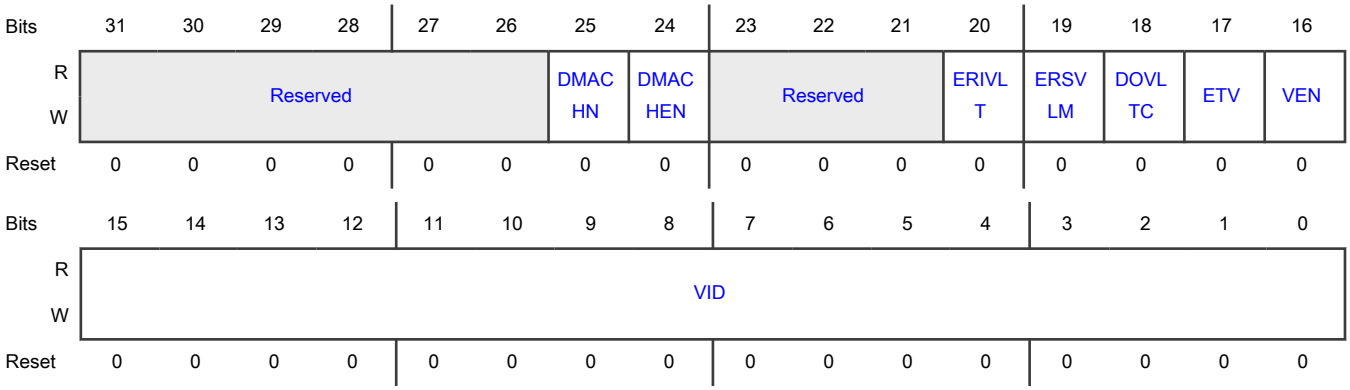
Register	Offset
MAC_VLAN_Tag_Data	54h

Function

Holds the read and write data for indirect access to the Per VLAN Tag registers.

- During read access, this register contains valid read data only after MAC_VLAN_Tag_Ctrl[OB] becomes 0.
- During write access, this register must become valid before you write 1 to MAC_VLAN_Tag_Ctrl[OB].

Diagram



Fields

Field	Function
31-26 —	Reserved
25 DMACHN	DMA Channel Number Indicates the DMA channel number to which the VLAN-tagged frame is to be routed if it passes the VLAN tag filter programmed using this field. If the routing based on VLAN tag filter is not necessary, this field does not need to be programmed.
24 DMACHEN	DMA Channel Number Enable Enables or disables the DMA channel number value programmed using the DMACHN field. If this field is 0, the routing does not occur based on the VLAN filter result. The frame is routed on the basis of DA-based DMA channel routing. 0b - Disabled 1b - Enabled
23-21 —	Reserved
20 ERIVLT	Enable Inner VLAN Tag Enables or disables inner VLAN tag comparison. • This field is valid only when the double VLAN tag enable of the filter is set. • If this field and the EDVLP field is 1, the MAC receiver enables operation on the inner VLAN tag (if present). • If this field is 0, the MAC receiver enables operation on the outer VLAN tag (if present).

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disabled 1b - Enabled
19 ERSVLM	Enable S-VLAN Match Enables or disables S-VLAN match for received frames. This field is valid only when the VLAN tag enable of the filter is set. • If this field is 1, the MAC receiver enables filtering or matching for S-VLAN (type = 88A8h) packets. • If this field is 0, the MAC receiver enables filtering or matching for C-VLAN (type = 8100h) packets. 0b - Disabled 1b - Enabled
18 DOVLTC	Disable VLAN Type Comparison Enables or disables VLAN type comparison. This field is valid only when the VEN field is 1. • If this field is 1, MAC does not check whether the VLAN tag that ERIVLT specifies is of type S-VLAN or C-VLAN. • If this field is 0, MAC filters or matches the VLAN tag that the ERIVLT field specifies only when the VLAN tag type is similar to the one that the ERSVLM field specifies. 0b - Enabled 1b - Disabled
17 ETV	VLAN Comparison Indicates 12-bit or 16-bit VLAN comparison. • This field is valid only when the VEN field is 1. • When the value of this field is 1, a 12-bit VLAN identifier is used for filter and comparison instead of the complete 16-bit VLAN tag. Bits [11:0] of the VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. 0b - 16-bit VLAN comparison 1b - 12-bit VLAN comparison
16 VEN	VLAN Tag Enable Enables or disables the VLAN tag. • If this field is 1, MAC compares the VLAN tag of the received packet with the VLAN tag ID. • If this field is 0, no comparison is performed irrespective of the programming of the other fields. 0b - Disabled 1b - Enabled
15-0	VLAN Tag ID

Table continues on the next page...

Table continued from the previous page...

Field	Function
VID	Holds the VLAN tag value that MAC uses for perfect comparison. The field is valid when VEN = 1.

75.17.11 MAC VLAN Tag Filter 0 (MAC_VLAN_Tag_Filter0)

Offset

Register	Offset
MAC_VLAN_Tag_Filter0	54h

Function
Contains VLAN tag filter 0 control information.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved						DMACHN	DMACHEN	Reserved			ERIVLT	ERSVLM	DOVLTC	ETV	VEN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	VID															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-26 —	Reserved
25 DMACHN	DMA Channel Number Indicates the DMA channel number to which the VLAN-tagged frame is to be routed if it passes the VLAN tag filter programmed using this field. If the routing based on VLAN tag filter is not necessary, this field does not need to be programmed.
24 DMACHEN	DMA Channel Number Enable Enables or disables the DMA channel number value programmed using the DMACHN field. If this field is 0, the routing does not occur based on the VLAN filter result. The frame is routed on the basis of DA-based DMA channel routing.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disabled 1b - Enabled
23-21 —	Reserved
20 ERIVLT	Enable Inner VLAN Tag Enables or disables inner VLAN tag comparison. • This field is valid only when the double VLAN tag enable of the filter is set. • If this field and the EDVLP field is 1, the MAC receiver enables operation on the inner VLAN tag (if present). • If this field is 0, the MAC receiver enables operation on the outer VLAN tag (if present). 0b - Disabled 1b - Enabled
19 ERSVLM	Enable S-VLAN Match Enables or disables S-VLAN match for received frames. This field is valid only when the VLAN tag enable of the filter is set. • If this field is 1, the MAC receiver enables filtering or matching for S-VLAN (type = 88A8h) packets. • If this field is 0, the MAC receiver enables filtering or matching for C-VLAN (type = 8100h) packets. 0b - Disabled 1b - Enabled
18 DOVLTC	Disable VLAN Type Comparison Enables or disables VLAN type comparison. This field is valid only when the VEN field is 1. • If this field is 1, MAC does not check whether the VLAN tag that ERIVLT specifies is of type S-VLAN or C-VLAN. • If this field is 0, MAC filters or matches the VLAN tag that the ERIVLT field specifies only when the VLAN tag type is similar to the one that the ERSVLM field specifies. 0b - Enabled 1b - Disabled
17 ETV	VLAN Comparison Indicates 12-bit or 16-bit VLAN comparison. • This field is valid only when the VEN field is 1. • When the value of this field is 1, a 12-bit VLAN identifier is used for filter and comparison instead of the complete 16-bit VLAN tag. Bits [11:0] of the VLAN tag are compared with the corresponding field in the received VLAN-tagged packet.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - 16-bit VLAN comparison 1b - 12-bit VLAN comparison
16 VEN	VLAN Tag Enable Enables or disables the VLAN tag. • If this field is 1, MAC compares the VLAN tag of the received packet with the VLAN tag ID. • If this field is 0, no comparison is performed irrespective of the programming of the other fields. 0b - Disabled 1b - Enabled
15-0 VID	VLAN Tag ID Holds the VLAN tag value that MAC uses for perfect comparison. The field is valid when VEN = 1.

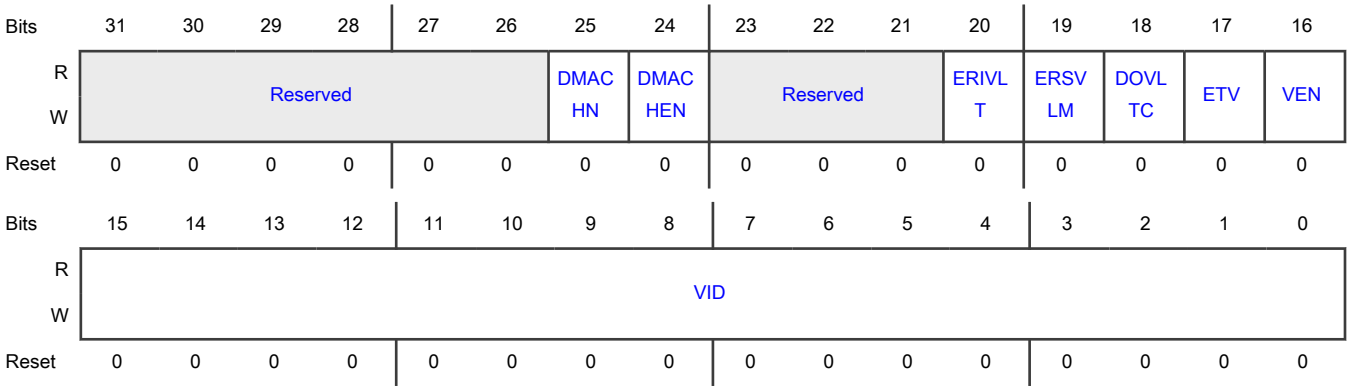
75.17.12 MAC VLAN Tag Filter 1 (MAC_VLAN_Tag_Filter1)

Offset

Register	Offset
MAC_VLAN_Tag_Filter1	54h

Function
Contains VLAN tag filter 1 control information.

Diagram



Fields

Field	Function
31-26 —	Reserved
25 DMACHN	DMA Channel Number Indicates the DMA channel number to which the VLAN-tagged frame is to be routed if it passes the VLAN tag filter programmed using this field. If the routing based on VLAN tag filter is not necessary, this field does not need to be programmed.
24 DMACHEN	DMA Channel Number Enable Enables or disables the DMA channel number value programmed using the DMACHN field. If this field is 0, the routing does not occur based on the VLAN filter result. The frame is routed on the basis of DA-based DMA channel routing. 0b - Disabled 1b - Enabled
23-21 —	Reserved
20 ERIVLT	Enable Inner VLAN Tag Enables or disables inner VLAN tag comparison. • This field is valid only when the double VLAN tag enable of the filter is set. • If this field and the EDVLP field is 1, the MAC receiver enables operation on the inner VLAN tag (if present). • If this field is 0, the MAC receiver enables operation on the outer VLAN tag (if present). 0b - Disabled 1b - Enabled
19 ERSVLM	Enable S-VLAN Match Enables or disables S-VLAN match for received frames. This field is valid only when the VLAN tag enable of the filter is set. • If this field is 1, the MAC receiver enables filtering or matching for S-VLAN (type = 88A8h) packets. • If this field is 0, the MAC receiver enables filtering or matching for C-VLAN (type = 8100h) packets. 0b - Disabled 1b - Enabled
18 DOVLTC	Disable VLAN Type Comparison Enables or disables VLAN type comparison. This field is valid only when the VEN field is 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• If this field is 1, MAC does not check whether the VLAN tag that ERIVLT specifies is of type S-VLAN or C-VLAN. • If this field is 0, MAC filters or matches the VLAN tag that the ERIVLT field specifies only when the VLAN tag type is similar to the one that the ERSVLM field specifies. 0b - Enabled 1b - Disabled
17 ETV	VLAN Comparison Indicates 12-bit or 16-bit VLAN comparison. • This field is valid only when the VEN field is 1. • When the value of this field is 1, a 12-bit VLAN identifier is used for filter and comparison instead of the complete 16-bit VLAN tag. Bits [11:0] of the VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. 0b - 16-bit VLAN comparison 1b - 12-bit VLAN comparison
16 VEN	VLAN Tag Enable Enables or disables the VLAN tag. • If this field is 1, MAC compares the VLAN tag of the received packet with the VLAN tag ID. • If this field is 0, no comparison is performed irrespective of the programming of the other fields. 0b - Disabled 1b - Enabled
15-0 VID	VLAN Tag ID Holds the VLAN tag value that MAC uses for perfect comparison. The field is valid when VEN = 1.

75.17.13 MAC VLAN Tag Filter 2 (MAC_VLAN_Tag_Filter2)

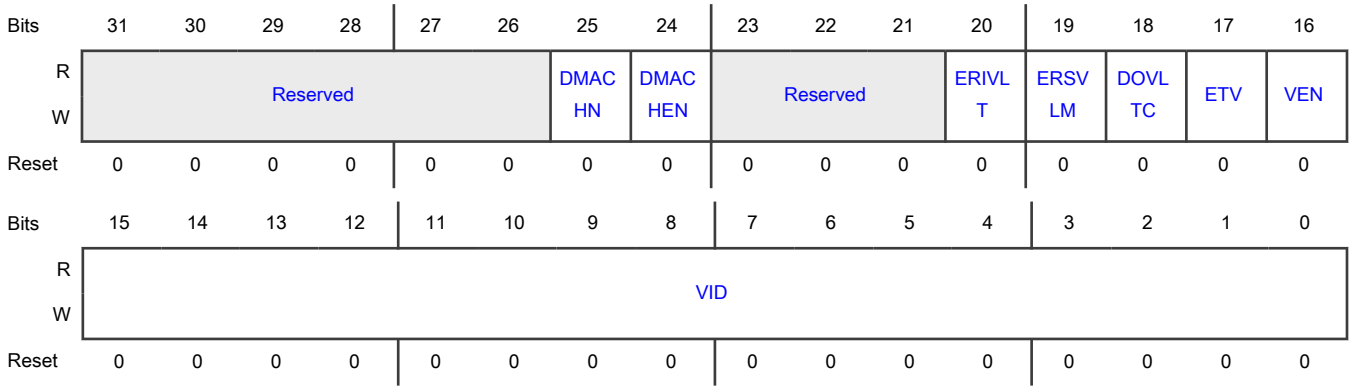
Offset

Register	Offset
MAC_VLAN_Tag_Filter2	54h

Function

Contains VLAN tag filter 2 control information.

Diagram



Fields

Field	Function
31-26 —	Reserved
25 DMACHN	DMA Channel Number Indicates the DMA channel number to which the VLAN-tagged frame is to be routed if it passes the VLAN tag filter programmed using this field. If the routing based on VLAN tag filter is not necessary, this field does not need to be programmed.
24 DMACHEN	DMA Channel Number Enable Enables or disables the DMA channel number value programmed using the DMACHN field. If this field is 0, the routing does not occur based on the VLAN filter result. The frame is routed on the basis of DA-based DMA channel routing. 0b - Disabled 1b - Enabled
23-21 —	Reserved
20 ERIVLT	Enable Inner VLAN Tag Enables or disables inner VLAN tag comparison. • This field is valid only when the double VLAN tag enable of the filter is set. • If this field and the EDVLP field is 1, the MAC receiver enables operation on the inner VLAN tag (if present). • If this field is 0, the MAC receiver enables operation on the outer VLAN tag (if present). 0b - Disabled 1b - Enabled
19	Enable S-VLAN Match

Table continues on the next page...

Table continued from the previous page...

Field	Function
ERSVLM	Enables or disables S-VLAN match for received frames. This field is valid only when the VLAN tag enable of the filter is set. • If this field is 1, the MAC receiver enables filtering or matching for S-VLAN (type = 88A8h) packets. • If this field is 0, the MAC receiver enables filtering or matching for C-VLAN (type = 8100h) packets. 0b - Disabled 1b - Enabled
18 DOVLTC	Disable VLAN Type Comparison Enables or disables VLAN type comparison. This field is valid only when the VEN field is 1. • If this field is 1, MAC does not check whether the VLAN tag that ERIVLT specifies is of type S-VLAN or C-VLAN. • If this field is 0, MAC filters or matches the VLAN tag that the ERIVLT field specifies only when the VLAN tag type is similar to the one that the ERSVLM field specifies. 0b - Enabled 1b - Disabled
17 ETV	VLAN Comparison Indicates 12-bit or 16-bit VLAN comparison. • This field is valid only when the VEN field is 1. • When the value of this field is 1, a 12-bit VLAN identifier is used for filter and comparison instead of the complete 16-bit VLAN tag. Bits [11:0] of the VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. 0b - 16-bit VLAN comparison 1b - 12-bit VLAN comparison
16 VEN	VLAN Tag Enable Enables or disables the VLAN tag. • If this field is 1, MAC compares the VLAN tag of the received packet with the VLAN tag ID. • If this field is 0, no comparison is performed irrespective of the programming of the other fields. 0b - Disabled 1b - Enabled
15-0 VID	VLAN Tag ID Holds the VLAN tag value that MAC uses for perfect comparison. The field is valid when VEN = 1.

75.17.14 MAC VLAN Tag Filter 3 (MAC_VLAN_Tag_Filter3)

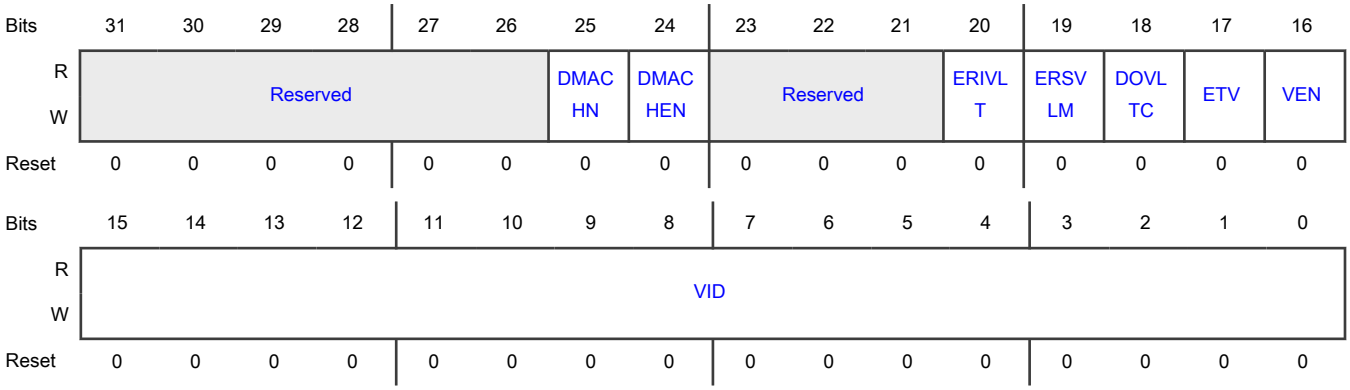
Offset

Register	Offset
MAC_VLAN_Tag_Filter3	54h

Function

Contains VLAN tag filter 3 control information.

Diagram



Fields

Field	Function
31-26 —	Reserved
25 DMACHN	DMA Channel Number Indicates the DMA channel number to which the VLAN-tagged frame is to be routed if it passes the VLAN tag filter programmed using this field. If the routing based on VLAN tag filter is not necessary, this field does not need to be programmed.
24 DMACHEN	DMA Channel Number Enable Enables or disables the DMA channel number value programmed using the DMACHN field. If this field is 0, the routing does not occur based on the VLAN filter result. The frame is routed on the basis of DA-based DMA channel routing. 0b - Disabled 1b - Enabled
23-21 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
20 ERIVLT	Enable Inner VLAN Tag Enables or disables inner VLAN tag comparison. • This field is valid only when the double VLAN tag enable of the filter is set. • If this field and the EDVLP field is 1, the MAC receiver enables operation on the inner VLAN tag (if present). • If this field is 0, the MAC receiver enables operation on the outer VLAN tag (if present). 0b - Disabled 1b - Enabled
19 ERSVLM	Enable S-VLAN Match Enables or disables S-VLAN match for received frames. This field is valid only when the VLAN tag enable of the filter is set. • If this field is 1, the MAC receiver enables filtering or matching for S-VLAN (type = 88A8h) packets. • If this field is 0, the MAC receiver enables filtering or matching for C-VLAN (type = 8100h) packets. 0b - Disabled 1b - Enabled
18 DOVLTC	Disable VLAN Type Comparison Enables or disables VLAN type comparison. This field is valid only when the VEN field is 1. • If this field is 1, MAC does not check whether the VLAN tag that ERIVLT specifies is of type S-VLAN or C-VLAN. • If this field is 0, MAC filters or matches the VLAN tag that the ERIVLT field specifies only when the VLAN tag type is similar to the one that the ERSVLM field specifies. 0b - Enabled 1b - Disabled
17 ETV	VLAN Comparison Indicates 12-bit or 16-bit VLAN comparison. • This field is valid only when the VEN field is 1. • When the value of this field is 1, a 12-bit VLAN identifier is used for filter and comparison instead of the complete 16-bit VLAN tag. Bits [11:0] of the VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. 0b - 16-bit VLAN comparison 1b - 12-bit VLAN comparison
16 VEN	VLAN Tag Enable Enables or disables the VLAN tag.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• If this field is 1, MAC compares the VLAN tag of the received packet with the VLAN tag ID. • If this field is 0, no comparison is performed irrespective of the programming of the other fields. 0b - Disabled 1b - Enabled
15-0 VID	VLAN Tag ID Holds the VLAN tag value that MAC uses for perfect comparison. The field is valid when VEN = 1.

75.17.15 MAC VLAN Hash Table (MAC_VLAN_Hash_Table)

Offset

Register	Offset
MAC_VLAN_Hash_Table	58h

Function

If MAC_VLAN_Tag_Ctrl[VTHM] = 1, the 16-bit VLAN hash table register is used for group address filtering based on the VLAN tag. For hash filtering, the content of the 16-bit VLAN tag or 12-bit VLAN ID (based on MAC_VLAN_Tag_Ctrl[ETV]) in the incoming packet is passed through the CRC logic. The upper four bits of the calculated CRC value are used to index the contents of the VLAN hash table. For example, hash value of 1000b selects bit 8 of the VLAN hash table.

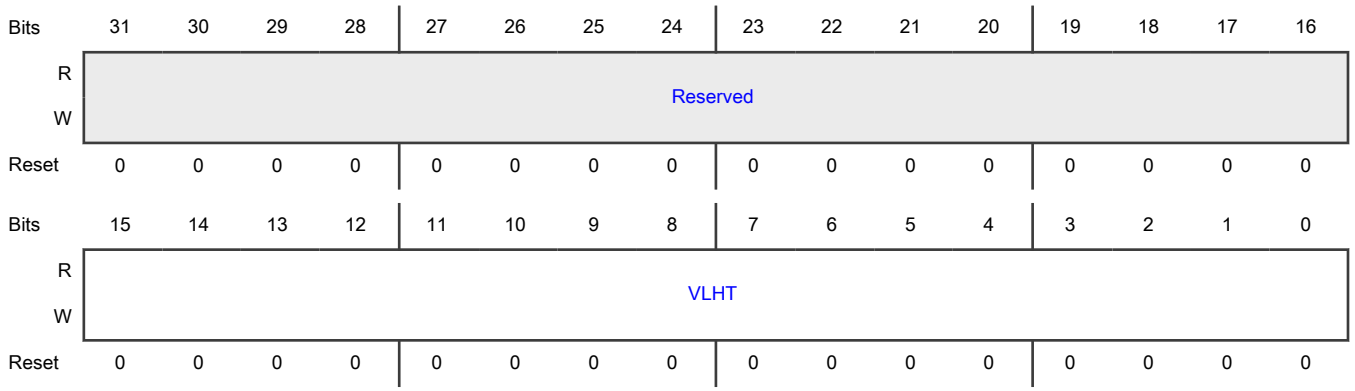
Perform these steps to calculate the hash value of the destination address:

1. Calculate the 32-bit CRC for DA (see section 3.2.8 of IEEE 802.3 for steps to calculate CRC32).
2. Perform bit-wise reversal for the value obtained in step 1.
3. Take the upper 4 bits from the value obtained in step 2.

If this register is configured to be double-synchronized with the (G)MII clock domain, the synchronization is triggered only when bits [15:8] (in Little-Endian mode) or bits [7:0] (in Big-Endian mode) of the register are written to.

If double-synchronization is enabled, consecutive writes to this register must be performed after at least four clock cycles in the destination clock domain.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 VLHT	VLAN Hash Table Contains the 16-bit VLAN hash table.

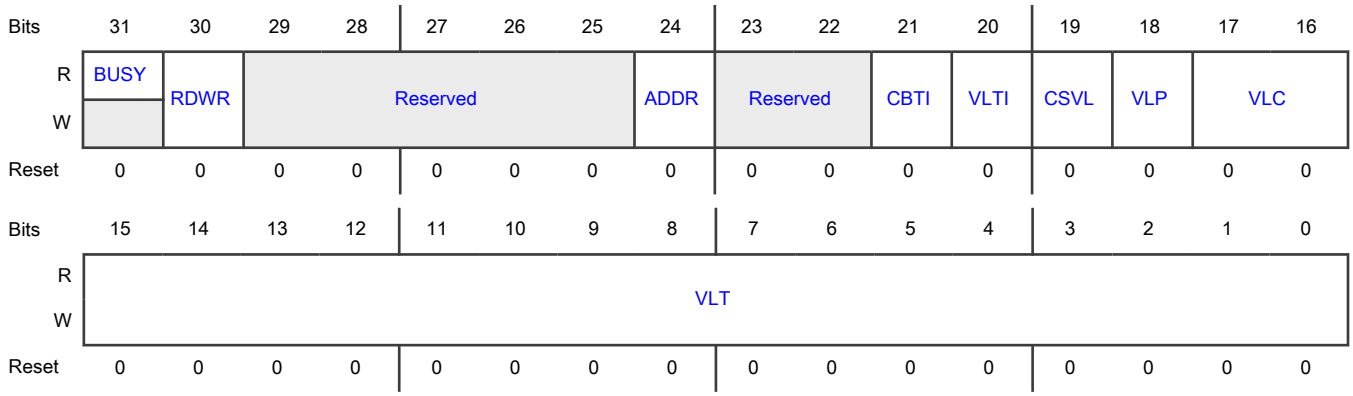
75.17.16 MAC VLAN Inclusion Or Replacement (MAC_VLAN_Incl)

Offset

Register	Offset
MAC_VLAN_Incl	60h

Function
Contains the VLAN tag for insertion or replacement in the transmit packets and the VLAN tag insertion controls.

Diagram



Fields

Field	Function
31 BUSY	Busy Indicates the status of the read or write operation of indirect access to the queue- or channel-specific VLAN inclusion register. The write operation to a register completes when this field becomes 0. For a read operation, the read data is valid when the field becomes 0. The application must make sure that this field becomes 0 before attempting subsequent accesses to this register.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Busy status not detected 1b - Busy status detected
30 RDWR	Read Write Control Controls the read or write operation for indirectly accessing the queue- or channel-specific VLAN inclusion register. • If this field is 1, it indicates the write operation. • If this field is 0, it indicates the read operation. The value of this field is not impacted when the CBTI field becomes 0. 0b - Read operation of indirect access 1b - Write operation of indirect access
29-25 —	Reserved
24 ADDR	Address Selects one of the queue- or channel-specific VLAN inclusion registers for read or write access. The value of this field is not impacted when the CBTI field is 0.
23-22 —	Reserved
21 CBTI	Channel-Based Tag Insertion Indicates the status of channel-based tag insertion. If this field is 1: • The outer VLAN tag is inserted for all the packets that MAC transmits. • The tag value is taken from the queue- or channel-specific VLAN tag register. Also, the VLTI, VLP, VLC, and VLT fields of the register are ignored. • A write operation to byte 3 of this register initiates the read or write access to the indirect register. If this field is 0, the outer VLAN operation is based on the settings of the VLTI, VLP, VLC, and VLT fields of this register. 0b - Disabled 1b - Enabled
20 VLTI	VLAN Tag Input Indicates the status of the VLAN tag input. The value of this field being 1 indicates that the VLAN tag to be inserted or replaced in the transmit packet must be taken from the transmit descriptor.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disabled 1b - Enabled
19 CSVL	C-VLAN Or S-VLAN Indicates whether C-VLAN or S-VLAN is inserted or replaced in the transmitted packets. • If this field is 1, S-VLAN type (88A8h) is inserted or replaced in the 13th and 14th bytes of the transmitted packets. • If this field is 0, C-VLAN type (8100h) is inserted or replaced in the 13th and 14th bytes of the transmitted packets. 0b - C-VLAN 1b - S-VLAN
18 VLP	VLAN Priority Control Indicates the status of VLAN priority control. • If this field is 1, the control bits [17:16] are used for VLAN deletion, insertion, or replacement. • If this field is 0, the mti_vlan_ctrl_i control input is used and bits [17:16] are ignored. 0b - Disabled 1b - Enabled
17-16 VLC	VLAN Tag Control Contains values for VLAN tag control in transmit packets. MAC: • Removes the VLAN type (bytes 13 and 14) and VLAN tag (bytes 15 and 16) of all the transmitted packets with VLAN tags. • Replaces VLT in bytes 15 and 16 of all the VLAN-type transmitted packets (bytes 13 and 14 are 8100h or 88a8h). MAC inserts the following into the packet in the order shown: 1. Type value (8100h or 88A8h) into bytes 13 and 14 2. MAC_VLAN_Tag_Data[VLT] into bytes 15 and 16 This operation is performed on all the transmitted packets, irrespective of whether they already have a VLAN tag. **NOTE** Changes to this field take effect only on the start of a packet. If you write to this field when a packet is being transmitted, only the subsequent packets can use the updated value. That is, the current packet does not use the updated value. 00b - No VLAN tag deletion, insertion, or replacement 01b - VLAN tag deletion

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10b - VLAN tag insertion 11b - VLAN tag replacement
15-0 VLT	VLAN Tag For Transmit Packets Contains the value of the VLAN tag to be inserted or replaced. This value must only be changed during the initialization phase or when the transmit lines are inactive. The following list describes the bits of this field: • Bits [15:13]: User priority • Bit 12: Canonical format indicator (CFI) or drop eligible indicator (DEI) • Bits [11:0]: MAC_VLAN_Tag_Data[VID]

75.17.17 MAC VLAN Inclusion 0 (MAC_VLAN_Incl0)

Offset

Register	Offset
MAC_VLAN_Incl0	60h

Function

Contains the VLAN tag for insertion in the transmit packets from transmit queue 0. The register also contains the VLAN tag insertion controls.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved												CSVL	Reserved		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	VLT															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-20	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
19 CSVL	C-VLAN or S-VLAN When this field is 1, S-VLAN type (0x88A8) is inserted in the 13th and 14th bytes of transmitted packets. When this field is reset, C-VLAN type (0x8100) is inserted in the 13th and 14th bytes of transmitted packets. 0b - C-VLAN type (0x8100) is inserted 1b - S-VLAN type (0x88A8) is inserted
18-16 —	Reserved
15-0 VLT	VLAN Tag for Transmit Packets Contains the value of the VLAN tag to be inserted. The value of this field must only be changed when the transmit lines are inactive or during the initialization phase. The following list describes the bits of this field: • Bits[15:13]: User Priority • Bit 12: Canonical Format Indicator (CFI) or Drop Eligible Indicator (DEI) • Bits[11:0]: VLAN Identifier (VID) field of VLAN tag

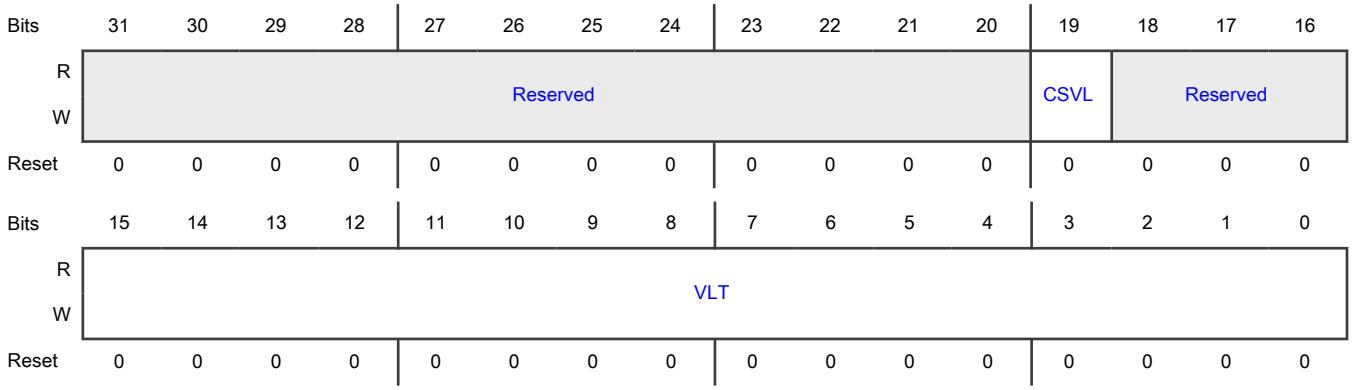
75.17.18 MAC VLAN Inclusion 1 (MAC_VLAN_Incl1)

Offset

Register	Offset
MAC_VLAN_Incl1	60h

Function
Contains the VLAN tag for insertion in the transmit packets from transmit queue 1. The register also contains the VLAN tag insertion controls.

Diagram



Fields

Field	Function
31-20 —	Reserved
19 CSVL	C-VLAN or S-VLAN When this field is 1, S-VLAN type (0x88A8) is inserted in the 13th and 14th bytes of transmitted packets. When this field is reset, C-VLAN type (0x8100) is inserted in the 13th and 14th bytes of transmitted packets. 0b - C-VLAN type (0x8100) is inserted 1b - S-VLAN type (0x88A8) is inserted
18-16 —	Reserved
15-0 VLT	VLAN Tag for Transmit Packets Contains the value of the VLAN tag to be inserted. The value of this field must only be changed when the transmit lines are inactive or during the initialization phase. The following list describes the bits of this field: • Bits[15:13]: User Priority • Bit 12: Canonical Format Indicator (CFI) or Drop Eligible Indicator (DEI) • Bits[11:0]: VLAN Identifier (VID) field of VLAN tag

75.17.19 MAC VLAN Inclusion 2 (MAC_VLAN_Incl2)

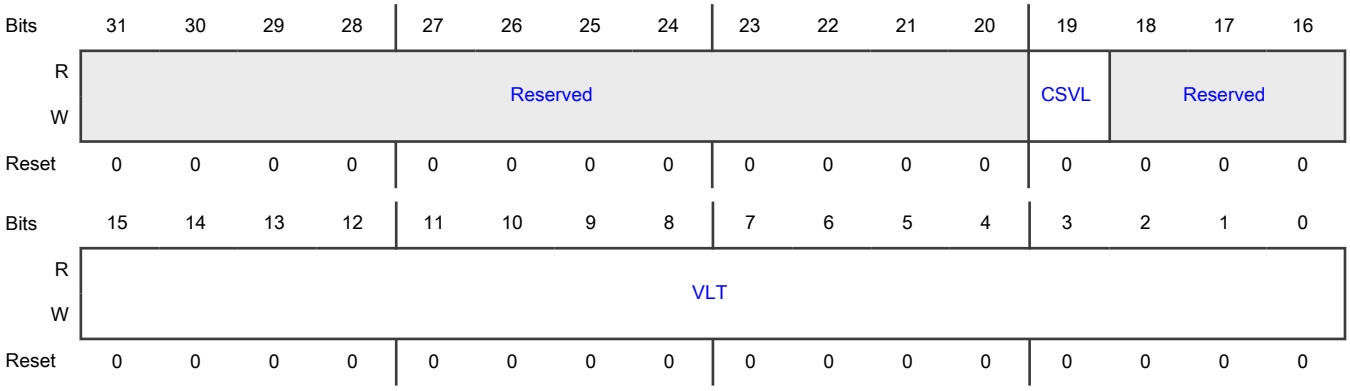
Offset

Register	Offset
MAC_VLAN_Incl2	60h

Function

Contains the VLAN tag for insertion in the transmit packets from transmit queue 2. The register also contains the VLAN tag insertion controls.

Diagram



Fields

Field	Function
31-20 —	Reserved
19 CSVL	C-VLAN or S-VLAN When this field is 1, S-VLAN type (0x88A8) is inserted in the 13th and 14th bytes of transmitted packets. When this field is reset, C-VLAN type (0x8100) is inserted in the 13th and 14th bytes of transmitted packets. 0b - C-VLAN type (0x8100) is inserted 1b - S-VLAN type (0x88A8) is inserted
18-16 —	Reserved
15-0 VLT	VLAN Tag for Transmit Packets Contains the value of the VLAN tag to be inserted. The value of this field must only be changed when the transmit lines are inactive or during the initialization phase. The following list describes the bits of this field: • Bits[15:13]: User Priority • Bit 12: Canonical Format Indicator (CFI) or Drop Eligible Indicator (DEI) • Bits[11:0]: VLAN Identifier (VID) field of VLAN tag

75.17.20 MAC VLAN Inclusion 3 (MAC_VLAN_Incl3)

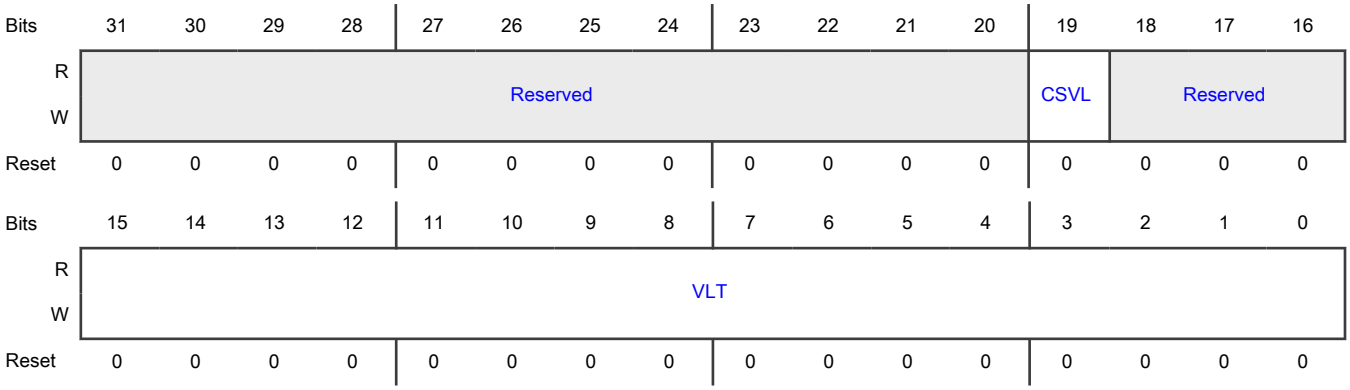
Offset

Register	Offset
MAC_VLAN_Incl3	60h

Function

Contains the VLAN tag for insertion in the transmit packets from transmit queue 3. The register also contains the VLAN tag insertion controls.

Diagram



Fields

Field	Function
31-20 —	Reserved
19 CSVL	C-VLAN or S-VLAN When this field is 1, S-VLAN type (0x88A8) is inserted in the 13th and 14th bytes of transmitted packets. When this field is reset, C-VLAN type (0x8100) is inserted in the 13th and 14th bytes of transmitted packets. 0b - C-VLAN type (0x8100) is inserted 1b - S-VLAN type (0x88A8) is inserted
18-16 —	Reserved
15-0 VLT	VLAN Tag for Transmit Packets Contains the value of the VLAN tag to be inserted. The value of this field must only be changed when the transmit lines are inactive or during the initialization phase. The following list describes the bits of this field:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• Bits[15:13]: User Priority • Bit 12: Canonical Format Indicator (CFI) or Drop Eligible Indicator (DEI) • Bits[11:0]: VLAN Identifier (VID) field of VLAN tag

75.17.21 MAC VLAN Inclusion 4 (MAC_VLAN_Incl4)

Offset

Register	Offset
MAC_VLAN_Incl4	60h

Function

Contains the VLAN tag for insertion in the transmit packets from transmit queue 4. The register also contains the VLAN tag insertion controls.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved												CSVL	Reserved		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	VLT															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-20 —	Reserved
19 CSVL	C-VLAN or S-VLAN When this field is 1, S-VLAN type (0x88A8) is inserted in the 13th and 14th bytes of transmitted packets. When this field is reset, C-VLAN type (0x8100) is inserted in the 13th and 14th bytes of transmitted packets. 0b - C-VLAN type (0x8100) is inserted 1b - S-VLAN type (0x88A8) is inserted

Table continues on the next page...

Table continued from the previous page...

Field	Function
18-16 —	Reserved
15-0 VLT	VLAN Tag for Transmit Packets Contains the value of the VLAN tag to be inserted. The value of this field must only be changed when the transmit lines are inactive or during the initialization phase. The following list describes the bits of this field: • Bits[15:13]: User Priority • Bit 12: Canonical Format Indicator (CFI) or Drop Eligible Indicator (DEI) • Bits[11:0]: VLAN Identifier (VID) field of VLAN tag

75.17.22 MAC VLAN Inclusion 5 (MAC_VLAN_Incl5)

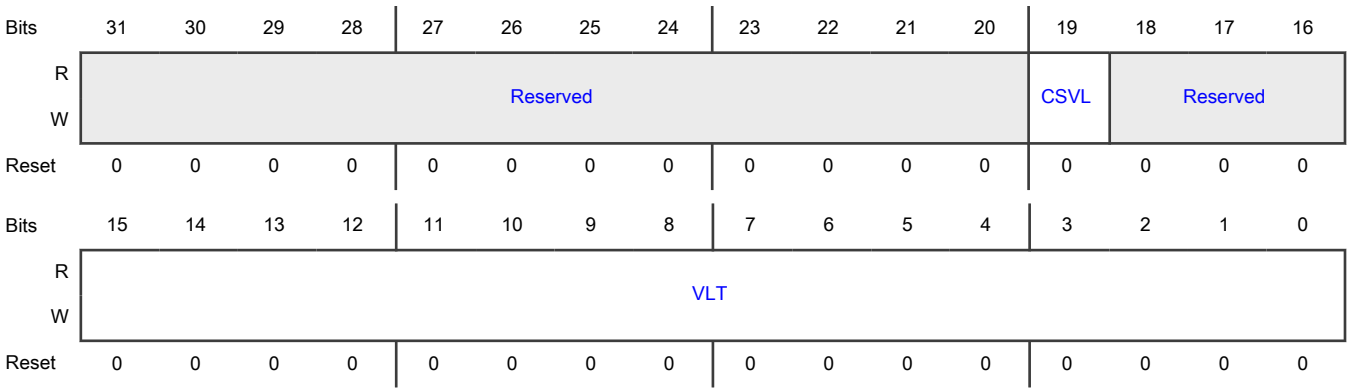
Offset

Register	Offset
MAC_VLAN_Incl5	60h

Function

Contains the VLAN tag for insertion in the transmit packets from transmit queue 5. The register also contains the VLAN tag insertion controls.

Diagram



Fields

Field	Function
31-20 —	Reserved
19 CSVL	C-VLAN or S-VLAN When this field is 1, S-VLAN type (0x88A8) is inserted in the 13th and 14th bytes of transmitted packets. When this field is reset, C-VLAN type (0x8100) is inserted in the 13th and 14th bytes of transmitted packets. 0b - C-VLAN type (0x8100) is inserted 1b - S-VLAN type (0x88A8) is inserted
18-16 —	Reserved
15-0 VLT	VLAN Tag for Transmit Packets Contains the value of the VLAN tag to be inserted. The value of this field must only be changed when the transmit lines are inactive or during the initialization phase. The following list describes the bits of this field: • Bits[15:13]: User Priority • Bit 12: Canonical Format Indicator (CFI) or Drop Eligible Indicator (DEI) • Bits[11:0]: VLAN Identifier (VID) field of VLAN tag

75.17.23 MAC VLAN Inclusion 6 (MAC_VLAN_Incl6)

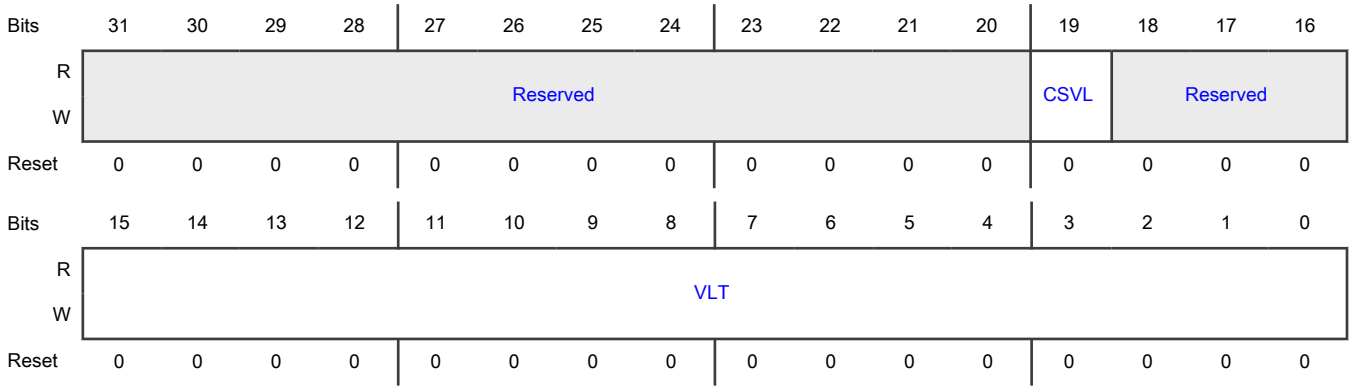
Offset

Register	Offset
MAC_VLAN_Incl6	60h

Function

Contains the VLAN tag for insertion in the transmit packets from transmit queue 6. The register also contains the VLAN tag insertion controls.

Diagram



Fields

Field	Function
31-20 —	Reserved
19 CSVL	C-VLAN or S-VLAN When this field is 1, S-VLAN type (0x88A8) is inserted in the 13th and 14th bytes of transmitted packets. When this field is reset, C-VLAN type (0x8100) is inserted in the 13th and 14th bytes of transmitted packets. 0b - C-VLAN type (0x8100) is inserted 1b - S-VLAN type (0x88A8) is inserted
18-16 —	Reserved
15-0 VLT	VLAN Tag for Transmit Packets Contains the value of the VLAN tag to be inserted. The value of this field must only be changed when the transmit lines are inactive or during the initialization phase. The following list describes the bits of this field: • Bits[15:13]: User Priority • Bit 12: Canonical Format Indicator (CFI) or Drop Eligible Indicator (DEI) • Bits[11:0]: VLAN Identifier (VID) field of VLAN tag

75.17.24 MAC VLAN Inclusion 7 (MAC_VLAN_Incl7)

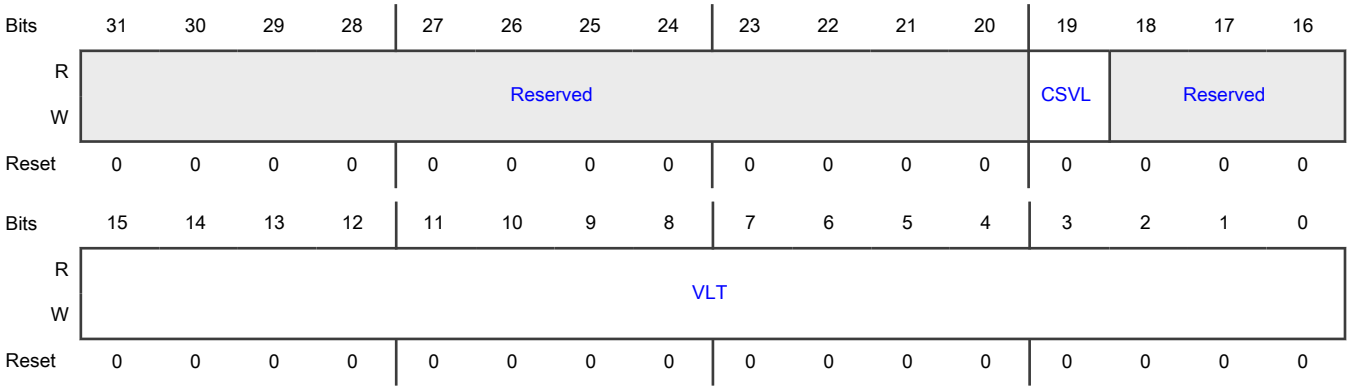
Offset

Register	Offset
MAC_VLAN_Incl7	60h

Function

Contains the VLAN tag for insertion in the transmit packets from transmit queue 7. The register also contains the VLAN tag insertion controls.

Diagram



Fields

Field	Function
31-20 —	Reserved
19 CSVL	C-VLAN or S-VLAN When this field is 1, S-VLAN type (0x88A8) is inserted in the 13th and 14th bytes of transmitted packets. When this field is reset, C-VLAN type (0x8100) is inserted in the 13th and 14th bytes of transmitted packets. 0b - C-VLAN type (0x8100) is inserted 1b - S-VLAN type (0x88A8) is inserted
18-16 —	Reserved
15-0 VLT	VLAN Tag for Transmit Packets Contains the value of the VLAN tag to be inserted. The value of this field must only be changed when the transmit lines are inactive or during the initialization phase. The following list describes the bits of this field: • Bits[15:13]: User Priority • Bit 12: Canonical Format Indicator (CFI) or Drop Eligible Indicator (DEI) • Bits[11:0]: VLAN Identifier (VID) field of VLAN tag

75.17.25 Inner VLAN Tag Inclusion Or Replacement (MAC_Inner_VLAN_Incl)

Offset

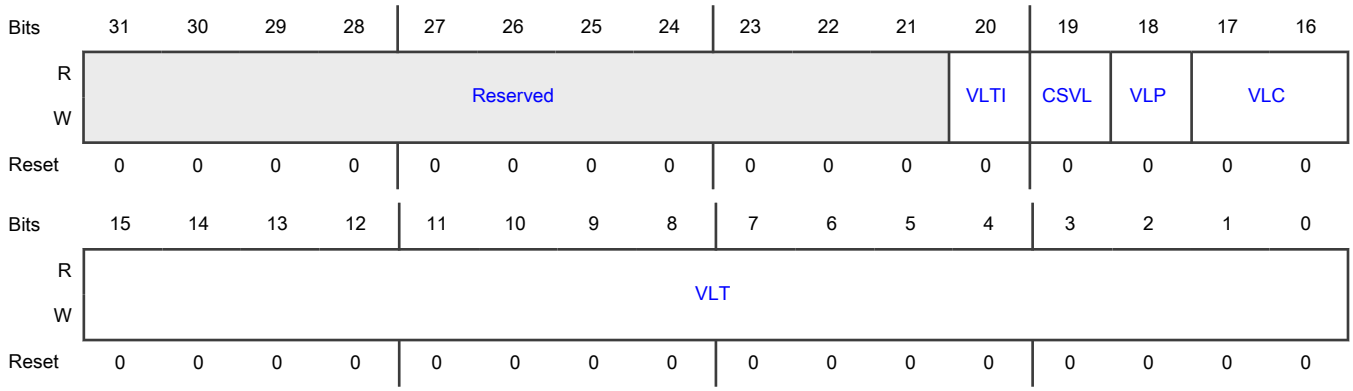
Register	Offset
MAC_Inner_VLAN_Incl	64h

Function

Contains:

- The inner VLAN tag to be inserted or replaced in the transmit packet
- The inner VLAN tag insertion controls

Diagram



Fields

Field	Function
31-21 —	Reserved
20 VLTI	VLAN Tag Input Indicates the status of VLAN tag input. If this field is 1, it indicates that the VLAN tag to be inserted or replaced in the transmit packet must be taken from the transmit descriptor. 0b - Disabled 1b - Enabled
19 CSVL	C-VLAN Or S-VLAN Controls the insertion or replacement type in the 17th and 18th bytes of the transmitted packets. 0b - C-VLAN type (8100h) 1b - S-VLAN type (88A8h)

Table continues on the next page...

Table continued from the previous page...

Field	Function
18 VLP	VLAN Priority Control Indicates the status of VLAN priority control. • If this field is 1, it's used for VLAN deletion, insertion, or replacement. • If this field is 0, the mti_vlan_ctrl_i control input is used and MAC_VLAN_Incl[VLC] is ignored. 0b - Disabled 1b - Enabled
17-16 VLC	VLAN Tag Control in Transmit Packets Indicates the value of VLAN tag control in transmit packets. The following list specifies these values: • 2'b00: No VLAN tag deletion, insertion, or replacement • 2'b01: VLAN tag deletion • 2'b10: VLAN tag insertion • 2'b11: VLAN tag replacement MAC: • Removes the VLAN type (bytes 17 and 18) and VLAN tag (bytes 19 and 20) of all the transmitted packets with VLAN tags • Inserts VLT in bytes 19 and 20 of the packet after inserting the Type value (8100h or 88a8h) in bytes 17 and 18. This operation is performed on all the transmitted packets, irrespective of whether they already have a VLAN tag. • Replaces VLT in bytes 19 and 20 for all the VLAN-type transmitted packets (bytes 17 and 18 are 8100h or 88a8h). **NOTE** Changes to this field take effect only on the start of a packet. If you write to the field when a packet is being transmitted, only the subsequent packets can use the updated value. That is, the current packet does not use the updated value. 00b - No VLAN tag deletion, insertion, or replacement 01b - VLAN tag deletion 10b - VLAN tag insertion 11b - VLAN tag replacement
15-0 VLT	VLAN Tag For Transmit Packets Contains the value of the VLAN tag to be inserted or replaced. The value must only be changed when the transmit lines are inactive or during the initialization phase. The following list describes the bits of this field: • Bits [15:13]: User priority

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• Bit 12: CFI or DEI • Bits [11:0]: MAC_VLAN_Tag_Data[VID]

75.17.26 MAC Q0 Tx Flow Control (MAC_Q0_Tx_Flow_Ctrl)

Offset

Register	Offset
MAC_Q0_Tx_Flow_Ctrl	70h

Function

Controls the generation and reception of the control (pause command) packets by MAC's flow control module. A write to the register with the Busy field = 1 triggers the flow control block to generate a pause packet. The 802.3x specification indicates the way the fields of the control packet are selected, and the pause time value from this register is used in the PT field of the control packet. The Busy field remains 1 until the control packet is transferred onto the cable. You must make sure that the Busy field is 0 before writing to this register.

If the value of the PFCE field in the MAC_Rx_Flow_Ctrl register is 1, this register controls the generation of priority flow control (PFC) frames with priorities mapped according to MAC_RxQ_Ctrl2[PSRQ0].

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PT															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								DZPQ	PLT			Reserved		TFE	FCB_ BPA
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 PT	Pause Time Holds the value to be used in this field in the transmit control packet. If the bits of this field are configured to be double-synchronized with the (G)MII clock domain, consecutive writes to this register must be performed only after at least four clock cycles in the destination clock domain.

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-8 —	Reserved
7 DZPQ	Disable Zero-Quanta Pause Enables or disables zero-quanta pause packet generation. • If this field is 1, it disables the automatic generation of the zero-quanta pause packets that the flow-control signal from the FIFO layer (MTL or external sideband flow control signal sbd_flowctrl_i or mti_flowctrl_i) deasserts. • If this field is 0, normal operations with automatic zero-quanta pause packet generation are enabled. 0b - Enabled 1b - Disabled
6-4 PLT	Pause Low Threshold Configures the threshold of the pause timer at which the input flow control signal mti_flowctrl_i (or sbd_flowctrl_i) is checked for automatic retransmission of the pause packet. The threshold values must always be less than the pause time configured in bits [31:16]. For example, if the PT field = 100H (256 slot times), and the PLT field = 001, a second pause packet automatically transmits if the mti_flowctrl_i signal is asserted at 228 (256-28) slot times after the first pause packet transmits. This list provides the threshold values for different values: • The slot time is defined as the time taken to transmit 512 bits (64 bytes) on GMII or MII. • This (approximate) computation is based on the packet size (64, 1518, 2000, 9018, 16384, or 32768) + two pause packet sizes + IPG in slot times. 000b - Pause time minus 4 slot times (PT is 4 slot times) 001b - Pause time minus 28 slot times (PT is 28 slot times) 010b - Pause time minus 36 slot times (PT is 36 slot times) 011b - Pause time minus 144 slot times (PT is 144 slot times) 100b - Pause time minus 256 slot times (PT is 256 slot times) 101b - Pause time minus 512 slot times (PT is 512 slot times) 110b - Reserved
3-2 —	Reserved
1 TFE	Transmit Flow Control Enable Indicates whether transmit flow control is enabled or disabled. • Full-Duplex mode: If this field is 1, MAC enables the flow control operation to the transmit pause packets. When the field becomes 0, MAC's flow control operation disables and MAC does not transmit any pause packets.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• Half-Duplex mode: If this field is 1, MAC enables the backpressure operation. When the field becomes 0, the backpressure operation disables. 0b - Disabled 1b - Enabled
0 FCB_BPA	Flow Control Busy Or Backpressure Activate Initiates if a pause packet is in Full-Duplex mode and activates the backpressure function in Half-Duplex mode if the TFE field is 1. • Full-Duplex mode: In this mode, FCB_BPA must be read as 1'b0 before writing to this register. To initiate a pause packet, the application must set this field to 1'b1. During control packet transfer, the field's value continues to be 1 to indicate that a packet transmission is in progress. When pause packet transmission is complete, MAC resets this field to 1'b0. You must not write to this register until this field becomes 0. • Half-Duplex mode: If FCB_BPA = 1 and TFE = 1 too, MAC asserts backpressure in this mode. During the backpressure function, when MAC receives a new packet, the transmitter starts sending a JAM pattern resulting in a collision. This control register field is logically ORed with the mti_flowctrl_i input signal for the backpressure function. When you configure MAC using Full-Duplex mode, this field automatically becomes 0. Access restrictions apply to this field: writing 1 sets it, writing 0 has no effect, and the field is self-clearing. 0b - Flow control busy or backpressure activate is disabled 1b - Flow control busy or backpressure activate is enabled

75.17.27 MAC Receive Flow Control (MAC_Rx_Flow_Ctrl)

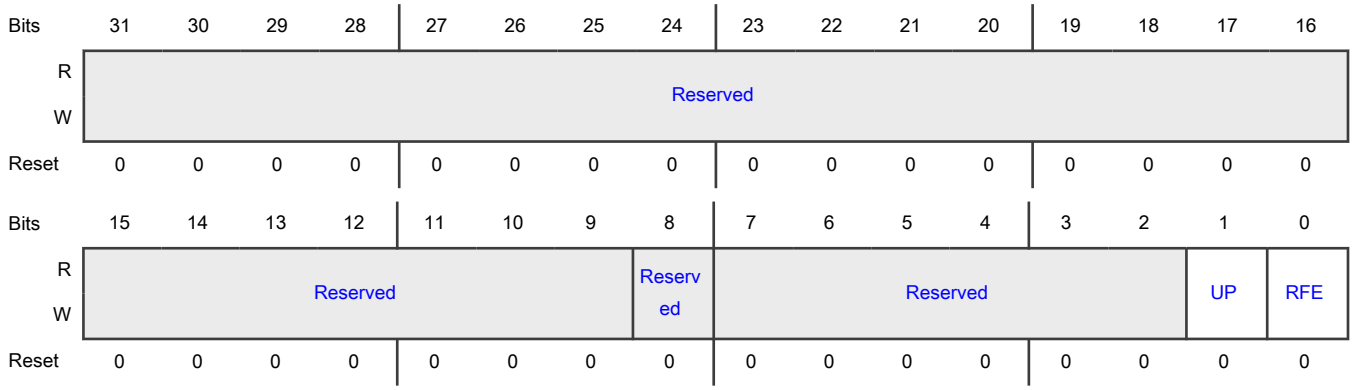
Offset

Register	Offset
MAC_Rx_Flow_Ctrl	90h

Function

Controls the pausing of MAC transmit based on the received pause packet.

Diagram



Fields

Field	Function
31-9 —	Reserved
8 —	Reserved
7-2 —	Reserved
1 UP	Unicast Pause Packet Detect Indicates whether the unicast pause packet is enabled or disabled. A pause packet is processed when it has the unique multicast address specified in IEEE 802.3. • If this field is 1, MAC can detect pause packets with a unicast address of the station. This address must comply with the specifications in MAC Address 0 High (MAC_Address0_High) and MAC Address 0 Low (MAC_Address0_Low). • If this field is 0, MAC only detects pause packets with a unique multicast address. **NOTE** MAC does not process a pause packet if the multicast address is different from the unique multicast address. This applies to the received PFC packet if PFC is enabled. The unique multicast address (01_80_C2_00_00_01h) complies with the IEEE 802.1 Qbb-2011 specifications. 0b - Disabled 1b - Enabled
0 RFE	Receive Flow Control Enable Indicates whether the receive flow control is enabled or disabled. • If this field is 1 and MAC is operating in Full-Duplex mode, MAC decodes the received pause packet and disables its transmitter for a specified (pause) time.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• If this field is 0 or if MAC is operating in Half-Duplex mode, the decode function of the pause packet disables. When PFC is enabled, the flow control for PFC packets is enabled too. MAC decodes the received PFC packet and disables the transmit queue, with matching priorities, for a duration of the received pause time. 0b - Disabled 1b - Enabled

75.17.28 MAC RxQ Control 4 (MAC_RxQ_Ctrl4)

Offset

Register	Offset
MAC_RxQ_Ctrl4	94h

Function

Controls the routing of unicast and multicast packets that fail the destination or source address filter to the receive queues.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved														VFFQ	VFFQE
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved						MFFQ	MFFQE	Reserved						UFFQ	UFFQE
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-18 —	Reserved
17 VFFQ	VLAN Tag Filter Fail Packets Queue Holds the receive queue number to which the tagged packets failing the destination or source address filter (and UFFQE and MFFQE not being 1) or failing the VLAN tag filter must be routed to. This field is valid only when the VFFQE field = 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
16 VFFQE	VLAN Tag Filter Fail Packets Queuing Enable Indicates whether VLAN tag filter fail packet queuing is enabled or disabled. • If this field is 1, the tagged packets that fail the destination or source address filter or fail the VLAN tag filter are routed to the receive queue number programmed using the VFFQ field. • If this field is 0, the tagged packets that fail the destination or source address filter or fail the VLAN tag filter are routed based on the other routing options. This field is valid only when MAC_Packet_Filter[RA] = 1. 0b - Disabled 1b - Enabled
15-10 —	Reserved
9 MFFQ	Multicast Address Filter Fail Packets Queue Holds the receive queue number to which the multicast packets failing the destination or source address filter are routed to. This field is valid only when the MFFQE field = 1.
8 MFFQE	Multicast Address Filter Fail Packets Queuing Enable Indicates whether multicast address filter fail packet queuing is enabled or disabled. • If this field is 1, the multicast packets that fail the destination or source address filter are routed to the receive queue number programmed using the MFFQ field. • If this field is 0, the multicast packets that fail the destination or source address filter are routed based on the other routing options. This field is valid only when MAC_Packet_Filter[RA] = 1. 0b - Disabled 1b - Enabled
7-2 —	Reserved
1 UFFQ	Unicast Address Filter Fail Packets Queue Holds the receive queue number to which the unicast packets failing the destination or source address filter are routed to. This field is valid only when the UFFQE field = 1.
0 UFFQE	Unicast Address Filter Fail Packets Queuing Enable Indicates whether unicast address filter fail packet queuing is enabled or disabled. • When the value of this field is 1, the unicast packets that fail the destination or source address filter are routed to the receive queue number programmed using the UFFQ field. • If this field is 0, the unicast packets that fail the destination or source address filter are routed based on the other routing options.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is valid only when MAC_Packet_Filter[RA] = 1. 0b - Disabled 1b - Enabled

75.17.29 MAC RxQ Control 0 (MAC_RxQ_Ctrl0)

Offset

Register	Offset
MAC_RxQ_Ctrl0	A0h

Function

Controls queue management in the MAC receiver.

NOTE

In multiple receive queues configuration, all the queues are disabled by default. Enable the receive queue by programming the corresponding field in this register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		RXQ1EN		RXQ0EN	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-14 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
13-12 —	Reserved
11-10 —	Reserved
9-8 —	Reserved
7-6 —	Reserved
5-4 —	Reserved
3-2 RXQ1EN	Receive Queue 1 Enable Performs in a manner similar to the RXQ0EN field and indicates if receive queue 1 is enabled for AV or DCB. 00b - Queue not enabled 01b - Queue enabled for AV 10b - Queue enabled for DCB/generic 11b - Reserved
1-0 RXQ0EN	Receive Queue 0 Enable Indicates whether receive queue 0 is enabled for AV or DCB. 00b - Queue not enabled 01b - Queue enabled for AV 10b - Queue enabled for DCB/generic 11b - Reserved

75.17.30 Receive Queue Control 1 (MAC_RxQ_Ctrl1)

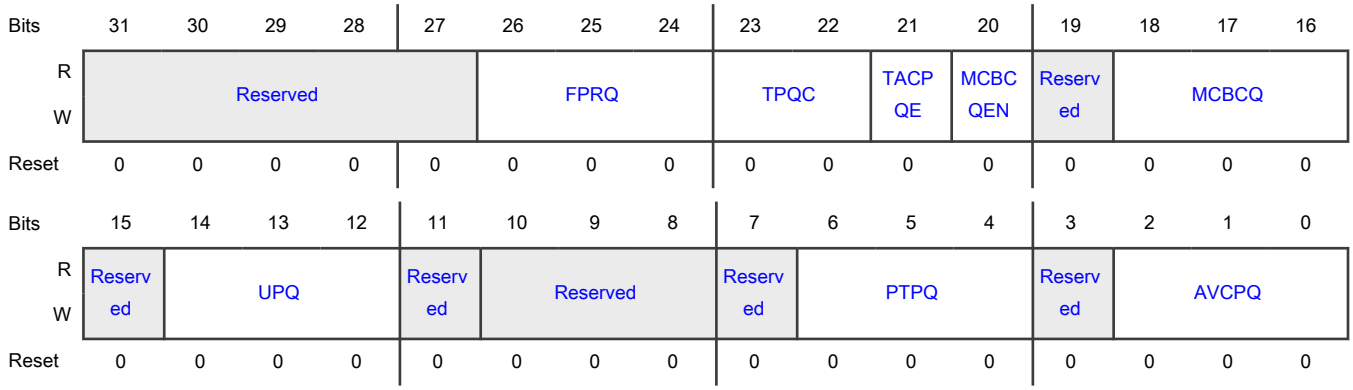
Offset

Register	Offset
MAC_RxQ_Ctrl1	A4h

Function

Controls the routing of multicast, broadcast, AV, DCB, and untagged packets to the receive queues.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-24 FPRQ	Frame Preemption Residue Queue Holds the receive queue number to which the residual preemption frames must be forwarded. Preemption frames that are tagged and pass the SA, DA, or VLAN filtering are routed based on the settings of the PSRQ fields in MAC RxQ Control 2 (MAC_RxQ_Ctrl2). All other frames are treated as residual frames and are routed to the queue number mentioned in this field. Queue 0 is used as the default queue for express frames, so you cannot write 0 to this field.
23-22 TPQC	Tagged PTP Over Ethernet Packets Queuing Control Controls the routing of the VLAN-tagged PTPoE packets. These are the allowed programmable options: • 2'b00: VLAN-tagged PTPoE packets are routed as generic VLAN-tagged packets (based on the PSRQ fields in MAC RxQ Control 2 (MAC_RxQ_Ctrl2) for only non-AV enabled receive queues). • 2'b01: VLAN-tagged PTPoE packets are routed to the receive queue that the PTPQ field of this register specifies (this Rx queue can be enabled for AV or non-AV traffic). • 2'b10: VLAN-tagged PTPoE packets are routed to only AV-enabled receive queues based on the settings of the PSRQ fields.
21 TACPQE	Tagged AV Control Packets Queuing Enable Indicates the status of tagged AV control packet queuing. • If this field is 1, MAC routes the received tagged AV control packets to the receive queue that the AVCPQ field specifies. • If this field is 0, MAC routes the received tagged AV control packets based on the tag priority matching the PSRQ fields in MAC RxQ Control 2 (MAC_RxQ_Ctrl2). 0b - Disabled 1b - Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
20 MCBCQEN	Multicast And Broadcast Queue Enable Specifies whether multicast or broadcast packet routing to the receive queue is enabled or disabled. The multicast or broadcast packets must be routed to the receive queue specified in the MCBCQ field. 0b - Disabled 1b - Enabled
19 —	Reserved
18-16 MCBCQ	Multicast And Broadcast Queue Specifies the receive queue onto which multicast or broadcast packets are routed. Any receive queue enabled for Generic, DCB, or AV traffic can be used to route the multicast or broadcast packets. 000b - Receive queue 0 001b - Receive queue 1 010b - Receive queue 2 011b - Receive queue 3 100b - Receive queue 4 101b - Receive queue 5 110b - Receive queue 6 111b - Receive queue 7
15 —	Reserved
14-12 UPQ	Untagged Packet Queue Indicates the receive queue to which the untagged packets need to be routed. Any receive queue enabled for generic, DCB, or AV traffic can be used to route the untagged packets. 000b - Receive queue 0 001b - Receive queue 1 010b - Receive queue 2 011b - Receive queue 3 100b - Receive queue 4 101b - Receive queue 5 110b - Receive queue 6 111b - Receive queue 7
11 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
10-8 —	Reserved
7 —	Reserved
6-4 PTPQ	PTP Packets Queue Specifies the receive queue on which the PTP packets sent over the Ethernet payload (not over IPv4 or IPv6) are routed. When MAC_Timestamp_Control[AV8021ASMEN] = 1, only untagged PTP over Ethernet packets are routed to a receive queue. If this field is not 1, then based on the settings of the TPQC field, both tagged and untagged PTPoE packets can be routed to this receive queue. 000b - Receive queue 0 001b - Receive queue 1 010b - Receive queue 2 011b - Receive queue 3 100b - Receive queue 4 101b - Receive queue 5 110b - Receive queue 6 111b - Receive queue 7
3 —	Reserved
2-0 AVCPQ	AV Untagged Control Packets Queue Specifies the receive queue on which the received AV tagged and untagged control packets are routed. The AV tagged (when the TACPQE field = 1) and untagged control packets are routed to the receive queue that this field specifies. 000b - Receive queue 0 001b - Receive queue 1 010b - Receive queue 2 011b - Receive queue 3 100b - Receive queue 4 101b - Receive queue 5 110b - Receive queue 6 111b - Receive queue 7

75.17.31 MAC RxQ Control 2 (MAC_RxQ_Ctrl2)

Offset

Register	Offset
MAC_RxQ_Ctrl2	A8h

Function

Controls the routing of tagged packets based on the settings of MAC_Ext_Configuration[USP] for packets received in receive queues 0-3.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved								Reserved							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PSRQ1								PSRQ0							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-16 —	Reserved
15-8 PSRQ1	Priorities Selected In Receive Queue 1 Determines the priorities assigned to receive queue 1. All the packets with priorities that match the values defined in this field are routed to receive queue 1. For example, if PSRQ1[4] = 1, packets with MAC_Ext_Configuration[USP] equal to 4 are routed to receive queue 1. You must ensure that the content of this field is mutually exclusive to the PSRQ fields for other queues. This means that the same priority is not mapped to multiple receive queues.
7-0 PSRQ0	Priorities Selected In Receive Queue 0 Determines the priorities assigned to receive queue 0. All the packets with priorities that match the values defined in this field are routed to receive queue 0. For example, if PSRQ0[5] = 1, packets with MAC_Ext_Configuration[USP] equal to 5 are routed to receive queue 0. You must ensure that the content of this field is mutually exclusive to the PSRQ fields for other queues. This means that the same priority is not mapped to multiple receive queues.

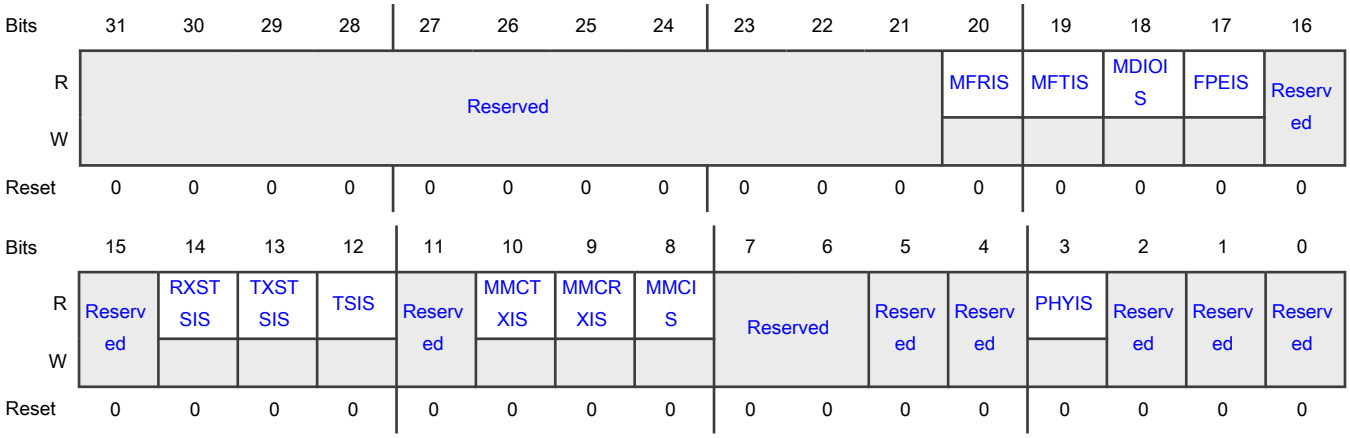
75.17.32 MAC Interrupt Status (MAC_Interrupt_Status)

Offset

Register	Offset
MAC_Interrupt_Status	B0h

Function
Contains the status of interrupts.

Diagram



Fields

Field	Function
31-21 —	Reserved
20 MFRIS	MMC FPE Receive Interrupt Status Indicates the status of MMC FPE receive interrupt. This field becomes 1 when any of the fields in MMC Receive Packet Assembly Error Counter Interrupt Status (MMC_FPE_Rx_Interrupt) is 1. The field becomes 0 when all the other fields in MMC Receive Packet Assembly Error Counter Interrupt Status (MMC_FPE_Rx_Interrupt) become 0 too. 0b - Inactive 1b - Active
19 MFTIS	MMC FPE Transmit Interrupt Status Indicates the status of MMC FPE transmit interrupt. This field becomes 1 when an interrupt generates in MMC Transmit FPE Fragment Counter Interrupt Status (MMC_FPE_Tx_Interrupt). The field becomes 0 when all the other fields in this register become 0 too. This field is valid only when you select the Enable MAC Management Counters (MMC) option along with FPE support.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Inactive 1b - Active
18 MDIOIS	MDIO Interrupt Status Indicates the status of an interrupt event after the completion of an MDIO operation. To reset this field, you must read this field or write 1 to it when MAC_CSR_SW_Ctrl[RCWE] = 1. Access restrictions apply to this field. It clears on a read or write of 1 when MAC_CSR_SW_Ctrl[RCWE] = 1. The field automatically becomes 1 on an internal event. 0b - Inactive 1b - Active
17 FPEIS	Frame Preemption Interrupt Status Indicates the status of an interrupt event during the frame preemption operation (the value of RVER, RRSP, TVER, and TRSP fields of MAC FPE Control STS (MAC_FPE_CTRL_STS) = 1). To reset this field, you must clear the event in MAC FPE Control STS (MAC_FPE_CTRL_STS) that caused the interrupt. 0b - Inactive 1b - Active
16 —	Reserved
15 —	Reserved
14 RXSTSIS	Receive Status Interrupt Indicates the status of received packets. This field becomes 1 when MAC_Rx_Tx_Status[RWT] = 1. The field becomes 0 when you read the corresponding interrupt source field in MAC Rx Transmit Status (MAC_Rx_Tx_Status), or if you write to this interrupt source field when MAC_CSR_SW_Ctrl[RCWE] = 1. The field becomes 0 in either of these circumstances: • You read the corresponding interrupt-source field in MAC Rx Transmit Status (MAC_Rx_Tx_Status). • You write 1 to the corresponding interrupt-source field in MAC Rx Transmit Status (MAC_Rx_Tx_Status) when MAC_CSR_SW_Ctrl[RCWE] = 1. 0b - Inactive 1b - Active
13 TXSTSIS	Transmit Status Interrupt Indicates the status of transmitted packets.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field becomes 1 when any of the following fields is 1 in MAC Rx Transmit Status (MAC_Rx_Tx_Status): • Excessive collision (EXCOL) • Late collision (LCOL) • Excessive deferral (EXDEF) • Loss of carrier (LCARR) • No carrier (NCARR) • Jabber timeout (TJT) This field becomes 0 in either of these circumstances: • You read the corresponding interrupt-source field in MAC Rx Transmit Status (MAC_Rx_Tx_Status). • You write 1 to the corresponding interrupt-source field in MAC Rx Transmit Status (MAC_Rx_Tx_Status) when MAC_CSR_SW_Ctrl[RCWE] = 1. 0b - Inactive 1b - Active
12 TSIS	Timestamp Interrupt Status Indicates the status of timestamp interrupt. If the timestamp feature is enabled, this field becomes 1 when any of the following conditions is true: • The system time value is equal to or exceeds the value specified in MAC PPS1 Target Time In Seconds (MAC_PPS1_Target_Time_Seconds) and MAC PPS1 Target Time In Nanoseconds (MAC_PPS1_Target_Time_Nanoseconds). • There is an overflow in MAC System Time In Seconds (MAC_System_Time_Seconds). • The target time error occurred, which means, the programmed target time already elapsed. In configurations other than EQOS_CORE, when the drop transmit status is enabled in MTL, this field becomes 1 when the captured transmit timestamp is updated in MAC Transmit Timestamp Status In Nanoseconds (MAC_Tx_Timestamp_Status_Nanoseconds) and MAC Transmit Timestamp Status In Seconds (MAC_Tx_Timestamp_Status_Seconds). The field becomes 0 in either of these circumstances: • You read the corresponding interrupt-source field in MAC Rx Transmit Status (MAC_Rx_Tx_Status). • You write 1 to the corresponding interrupt-source field in MAC Rx Transmit Status (MAC_Rx_Tx_Status) when MAC_CSR_SW_Ctrl[RCWE] = 1. 0b - Inactive 1b - Active
11 —	Reserved
10 MMCTXIS	MMC Transmit Interrupt Status Indicates the status of MMC transmit interrupt.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field: • Becomes 1 when any field in MMC Transmit Interrupt (MMC_Tx_Interrupt) becomes 1. • Becomes 0 when all the fields in MMC Transmit Interrupt (MMC_Tx_Interrupt)return to 0. This field is valid only when you select the Enable MAC Management Counters (MMC) option. 0b - Inactive 1b - Active
9 MMCRXIS	MMC Receive Interrupt Status Indicates the status of MMC receive interrupt. This field: • Becomes 1 when any field in MMC Receive Interrupt (MMC_Rx_Interrupt) becomes 1. • Becomes 0 when all the fields in MMC Receive Interrupt (MMC_Rx_Interrupt)return to 0. This field is valid only when you select the Enable MAC Management Counters (MMC) option. 0b - Inactive 1b - Active
8 MMCIS	MMC Interrupt Status Indicates the status of MMC interrupt. This field becomes 1 when MMCRXIS = MMCTXIS = 1, and the field becomes 0 only when MMCRXIS = MMCTXIS = 0. MMCIS is valid only when you select the Enable MAC Management Counters (MMC) option. 0b - Inactive 1b - Active
7-6 —	Reserved
5 —	Reserved
4 —	Reserved
3 PHYIS	PHY Interrupt Indicates whether the PHY interrupt is detected. This field becomes 1 when rising edge is detected on the phy_intr_i input signal. The field becomes 0 when you read this register (or if you write 1 to this field when MAC_CSR_SW_Ctrl[RCWE] = 1). 0b - Not detected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Detected
2 —	Reserved
1 —	Reserved
0 —	Reserved

75.17.33 MAC Interrupt Enable (MAC_Interrupt_Enable)

Offset

Register	Offset
MAC_Interrupt_Enable	B4h

Function
Contains masks for generating MAC interrupts.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved													MDIOIE	FPEIE	Reserved
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved	RXSTSIE	TXSTSIE	TSIE	Reserved						Reserved	Reserved	PHYIE	Reserved	Reserved	Reserved
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-19 —	Reserved
18	MDIO Interrupt Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
MDIOIE	Indicates whether the MDIO interrupt is enabled or disabled. If this field is 1, it enables the assertion of the interrupt when MAC_Interrupt_Status[MDIOIS] = 1. 0b - Disabled 1b - Enabled
17 FPEIE	Frame Preemption Interrupt Enable Indicates whether the frame preemption interrupt is enabled or disabled. If this field is 1, it enables the assertion of the interrupt when MAC_Interrupt_Status[FPEIS] = 1. 0b - Disabled 1b - Enabled
16 —	Reserved
15 —	Reserved
14 RXSTSIE	Receive Status Interrupt Enable Indicates whether the receive status interrupt is enabled or disabled. If this field is 1, it enables the assertion of the interrupt signal because of the setting of MAC_Interrupt_Status[RXSTSIS]. 0b - Disabled 1b - Enabled
13 TXSTSIE	Transmit Status Interrupt Enable Indicates whether the timestamp status interrupt is enabled or disabled. If this field is 1, it enables the assertion of the interrupt signal because of the setting of MAC_Interrupt_Status[TXSTSIS]. 0b - Disabled 1b - Enabled
12 TSIE	Timestamp Interrupt Enable Indicates whether the timestamp interrupt is enabled or disabled. If this field is 1, it enables the assertion of the interrupt signal because of the setting MAC_Interrupt_Status[TSIS]. 0b - Disabled 1b - Enabled
11-6	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
5 —	Reserved
4 —	Reserved
3 PHYIE	PHY Interrupt Enable Indicates whether the assertion of the interrupt signal is enabled or disabled. If this field is 1, it enables the assertion of the interrupt signal because of the setting of MAC_Interrupt_Status[PHYIS]. 0b - Disabled 1b - Enabled
2 —	Reserved
1 —	Reserved
0 —	Reserved

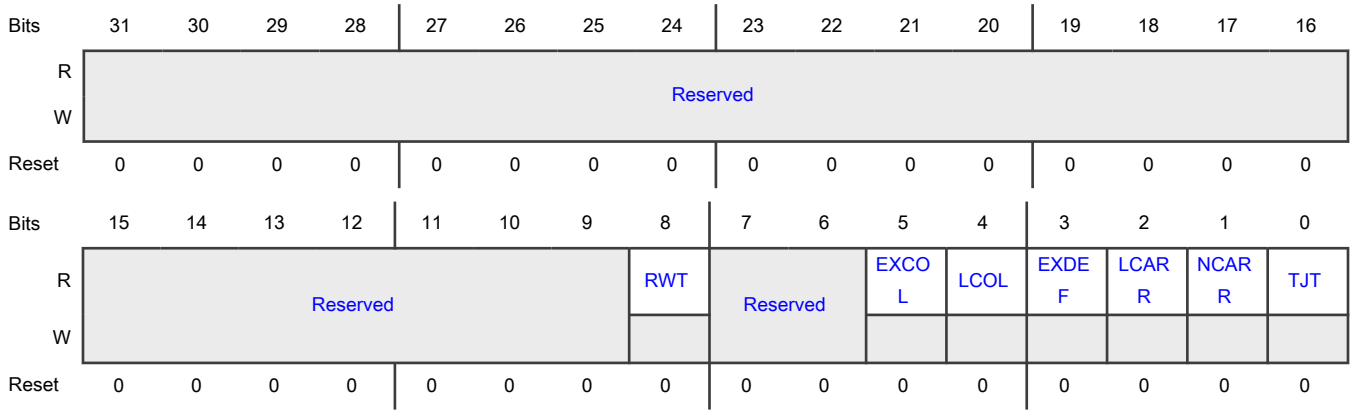
75.17.34 MAC Rx Transmit Status (MAC_Rx_Tx_Status)

Offset

Register	Offset
MAC_Rx_Tx_Status	B8h

Function
Contains the receive and transmit error status.

Diagram



Fields

Field	Function
31-9 —	Reserved
8 RWT	Receive Watchdog Timeout Indicates the status of receive watchdog. This field becomes 1 when a packet with a length greater than 2,048 bytes is received (10,240 bytes when Jumbo Packet mode is enabled) and MAC_Configuration[WD] = 0. RWT becomes 1 when a packet with length greater than 16,383 bytes is received and MAC_Configuration[WD] = 1. Access restrictions apply to RWT. It becomes 0 on read or write of 1 when MAC_CSR_SW_Ctrl[RCWE] = 1. RWT automatically becomes 1 on an internal event occurrence. 0b - No receive watchdog timed out 1b - Receive watchdog timed out
7-6 —	Reserved
5 EXCOL	Excessive Collisions Indicates the status of excessive collision. If MTL_Operation_Mode[DTXSTS] = 1, this field, EXCOL, indicates that the transmission aborted after 16 successive collisions while attempting to transmit the current packet. If MAC_Configuration[DR] = 1, EXCOL becomes 1 after the first collision and the packet transmission is aborted. Access restrictions apply to EXCOL. It becomes 0 on read or write of 1 when MAC_CSR_SW_Ctrl[RCWE] = 1. EXCOL automatically becomes 1 on an internal event occurrence. 0b - No collision 1b - Excessive collision is sensed
4	Late Collision

Table continues on the next page...

Table continued from the previous page...

Field	Function
LCOL	Indicates the status of late collision. If MTL_Operation_Mode[DTXSTS] = 1, this field, LCOL, indicates that the packet transmission aborted because a collision occurred after the collision window (64 bytes including preamble in MII mode; 512 bytes including preamble and carrier extension in GMII mode). This field is invalid if an underflow error occurs. Access restrictions apply to LCOL. It becomes 0 on read or write of 1 when MAC_CSR_SW_Ctrl[RCWE] = 1. LCOL automatically becomes 1 on an internal event occurrence. 0b - No collision 1b - Late collision is sensed
3 EXDEF	Excessive Deferral Indicates whether the transmission ended because of excessive deferral. If MTL_Operation_Mode[DTXSTS] = 1 and MAC_Configuration[DC] = 1 too, this field, EXDEF, indicates that the transmission ended because of excessive deferral of over 24,288 bit times (155,680 in 1000/2500 Mbit/s mode or when the jumbo packet is enabled). Access restrictions apply to EXDEF. It becomes 0 on read or write of 1 when MAC_CSR_SW_Ctrl[RCWE] = 1. EXDEF automatically becomes 1 on an internal event occurrence. 0b - No excessive deferral 1b - Excessive deferral
2 LCARR	Loss of Carrier Indicates the status of carrier signal. If MTL_Operation_Mode[DTXSTS] = 1, this field, LCARR, indicates that the loss of carrier occurred during packet transmission, which means, the PHY_CRS_I signal was inactive for one or more transmission clock periods during packet transmission. The field is valid only for packets transmitted without collision. Access restrictions apply to LCARR. It becomes 0 on read or write of 1 when MAC_CSR_SW_Ctrl[RCWE] = 1. LCARR automatically becomes 1 on an internal event occurrence. 0b - Carrier is present 1b - Loss of carrier
1 NCARR	No Carrier Indicates whether the carrier signal is present from the PHY at the end of preamble transmission. If MTL_Operation_Mode[DTXSTS] = 1, NCARR indicates that the carrier signal from the PHY is absent at the end of preamble transmission. Access restrictions apply to NCARR. It becomes 0 after either of these actions: • You read NCARR. • You write 1 to NCARR when MAC_CSR_SW_Ctrl[RCWE] = 1. NCARR automatically becomes 1 after an internal event occurs.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Present 1b - Absent
0 TJT	Transmit Jabber Timeout Indicates whether a transmit-jabber timer timeout occurred. The timeout occurs after any of these events: • MAC_Configuration[JD] = 0 and the normal packet size exceeds 2,048 bytes. • MAC_Configuration[JD] = 0 and the jumbo packet size exceeds 10240 bytes. • MAC_Configuration[JD] = 1 and the normal packet size exceeds 16383 bytes. Access restrictions apply to TJT. It becomes 0 after either of these actions: • You read TJT. • You write 1 to TJT when MAC_CSR_SW_Ctrl[RCWE] = 1. TJT automatically becomes 1 after an internal event occurs. 0b - No transmit jabber timeout occurred 1b - Transmit jabber timeout occurred

75.17.35 MAC Version (MAC_Version)

Offset

Register	Offset
MAC_Version	110h

Function
Identifies the version of the module.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CFGVER								IPVER							
W																
Reset	0	0	0	1	0	0	0	0	0	1	0	1	0	0	0	1

Fields

Field	Function
31-16 —	Reserved
15-8 CFGVER	IP Configuration Version
7-0 IPVER	IP Version

75.17.36 MAC Debug (MAC_Debug)

Offset

Register	Offset
MAC_Debug	114h

Function
Provides the debug status of the various MAC blocks.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved													TFCSTS		TPESTS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved													RFCFCSTS		RPESTS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-19 —	Reserved
18-17	MAC Transmit Packet Controller Status

Table continues on the next page...

Table continued from the previous page...

Field	Function
TFCSTS	Indicates the state of the MAC transmit packet controller module. 00b - Idle state 01b - Waiting for one of these: status of the previous packet or IPG, or for the back-off period to be over 10b - Generating and transmitting a pause control packet (in Full-Duplex mode) 11b - Transferring input packet for transmission
16 TPESTS	MAC GMII Or MII Transmit Protocol Engine Status Indicates whether MAC GMII or MII transmit protocol engine is detected. If this field is 1, it indicates that the MAC GMII or MII transmit protocol engine is actively transmitting data and is not in an idle state. 0b - Not detected 1b - Detected
15-3 —	Reserved
2-1 RFCFCSTS	MAC Receive Packet Controller FIFO Status Indicates the status of the small FIFO read and write controllers of the MAC receive packet controller module. If this field is 1, it indicates that the small FIFO read and write controllers of the MAC receive packet controller module are in an active state. 00b - Inactive 01b - Active
0 RPESTS	Receive Protocol Engine Status Indicates whether MAC GMII or MII receive protocol engine is detected. If this field is 1, it indicates that the MAC GMII or MII receive protocol engine is actively receiving data and is not in an idle state. 0b - Not detected 1b - Detected

75.17.37 MAC Hardware Feature 0 (MAC_HW_Feature0)

Offset

Register	Offset
MAC_HW_Feature0	11Ch

Function

Indicates the presence of the first set of EMAC optional features or functions. You can use this register to dynamically enable or disable the programs related to these features.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved	ACTPHYSEL			SAVLANI...	TSSTSSEL		MACADR6...	MACADR3...	ADDMACADRSEL					Reserved	RXCOESEL
W																
Reset	0	0	0	0	1	1	1	0	0	0	0	0	1	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	TXCOESEL	EEESEL	TSSEL	Reserved		ARPOFFS...	MMCSEL	MGKSEL	RWKSEL	SMASEL	VLHASH	PCSSEL	HDSEL	GMIISEL	MIISEL
W																
Reset	0	1	0	1	0	0	0	1	0	0	1	1	0	1	0	1

Fields

Field	Function
31 —	Reserved
30-28 ACTPHYSEL	Active PHY Feature Indicates the selected active PHY interface. If you have multiple PHY interfaces (GMII, MII, RevMII, RGMII, RMII, RTBI, SGMII, SGMII, or TBI) in your configuration, this field indicates the sampled value of the PHY_INTF_SEL_I signal during reset deassertion. 000b - GMII or MII 001b - RGMII 010b - SGMII 011b - TBI 100b - RMII 101b - RTBI 110b - SMII 111b - RevMII
27 SAVLANINS	SA or VLAN Insertion Feature Indicates whether the feature that enables SA or VLAN insertion on transmit is selected. This field becomes 1 if you select the "enable SA and VLAN insertion on Tx" option. 0b - Not selected 1b - Selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
26-25 TSSTSSEL	Timestamp System Time Source Feature Indicates the source of the timestamp system time. This field becomes 1 if you select the "enable IEEE 1588 timestamp support" option. 00b - Internal 01b - External 10b - Both internal and external 11b - Reserved
24 MACADR64SEL	MAC Addresses 64-127 Indicates whether the feature that enables additional 64 MAC address registers (64-127) is selected. This field becomes 1 if you enable the "MAC addresses (64-127) select" option. 0b - Not selected 1b - Selected
23 MACADR32SEL	MAC Addresses 32-63 Indicates whether the feature that enables additional 32 MAC address registers (32-63) is selected. This field becomes 1 if you select the "MAC addresses 32-63 select" option. 0b - Not selected 1b - Selected
22-18 ADDMACADRSEL	MAC Addresses 1-31 Indicates whether the feature that enables additional 1-31 MAC address registers is selected. This field becomes 1 if you select a non-zero value for enabling the "MAC addresses 1-31 select" option. 0_0000b - Not selected 0_0001b - Selected
17 —	Reserved
16 RXCOESEL	Receive Checksum Offload Feature Indicates whether the feature that enables receive TCP/IP checksum offload is selected. This field becomes 1 if you select the "enable receive TCP/IP checksum check" option. 0b - Not selected 1b - Selected
15 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
14 TXCOESEL	Transmit Checksum Offload Feature Indicates whether the feature that enables transmit TCP/IP checksum insertion is selected. This field becomes 1 if you select the "enable transmit TCP/IP checksum insertion" option. 0b - Not selected 1b - Selected
13 EEESEL	Energy Efficient Ethernet (EEE) Feature Indicates whether the feature that enables EEE is selected. This field becomes 1 if you select the "enable EEE" option. 0b - Not selected 1b - Selected
12 TSSEL	IEEE 1588-2008 Timestamp Feature Indicates whether the option that enables IEEE 1588-2008 timestamp is selected. This field becomes 1 if you select the "enable IEEE 1588 timestamp support" option. 0b - Not selected 1b - Selected
11-10 —	Reserved
9 ARPOFFSEL	ARP Offload Feature Indicates whether the feature that enables ARP offload is selected. This field becomes 1 if you select the "enable IPv4 ARP offload" option. 0b - Not selected 1b - Selected
8 MMCSEL	MAC Management Counters (MMC) Feature Indicates whether the feature that enables MMC is selected. This field becomes 1 if you select the "enable MMC" option. 0b - Not selected 1b - Selected
7 MGKSEL	PMT Magic Packet Feature Indicates whether the feature that enables PMT magic detection is selected. This field becomes 1 if you select the "enable magic packet detection" option. 0b - Not selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Selected
6 RWKSEL	PMT Remote Wake-Up Packet Feature Indicates whether the feature that enables PMT remote wake-up packet detection is selected. This field becomes 1 if you select the "enable remote wake-up packet detection" option. 0b - Not selected 1b - Selected
5 SMASEL	SMA (MDIO) Interface Feature Indicates whether the feature that enables station management (MDIO interface) is selected. This field becomes 1 if you select the "enable station management (MDIO interface)" option. 0b - Not selected 1b - Selected
4 VLHASH	VLAN Hash Filter Feature Indicates whether the filtering option based on VLAN hash table is selected. This field becomes 1 if you select the "enable VLAN hash table based filtering" option. 0b - Not selected 1b - Selected
3 PCSSEL	PCS Select Indicates if either of the TBI, SGMII, or RTBI PHY interface options is selected. This field becomes 1 if you select any one of the TBI, SGMII, or RTBI PHY interface options. 0b - No 1b - Yes
2 HDSEL	Half-Duplex Support Feature Indicates whether half-duplex support is available as the mode of operation. This field becomes 1 if you select Half-Duplex as the mode of operation. 0b - Unavailable 1b - Available
1 GMIISEL	1000 Mbit/s Support Feature Indicates whether 1000 Mbit/s support is available as the mode of operation. This field becomes 1 if you select 1000 Mbit/s as the mode of operation. 0b - Unavailable 1b - Available

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 MIISEL	10 or 100 Mbit/s Support Feature Indicates whether 10 or 100 Mbit/s support is available as the mode of operation. This field becomes 1 if you select 10/100 Mbit/s as the mode of operation. 0b - Unavailable 1b - Available

75.17.38 MAC Hardware Feature 1 (MAC_HW_Feature1)

Offset

Register	Offset
MAC_HW_Feature1	120h

Function

Indicates the presence of the second set of optional features or functions of the module. You can use this register to dynamically enable or disable the programs related to these features.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved	L3L4FNUM			Reserved	HASHTBLSZ			POUOST	Reserved	RAVSEL	AVSEL	DBGMEMA	TSOEN	SPHEN	DCBEN
W																
Reset	0	0	1	0	0	0	0	1	0	0	0	1	1	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ADDR64		ADVTHWO...	PTOEN	OSTEN	TXFIFOSIZE					SPRAM	RXFIFOSIZE				
W																
Reset	0	0	1	0	1	0	0	1	1	0	1	0	0	1	1	0

Fields

Field	Function
31 —	Reserved
30-27	L3 Or L4 Filter Number

Table continues on the next page...

Table continued from the previous page...

Field	Function
L3L4FNUM	Indicates the total number of L3 or L4 filters. 0000b - No filters (0) 0001b - 1 0010b - 2 0011b - 3 0100b - 4 0101b - 5 0110b - 6 0111b - 7 1000b - 8
26 —	Reserved
25-24 HASHTBLSZ	Hash Table Size Indicates the size of the hash table. 00b - No hash table 01b - 64 10b - 128 11b - 256
23 POUOST	One Step For PTP Over UDP/IP Feature Indicates whether the feature that enables one step for PTP over UDP/IP is selected. This field becomes 1 if you select the "enable one-step timestamp for PTP over UDP/IP" option. 0b - Not selected 1b - Selected
22 —	Reserved
21 RAVSEL	Receive Side-Only AV Feature Indicates whether the feature that enables audio-video bridging only on the receive side is selected. This field becomes 1 if you select the "enable audio-video bridging on receive side only" option. 0b - Not selected 1b - Selected
20	AV Feature Indicates whether the feature that enables audio-video bridging is selected.

Table continues on the next page...

Table continued from the previous page...

Field	Function
AVSEL	This field becomes 1 if you select the "enable audio video bridging" option. 0b - Not selected 1b - Selected
19 DBGMEMA	DMA Debug Registers Enable Feature Indicates whether the feature that enables DMA debug registers is selected. This field becomes 1 if you select the "enable DMA debug mode" option. 0b - Not selected 1b - Selected
18 TSOEN	TCP Segmentation Offload Enable Feature Indicates whether the feature that enables TCP segmentation offload for TCP/IP packets is selected. This field becomes 1 if you select the "enable TCP segmentation offloading for TCP/IP packets" option. 0b - Not selected 1b - Selected
17 SPHEN	Split Header Enable Feature Indicates whether the feature that enables a split header structure is selected. This field becomes 1 if you select the "enable split header structure" option. 0b - Not selected 1b - Selected
16 DCBEN	DCB Enable Feature Indicates whether the feature that enables data center bridging is selected. This field becomes 1 if you select the "enable data center bridging" option. 0b - Not selected 1b - Selected
15-14 ADDR64	Address Width Feature Indicates the configured address width. 00b - 32 01b - 40 10b - 48 11b - Reserved
13 ADVTHWORD	IEEE 1588 High-Word Feature Indicates whether the feature that enables the IEEE 1588 high-word register is selected. This field becomes 1 if you select the "enable/add IEEE 1588 higher-word register" option.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Not selected 1b - Selected
12 PTOEN	PTP Offload Enable Feature Indicates whether the feature that enables PTP offload is selected. This field becomes 1 if you select the "enable PTP timestamp offload" option. 0b - Not selected 1b - Selected
11 OSTEN	One-Step Timestamping Enable Feature Indicates whether the feature that enables one-step timestamping is selected. This field becomes 1 if you select the "enable one-step timestamp" option. 0b - Not selected 1b - Selected
10-6 TXFIFOSIZE	MTL Transmit FIFO Size Feature Contains the configured value of MTL transmit FIFO (in bytes) expressed as log to base 2 minus 7, which means, Log 2 (TXFIFO_SIZE) - 7. 0_0000b - 128 bytes 0_0001b - 256 bytes 0_0010b - 512 bytes 0_0011b - 1024 bytes 0_0100b - 2048 bytes 0_0101b - 4096 bytes 0_0110b - 8192 bytes 0_0111b - 16384 bytes 0_1000b - 32 KB 0_1001b - 64 KB 0_1010b - 128 KB 0_1011b - Reserved
5 SPRAM	Single Port RAM Feature Indicates whether the single-port RAM feature is selected. This field becomes 1 if you select the option that enables the "use single port RAM" feature. 0b - Not selected 1b - Selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
4-0 RXFIFOSIZE	MTL Receive FIFO Size Feature Contains the configured value of MTL receive FIFO (in bytes) expressed as log to base 2 minus 7, which means, Log 2 (RXFIFO_SIZE) - 7. 0_0000b - 128 bytes 0_0001b - 256 bytes 0_0010b - 512 bytes 0_0011b - 1024 bytes 0_0100b - 2048 bytes 0_0101b - 4096 bytes 0_0110b - 8192 bytes 0_0111b - 16384 bytes 0_1000b - 32 KB 0_1001b - 64 KB 0_1010b - 128 KB 0_1011b - 256 KB 0_1100b - Reserved

75.17.39 MAC Hardware Feature 2 (MAC_HW_Feature2)

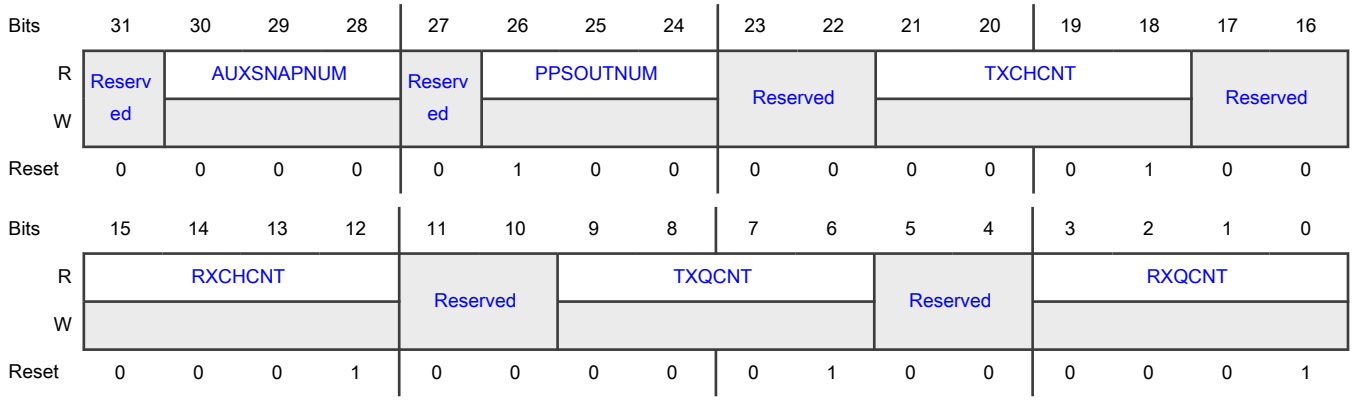
Offset

Register	Offset
MAC_HW_Feature2	124h

Function

Indicates the presence of the third set of optional features or functions of the module. You can use this register to dynamically enable or disable the programs related to these features.

Diagram



Fields

Field	Function
31 —	Reserved
30-28 AUXSNAPNUM	Number Of Auxiliary Snapshot Inputs Indicates the number of auxiliary snapshot inputs. 000b - No auxiliary input (0) 001b - 1 010b - 2 011b - 3 100b - 4 101b - Reserved
27 —	Reserved
26-24 PPSOUTNUM	Number Of PPS Outputs Indicates the number of PPS outputs. 000b - No PPS output (0) 001b - 1 010b - 2 011b - 3 100b - 4 101b - Reserved
23-22 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
21-18 TXCHCNT	Number Of DMA Transmit Channels Indicates the number of DMA transmit channels. 0000b - 1 0001b - 2 0010b - 3 0011b - 4 0100b - 5 0101b - 6 0110b - 7 0111b - 8
17-16 —	Reserved
15-12 RXCHCNT	Number Of DMA Receive Channels Indicates the number of DMA receive channels. 0000b - 1 0001b - 2 0010b - 3 0011b - 4 0100b - 5 0101b - 6 0110b - 7 0111b - 8
11-10 —	Reserved
9-6 TXQCNT	Number Of MTL Transmit Queues Indicates the number of MTL transmit queues. 0000b - 1 0001b - 2 0010b - 3 0011b - 4 0100b - 5 0101b - 6

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0110b - 7 0111b - 8
5-4 —	Reserved
3-0 RXQCNT	Number Of MTL Receive Queues Indicates the number of MTL receive queues. 0000b - 1 0001b - 2 0010b - 3 0011b - 4 0100b - 5 0101b - 6 0110b - 7 0111b - 8

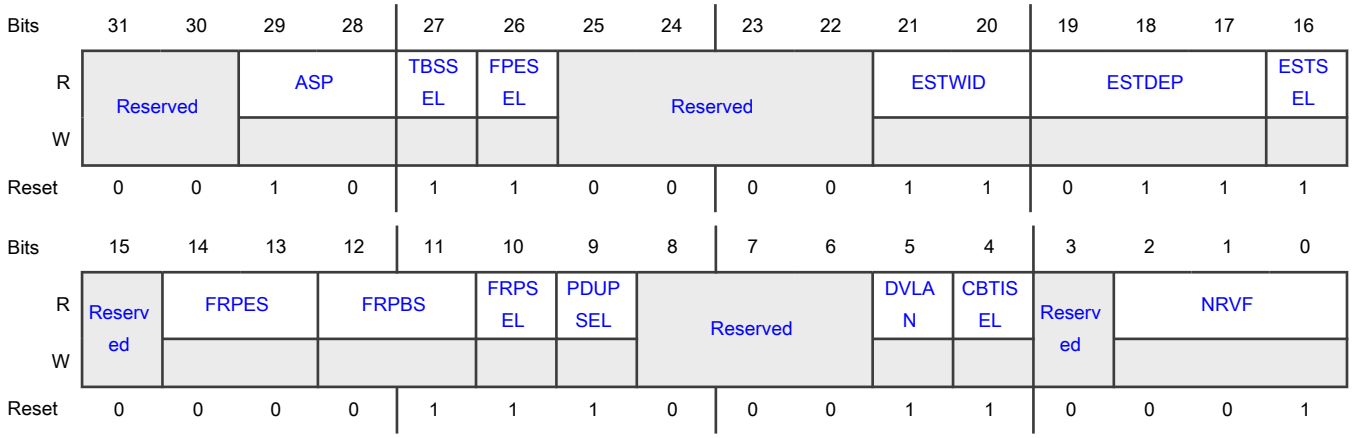
75.17.40 MAC Hardware Feature 3 (MAC_HW_Feature3)

Offset

Register	Offset
MAC_HW_Feature3	128h

Function
Indicates the presence of the fourth set of optional features or functions of the module. You can use this register to dynamically enable or disable the programs related to these features.

Diagram



Fields

Field	Function
31-30 —	Reserved
29-28 ASP	Automotive Safety Package Indicates the encoding for the different automotive safety features. 00b - No safety features selected 01b - Only "ECC protection for external memory" feature is selected 10b - All the safety features are selected without the "parity port enable for external interface" feature 11b - All the safety features are selected with the "parity port enable for external interface" feature
27 TBSSEL	Time-Based Scheduling Feature Indicates whether the time-based scheduling feature is selected. This field becomes 1 if you select the option that enables the time-based scheduling feature. 0b - Selected 1b - Selected
26 FPESEL	Frame Preemption Feature Indicates whether the feature that enables frame preemption is selected. This field becomes 1 if you select the "enable frame preemption" option. 0b - Not selected 1b - Selected
25-22 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
21-20 ESTWID	Estimated Time Interval Width Indicates the width configured for the time interval field in the gate control list. 00b - Width not configured 01b - 16 bits 10b - 20 bits 11b - 24 bits
19-17 ESTDEP	Depth Of Gate Control List Indicates the depth of gate control list expressed as Log 2 (DWC_EQOS_EST_DEP) - 5. 000b - No depth configured 001b - 64 bytes 010b - 128 bytes 011b - 256 bytes 100b - 512 bytes 101b - 1024 bytes 110b - Reserved
16 ESTSEL	Enhancements To Scheduling Traffic Feature Indicates whether the "enhancements to scheduling traffic" feature is selected. This field becomes 1 if you select the "enable enhancements to scheduling traffic" option. 0b - Not selected 1b - Selected
15 —	Reserved
14-13 FRPES	Flexible Receive Parser Table Entry Size Indicates the maximum number of parser entries that the flexible receive parser supports. 00b - 64 01b - 128 10b - 256 11b - Reserved
12-11 FRPBS	Flexible Receive Parser Buffer Size Indicates the maximum number of packet data bytes that the flexible receive parser supports. 00b - 64

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01b - 128 10b - 256 11b - Reserved
10 FRPSEL	Flexible Receive Parser Feature Indicates whether the flexible receive parser feature is selected. This field becomes 1 if you select the "enable flexible programmable receive parser" option. 0b - Not selected 1b - Selected
9 PDUPSEL	Broadcast/Multicast Packet Duplication Feature Indicates whether the broadcast or multicast duplication feature is selected. This field becomes 1 if you select the "broadcast or multicast packet duplication" option. 0b - Not selected 1b - Selected
8-6 —	Reserved
5 DVLAN	Double VLAN Tag Processing Feature Indicates whether the double VLAN processing feature is selected. 0b - Not selected 1b - Selected
4 CBTISEL	Queue/Channel Based VLAN Tag Insertion On Transmit Feature Indicates whether the "queue- or channel-based VLAN tag insertion on Tx" feature is selected. This field becomes 1 if you select the "enable queue- or channel-based VLAN tag insertion on Tx" option. 0b - Not selected 1b - Selected
3 —	Reserved
2-0 NRVF	Number Of Extended VLAN Tag Filters Indicates the number of selected extended VLAN tag filters. 000b - No filters (0) 001b - 4 010b - 8

Table continues on the next page...

Table continued from the previous page...

Field	Function
	011b - 16 100b - 24 101b - 32 110b - Reserved

75.17.41 MAC DPP FSM Interrupt Status (MAC_DPP_FSM_Interrupt_Status)

Offset

Register	Offset
MAC_DPP_FSM_Interrupt_Status	140h

Function

Contains the status of automative-safety related data path parity errors, interface timeout errors, FSM state parity errors, and FSM state timeout errors.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved							FSMPES	Reserved						Reserved	MSTTES
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	Reserved	Reserved	PTES	ATES	Reserved	RTES	TTES	Reserved	Reserved	ARPES	MTSPES	MPES	RDPES	Reserved	Reserved
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-25 —	Reserved
24 FSMPES	FSM State Parity Error Status Indicates whether a parity error is detected on one of the FSM state registers.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	After a read operation, the field's value clears to 0. 0b - Not detected 1b - Detected
23-18 —	Reserved
17 —	Reserved
16 MSTTES	Master Read Or Write Timeout Error Status Indicates whether an application or CSR timeout error is detected on the master (AXI, AHB, ARI, or ATI) interface. After a read operation, the field's value clears to 0. 0b - Not detected 1b - Detected
15 —	Reserved
14 —	Reserved
13 —	Reserved
12 PTES	PTP FSM Timeout Error Status Indicates whether one of the PTP FSM timeout errors is detected. After a read operation, the field's value clears to 0. 0b - Not detected 1b - Detected
11 ATES	APP FSM Timeout Error Status Indicates whether one of the APP FSM timeout errors is detected. After a read operation, the field's value clears to 0. 0b - Not detected 1b - Detected
10 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
9 RTES	Receive FSM Timeout Error Status Indicates whether one of the receive FSM timeout errors is detected. After a read operation, the field's value clears to 0. 0b - Not detected 1b - Detected
8 TTES	Transmit FSM Timeout Error Status Indicates whether one of the transmit FSM timeout errors is detected. After a read operation, the field's value clears to 0. 0b - Not detected 1b - Detected
7 —	Reserved
6 —	Reserved
5 ARPES	Application Receive Interface Data Path Parity Error Status Indicates whether the parity checker detected a parity error on the application receive interface (or on PC6 as shown in the transmit data path parity protection diagram). After a read operation, the field's value clears to 0. 0b - Not detected 1b - Detected
4 MTSPES	MTL Transmit Status Data Path Parity Checker Error Status Indicates whether the parity checker detected a parity error for the MTL transmit status data on ATI (or on PC5 as shown in the transmit data path parity protection diagram). After a read operation, the field's value clears to 0. 0b - Not detected 1b - Detected
3 MPES	MTL Data Path Parity Checker Error Status Indicates whether the parity checker detected a parity error on the MTL transmit write controller interface (or on PC4 as shown in the transmit data path parity protection diagram). After a read operation, the field's value clears to 0. 0b - Not detected 1b - Detected

Table continues on the next page...

Table continued from the previous page...

Field	Function
2 RDPES	Read Descriptor Parity Checker Error Status Indicates whether the parity checker detected a parity error on the DMA read descriptor interface (or on PC3 as shown in the transmit data path parity protection diagram). After a read operation, the field's value clears to 0. 0b - Not detected 1b - Detected
1 —	Reserved
0 —	Reserved

75.17.42 MAC FSM Control (MAC_FSM_Control)

Offset

Register	Offset
MAC_FSM_Control	148h

Function

Is used to control the FSM state parity and timeout error injection in Debug mode.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved	Reserved	Reserved	PLGRNML	ALGRNML	Reserved	RLGRNML	TLGRNML	Reserved	Reserved	Reserved	PPEIN	APEIN	Reserved	RPEIN	TPEIN
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved	Reserved	Reserved	PTEIN	ATEIN	Reserved	RTEIN	TTEIN	Reserved						PRTYEN	TMOUTEN
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
30 —	Reserved
29 —	Reserved
28 PLGRNML	PTP Large Or Normal Mode Select Indicates which mode of tic generation is used for the PTP domain. 0b - Normal mode 1b - Large mode
27 ALGRNML	APP Large Or Normal Mode Select Indicates which mode of tic generation is used for the APP domain. 0b - Normal mode 1b - Large mode
26 —	Reserved
25 RLGRNML	Receive Large Or Normal Mode Select Indicates which mode of tic generation is used for the receive domain. 0b - Normal mode 1b - Large mode
24 TLGRNML	Transmit Large Or Normal Mode Select Indicates which mode of tic generation is used for the transmit domain. 0b - Normal mode 1b - Large mode
23 —	Reserved
22 —	Reserved
21 —	Reserved
20	PTP FSM Parity Error Injection

Table continues on the next page...

Table continued from the previous page...

Field	Function
PPEIN	Indicates the status of error injection for PTP FSM parity. 0b - Disabled 1b - Enabled
19 APEIN	APP FSM Parity Error Injection Indicates the status of error injection for APP FSM parity. 0b - Disabled 1b - Enabled
18 —	Reserved
17 RPEIN	Receive FSM Parity Error Injection Indicates the status of error injection for receive FSM parity. 0b - Disabled 1b - Enabled
16 TPEIN	Transmit FSM Parity Error Injection Indicates the status of error injection for transmit FSM parity. 0b - Disabled 1b - Enabled
15 —	Reserved
14 —	Reserved
13 —	Reserved
12 PTEIN	PTP FSM Timeout Error Injection Indicates the status of error injection for PTP FSM timeout. 0b - Disabled 1b - Enabled
11 ATEIN	APP FSM Timeout Error Injection Indicates the status of APP FSM timeout error injection. 0b - Disabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Enabled
10 —	Reserved
9 RTEIN	Receive FSM Timeout Error Injection Indicates the status of receive FSM timeout error injection. 0b - Disabled 1b - Enabled
8 TTEIN	Transmit FSM Timeout Error Injection Indicates the status of transmit FSM timeout eror injection. 0b - Disabled 1b - Enabled
7-2 —	Reserved
1 PRTYEN	Parity Enable Indicates whether the FSM parity feature is enabled. 0b - Disabled 1b - Enabled
0 TMOUTEN	Time Out Enable Indicates the status of the FSM timeout feature. 0b - Disabled 1b - Enabled

75.17.43 MAC FSM ACT Timer (MAC_FSM_ACT_Timer)

Offset

Register	Offset
MAC_FSM_ACT_Timer	14Ch

Function
Is used to select the FSM and interface timeout values.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved								LTMRMD				NTMRMD			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved						TMR									
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-20 LTMRMD	Large Mode Timeout Value Provides the timeout duration value to be used for large mode FSM and other interface timeouts. 0000b - Timer disabled 0001b - 1 us 0010b - 1.024 ms (~4 ms) 0011b - 16.384 ms (~16 ms) 0100b - 65.536 ms (~64 ms) 0101b - 262.144 ms (~256 ms) 0110b - 1.048 sec (~1 sec) 0111b - 4.194 sec (~4sec) 1000b - 16.777 sec (~16 sec) 1001b - 33.554 sec (~32 sec) 1010b - 67.108 sec (~64 sec) 1011b - Reserved
19-16 NTMRMD	Normal Mode Timeout Value Provides the timeout duration value to be used for normal mode FSM and other interface timeouts. 0000b - Timer disabled 0001b - 1 us 0010b - 1.024 ms (~4 ms) 0011b - 16.384 ms (~16 ms)

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0100b - 65.536 ms (~64 ms) 0101b - 262.144 ms (~256 ms) 0110b - 1.048 sec (~1 sec) 0111b - 4.194 sec (~4 sec) 1000b - 16.777 sec (~16 sec) 1001b - 33.554 sec (~32 sec) 1010b - 67.108 sec (~64 sec) 1011b - Reserved
15-10 —	Reserved
9-0 TMR	CSR Clocks For 1 µs Tic Indicates the number of CSR clocks required to generate 1 µs tic.

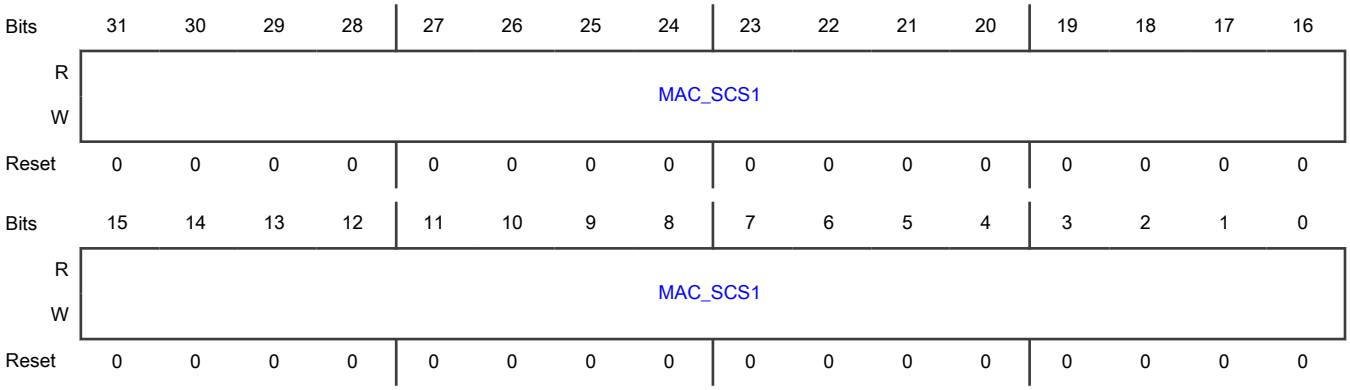
75.17.44 SCS_REG 1 (SCS_REG1)

Offset

Register	Offset
SCS_REG1	150h

Function
Is an NXP-reserved register for internal use.

Diagram



Fields

Field	Function
31-0 MAC_SCS1	MAC SCS 1 Is reserved for NXP internal use. The value of this field must always be 0 unless NXP instructs you otherwise. Writing 1 to any of the bits of this field might cause expected chip behavior.

75.17.45 MAC MDIO Address (MAC_MDIO_Address)

Offset

Register	Offset
MAC_MDIO_Address	200h

Function
Controls the management cycles to external PHY through a management interface.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved				PSE	BTB	PA						RDA			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved	NTC			CR				Reserved			SKAP	GOC_1	GOC_0	C45E	GB
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved
27 PSE	Preamble Suppression Enable Indicates the status of preamble suppression. • If this field is 1, SMA suppresses the 32-bit preamble and transmits MDIO frames with only 1 preamble bit. • If this field is 0, the MDIO frame always includes 32 bits of preamble as defined in the IEEE specification.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disabled 1b - Enabled
26 BTB	Back-To-Back Transactions Indicates the status of back-to-back transactions. When BTB = 1 and NTC > 0, MAC indicates the completion of a read or write command at the end of a frame transfer (before the trailing clocks are transmitted). Therefore, you can initiate the next command, and that command executes immediately irrespective of the number of trailing clocks generated for the previous frame. When BTB = 0, the read or write command completes (GB = 0) only after the trailing clocks generate. In this mode, you must ensure that: • NTC is always generated after each frame. • BTB must not be equal to 1 when NTC = 0. 0b - Disabled 1b - Enabled
25-21 PA	Physical Layer Address Indicates whether MAC accesses clause 22 PHY devices (out of 32) or clause 45 capable PHYs (out of 32).
20-16 RDA	Register Or Device Address Indicates whether MAC accesses clause 22 PHY devices (out of 32) or clause 45 capable PHYs (out of 32).
15 —	Reserved
14-12 NTC	Number Of Trailing Clocks Controls the number of trailing clock cycles generated on the GMII_MDC_O (MDC) signal after the end of MDIO frame transmission. The valid values can range from 0 to 7. Writing 3h to this field indicates that there are additional three clock cycles on the MDC line after the end of MDIO frame transfer.
11-8 CR	CSR Clock Range Determines the frequency of the MDC clock according to the CSR clock frequencies used in your design: • 0000: CSR clock = 60-100 MHz; MDC clock = CSR clock/42 • 0001: CSR clock = 100-150 MHz; MDC clock = CSR clock/62 • 0010: CSR clock = 20-35 MHz; MDC clock = CSR clock/16 • 0011: CSR clock = 35-60 MHz; MDC clock = CSR clock/26 • 0100: CSR clock = 150-250 MHz; MDC clock = CSR clock/102 • 0101: CSR clock = 250-300 MHz; MDC clock = CSR clock/124

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• 0110: CSR clock = 300-500 MHz; MDC clock = CSR clock/204 • 0111: CSR clock = 500-800 MHz; MDC clock = CSR clock/324 When bit 3 of this field is 0, the suggested CSR clock-frequency range applicable to each value ensures that the MDC clock frequency is approximately between 1.0 MHz and 2.5 MHz. When bit 3 of this field is 1, you can achieve a higher MDC clock frequency than the IEEE 802.3 frequency limit of 2.5 MHz, and program a clock divider of a lower value. For example, if the CSR clock frequency is 100 MHz and you write 1010 to this field, the resultant MDC clock frequency equals 12.5 MHz, which is above the range specified in IEEE 802.3. Program these values only if the interfacing chips support faster MDC clocks: • 1000: CSR clock/4 • 1001: CSR clock/6 • 1010: CSR clock/8 • 1011: CSR clock/10 • 1100: CSR clock/12 • 1101: CSR clock/14 • 1110: CSR clock/16 • 1111: CSR clock/18 These bits are not used for accessing RevMII. They are read-only if the RevMII interface is selected as a single PHY interface.
7-5 —	Reserved
4 SKAP	Skip Address Packet Indicates the status of the skip address packet. If this field is 1, SMA does not send the address packets before read, write, or post-read increment address packets. This field is valid only when C45E = 1. 0b - Disabled 1b - Enabled
3 GOC_1	GMII Operation Command 1 Indicates the status of GMII operation command 1. GMII operation command 1 represents the higher bit of the operation command to PHY or RevMII. The fields GOC_1 and GOC_O indicate these values: • 00: Reserved • 01: Write • 10: Post-read increment address for clause 45 PHY • 11: Read

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When clause 22 PHY or RevMII is enabled, only write and read commands are valid. 0b - Disabled 1b - Enabled
2 GOC_0	GMII Operation Command 0 Indicates the status of GMII operation command 0. GMII operation command 0 represents the lower bit of the operation command to PHY or RevMII. When in SMA mode (MDIO master), this field, along with the GOC_1 field, determines the operation to be performed to PHY. This field is read-only and tied to 1 when only RevMII is selected in configuration. 0b - Disabled 1b - Enabled
1 C45E	Clause 45 PHY Enable Indicates the status of clause 45 PHY. • If this field is 1, clause 45 capable PHY is connected to MDIO. • If this field is 0, clause 22 capable PHY is connected to MDIO. 0b - Disabled 1b - Enabled
0 GB	GMII Busy Indicates the status of GMII busy. Writing 1 to this field instructs SMA to initiate a read or write access to the MDIO slave. MAC writes 0 to the field after the MDIO frame transfer is complete. Therefore, you must not write or change any of the fields in the MAC_MDIO_Address and MAC_MDIO_Data registers as long as the value of this field is 1. For write transfers, before writing 1 to this field, you must write a 16-bit data to MAC_MDIO_Data[GD] and MAC_MDIO_Data[RA] if C45E = 1. If C45E = 1, you must also write to MAC_MDIO_Data[RA] before initiating a read transfer. After a read transfer completes (GB = 0), a data read from the PHY register is valid in MAC_MDIO_Data[GD]. **NOTE** Even if the addressed PHY is not present, there is no change in the functionality of this field. Access restrictions apply to this field. It clears automatically and writing 0 to it has no effect. 0b - Disabled 1b - Enabled

75.17.46 MAC MDIO Data (MAC_MDIO_Data)

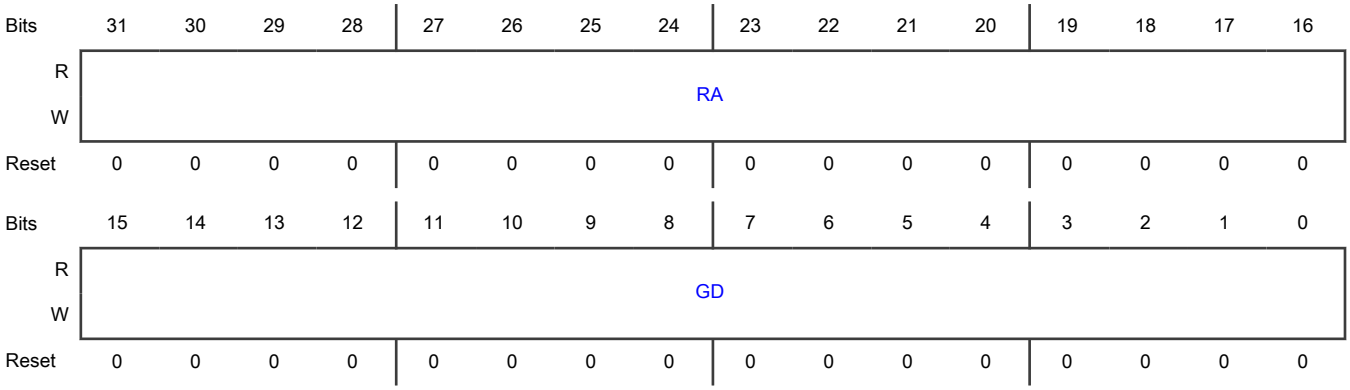
Offset

Register	Offset
MAC_MDIO_Data	204h

Function

Is used to store the write data to be written to the PHY register located at the address specified in MAC MDIO Address (MAC_MDIO_Address). This register also stores the read data from the PHY register located at the address that MAC MDIO Address (MAC_MDIO_Address) specifies.

Diagram



Fields

Field	Function
31-16 RA	Register Address Contains the PHY register address intended for the MDIO frame. This field is valid only when MAC_MDIO_Address[C45E] = 1.
15-0 GD	GMII Data Contains the 16-bit data value read from PHY or RevMII after a management read operation or the 16-bit data value to be written to PHY or RevMII before a management write operation.

75.17.47 MAC CSR Software Control (MAC_CSR_SW_Ctrl)

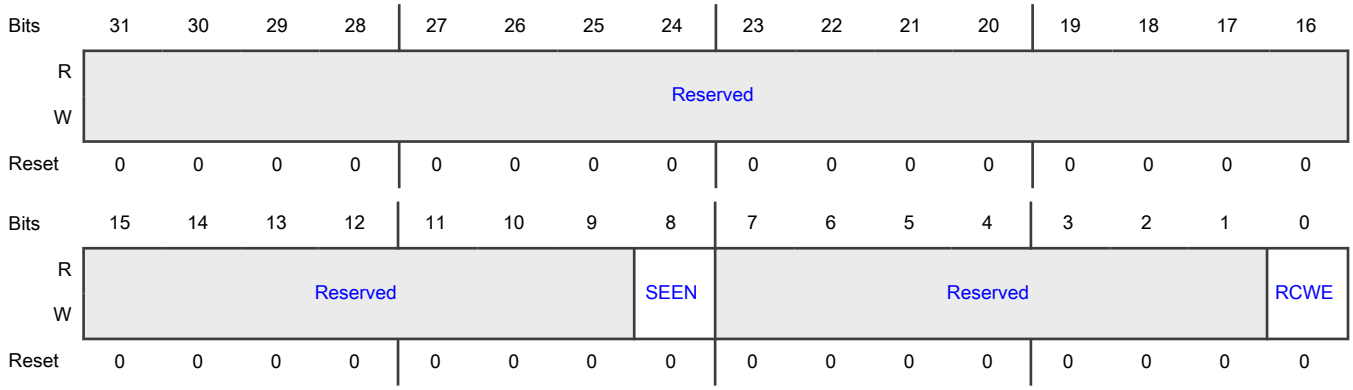
Offset

Register	Offset
MAC_CSR_SW_Ctrl	230h

Function

Contains software programmable controls for changing the CSR access response and clearing the status bits on read.

Diagram



Fields

Field	Function
31-9 —	Reserved
8 SEEN	Slave Error Response Enable Indicates the status of the slave error response. • If this field is 1, MAC responds with a slave error for accesses to reserved registers in the CSR space. • If this field is 0, MAC sends an "okay" response to registers accessed from the CSR space. 0b - Disabled 1b - Enabled
7-1 —	Reserved
0 RCWE	Enable Register Write 1 To Clear (W1C) Enables and disables the W1C feature for some registers. When this field is 1, the access mode of some of the other fields changes from R2C to W1C. When this field is 0, the access mode of those fields remains R2C. 0b - Disabled 1b - Enabled

75.17.48 MAC FPE Control STS (MAC_FPE_CTRL_STS)

Offset

Register	Offset
MAC_FPE_CTRL_STS	234h

Function

Controls the operation of frame preemption.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved												TRSP	TVER	RRSP	RVER
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved												S1_SET_0	SRSP	SVER	EFPE
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-20 —	Reserved
19 TRSP	Transmitted Respond Frame Indicates whether the respond frame is transmitted. This field becomes 1 when a respond frame is transmitted (triggered by writing 1 to the SRSP field). An interrupt can be generated for this event if you write 1 to MAC_Interrupt_Enable[FPEIE]. Access restrictions apply to this field. TRSP becomes a W1C field when MAC_CSR_SW_Ctrl[RCWE] = 1, and becomes an R2C field when MAC_CSR_SW_Ctrl[RCWE] = 0. Thus, TRSP becomes 0 sometimes when you read it and sometimes when you write 1 to it. It automatically becomes 1 on an internal event. 0b - Not transmitted 1b - Transmitted
18 TVER	Transmitted Verify Frame Indicates whether the verify frame is transmitted. This field becomes 1 when a verify frame is transmitted (triggered by writing 1 to the SVER field). An interrupt can be generated for this event if you write 1 to MAC_Interrupt_Enable[FPEIE]. Access restrictions apply to this field. The field clears on read or write of 1 when MAC_CSR_SW_Ctrl[RCWE] = 1. It automatically becomes 1 on an internal event. 0b - Not transmitted 1b - Transmitted
17 RRSP	Received Respond Frame Indicates whether the respond frame is received.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field becomes 1 when a respond frame is received. An interrupt can be generated for this event if you write 1 to MAC_Interrupt_Enable[FPEIE]. Access restrictions apply to this field. The field clears on read or write of 1 when MAC_CSR_SW_Ctrl[RCWE] = 1. It automatically becomes 1 on an internal event. 0b - Not received 1b - Received
16 RVER	Received Verify Frame Indicates whether the verify frame is received. This field becomes 1 when a verify frame is received. An interrupt can be generated on the transmission of a packet if you write 1 to MAC_Interrupt_Enable[FPEIE]. Access restrictions apply to this field. The field clears on read or write of 1 when MAC_CSR_SW_Ctrl[RCWE] = 1. It automatically becomes 1 on an internal event. 0b - Not received 1b - Received
15-4 —	Reserved
3 S1_SET_0	S1 SET 0 Is reserved for NXP internal use and must always be 0 unless instructed by NXP. Writing 1 to this field may cause unexpected behavior.
2 SRSP	Send Respond mPacket Indicates the status of the send respond mPacket. If this field is 1, it indicates the hardware to send a respond mPacket. The hardware resets the field after sending the respond mPacket. Access restrictions apply to this field. It clears automatically and writing 0 to it has no effect. 0b - Disabled 1b - Enabled
1 SVER	Send Verify mPacket Enables and disables the sending of a verify mPacket. Write 1 to this field to command EMAC to send a verify mPacket. EMAC writes 0 to this field after sending the verify mPacket. Access restrictions apply to this field. It clears automatically and writing 0 to it has no effect. 0b - Disabled 1b - Enabled
0	Enable Transmit Frame Preemption

Table continues on the next page...

Table continued from the previous page...

Field	Function
EFPE	Enables or disables the frame preemption transmit functionality. 0b - Disabled 1b - Enabled

75.17.49 MAC Presentation Time (MAC_Presn_Time_ns)

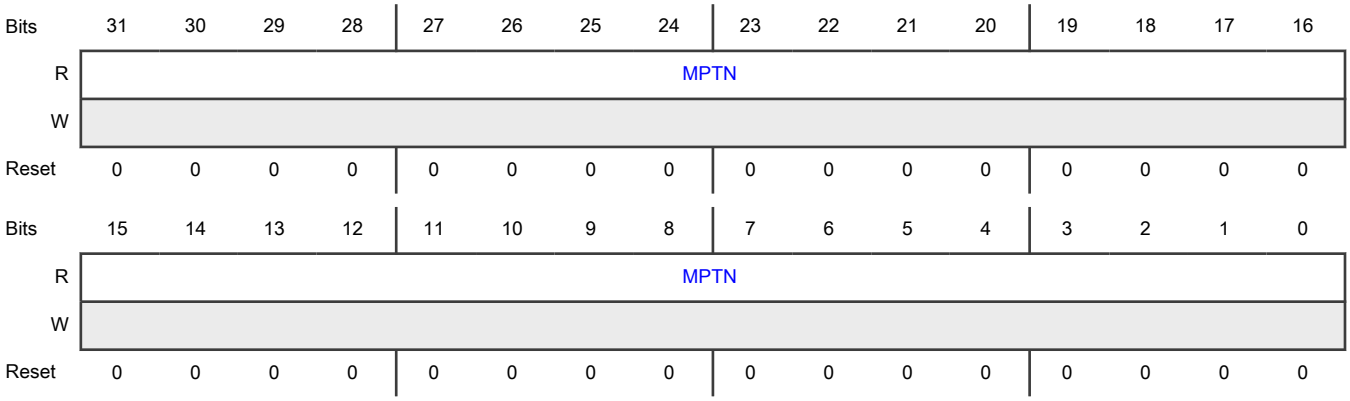
Offset

Register	Offset
MAC_Presn_Time_ns	240h

Function

Contains the 32-bit binary rollover equivalent time of the PTP system time (in nanoseconds) and exists when DWC_EQOS_FLEXI_PPS_OUT_EN is configured.

Diagram



Fields

Field	Function
31-0 MPTN	MAC 1722 Presentation Time (In Nanoseconds) Indicates the value of the 32-bit binary rollover equivalent time of the PTP System Time (in ns).

75.17.50 MAC Presentation Time Update (MAC_Presn_Time_Updt)

Offset

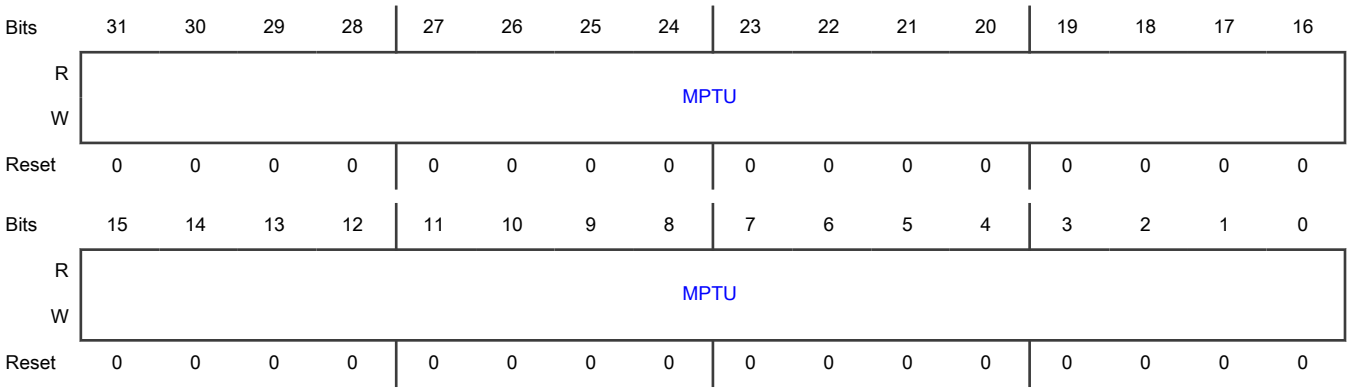
Register	Offset
MAC_Presn_Time_Updt	244h

Function

Holds the 32-bit value of MAC 1722 presentation time (in ns), which must be added to the current presentation time counter value. Initialization happens when MAC_Timestamp_Control[TSINIT] = 1, and an update happens when MAC_Timestamp_Control[TSUPDT] = 1.

Exists when DWC_EQOS_FLEXI_PPS_OUT_EN is configured.

Diagram



Fields

Field	Function
31-0 MPTU	MAC 1722 Presentation Time Update Holds the initial value or the update value for the presentation time. When used for an update, this field holds the 32-bit value (in ns), which must be added to the current presentation time counter value. Initialization happens when MAC_Timestamp_Control[TSINIT] = 1, and an update happens when MAC_Timestamp_Control[TSUPDT] = 1. When MAC_System_Time_Nanoseconds_Update[ADDSUB] = 1, this value is directly used for subtraction.

75.17.51 MAC Address 0 High (MAC_Address0_High)

Offset

Register	Offset
MAC_Address0_High	300h

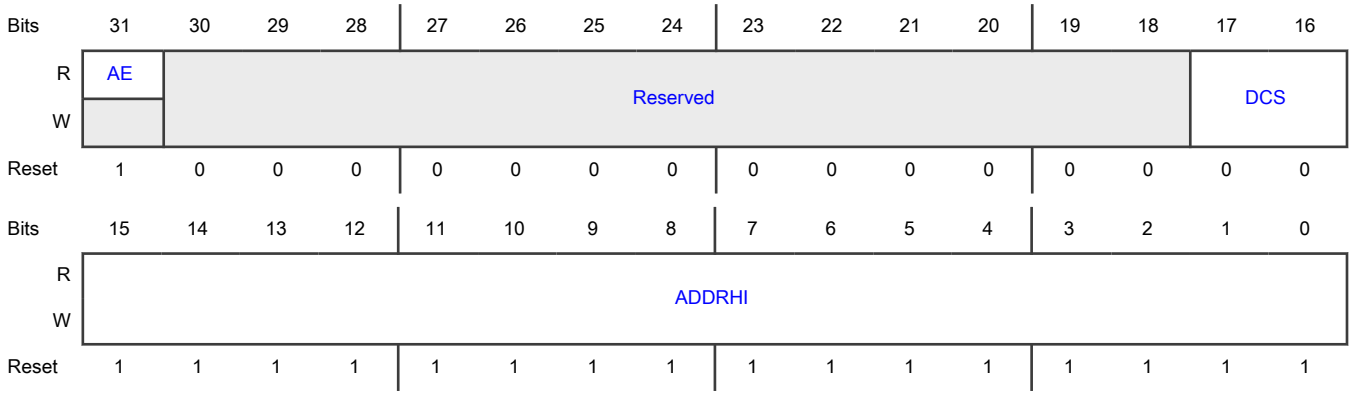
Function

Holds the upper 16 bits of the first 6-byte MAC address of the station.

The first DA byte that is received on the (G)MII interface corresponds to the LS byte (bits [7:0]) of MAC Address 0 Low (MAC_Address0_Low). For example, if 112233445566h is received (11h in lane 0 of the first column) on (G)MII as the destination address, MAC Address 0 Low (MAC_Address0_Low)[47:0] is compared to 665544332211h.

If MAC address registers are configured to be double-synchronized with the (G)MII clock domains, the synchronization is triggered only when bits [31:24] (in Little-Endian mode) or bits [7:0] (in Big-Endian mode) of MAC Address 0 Low (MAC_Address0_Low) are written to. For proper synchronization updates, the consecutive writes to MAC Address 0 Low (MAC_Address0_Low) must be performed after at least four clock cycles in the destination clock domain.

Diagram



Fields

Field	Function
31 AE	Address Enable Enables MAC address 0b - Disabled and invalid (the field's value must always be 1) 1b - Enabled
30-18 —	Reserved
17-16 DCS	DMA Channel Select If MAC_Ext_Configuration[PDC] = 0, this field contains the binary representation of the DMA channel number to which a receive packet whose DA matches MAC Address 0 Low (MAC_Address0_Low) content is routed. If MAC_Ext_Configuration[PDC] = 1, this field contains the one-hot representation of one or more DMA channel numbers to which a receive packet whose DA matches MAC Address 0 Low (MAC_Address0_Low) content is routed.
15-0 ADDRHI	MAC Address 0 [47:32] Contains the upper 16 bits [47:32] of the first 6-byte MAC address. MAC uses this field for filtering the received packets and inserting the MAC address in the transmit flow control (pause) packets.

75.17.52 MAC Address 0 Low (MAC_Address0_Low)

Offset

Register	Offset
MAC_Address0_Low	304h

Function

Holds the lower 32 bits of the 6-byte first MAC address of the station.

Diagram



Fields

Field	Function
31-0 ADDRLO	MAC Address 0 [31:0] Contains the lower 32 bits of the first 6-byte MAC address. MAC uses this field to filter the received packets and insert the MAC address in the transmit flow control (pause) packets.

75.17.53 MAC Address 1 High (MAC_Address1_High)

Offset

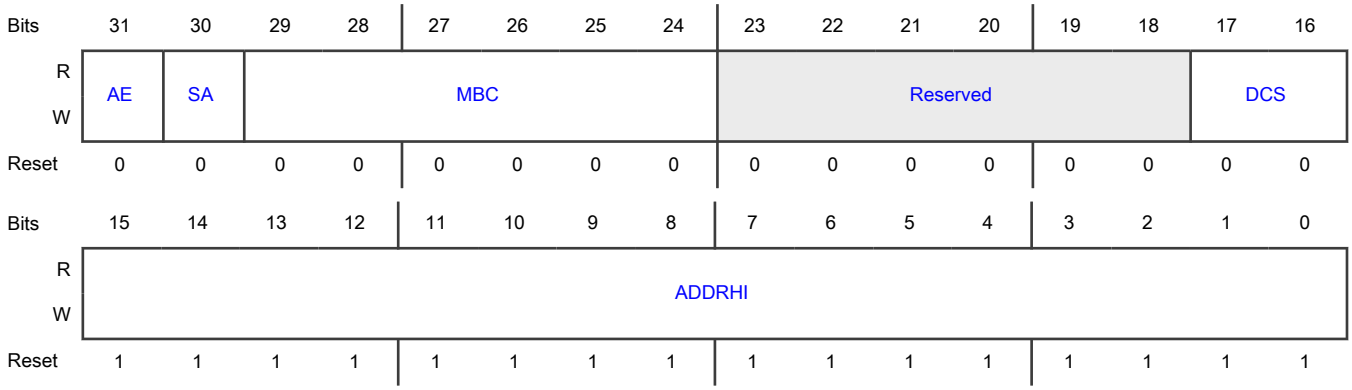
Register	Offset
MAC_Address1_High	308h

Function

Holds the upper 16 bits of the second 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized with the (G)MII clock domains, the synchronization is triggered only when bits [31:24] (in Little-Endian mode) or bits [7:0] (in Big-Endian mode) of MAC Address 1 Low (MAC_Address1_Low) are written to. For proper synchronization updates, the consecutive writes to MAC Address 1 Low (MAC_Address1_Low) must be performed after at least four clock cycles in the destination clock domain.

Diagram



Fields

Field	Function
31 AE	Address Enable Indicates whether the MAC address is enabled or ignored. If this field is 1, the address filter module uses the second MAC address for perfect filtering. If this field is 0, the address filter module ignores the address used for filtering. 0b - Ignored 1b - Enabled
30 SA	Source Address Indicates whether the MAC address is compared to the source or destination address. If this field is 1, MAC Address 1[47:0] is compared to the SA fields of the received packet. If this field is 0, MAC Address 1[47:0] is compared to the DA fields of the received packet. 0b - Destination address 1b - Source address
29-24 MBC	Mask Byte Control Contains mask control bits for comparing each of the MAC address bytes. If this field is 1, MAC does not compare the corresponding byte of the received DA or SA with the contents of the MAC address 1 registers. Each bit controls the masking of the bytes as follows: • Bit 29: MAC Address 1 High[15:8] • Bit 28: MAC Address 1 High[7:0] • Bit 27: MAC Address 1 Low[31:24] • - .. • Bit 24: MAC Address 1 Low[7:0] You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address.
23-18	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
17-16 DCS	DMA Channel Select If MAC_Ext_Configuration[PDC] = 0, this field contains the binary representation of the DMA channel number to which a receive packet whose DA matches MAC Address 1 Low (MAC_Address1_Low) content is routed. If MAC_Ext_Configuration[PDC] = 1, this field contains the one-hot representation of one or more DMA Channel numbers to which a receive packet whose DA matches MAC Address 1 Low (MAC_Address1_Low) content is routed.
15-0 ADDRHI	MAC Address 1 [47:32] Contains the upper 16 bits [47:32] of the second 6-byte MAC address.

75.17.54 MAC Address 1 Low (MAC_Address1_Low)

Offset

Register	Offset
MAC_Address1_Low	30Ch

Function

Holds the lower 32 bits of the second 6-byte MAC address of the station.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	ADDRLO															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ADDRLO															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-0	MAC Address 1 [31:0] Contains the lower 32 bits of the second 6-byte MAC address.

Table continues on the next page...

Field	Function
ADDRLO	The contents of this field are undefined until you load them after the initialization process.

75.17.55 MAC Address 2 High (MAC_Address2_High)

Offset

Register	Offset
MAC_Address2_High	310h

Function

Holds the upper 16 bits of the second 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized with the (G)MII clock domains, the synchronization is triggered only when bits [31:24] (in Little-Endian mode) or bits [7:0] (in Big-Endian mode) of MAC Address 1 Low (MAC_Address1_Low) are written to. For proper synchronization updates, the consecutive writes to MAC Address 1 Low (MAC_Address1_Low) must be performed after at least four clock cycles in the destination clock domain.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	AE	SA	MBC						Reserved						DCS	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ADDRHI															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31 AE	Address Enable Indicates whether the MAC address is enabled or ignored. If this field is 1, the address filter module uses the second MAC address for perfect filtering. If this field is 0, the address filter module ignores the address for filtering. 0b - Ignored 1b - Enabled
30 SA	Source Address Indicates whether the MAC address is compared to the source or destination address.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	If this field is 1, MAC Address 1[47:0] is compared to the SA fields of the received packet. If this field is 0, MAC Address 1[47:0] is compared to the DA fields of the received packet. 0b - Destination address 1b - Source address
29-24 MBC	Mask Byte Control Contains mask control bits for comparing each of the MAC address bytes. If this field is 1, MAC does not compare the corresponding byte of the received DA or SA with the contents of MAC address 1 registers. Each bit controls the masking of the bytes as follows: • Bit 29: MAC Address 1 High[15:8] • Bit 28: MAC Address 1 High[7:0] • Bit 27: MAC Address 1 Low[31:24] • - .. • Bit 24: MAC Address 1 Low[7:0] You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address.
23-18 —	Reserved
17-16 DCS	DMA Channel Select If MAC_Ext_Configuration[PDC] = 0, this field contains the binary representation of the DMA channel number to which a receive packet whose DA matches MAC Address 2 Low (MAC_Address2_Low) content is routed. If MAC_Ext_Configuration[PDC] = 1, this field contains the one-hot representation of one or more DMA Channel numbers to which a receive packet whose DA matches MAC Address 2 Low (MAC_Address2_Low) content is routed.
15-0 ADDRHI	MAC Address 1 [47:32] Contains the upper 16 bits [47:32] of the second 6-byte MAC address.

75.17.56 MAC Address 2 Low (MAC_Address2_Low)

Offset

Register	Offset
MAC_Address2_Low	314h

Function

Holds the lower 32 bits of the second 6-byte MAC address of the station.

Diagram



Fields

Field	Function
31-0 ADDRLO	MAC Address 1 [31:0] Contains the lower 32 bits of the second 6-byte MAC address. The content of this field remains undefined until you load it after the initialization process.

75.17.57 MMC Control (MMC_Control)

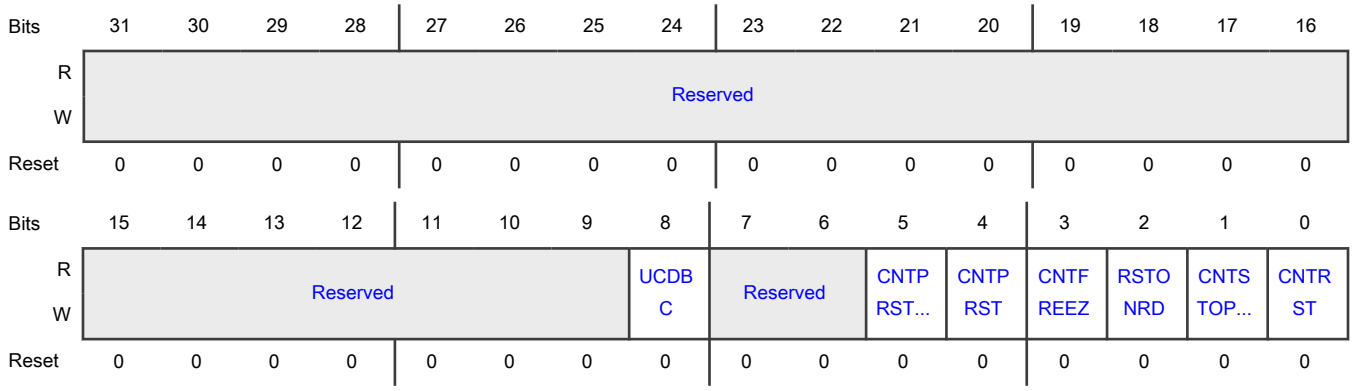
Offset

Register	Offset
MMC_Control	700h

Function

Establishes the operating mode for MMC.

Diagram



Fields

Field	Function
31-9 —	Reserved
8 UCDBC	Update MMC Counters For Dropped Broadcast Packets Indicates the status of MMC counters for dropped broadcast packets. • If this field is 1, MAC updates all the related MMC counters for broadcast packets that are dropped because MAC_Packet_Filter[DBF] = 1. • If this field is 0, the MMC counters are not updated for dropped broadcast packets. **NOTE** The CNTRST field has a higher priority than the CNTPRST field. Therefore, if you write 1 to both these fields in the same write cycle, all the counters are cleared and the CNTPRST field becomes 0. 0b - Disabled 1b - Enabled
7-6 —	Reserved
5 CNTPRSTLVL	Full-Half Preset Indicates whether full-half preset is enabled or disabled. • If this field is 0 and the CNTPRST field is 1, all the MMC counters are preset to the almost-half value. All octet counters are preset to 7FFF_F800h (half 2 Kbytes) and all packet counters are preset to 7FFF_FFF0h (half 16). • If this field is 1 and the CNTPRST field is 1 too, all MMC counters are preset to the almost-full value. All octet counters are preset to FFFF_F800h (full 2 Kbytes) and all packet counters are preset to FFFF_FFF0h (full 16). For 16-bit counters, the almost-half preset values are 7800h and 7FF0h for the respective octet and packet counters. Similarly, the almost-full preset values for the 16-bit counters are 0xF800 and 0xFFF0. 0b - Disabled 1b - Enabled
4 CNTPRST	Counters Preset Indicates whether MMC counter preset is enabled or disabled. If this field is 1, all the counters are initialized or preset to almost full or almost half according to the settings of the CNTPRSTLVL field. This field clears automatically after 1 clock cycle, and along with the CNTPRSTLVL field, is useful for debugging and testing the assertion of interrupts because of the MMC counter becoming half-full or full. Access restrictions apply to this field, which clears automatically. 0b - Disabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Enabled
3 CNTFREEZ	MMC Counter Freeze Indicates the status of MMC counter freeze. If this field is 1, the field freezes all the MMC counters to their current values. Until this field becomes 0, no MMC counter is updated because of any transmitted or received packets. If an MMC counter is read when RSTONRD = 1, that counter is also cleared in this mode. 0b - Disabled 1b - Enabled
2 RSTONRD	Reset On Read Indicates whether the reset on read for MMC counters is enabled or disabled. If this field is 1, MMC counters are reset to 0 after read (they clear automatically after reset). The counters are cleared when the least significant byte lane (bits [7:0]) is read. 0b - Disabled 1b - Enabled
1 CNTSTOPRO	Counter Stop Rollover Indicates the status of counter stop rollover. If this field is 1, the counter does not roll over to 0 after reaching the maximum value. 0b - Disabled 1b - Enabled
0 CNTRST	Counters Reset Indicates whether MMC counters are reset. If this field is 1, all the counters are reset. This field clears automatically after 1 clock cycle. Access restrictions apply to this field that clears automatically. 0b - Counters are not reset 1b - All counters are reset

75.17.58 MMC Receive Interrupt (MMC_Rx_Interrupt)

Offset

Register	Offset
MMC_Rx_Interrupt	704h

Function

Maintains the interrupts generated from all receive statistics counters when the following conditions occur:

- Receive statistic counters reach half of their maximum values (8000_0000h for a 32-bit counter and 8000h for a 16-bit counter).
- Receive statistic counters cross their maximum values (FFFF_FFFFh for a 32-bit counter and FFFFh for a 16-bit counter).

If MMC_Control[CNTSTOPRO] = 1, interrupts are set but the counter remains at all-ones. This is a 32-bit register. An interrupt bit is cleared when the respective MMC counter that caused the interrupt is read. The least significant byte lane (bits [7:0]) of the respective counter must be read to clear the interrupt bit.

NOTE

R_SS_RC means that this register bit is set internally, and is cleared when the counter register is read.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved				Reserved	Reserved	RXCTRLP...	RXRCVER...	RXWDOGP...	RXVLANG...	RXFOVPIS	RXPAUSP...	RXORANG...	RXLENER...	RXUCGPIS	RX1024T...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RX512T1...	RX256T5...	RX128T2...	RX65T12...	RX64OCT...	RXOSIZE...	RXUSIZE...	RXJABER...	RXRUNTP...	RXALGNE...	RXCRCER...	RXMCGPIS	RXBCGPIS	RXGOCTIS	RXGBOCT...	RXGBPKT...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved
27 —	Reserved
26 —	Reserved
25 RXCTRLPIS	MMC Receive Control Packet Counter Interrupt Status Indicates whether the status of the MMC receive control packet counter interrupt is detected. This field becomes 1 when the rxctrlpackets_g counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected

Table continues on the next page...

Table continued from the previous page...

Field	Function
24 RXRCVERRPIS	MMC Receive Error Packet Counter Interrupt Status Indicates whether the status of the MMC receive error packet counter interrupt is detected. This field becomes 1 when the rxrcverror counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
23 RXWDOGPIS	MMC Receive Watchdog Error Packet Counter Interrupt Status Indicates whether the status of the MMC receive watchdog error packet counter interrupt is detected. This field becomes 1 when the rxwatchdog error counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
22 RXVLANGBPIS	MMC Receive VLAN Good Bad Packet Counter Interrupt Status Indicates whether the status of the MMC receive VLAN good bad packet counter interrupt is detected. This field becomes 1 when the rxvlanpackets_gb counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
21 RXFOVPIS	MMC Receive FIFO Overflow Packet Counter Interrupt Status Indicates whether the status of the MMC receive FIFO overflow packet counter interrupt is detected. This field becomes 1 when the rxfifooverflow counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
20 RXPAUSPIS	MMC Receive Pause Packet Counter Interrupt Status Indicates whether the status of the MMC receive pause packet counter interrupt is detected. This field becomes 1 when the rxpausepackets counter reaches half of its maximum value or the maximum value.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
19 RXORANGEPIS	MMC Receive Out-Of-Range Error Packet Counter Interrupt Status Indicates whether the status of the MMC receive out-of-range error packet counter interrupt is detected. This field becomes 1 when the rxoutofrangetype counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
18 RXLENERPIS	MMC Receive Length Error Packet Counter Interrupt Status Indicates whether the status of the MMC receive length error packet counter interrupt is detected. This field becomes 1 when the rxlengtherror counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
17 RXUCGPIS	MMC Receive Unicast Good Packet Counter Interrupt Status Indicates whether the status of the MMC receive unicast good packet counter interrupt is detected. This field becomes 1 when the rxunicastpackets_g counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
16 RX1024TMAXOCTGBPIS	MMC Receive 1024 To Maximum Octet Good Bad Packet Counter Interrupt Status Indicates whether the status of the MMC receive 1024 to maximum octet good bad packet counter interrupt is detected. This field becomes 1 when the rx1024tomaxoctets_gb counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Not detected 1b - Detected
15 RX512T1023OCTGBPIS	MMC Receive 512 To 1023 Octet Good Bad Packet Counter Interrupt Status Indicates whether the status of the MMC receive 512 to 1023 octet good bad packet counter interrupt is detected. This field becomes 1 when the rx512to1023octets_gb counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
14 RX256T511OCTGBPIS	MMC Receive 256 To 511 Octet Good Bad Packet Counter Interrupt Status Indicates whether the status of the MMC receive 256 to 511 octet good bad packet counter interrupt is detected. This field becomes 1 when the rx256to511octets_gb counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
13 RX128T255OCTGBPIS	MMC Receive 128 To 255 Octet Good Bad Packet Counter Interrupt Status Indicates whether the status of the MMC receive 128 to 255 octet good bad packet counter interrupt is detected. This field becomes 1 when the rx128to255octets_gb counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
12 RX65T127OCTGBPIS	MMC Receive 65 To 127 Octet Good Bad Packet Counter Interrupt Status Indicates whether the status of the MMC receive 65 to 127 octet good bad packet counter interrupt is detected. The field becomes 1 when the rx65to127octets_gb counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Not detected 1b - Detected
11 RX64OCTGBPIS	MMC Receive 64 Octet Good Bad Packet Counter Interrupt Status Indicates whether the status of the MMC receive 64-octet good bad packet counter interrupt is detected. The field becomes 1 when the rx64octets_gb counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
10 RXOSIZEGPIS	MMC Receive Oversize Good Packet Counter Interrupt Status Indicates whether the status of the MMC receive oversize good packet counter interrupt is detected. This field becomes 1 when the rxoversize_g counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
9 RXUSIZEGPIS	MMC Receive Undersize Good Packet Counter Interrupt Status Indicates whether the status of MMC receive undersize good packet counter interrupt is detected. This field becomes 1 when the rxundersize_g counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
8 RXJABERPIS	MMC Receive Jabber Error Packet Counter Interrupt Status Indicates whether the status of MMC receive jabber error packet counter interrupt is detected. This field becomes 1 when the rxjabbererror counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected

Table continues on the next page...

Table continued from the previous page...

Field	Function
7 RXRUNTPIS	MMC Receive Runt Packet Counter Interrupt Status Indicates whether the status of the MMC receive runt packet counter interrupt is detected. This field becomes 1 when the rxrunterror counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
6 RXALGNERPIS	MMC Receive Alignment Error Packet Counter Interrupt Status Indicates whether the status of the MMC receive alignment error packet counter interrupt is detected. This field becomes 1 when the rxalignmenterror counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
5 RXCRCERPIS	MMC Receive CRC Error Packet Counter Interrupt Status Indicates whether the status of the MMC receive CRC error packet counter interrupt is detected. This field becomes 1 when the rxcrcerror counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
4 RXMCGPIS	MMC Receive Multicast Good Packet Counter Interrupt Status Indicates whether the status of the MMC receive multicast good packet counter interrupt is detected. This field becomes 1 when the rxmulticastpackets_g counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
3 RXBCGPIS	MMC Receive Broadcast Good Packet Counter Interrupt Status Indicates whether the status of the MMC receive broadcast good packet counter interrupt is detected. This field becomes 1 when the rxbroadcastpackets_g counter reaches half of its maximum value or the maximum value.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
2 RXGOCTIS	MMC Receive Good Octet Counter Interrupt Status Indicates whether the status of the MMC receive good octet counter interrupt is detected. This field becomes 1 when the rxoctetcount_g counter reaches half of its maximum value or its maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
1 RXGBOCTIS	MMC Receive Good Bad Octet Counter Interrupt Status Indicates whether the status of the MMC receive good bad octet counter interrupt is detected. This field becomes 1 when the rxoctetcount_gb counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
0 RXGBPKTIS	MMC Receive Good Bad Packet Counter Interrupt Status Indicates whether the status of the MMC receive good bad packet counter interrupt is detected. This field becomes 1 when the rxpacketcount_gb counter reaches half of its maximum value or its maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected

75.17.59 MMC Transmit Interrupt (MMC_Tx_Interrupt)

Offset

Register	Offset
MMC_Tx_Interrupt	708h

Function

Maintains the interrupts generated from all the transmit statistics counters.

This register maintains those interrupts that are generated when transmit statistic counters:

- Reach half of their maximum values (8000_0000h for a 32-bit counter and 0x8000 for a 16-bit counter)
- Cross their maximum values (FFFF_FFFFh for a 32-bit counter and FFFFh for a 16-bit counter)

If MMC_Control[CNTSTOPRO] = 1, the interrupts are set but the counter remains at all-ones.

This is a 32 bit register. An interrupt bit is cleared when the respective MMC counter that caused the interrupt is read.

The least significant byte lane (bits [7:0]) of the respective counter must be read to clear the interrupt bit.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved				Reserved	Reserved	TXOSIZE...	TXVLANG...	TXPAUSP...	TXEXDEF...	TXGPKTIS	TXGOCTIS	TXCARER...	TXEXCOL...	TXLATCO...	TXDEFPIS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TXMCOLG...	TXSCOLG...	TXUFLOW...	TXBCGBP...	TXMCGBP...	TXUCGBP...	TX1024T...	TX512T1...	TX256T5...	TX128T2...	TX65T12...	TX64OCT...	TXMCGPIS	TXBCGPIS	TXGBPKT...	TXGBOCT...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved
27 —	Reserved
26 —	Reserved
25 TXOSIZEGPIS	MMC Transmit Oversize Good Packet Counter Interrupt Status Indicates whether the status of the MMC transmit oversize good packet counter interrupt is detected. This field becomes 1 when the txoversize_g counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Detected
24 TXVLANGPIS	MMC Transmit VLAN Good Packet Counter Interrupt Status Indicates whether the status of the MMC transmit VLAN good packet counter interrupt is detected. This field becomes 1 when the txvlanpackets_g counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
23 TXPAUSPIS	MMC Transmit Pause Packet Counter Interrupt Status Indicates whether the status of the MMC transmit pause packet counter interrupt is detected. This field becomes 1 when the txpausepacketserror counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
22 TXEXDEFPIS	MMC Transmit Excessive Deferral Packet Counter Interrupt Status Indicates whether the status of the MMC transmit excessive deferral packet counter interrupt is detected. This field becomes 1 when the txexcessdef counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
21 TXGPKTIS	MMC Transmit Good Packet Counter Interrupt Status Indicates whether the status of the MMC transmit good packet counter interrupt is detected. This field becomes 1 when the txpacketcount_g counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
20	MMC Transmit Good Octet Counter Interrupt Status

Table continues on the next page...

Table continued from the previous page...

Field	Function
TXGOCTIS	Indicates whether the status of the MMC transmit good octet counter interrupt is detected. This field becomes 1 when the txoctetcount_g counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
19 TXCARERPIS	MMC Transmit Carrier Error Packet Counter Interrupt Status Indicates whether the status of the MMC transmit carrier error packet counter interrupt is detected. This field becomes 1 when the txcarriererror counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
18 TXEXCOLPIS	MMC Transmit Excessive Collision Packet Counter Interrupt Status Indicates whether the status of the MMC transmit excessive collision packet counter interrupt is detected. This field becomes 1 when the txexesscol counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
17 TXLATCOLPIS	MMC Transmit Late Collision Packet Counter Interrupt Status Indicates whether the status of the MMC transmit late collision packet counter interrupt is detected. This field becomes 1 when the txlatecol counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
16 TXDEFPIS	MMC Transmit Deferred Packet Counter Interrupt Status Indicates whether the status of the MMC transmit deferred packet counter interrupt is detected. This field becomes 1 when the txdeferred counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Not detected 1b - Detected
15 TXMCOLGPIS	MMC Transmit Multiple Collision Good Packet Counter Interrupt Status Indicates whether the status of the MMC transmit multiple collision good packet counter interrupt is detected. This field becomes 1 when the txmulticol_g counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
14 TXSCOLGPIS	MMC Transmit Single Collision Good Packet Counter Interrupt Status Indicates whether the status of the MMC transmit single collision good packet counter interrupt is detected. This field becomes 1 when the txsinglecol_g counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
13 TXUFLOWERPIS	MMC Transmit Underflow Error Packet Counter Interrupt Status Indicates whether the status of the MMC transmit underflow error packet counter interrupt is detected. This field becomes 1 when the txunderflowerror counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
12 TXBCGBPIS	MMC Transmit Broadcast Good Bad Packet Counter Interrupt Status Indicates whether the status of the MMC transmit broadcast good bad packet counter interrupt is detected. This field becomes 1 when the txbroadcastpackets_gb counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected

Table continues on the next page...

Table continued from the previous page...

Field	Function
11 TXMCGBPIS	MMC Transmit Multicast Good Bad Packet Counter Interrupt Status Indicates whether the status of the MMC transmit multicast good bad packet counter interrupt is detected. This field becomes 1 when the txmulticastpackets_gb counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
10 TXUCGBPIS	MMC Transmit Unicast Good Bad Packet Counter Interrupt Status Indicates whether the status of the MMC transmit unicast good bad packet counter interrupt is detected. This field becomes 1 when the txunicastpackets_gb counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
9 TX1024TMAXOCTGBPIS	MMC Transmit 1024 To Maximum Octet Good Bad Packet Counter Interrupt Status Indicates whether the status of the MMC transmit 1024 to maximum octet good bad packet counter interrupt is detected. This field becomes 1 when the tx1024tomaxoctets_gb counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
8 TX512T1023OCTGBPIS	MMC Transmit 512 To 1023 Octet Good Bad Packet Counter Interrupt Status Indicates whether the status of the MMC transmit 512 to 1023 octet good bad packet counter interrupt is detected. This field becomes 1 when the tx512to1023octets_gb counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
7	MMC Transmit 256 To 511 Octet Good Bad Packet Counter Interrupt Status

Table continues on the next page...

Table continued from the previous page...

Field	Function
TX256T511OCTGBPIS	Indicates whether the status of the MMC transmit 256 to 511 octet good bad packet counter interrupt is detected. This field becomes 1 when the tx256to511octets_gb counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
6 TX128T255OCTGBPIS	MMC Transmit 128 To 255 Octet Good Bad Packet Counter Interrupt Status Indicates whether the status of the MMC transmit 128 to 255 octet good bad packet counter interrupt is detected. This field becomes 1 when the tx128to255octets_gb counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
5 TX65T127OCTGBPIS	MMC Transmit 65 To 127 Octet Good Bad Packet Counter Interrupt Status Indicates whether the status of the MMC transmit 65 to 127 octet good bad packet counter interrupt is detected. This field becomes 1 when the tx65to127octets_gb counter reaches half of its maximum value, and also when it reaches the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
4 TX64OCTGBPIS	MMC Transmit 64-Octet Good Bad Packet Counter Interrupt Status Indicates whether the status of the MMC transmit 64-octet good bad packet counter interrupt is detected. This field becomes 1 when the tx64octets_gb counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
3 TXMCGPIS	MMC Transmit Multicast Good Packet Counter Interrupt Status Indicates whether the status of the MMC transmit multicast good packet counter interrupt is detected.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field becomes 1 when the txmulticastpackets_g counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
2 TXBCGPIS	MMC Transmit Broadcast Good Packet Counter Interrupt Status Indicates whether the status of the MMC transmit broadcast good packet counter interrupt is detected. This field becomes 1 when the txbroadcastpackets_g counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
1 TXGBPKTIS	MMC Transmit Good Bad Packet Counter Interrupt Status Indicates whether the status of the MMC transmit good bad packet counter interrupt is detected. This field becomes 1 when the txpacketcount_gb counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
0 TXGBOCTIS	MMC Transmit Good Bad Octet Counter Interrupt Status Indicates whether the status of the MMC transmit good bad octet counter interrupt is detected. This field becomes 1 when the txoctetcount_gb counter reaches half of its maximum value or the maximum value. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected

75.17.60 MMC Receive Interrupt Mask (MMC_Rx_Interrupt_Mask)

Offset

Register	Offset
MMC_Rx_Interrupt_Mask	70Ch

Function

Maintains the masks for interrupts generated from all the receive statistic counters. These interrupts are generated when the receive statistic counters reach half of their maximum values or the maximum values.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved				Reserved	Reserved	RXCTRLP...	RXRCVER...	RXWDOGP...	RXVLANG...	RXFOVPIM	RXPAUSP...	RXORANG...	RXLENER...	RXUCGPIM	RX1024T...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RX512T1...	RX256T5...	RX128T2...	RX65T12...	RX64OCT...	RXOSIZE...	RXUSIZE...	RXJABER...	RXRUNTP...	RXALGNE...	RXCRCER...	RXMCGPIM	RXBCGPIM	RXGOCTIM	RXGBOCT...	RXGBPKT...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved
27 —	Reserved
26 —	Reserved
25 RXCTRLPIM	MMC Receive Control Packet Counter Interrupt Mask Indicates the status of the MMC receive control packet counter interrupt mask. Writing 1 to this field masks the interrupt when the rxctrlpackets_g counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
24 RXRCVERRPIM	MMC Receive Error Packet Counter Interrupt Mask Indicates the status of the MMC receive error packet counter interrupt mask. Writing 1 to this field masks the interrupt when the rxrcverror counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
23	MMC Receive Watchdog Error Packet Counter Interrupt Mask

Table continues on the next page...

Table continued from the previous page...

Field	Function
RXWDOGPIM	Indicates the status of the MMC receive watchdog error packet counter interrupt mask. Writing 1 to this field masks the interrupt when the rxwatchdog counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
22 RXVLANGBPIM	MMC Receive VLAN Good Bad Packet Counter Interrupt Mask Indicates the status of the MMC receive VLAN good bad packet counter interrupt mask. Writing 1 to this field masks the interrupt when the rxvlanpackets_gb counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
21 RXFOVPIM	MMC Receive FIFO Overflow Packet Counter Interrupt Mask Indicates the status of the MMC receive FIFO overflow packet counter interrupt mask. Writing 1 to this field masks the interrupt when the rxfifooverflow counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
20 RXPAUSPIM	MMC Receive Pause Packet Counter Interrupt Mask Indicates the status of the MMC receive pause packet counter interrupt mask. Writing 1 to this field masks the interrupt when the rxpausepackets counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
19 RXORANGEPIM	MMC Receive Out-Of-Range Error Packet Counter Interrupt Mask Indicates the status of the MMC receive out-of-range error packet counter interrupt mask. Writing 1 to this field masks the interrupt when the rxoutofrangetype counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
18 RXLENERPIM	MMC Receive Length Error Packet Counter Interrupt Mask Indicates the status of the MMC receive length error packet counter interrupt mask. Writing 1 to this field masks the interrupt when the rxlengtherror counter reaches half of its maximum value or the maximum value.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disabled 1b - Enabled
17 RXUCGPIM	MMC Receive Unicast Good Packet Counter Interrupt Mask Indicates the status of the MMC receive unicast good packet counter interrupt mask. Writing 1 to this field masks the interrupt when the rxunicastpackets_g counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
16 RX1024TMAXOCTGBPIM	MMC Receive 1024 To Maximum Octet Good Bad Packet Counter Interrupt Mask Indicates the status of the MMC receive 1024 to maximum octet good bad packet counter interrupt mask. Writing 1 to this field masks the interrupt when the rx1024tomaxoctets_gb counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
15 RX512T1023OCTGBPIM	MMC Receive 512 To 1023 Octet Good Bad Packet Counter Interrupt Mask Indicates the status of the MMC receive 512 to 1023 octet good bad packet counter interrupt mask. Writing 1 to this field masks the interrupt when the rx512to1023octets_gb counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
14 RX256T511OCTGBPIM	MMC Receive 256 To 511 Octet Good Bad Packet Counter Interrupt Mask Indicates the status of the MMC receive 256 to 511 octet good bad packet counter interrupt mask. Writing 1 to this field masks the interrupt when the rx256to511octets_gb counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
13 RX128T255OCTGBPIM	MMC Receive 128 To 255 Octet Good Bad Packet Counter Interrupt Mask Indicates the status of the MMC receive 128 to 255 octet good bad packet counter interrupt mask. Writing 1 to this field masks the interrupt when the rx128to255octets_gb counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
12	MMC Receive 65 To 127 Octet Good Bad Packet Counter Interrupt Mask

Table continues on the next page...

Table continued from the previous page...

Field	Function
RX65T127OCTGBPIM	Indicates the status of the MMC receive 65 to 127 octet good bad packet counter interrupt mask. Writing 1 to this field masks the interrupt when the rx65to127octets_gb counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
11 RX64OCTGBPIM	MMC Receive 64-Octet Good Bad Packet Counter Interrupt Mask Indicates the status of the MMC receive 64-octet good bad packet counter interrupt mask. Writing 1 to this field masks the interrupt when the rx64octets_gb counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
10 RXOSIZEGPIM	MMC Receive Oversize Good Packet Counter Interrupt Mask Indicates the status of the MMC receive oversize good packet counter interrupt mask. Writing 1 to this field masks the interrupt when the rxoversize_g counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
9 RXUSIZEGPIM	MMC Receive Undersize Good Packet Counter Interrupt Mask Indicates the status of the MMC receive undersize good packet counter interrupt mask. Writing 1 to this field masks the interrupt when the rxundersize_g counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
8 RXJABERPIM	MMC Receive Jabber Error Packet Counter Interrupt Mask Indicates the status of the MMC receive jabber error packet counter interrupt mask. Writing 1 to this field masks the interrupt when the rxjabbererror counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
7 RXRUNTPIM	MMC Receive Runt Packet Counter Interrupt Mask Indicates the status of the MMC receive runt packet counter interrupt mask. Writing 1 to this field masks the interrupt when the rxrunterror counter reaches half of its maximum value or the maximum value.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disabled 1b - Enabled
6 RXALGNERPIM	MMC Receive Alignment Error Packet Counter Interrupt Mask Indicates the status of the MMC receive alignment error packet counter interrupt mask. Writing 1 to this field masks the interrupt when the rxalignmenterror counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
5 RXCRCERPIM	MMC Receive CRC Error Packet Counter Interrupt Mask Indicates the status of the MMC receive CRC error packet counter interrupt mask. Writing 1 to this field masks the interrupt when the rxcrcerror counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
4 RXMCGPIM	MMC Receive Multicast Good Packet Counter Interrupt Mask Indicates the status of the MMC receive multicast good packet counter interrupt mask. Writing 1 to this field masks the interrupt when the rxmulticastpackets_g counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
3 RXBCGPIM	MMC Receive Broadcast Good Packet Counter Interrupt Mask Indicates the status of the MMC receive broadcast good packet counter interrupt mask. Writing 1 to this field masks the interrupt when the rxbroadcastpackets_g counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
2 RXGOCTIM	MMC Receive Good Octet Counter Interrupt Mask Indicates the status of the MMC receive good octet counter interrupt mask. Writing 1 to this field masks the interrupt when the rxoctetcount_g counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
1	MMC Receive Good Bad Octet Counter Interrupt Mask

Table continues on the next page...

Table continued from the previous page...

Field	Function
RXGBOCTIM	Indicates the status of the MMC receive good bad octet counter interrupt mask. Writing 1 to this field masks the interrupt when the rxoctetcount_gb counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
0 RXGBPKTIM	MMC Receive Good Bad Packet Counter Interrupt Mask Indicates the status of the MMC receive good bad packet counter interrupt mask. Writing 1 to this field masks the interrupt when the rxpacketcount_gb counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled

75.17.61 MMC Transmit Interrupt Mask (MMC_Tx_Interrupt_Mask)

Offset

Register	Offset
MMC_Tx_Interrupt_Mask	710h

Function

Maintains the masks for interrupts that are generated when the transmit statistic counters reach half of their maximum values or their maximum values.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved				Reserved	Reserved	TXOSIZE...	TXVLANG...	TXPAUSP...	TXEXDEF...	TXGPKTIM	TXGOCTIM	TXCARER...	TXEXCOL...	TXLATCO...	TXDEFPIM
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	TXMCOLG...	TXSCOLG...	TXUFLOW...	TXBCGBP...	TXMCGBP...	TXUCGBP...	TX1024T...	TX512T1...	TX256T5...	TX128T2...	TX65T12...	TX64OCT...	TXMCGPIM	TXBCGPIM	TXGBPKT...	TXGBOCT...
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved
27 —	Reserved
26 —	Reserved
25 TXOSIZEGPIM	MMC Transmit Oversize Good Packet Counter Interrupt Mask Indicates the status of the MMC transmit oversize good packet counter interrupt mask. Writing 1 to this field masks the interrupt when the txoversize_g counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
24 TXVLANGPIM	MMC Transmit VLAN Good Packet Counter Interrupt Mask Indicates the status of the MMC transmit VLAN good packet counter interrupt mask. Writing 1 to this field masks the interrupt when the txvlanpackets_g counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
23 TXPAUSPIM	MMC Transmit Pause Packet Counter Interrupt Mask Indicates the status of the MMC transmit pause packet counter interrupt mask. Writing 1 to this field masks the interrupt when the txpausepackets counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
22 TXEXDEFPIM	MMC Transmit Excessive Deferral Packet Counter Interrupt Mask Indicates the status of the MMC transmit excessive deferral packet counter interrupt mask. Writing 1 to this field masks the interrupt when the txexcessdef counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
21 TXGPKTIM	MMC Transmit Good Packet Counter Interrupt Mask Indicates the status of the MMC transmit good packet counter interrupt mask.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Writing 1 to this field masks the interrupt when the txpacketcount_g counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
20 TXGOCTIM	MMC Transmit Good Octet Counter Interrupt Mask Indicates the status of the MMC transmit good octet counter interrupt mask. Writing 1 to this field masks the interrupt when the txoctetcount_g counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
19 TXCARERPIM	MMC Transmit Carrier Error Packet Counter Interrupt Mask Indicates the status of the MMC transmit carrier error packet counter interrupt mask. Writing 1 to this field masks the interrupt when the txcarriererror counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
18 TXEXCOLPIM	MMC Transmit Excessive Collision Packet Counter Interrupt Mask Indicates the status of the MMC transmit excessive collision packet counter interrupt mask. Writing 1 to this field masks the interrupt when the txexcesscol counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
17 TXLATCOLPIM	MMC Transmit Late Collision Packet Counter Interrupt Mask Indicates the status of the MMC transmit late collision packet counter interrupt mask. Writing 1 to this field masks the interrupt when the txlatecol counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
16 TXDEFPIM	MMC Transmit Deferred Packet Counter Interrupt Mask Indicates the status of the MMC transmit deferred packet counter interrupt mask. Writing 1 to this field masks the interrupt when the txdeferred counter reaches half of the maximum value or the maximum value. 0b - Disabled 1b - Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
15 TXMCOLGPIM	MMC Transmit Multiple Collision Good Packet Counter Interrupt Mask Indicates the status of the MMC transmit multiple collision good packet counter interrupt mask. Writing 1 to this field masks the interrupt when the txmulticol_g counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
14 TXSCOLGPIM	MMC Transmit Single Collision Good Packet Counter Interrupt Mask Indicates the status of the MMC transmit single collision good packet counter interrupt mask. Writing 1 to this field masks the interrupt when the txsinglecol_g counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
13 TXUFLOWERPIM	MMC Transmit Underflow Error Packet Counter Interrupt Mask Indicates the status of the MMC transmit underflow error packet counter interrupt mask. Writing 1 to this field masks the interrupt when the txunderflowerror counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
12 TXBCGBPIM	MMC Transmit Broadcast Good Bad Packet Counter Interrupt Mask Indicates the status of the MMC transmit broadcast good bad packet counter interrupt mask. Writing 1 to this field masks the interrupt when the txbroadcastpackets_gb counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
11 TXMCGBPIM	MMC Transmit Multicast Good Bad Packet Counter Interrupt Mask Indicates the status of the MMC transmit multicast good bad packet counter interrupt mask. Writing 1 to this field masks the interrupt when the txmulticastpackets_gb counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
10 TXUCGBPIM	MMC Transmit Unicast Good Bad Packet Counter Interrupt Mask Indicates the status of the MMC transmit unicast good bad packet counter interrupt mask. Writing 1 to this field masks the interrupt when the txunicastpackets_gb counter reaches half of its maximum value or the maximum value.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disabled 1b - Enabled
9 TX1024TMAXOCTGBPIM	MMC Transmit 1024 To Maximum Octet Good Bad Packet Counter Interrupt Mask Indicates the status of the MMC transmit 1024 to maximum octet good bad packet counter interrupt mask. Writing 1 to this field masks the interrupt when the tx1024tomaxoctets_gb counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
8 TX512T1023OCTGBPIM	MMC Transmit 512 To 1023 Octet Good Bad Packet Counter Interrupt Mask Indicates the status of the MMC transmit 512 to 1023 octet good bad packet counter interrupt mask. Writing 1 to this field masks the interrupt when the tx512to1023octets_gb counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
7 TX256T511OCTGBPIM	MMC Transmit 256 To 511 Octet Good Bad Packet Counter Interrupt Mask Indicates the status of the MMC transmit 256 to 511 octet good bad packet counter interrupt mask. Writing 1 to this field masks the interrupt when the tx256to511octets_gb counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
6 TX128T255OCTGBPIM	MMC Transmit 128 To 255 Octet Good Bad Packet Counter Interrupt Mask Indicates the status of the MMC transmit 128 to 255 octet good bad packet counter interrupt mask. Writing 1 to this field masks the interrupt when the tx128to255octets_gb counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
5 TX65T127OCTGBPIM	MMC Transmit 65 To 127 Octet Good Bad Packet Counter Interrupt Mask Indicates the status of the MMC transmit 65 to 127 octet good bad packet counter interrupt mask. Writing 1 to this field masks the interrupt when the tx65to127octets_gb counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
4	MMC Transmit 64-Octet Good Bad Packet Counter Interrupt Mask

Table continues on the next page...

Table continued from the previous page...

Field	Function
TX64OCTGBPIM	Indicates the status of the MMC transmit 64-octet good bad packet counter interrupt mask. Writing 1 to this field masks the interrupt when the tx64octets_gb counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
3 TXMCGPIM	MMC Transmit Multicast Good Packet Counter Interrupt Mask Indicates the status of the MMC transmit multicast good packet counter interrupt mask. Writing 1 to this field masks the interrupt when the txmulticastpackets_g counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
2 TXBCGPIM	MMC Transmit Broadcast Good Packet Counter Interrupt Mask Indicates the status of the MMC transmit broadcast good packet counter interrupt mask. Writing 1 to this field masks the interrupt when the txbroadcastpackets_g counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
1 TXGBPKTIM	MMC Transmit Good Bad Packet Counter Interrupt Mask Indicates the status of the MMC transmit good bad packet counter interrupt mask. Writing 1 to this field masks the interrupt when the txpacketcount_gb counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled
0 TXGBOCTIM	MMC Transmit Good Bad Octet Counter Interrupt Mask Indicates the status of the MMC transmit good bad octet counter interrupt mask. Writing 1 to this field masks the interrupt when the txoctetcount_gb counter reaches half of its maximum value or the maximum value. 0b - Disabled 1b - Enabled

75.17.62 Transmit Octet Count Good Bad (Tx_Octet_Count_Good_Bad)

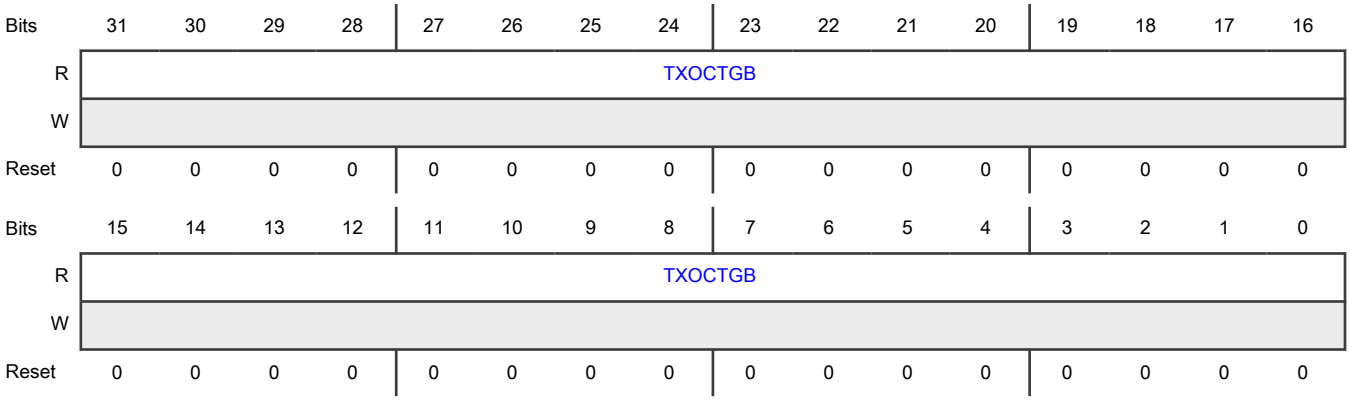
Offset

Register	Offset
Tx_Octet_Count_Good_Bad	714h

Function

Provides the number of bytes that the module transmitted, exclusive of preamble and retried bytes, in good and bad packets.

Diagram



Fields

Field	Function
31-0 TXOCTGB	Transmit Octet Count Good Bad Indicates the number of bytes transmitted, exclusive of preamble and retried bytes, in good and bad packets.

75.17.63 Transmit Packet Count Good Bad (Tx_Packet_Count_Good_Bad)

Offset

Register	Offset
Tx_Packet_Count_Good_Bad	718h

Function

Provides the number of good and bad packets that the module transmitted, exclusive of retried packets.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TXPKTGB															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TXPKTGB															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TXPKTGB	Transmit Packet Count Good Bad Indicates the number of good and bad packets transmitted, exclusive of retried packets.

75.17.64 Transmit Broadcast Packets Good (Tx_Broadcast_Packets_Good)

Offset

Register	Offset
Tx_Broadcast_Packets_Good	71Ch

Function

Provides the number of good broadcast packets that the module transmitted.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TXBCASTG															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TXBCASTG															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TXBCASTG	Transmit Broadcast Packets Good Indicates the number of good broadcast packets transmitted.

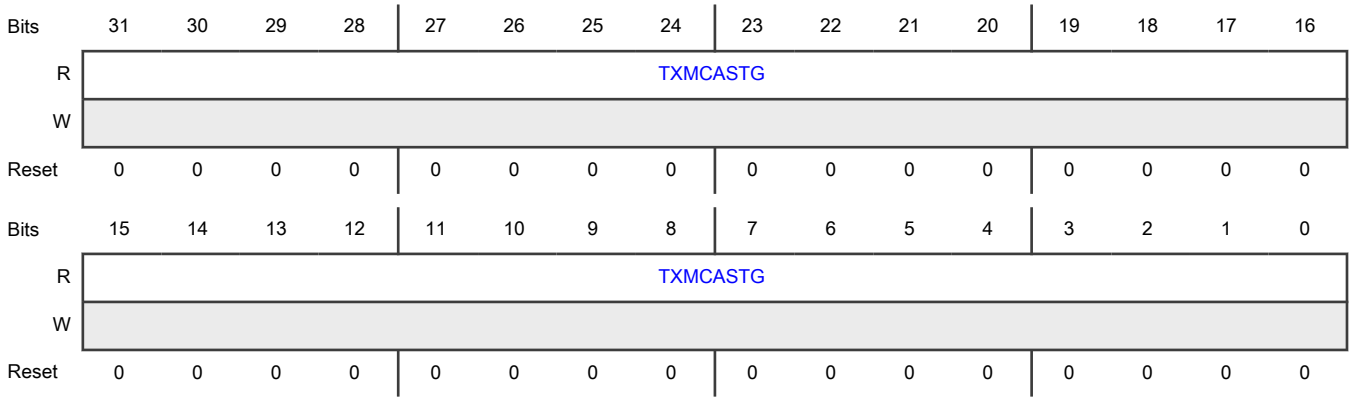
75.17.65 Transmit Multicast Packets Good (Tx_Multicast_Packets_Good)

Offset

Register	Offset
Tx_Multicast_Packets_Good	720h

Function
Provides the number of good multicast packets that the module transmitted.

Diagram



Fields

Field	Function
31-0 TXMCASTG	Transmit Multicast Packets Good Indicates the number of good multicast packets transmitted.

75.17.66 Transmit 64-Octet Packets Good Bad (Tx_64Octets_Packets_Good_Bad)

Offset

Register	Offset
Tx_64Octets_Packets_Good_Bad	724h

Function

Provides the number of 64-byte good and bad packets that the module transmitted, exclusive of preamble and retried packets.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TX64OCTGB															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TX64OCTGB															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TX64OCTGB	Transmit 64-Octet Packets Good Bad Indicates the number of transmitted 64-byte good and bad packets, exclusive of preamble and retried packets.

75.17.67 Transmit 65 To 127 Octet Packets Good Bad (Tx_65To127Octets_Packets_Good_Bad)

Offset

Register	Offset
Tx_65To127Octets_Packets_Good_Bad	728h

Function

Provides the number of good and bad packets, having length between 65 and 127 (inclusive) bytes, that the module transmitted, exclusive of preamble and retried packets.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TX65_127OCTGB															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TX65_127OCTGB															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TX65_127OCTGB	Transmit 65 To 127 Octet Packets Good Bad Indicates the number of good and bad packets transmitted, having length between 65 and 127 (inclusive) bytes, exclusive of preamble and retried packets.

75.17.68 Transmit 128 To 255 Octet Packets Good Bad (Tx_128To255Octets_Packets_Good_Bad)

Offset

Register	Offset
Tx_128To255Octets_Packets_Good_Bad	72Ch

Function

Provides the number of good and bad packets that the module transmitted, having length between 128 to 255 (inclusive) bytes, exclusive of preamble and retried packets.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TX128_255OCTGB															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TX128_255OCTGB															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TX128_255OCTGB	Transmit 128 To 255 Octet Packets Good Bad Indicates the number of good and bad packets transmitted, having length between 128 and 255 (inclusive) bytes, exclusive of preamble and retried packets.

75.17.69 Transmit 256 To 511 Octet Packets Good Bad (Tx_256To511Octets_Packets_Good_Bad)

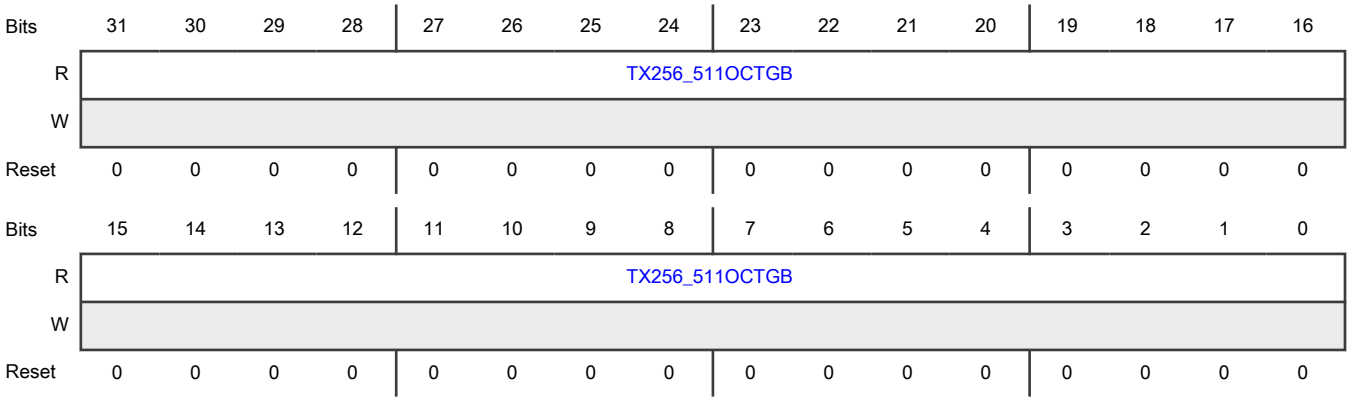
Offset

Register	Offset
Tx_256To511Octets_Packets_Good_Bad	730h

Function

Provides the number of good and bad packets that the module transmitted, having length between 256 to 511 (inclusive) bytes, exclusive of preamble and retried packets.

Diagram



Fields

Field	Function
31-0 TX256_511OCTGB	Transmit 256 To 511 Octet Packets Good Bad Indicates the number of good and bad packets transmitted, having length between 256 and 511 (inclusive) bytes, exclusive of preamble and retried packets.

75.17.70 Transmit 512 To 1023 Octet Packets Good Bad (Tx_512To1023Octets_Packets_Good_Bad)

Offset

Register	Offset
Tx_512To1023Octets_Packets_Good_Bad	734h

Function

Provides the number of good and bad packets that the module transmitted, having length between 512 to 1023 (inclusive) bytes, exclusive of preamble and retried packets.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TX512_1023OCTGB															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TX512_1023OCTGB															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TX512_1023OCTGB	Transmit 512 To 1023 Octet Packets Good Bad Indicates the number of good and bad packets transmitted, having length between 512 and 1023 (inclusive) bytes, exclusive of preamble and retried packets.

75.17.71 Transmit 1024 To Max Octet Packets Good Bad (Tx_1024ToMaxOctets_Packets_Good_Bad)

Offset

Register	Offset
Tx_1024ToMaxOctets_Packets_Good_Bad	738h

Function

Provides the number of good and bad packets that the module transmitted, having a length between 1024 bytes (inclusive) and the maximum size, exclusive of preamble and retried packets.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TX1024_MAXOCTGB															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TX1024_MAXOCTGB															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TX1024_MAXOCTGB	Transmit 1024 To Max Octet Packets Good Bad Indicates the number of good and bad packets transmitted, having a length between 1024 bytes (inclusive) and the maximum size, exclusive of preamble and retried packets.

75.17.72 Transmit Unicast Packets Good Bad (Tx_Unicast_Packets_Good_Bad)

Offset

Register	Offset
Tx_Unicast_Packets_Good_Bad	73Ch

Function

Provides the number of good and bad unicast packets that the module transmitted.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TXUCASTGB															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TXUCASTGB															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TXUCASTGB	Transmit Unicast Packets Good Bad Indicates the number of good and bad unicast packets transmitted.

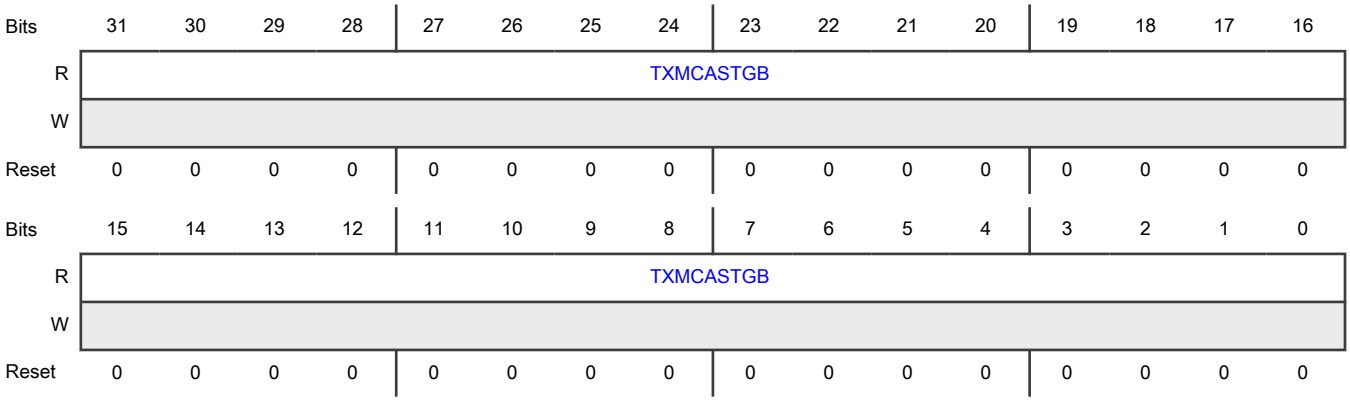
75.17.73 Transmit Multicast Packets Good Bad (Tx_Multicast_Packets_Good_Bad)

Offset

Register	Offset
Tx_Multicast_Packets_Good_Bad	740h

Function
Provides the number of good and bad multicast packets that the module transmitted.

Diagram



Fields

Field	Function
31-0 TXMCASTGB	Transmit Multicast Packets Good Bad Indicates the number of good and bad multicast packets transmitted.

75.17.74 Transmit Broadcast Packets Good Bad (Tx_Broadcast_Packets_Good_Bad)

Offset

Register	Offset
Tx_Broadcast_Packets_Good_Bad	744h

Function

Provides the number of good and bad broadcast packets that the module transmitted.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TXBCASTGB															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TXBCASTGB															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TXBCASTGB	Transmit Broadcast Packets Good Bad Indicates the number of good and bad broadcast packets transmitted.

75.17.75 Transmit Underflow Error Packets (Tx_Underflow_Error_Packets)

Offset

Register	Offset
Tx_Underflow_Error_Packets	748h

Function

Provides the number of packets that the module aborted because of a packet underflow error.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TXUNDRFLW															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TXUNDRFLW															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TXUNDRFLW	Transmit Underflow Error Packets Indicates the number of packets aborted because of a packet underflow error.

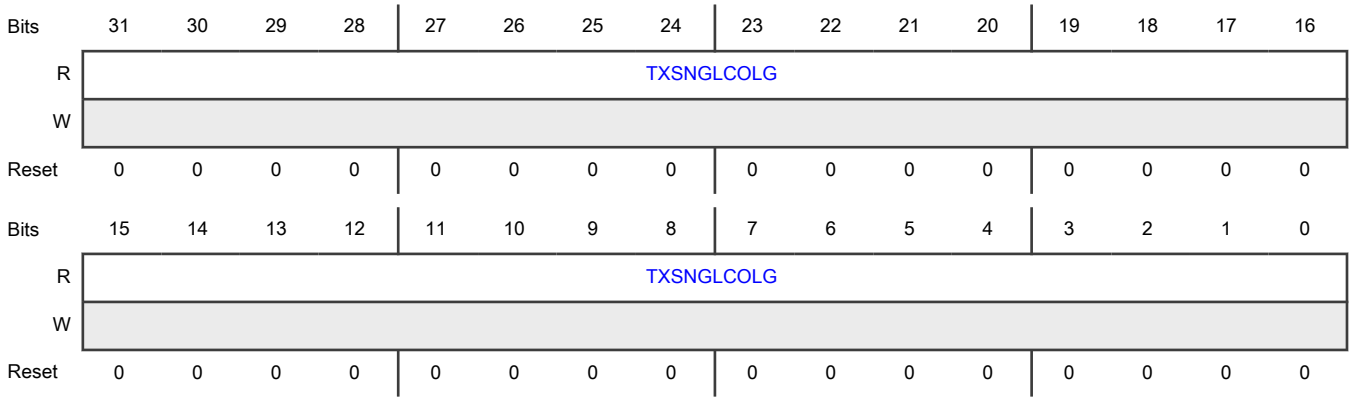
75.17.76 Transmit Single Collision Good Packets (Tx_Single_Collision_Good_Packets)

Offset

Register	Offset
Tx_Single_Collision_Good_Packets	74Ch

Function
Provides the number of packets that the module successfully transmitted after a single collision in Half-Duplex mode.

Diagram



Fields

Field	Function
31-0 TXSNGLCOLG	Transmit Single Collision Good Packets Indicates the number of successfully transmitted packets after a single collision in Half-Duplex mode.

75.17.77 Transmit Multiple Collision Good Packets (Tx_Multiple_Collision_Good_Packets)

Offset

Register	Offset
Tx_Multiple_Collision_Good_Packets	750h

Function

Provides the number of packets that the module successfully transmitted after multiple collisions in Half-Duplex mode.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TXMULTCOLG															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TXMULTCOLG															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TXMULTCOLG	Transmit Multiple Collision Good Packets Indicates the number of successfully transmitted packets after multiple collisions in Half-Duplex mode.

75.17.78 Transmit Deferred Packets (Tx_Deferred_Packets)

Offset

Register	Offset
Tx_Deferred_Packets	754h

Function

Provides the number of packets that the module successfully transmitted after a deferral in Half-Duplex mode.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TXDEFRD															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TXDEFRD															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TXDEFRD	Transmit Deferred Packets Indicates the number of successfully transmitted packets after a deferral in Half-Duplex mode.

75.17.79 Transmit Late Collision Packets (Tx_Late_Collision_Packets)

Offset

Register	Offset
Tx_Late_Collision_Packets	758h

Function
Provides the number of packets that the module aborted because of a late collision error.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TXLATECOL															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TXLATECOL															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TXLATECOL	Transmit Late Collision Packets Indicates the number of packets aborted because of a late collision error.

75.17.80 Transmit Excessive Collision Packets (Tx_Excessive_Collision_Packets)

Offset

Register	Offset
Tx_Excessive_Collision_Packets	75Ch

Function

Provides the number of packets that the module aborted because of excessive (16) collision errors.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TXEXSCOL															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TXEXSCOL															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TXEXSCOL	Transmit Excessive Collision Packets Indicates the number of packets aborted because of excessive (16) collision errors.

75.17.81 Transmit Carrier Error Packets (Tx_Carrier_Error_Packets)

Offset

Register	Offset
Tx_Carrier_Error_Packets	760h

Function

Provides the number of packets that the module aborted because of a carrier sense error (such as no carrier or loss of carrier).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TXCARR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TXCARR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TXCARR	Transmit Carrier Error Packets Indicates the number of packets aborted because of a carrier sense error (such as no carrier or loss of carrier).

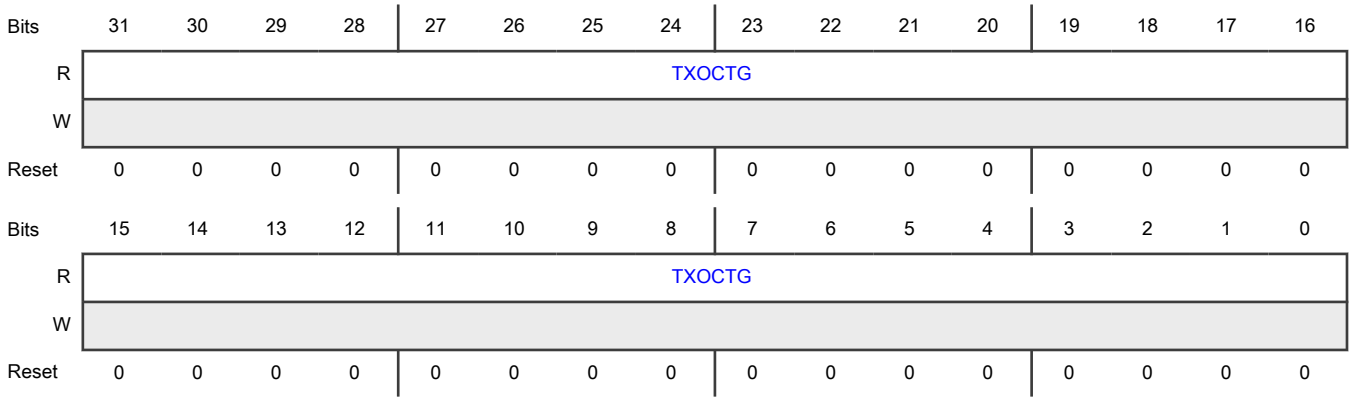
75.17.82 Transmit Octet Count Good (Tx_Octet_Count_Good)

Offset

Register	Offset
Tx_Octet_Count_Good	764h

Function
Provides the number of bytes that the module transmitted, exclusive of preamble, only in good packets.

Diagram



Fields

Field	Function
31-0 TXOCTG	Transmit Octet Count Good Indicates the number of bytes transmitted, exclusive of preamble, only in good packets.

75.17.83 Transmit Packet Count Good (Tx_Packet_Count_Good)

Offset

Register	Offset
Tx_Packet_Count_Good	768h

Function

Provides the number of good packets that the module transmitted.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TXPKTG															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TXPKTG															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TXPKTG	Transmit Packet Count Good Indicates the number of good packets transmitted.

75.17.84 Transmit Excessive Deferral Error (Tx_Excessive_Deferral_Error)

Offset

Register	Offset
Tx_Excessive_Deferral_Error	76Ch

Function

Provides the number of packets that the module aborted because of an excessive deferral error (deferred for more than two max-sized packet times).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TXEXSDEF															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TXEXSDEF															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TXEXSDEF	Transmit Excessive Deferral Error Indicates the number of packets aborted because of excessive deferral error (deferred for more than two max-sized packet times).

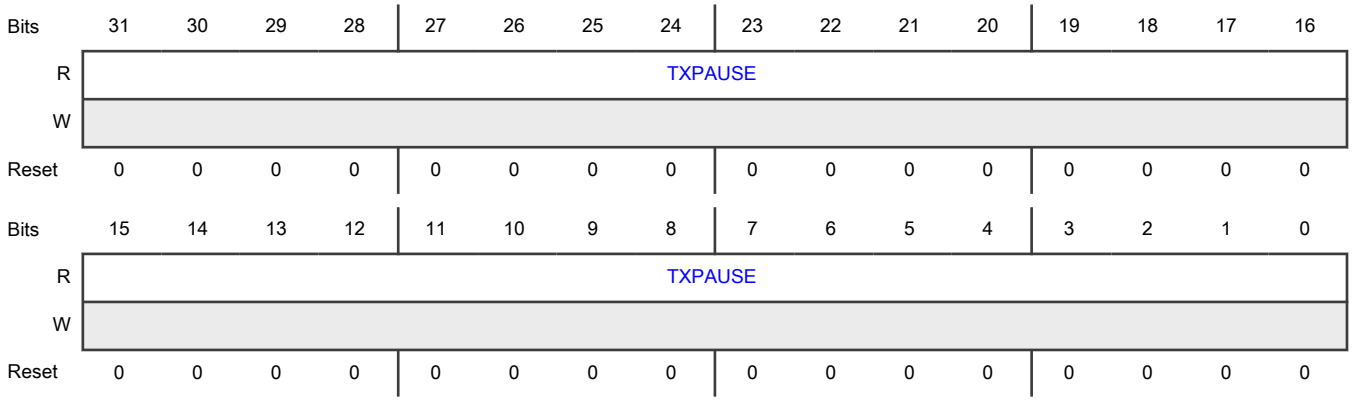
75.17.85 Transmit Pause Packets (Tx_Pause_Packets)

Offset

Register	Offset
Tx_Pause_Packets	770h

Function
Provides the number of good pause packets that the module transmitted.

Diagram



Fields

Field	Function
31-0 TXPAUSE	Transmit Pause Packets Indicates the number of good pause packets transmitted.

75.17.86 Transmit VLAN Packets Good (Tx_VLAN_Packets_Good)

Offset

Register	Offset
Tx_VLAN_Packets_Good	774h

Function

Provides the number of good VLAN packets that the module transmitted.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TXVLANG															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TXVLANG															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TXVLANG	Transmit VLAN Packets Good Provides the number of good VLAN packets transmitted.

75.17.87 Transmit O Size Packets Good (Tx_OSize_Packets_Good)

Offset

Register	Offset
Tx_OSize_Packets_Good	778h

Function

Provides the number of packets that the module transmitted without errors, and having a length greater than the maximum size, which is 1,518 or 1,522 bytes for VLAN-tagged packets. This size is 2000 bytes if MAC_Configuration[S2KP] = 1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TXOSIZG															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TXOSIZG															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TXOSIZG	Transmit O Size Packets Good Indicates the number of packets transmitted without errors, and having a length greater than the maximum size, which is 1,518 or 1,522 bytes for VLAN-tagged packets. This size is 2000 bytes if MAC_Configuration[S2KP] = 1.

75.17.88 Receive Packets Count Good Bad (Rx_Packets_Count_Good_Bad)

Offset

Register	Offset
Rx_Packets_Count_Good_Bad	780h

Function
Provides the number of good and bad packets that the module received.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RXPKTGB															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXPKTGB															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 RXPKTGB	Receive Packets Count Good Bad Indicates the number of good and bad packets received.

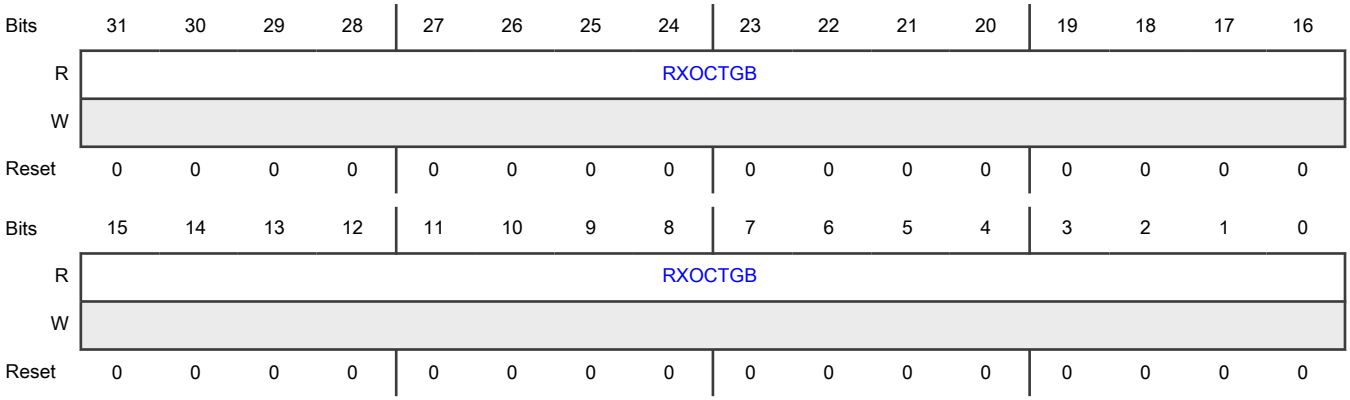
75.17.89 Receive Octet Count Good Bad (Rx_Octet_Count_Good_Bad)

Offset

Register	Offset
Rx_Octet_Count_Good_Bad	784h

Function
Provides the number of bytes that DWC_ther_qos received, exclusive of preamble, in good and bad packets.

Diagram



Fields

Field	Function
31-0 RXOCTGB	Receive Octet Count Good Bad Indicates the number of bytes received, exclusive of preamble, in good and bad packets.

75.17.90 Receive Octet Count Good (Rx_Octet_Count_Good)

Offset

Register	Offset
Rx_Octet_Count_Good	788h

Function
Provides the number of bytes that the module received, exclusive of preamble, only in good packets.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RXOCTG															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXOCTG															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 RXOCTG	Receive Octet Count Good Indicates the number of bytes received, exclusive of preamble, only in good packets.

75.17.91 Receive Broadcast Packets Good (Rx_Broadcast_Packets_Good)

Offset

Register	Offset
Rx_Broadcast_Packets_Good	78Ch

Function

Provides the number of good broadcast packets that the module received.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RXBCASTG															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXBCASTG															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 RXBCASTG	Receive Broadcast Packets Good Indicates the number of good broadcast packets received.

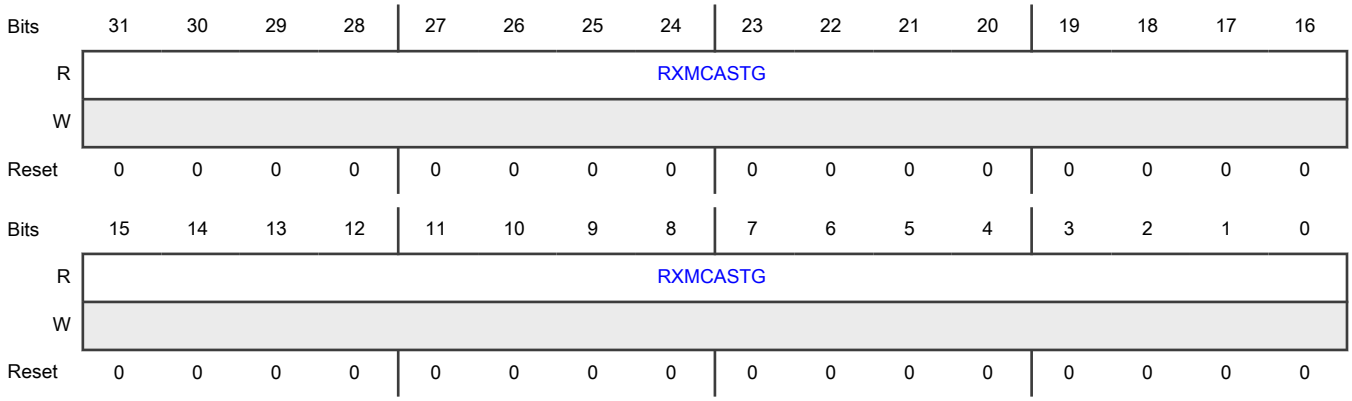
75.17.92 Receive Multicast Packets Good (Rx_Multicast_Packets_Good)

Offset

Register	Offset
Rx_Multicast_Packets_Good	790h

Function
Provides the number of good multicast packets that the module received.

Diagram



Fields

Field	Function
31-0 RXMCASTG	Receive Multicast Packets Good Indicates the number of good multicast packets received.

75.17.93 Receive CRC Error Packets (Rx_CRC_Error_Packets)

Offset

Register	Offset
Rx_CRC_Error_Packets	794h

Function

Provides the number of packets that the module received with a CRC error.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RXCRCERR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXCRCERR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 RXCRCERR	Receive CRC Error Packets Indicates the number of packets received with a CRC error.

75.17.94 Receive Alignment Error Packets (Rx_Alignment_Error_Packets)

Offset

Register	Offset
Rx_Alignment_Error_Packets	798h

Function

Provides the number of packets that the module received with an alignment (dribble) error. It is valid only in 10/100 mode.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RXALGNERR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXALGNERR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 RXALGNERR	Receive Alignment Error Packets Indicates the number of packets received with alignment (dribble) error. It is valid only in 10/100 mode.

75.17.95 Receive Runt Error Packets (Rx_Runt_Error_Packets)

Offset

Register	Offset
Rx_Runt_Error_Packets	79Ch

Function
Provides the number of runt packets, which have a length less than 64 bytes and a CRC error, that the module received.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RXRUNTERR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXRUNTERR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 RXRUNTERR	Receive Runt Error Packets Indicates the number of runt packets received.

75.17.96 Receive Jabber Error Packets (Rx_Jabber_Error_Packets)

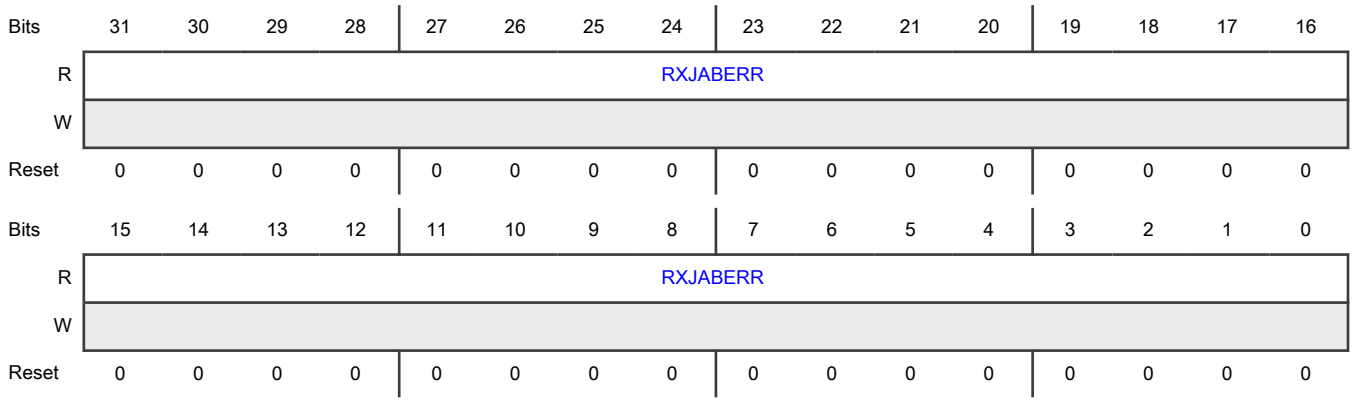
Offset

Register	Offset
Rx_Jabber_Error_Packets	7A0h

Function

Provides the number of giant packets that the module received, having a length (including CRC) greater than 1,518 bytes (1,522 bytes for VLAN-tagged packets), and with a CRC error. If Jumbo Packet mode is enabled, packets of length greater than 9,018 bytes (9,022 bytes for VLAN-tagged packets) are considered as giant packets.

Diagram



Fields

Field	Function
31-0 RXJABERR	Receive Jabber Error Packets Indicates the number of giant packets received, having a length (including CRC) greater than 1,518 bytes (1,522 bytes for VLAN-tagged packets), and with a CRC error. If Jumbo Packet mode is enabled, packets of length greater than 9,018 bytes (9,022 bytes for VLAN-tagged packets) are considered as giant packets.

75.17.97 Receive Undersize Packets Good (Rx_Undersize_Packets_Good)

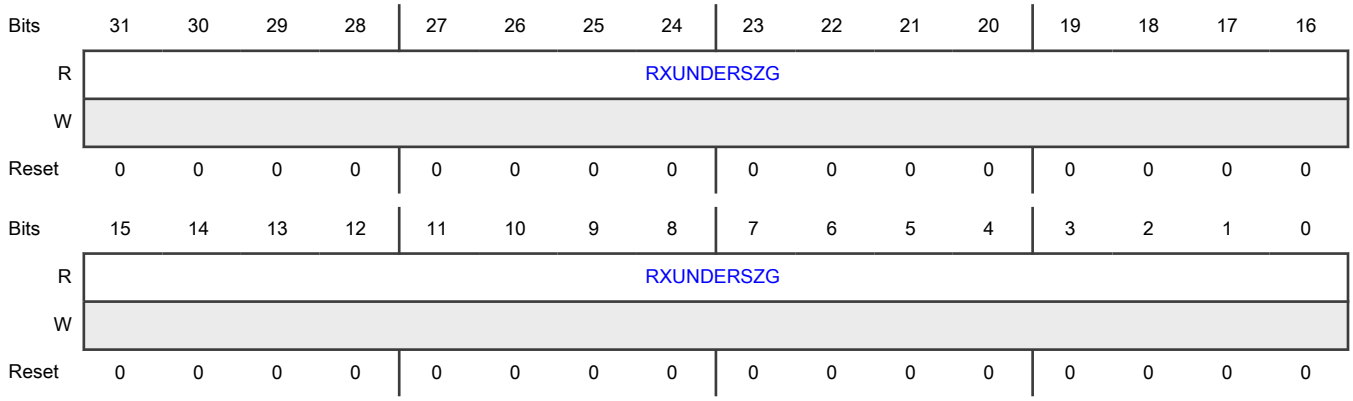
Offset

Register	Offset
Rx_Undersize_Packets_Good	7A4h

Function

Provides the number of packets that the module received, having a length less than 64 bytes, but without any errors.

Diagram



Fields

Field	Function
31-0 RXUNDERSZG	Receive Undersize Packets Good Indicates the number of packets received, having a length less than 64 bytes, but without any errors.

75.17.98 Receive Oversize Packets Good (Rx_Oversize_Packets_Good)

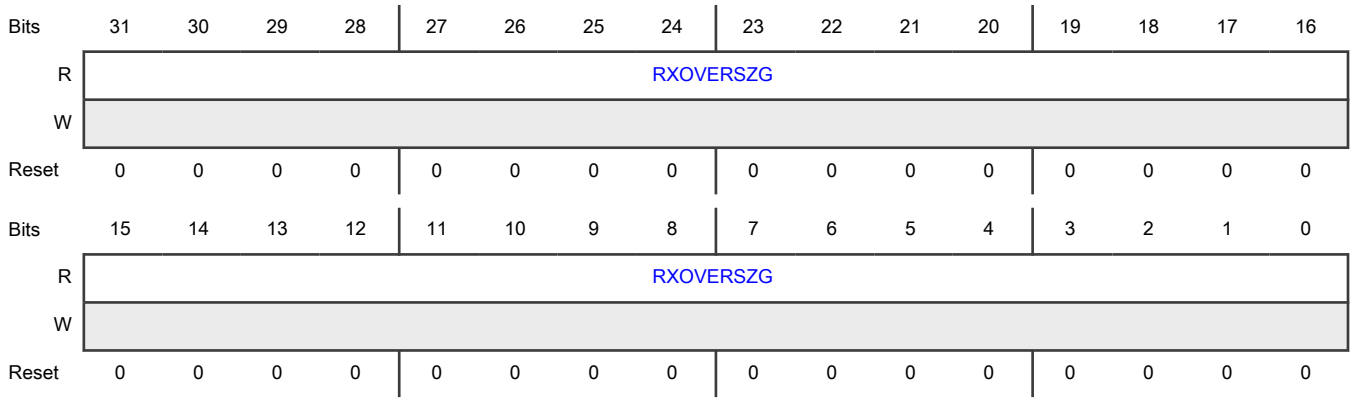
Offset

Register	Offset
Rx_Oversize_Packets_Good	7A8h

Function

Provides the number of packets that the module received without any errors, having a length greater than the maximum size (1,518 bytes or 1,522 bytes for VLAN-tagged packets; 2000 bytes if MAC_Configuration[S2KP] = 1).

Diagram



Fields

Field	Function
31-0 RXOVERSZG	Receive Oversize Packets Good Indicates the number of packets received without errors, with a length greater than the maximum size (1,518 bytes or 1,522 bytes for VLAN-tagged packets; 2000 bytes if MAC_Configuration[S2KP] = 1).

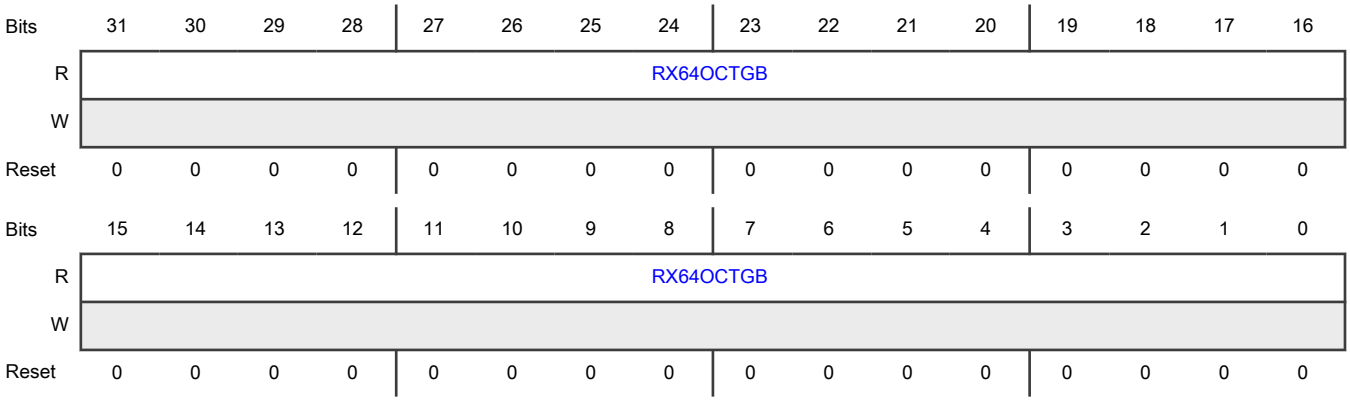
75.17.99 Receive 64 Octets Packets Good Bad (Rx_64Octets_Packets_Good_Bad)

Offset

Register	Offset
Rx_64Octets_Packets_Good_Bad	7ACh

Function
Provides the number of 64-byte good and bad packets that the module received, exclusive of the preamble.

Diagram



Fields

Field	Function
31-0 RX64OCTGB	Receive 64 Octets Packets Good Bad Provides the number of 64-byte good and bad packets that the module received, exclusive of the preamble.

75.17.100 Receive 65-127 Octets Packets Good Bad (Rx_65To127Octets_Packets_Good_Bad)

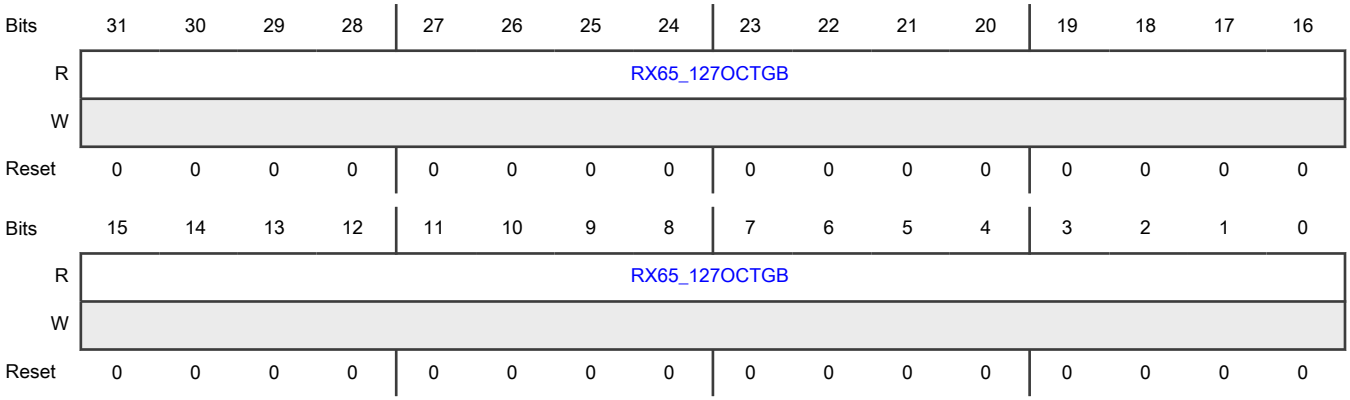
Offset

Register	Offset
Rx_65To127Octets_Packets_Good_Bad	7B0h

Function

Provides the number of good and bad packets that the module received, ranging between 65 and 127 (inclusive) bytes, exclusive of the preamble.

Diagram



Fields

Field	Function
31-0 RX65_127OCTGB	Receive 65-127 Octets Packets Good Bad Indicates the number of good and bad packets received, ranging between 65 and 127 (inclusive) bytes, exclusive of the preamble.

75.17.101 Receive 128-255 Octets Packets Good Bad (Rx_128To255Octets_Packets_Good_Bad)

Offset

Register	Offset
Rx_128To255Octets_Packets_Good_Bad	7B4h

Function

Provides the number of good and bad packets that the module received, with packet length between 128 and 255 (inclusive) bytes, exclusive of the preamble.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RX128_255OCTGB															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RX128_255OCTGB															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 RX128_255OCTGB	Receive 128-255 Octets Packets Good Bad Indicates the number of good and bad packets received, with packet length between 128 and 255 (inclusive) bytes, exclusive of the preamble.

75.17.102 Receive 256-511 Octets Packets Good Bad (Rx_256To511Octets_Packets_Good_Bad)

Offset

Register	Offset
Rx_256To511Octets_Packets_Good_Bad	7B8h

Function

Provides the number of good and bad packets that the module received, having a packet length between 256 and 511 (inclusive) bytes, exclusive of the preamble.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RX256_511OCTGB															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RX256_511OCTGB															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 RX256_511OCTGB	Receive 256-511 Octets Packets Good Bad Indicates the number of good and bad packets received, having a packet length between 256 and 511 (inclusive) bytes, exclusive of the preamble.

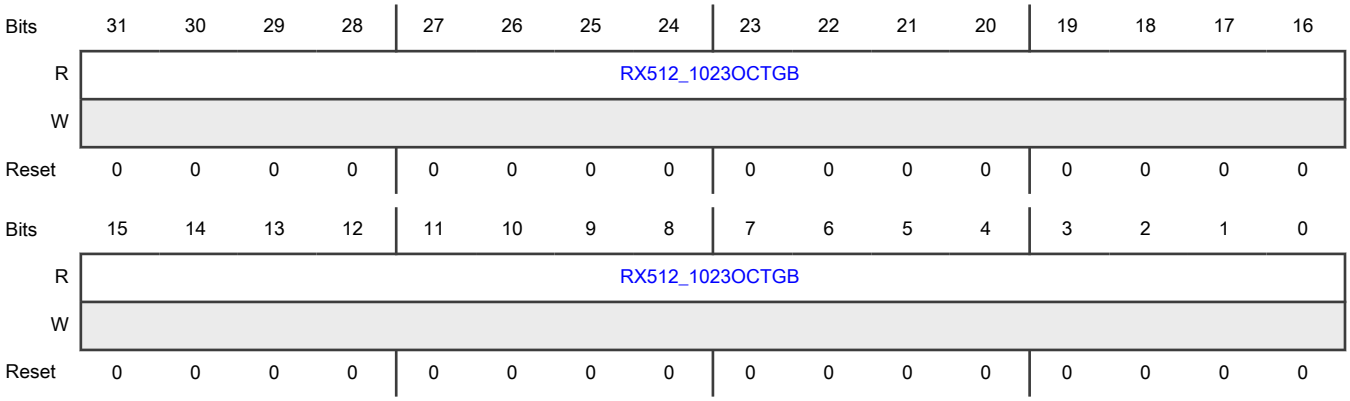
75.17.103 Receive 512-1023 Octets Packets Good Bad (Rx_512To1023Octets_Packets_Good_Bad)

Offset

Register	Offset
Rx_512To1023Octets_Packets_Good_Bad	7BCh

Function
Provides the number of good and bad packets that the module received, having a packet length between 512 and 1023 (inclusive) bytes, exclusive of the preamble.

Diagram



Fields

Field	Function
31-0 RX512_1023OCTGB	Receive 512-1023 Octets Packets Good Bad Indicates the number of good and bad packets received, having a packet length between 512 and 1023 (inclusive) bytes, exclusive of the preamble.

75.17.104 Receive 1024 To Max Octets Good Bad (Rx_1024ToMaxOctets_Packets_Good_Bad)

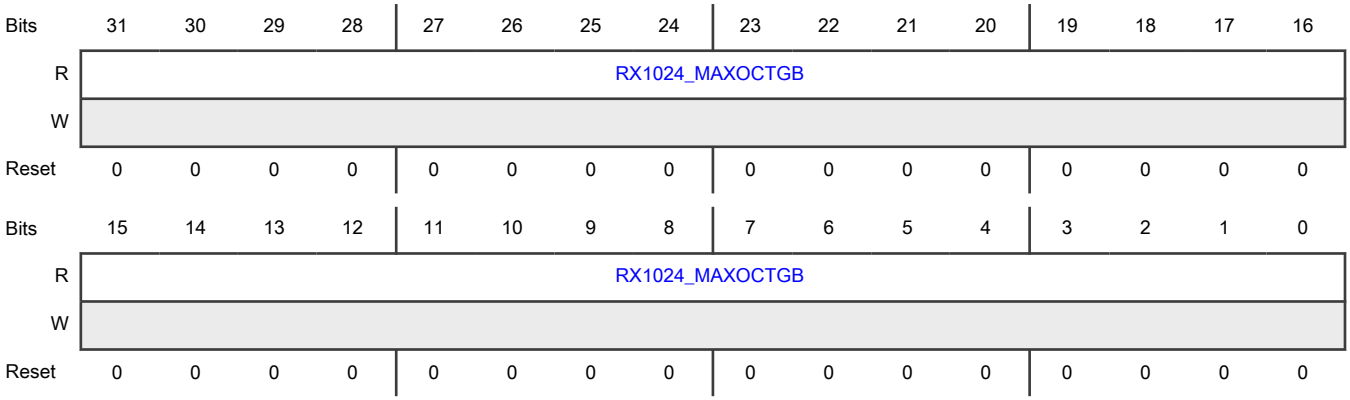
Offset

Register	Offset
Rx_1024ToMaxOctets_Packets_Good_Bad	7C0h

Function

Provides the number of good and bad packets that the module received, having an inclusive packet length between 1024 bytes and the maximum byte size, exclusive of the preamble.

Diagram



Fields

Field	Function
31-0 RX1024_MAXOCTGB	Receive 1024-Max Octets Good Bad Indicates the number of good and bad packets received with length between 1024 and the maximum size (inclusive) bytes, exclusive of the preamble.

75.17.105 Receive Unicast Packets Good (Rx_Unicast_Packets_Good)

Offset

Register	Offset
Rx_Unicast_Packets_Good	7C4h

Function

Provides the number of good unicast packets that the module received.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RXUCASTG															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXUCASTG															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 RXUCASTG	Receive Unicast Packets Good Indicates the number of good unicast packets received.

75.17.106 Receive Length Error Packets (Rx_Length_Error_Packets)

Offset

Register	Offset
Rx_Length_Error_Packets	7C8h

Function

Provides the number of packets that the module received with a length error (the Length/Type (LT) field of the RDES3 normal descriptor not equal to the packet size) for all the packets with a valid length field. For more information on the LT field, see the "RDES3 normal descriptor (write-back format)" section.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RXLENERR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXLENERR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 RXLENERR	Receive Length Error Packets Indicates the number of packets received with a length error (the Length/Type (LT) field of the RDES3 normal descriptor not equal to the packet size) for all the packets with a valid length field. For more information on the LT field, see the "RDES3 normal descriptor (write-back format)" section.

75.17.107 Receive Out of Range Type Packet (Rx_Out_Of_Range_Type_Packets)

Offset

Register	Offset
Rx_Out_Of_Range_Type_Packets	7CCh

Function

Provides the number of packets that the module received, with the length field not equal to the valid packet size (greater than 1,500 but less than 1,536).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RXOUTOFRNG															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXOUTOFRNG															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 RXOUTOFRNG	Receive Out of Range Type Packet Indicates the number of packets received with length field not equal to the valid packet size (greater than 1,500 but less than 1,536).

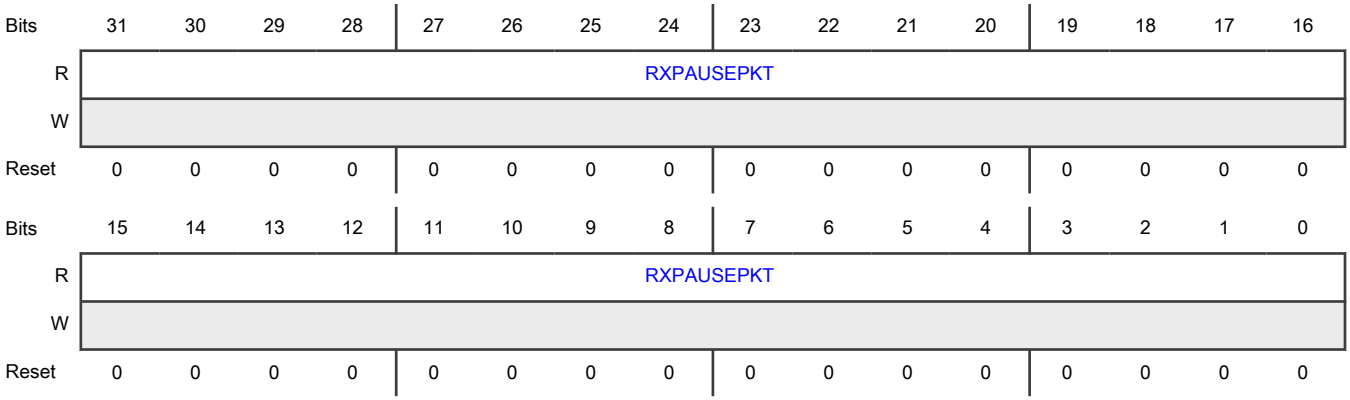
75.17.108 Receive Pause Packets (Rx_Pause_Packets)

Offset

Register	Offset
Rx_Pause_Packets	7D0h

Function
Provides the number of good and valid pause packets that the module received.

Diagram



Fields

Field	Function
31-0 RXPAUSEPKT	Receive Pause Packets Indicates the number of good and valid pause packets received.

75.17.109 Receive FIFO Overflow Packets (Rx_FIFO_Overflow_Packets)

Offset

Register	Offset
Rx_FIFO_Overflow_Packets	7D4h

Function
Provides the number of missed received packets because of FIFO overflow in the module.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RXFIFOOVFL															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXFIFOOVFL															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 RXFIFOOVFL	Receive FIFO Overflow Packets Indicates the number of missed received packets because of FIFO overflow.

75.17.110 Receive VLAN Packets Good Bad (Rx_VLAN_Packets_Good_Bad)

Offset

Register	Offset
Rx_VLAN_Packets_Good_Bad	7D8h

Function

Provides the number of good and bad VLAN packets that the module received.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RXVLANPKTGB															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXVLANPKTGB															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 RXVLANPKTGB	Receive VLAN Packets Good Bad Indicates the number of good and bad VLAN packets received.

75.17.111 Receive Watchdog Error Packets (Rx_Watchdog_Error_Packets)

Offset

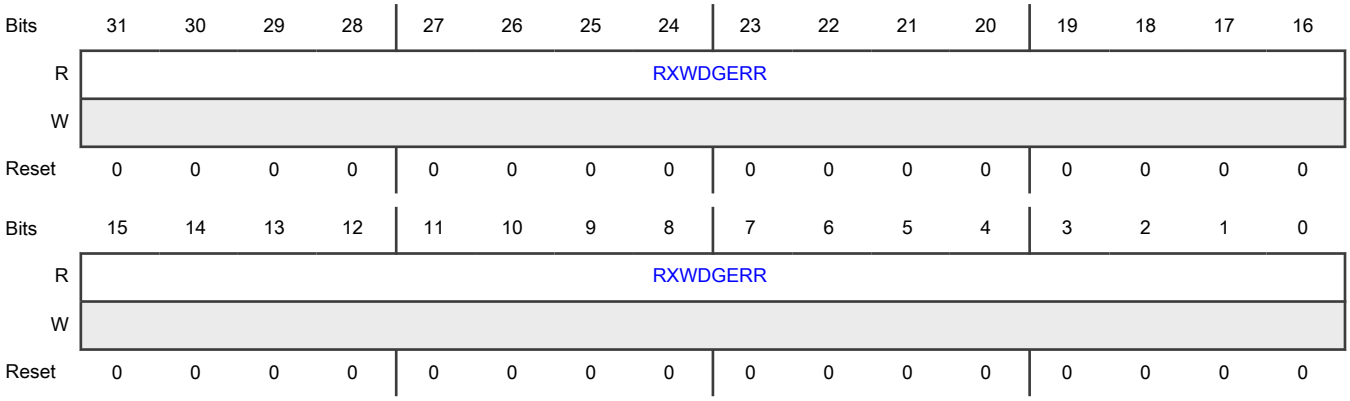
Register	Offset
Rx_Watchdog_Error_Packets	7DCh

Function

Provides the number of packets that the module received with a watchdog timeout error. Packets with a data load larger than the following are received:

- 2,048 bytes when MAC_Configuration[JE] = 0 and MAC_Configuration[WD] = 0
- 10,240 bytes when MAC_Configuration[JE] = 1 and MAC_Configuration[WD] = 0
- 16,384 bytes when MAC_Configuration[WD] = 1 or the value programmed in MAC Watchdog Timeout (MAC_Watchdog_Timeout)

Diagram



Fields

Field	Function
31-0 RXWDGERR	Receive Watchdog Error Packets Indicates the number of packets received with a watchdog timeout error. Packets with a data load larger than the following are received:

Table continues on the next page...

Field	Function
	• 2,048 bytes when MAC_Configuration[JE] = 0 and MAC_Configuration[WD] = 0 • 10,240 bytes when MAC_Configuration[JE] = 1 and MAC_Configuration[WD] = 0 • 16,384 bytes when MAC_Configuration[WD] = 1 or the value programmed in MAC Watchdog Timeout (MAC_Watchdog_Timeout)

75.17.112 Receive Error Packets (Rx_Receive_Error_Packets)

Offset

Register	Offset
Rx_Receive_Error_Packets	7E0h

Function

Provides the number of packets that the module received with a receive or packet extension error on GMII or MII.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RXRCVERR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXRCVERR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 RXRCVERR	Receive Error Packets Indicates the number of packets received with a receive or packet extension error on GMII or MII.

75.17.113 Receive Control Packets Good (Rx_Control_Packets_Good)

Offset

Register	Offset
Rx_Control_Packets_Good	7E4h

Function

Provides the number of good control packets that the module received.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RXCTRLG															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXCTRLG															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 RXCTRLG	Receive Control Packets Good Indicates the number of good control packets received.

75.17.114 MMC Transmit FPE Fragment Counter Interrupt Status (MMC_FPE_Tx_Interrupt)

Offset

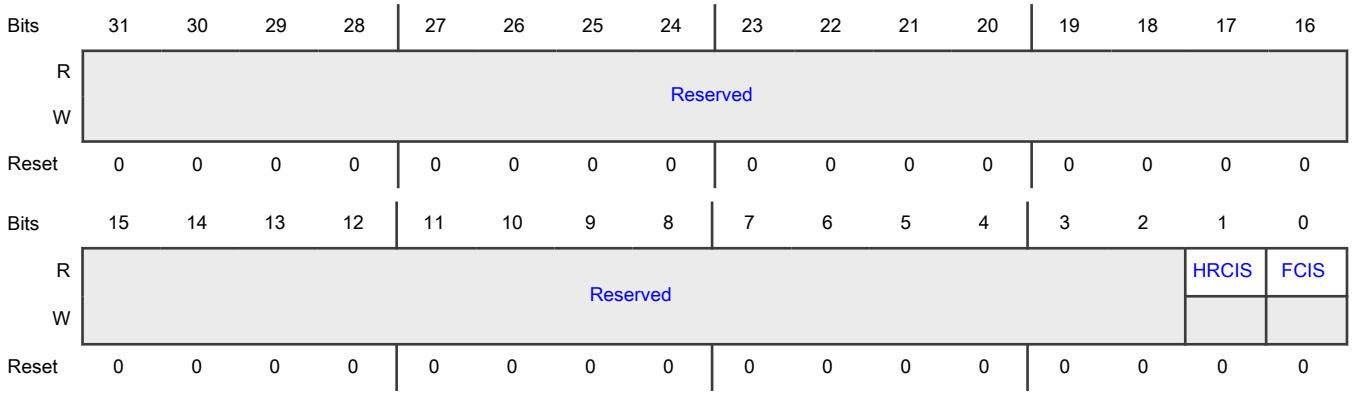
Register	Offset
MMC_FPE_Tx_Interrupt	8A0h

Function

Maintains the interrupts generated from all the FPE-related transmit statistic counters when they reach half of their maximum values (8000_0000h for a 32-bit counter and 8000h for a 16-bit counter), and when they cross their maximum values (FFFF_FFFFh for a 32-bit counter and FFFFh for a 16-bit counter). When MMC_Control[CNTSTOPRO] = 1, the interrupts are set but the counter remains at all-ones.

An interrupt bit of this register becomes 0 when the respective MMC counter that caused the interrupt is read. The least significant byte lane (bits [7:0]) of the respective counter must be read to clear the interrupt bit.

Diagram



Fields

Field	Function
31-2 —	Reserved
1 HRCIS	MMC Transmit Hold Request Counter Interrupt Status Indicates whether the MMC Transmit hold request counter interrupt status is detected. This field: • Becomes 1 when the Tx_Hold_Req_Cntr counter reaches half of the maximum value or the maximum value. • Exists when any one of the receive or transmit MMC counters is enabled during FPE with AV_EST-enabled configuration. Access restrictions apply to this field that clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
0 FCIS	MMC Transmit FPE Fragment Counter Interrupt Status Indicates whether the MMC Transmit FPE fragment counter interrupt status is detected. This field: • Becomes 1 when the Tx_FPE_Fragment_Cntr counter reaches half of the maximum value or the maximum value. • Exists when any one of the receive or transmit MMC counters is enabled during FPE-enabled configuration. Access restrictions apply to this field that clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected

75.17.115 MMC FPE Transmit Interrupt Mask (MMC_FPE_Tx_Interrupt_Mask)

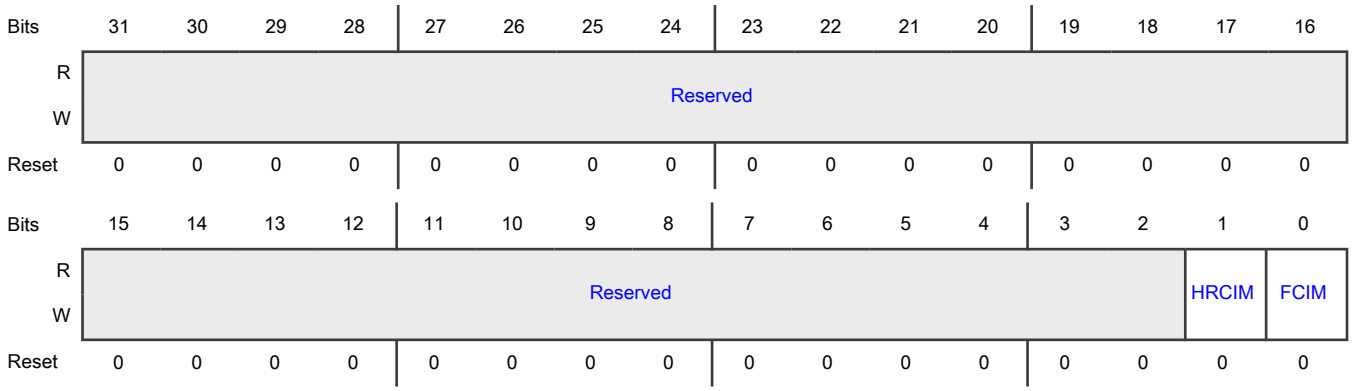
Offset

Register	Offset
MMC_FPE_Tx_Interrupt_Mask	8A4h

Function

Maintains the masks for interrupts generated from all FPE-related transmit statistics counters. MMC Receive Interrupt Mask (MMC_Rx_Interrupt_Mask) maintains the masks for the interrupts generated when FPE-related receive statistic counters reach half of their maximum value or the maximum value.

Diagram



Fields

Field	Function
31-2 —	Reserved
1 HRCIM	MMC Transmit Hold Request Counter Interrupt Mask Indicates the status of the MMC transmit hold request counter interrupt mask. Writing 1 to this field masks the interrupt when the Tx_Hold_Req_Cntr counter reaches half of its maximum value or the maximum value. The field exists when any one of the receive, transmit, or MMC counters is enabled during FPE with AV_EST-enabled configuration. 0b - Disabled 1b - Enabled
0 FCIM	MMC Transmit Fragment Counter Interrupt Mask Indicates the status of the MMC transmit fragment counter interrupt mask.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Writing 1 to this field masks the interrupt when the Tx_FPE_Fragment_Cntr counter reaches half of its maximum value or the maximum value. The field exists when any one of the receive, transmit, or MMC counters is enabled during FPE-enabled configuration. 0b - Disabled 1b - Enabled

75.17.116 Transmit FPE Fragment Counter (MMC_Tx_FPE_Fragment_Cntr)

Offset

Register	Offset
MMC_Tx_FPE_Fragment_Cntr	8A8h

Function

Provides the number of additional mPackets transmitted because of preemption.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TXFFC															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TXFFC															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TXFFC	Transmit FPE Fragment Counter Indicates the number of additional mPackets transmitted because of preemption. Exists when any one of the receive, transmit, or MMC counters is enabled during FPE-enabled configuration.

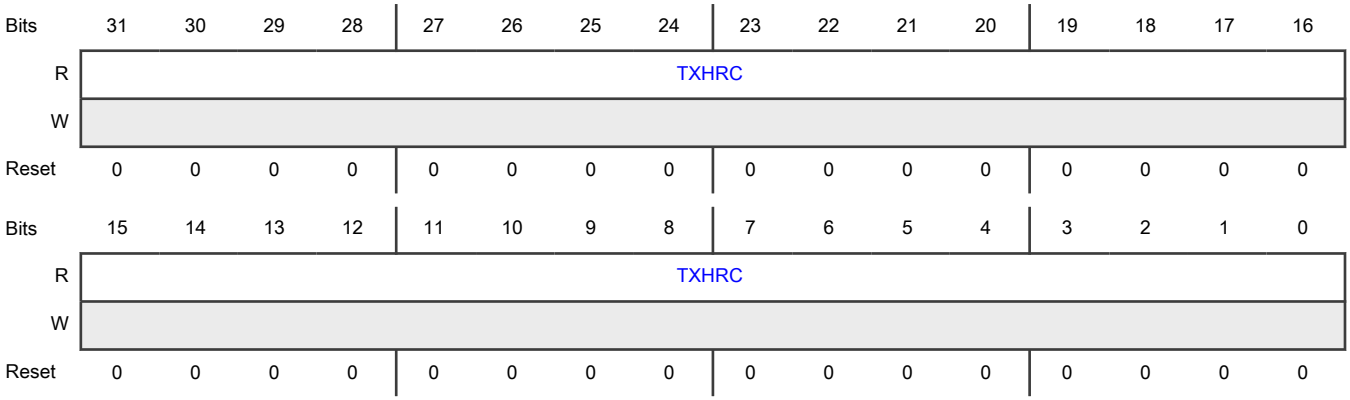
75.17.117 Transmit Hold Request Counter (MMC_Tx_Hold_Req_Cntr)

Offset

Register	Offset
MMC_Tx_Hold_Req_Cntr	8ACh

Function
Provides the count of times that a hold request is given to MAC.

Diagram



Fields

Field	Function
31-0 TXHRC	Transmit Hold Request Counter Indicates the count of times that a hold request is given to MAC. Exists when any one of the receive, transmit, or MMC counters is enabled during FPE with AV_EST-enabled configuration.

75.17.118 MMC Receive Packet Assembly Error Counter Interrupt Status (MMC_FPE_Rx_Interrupt)

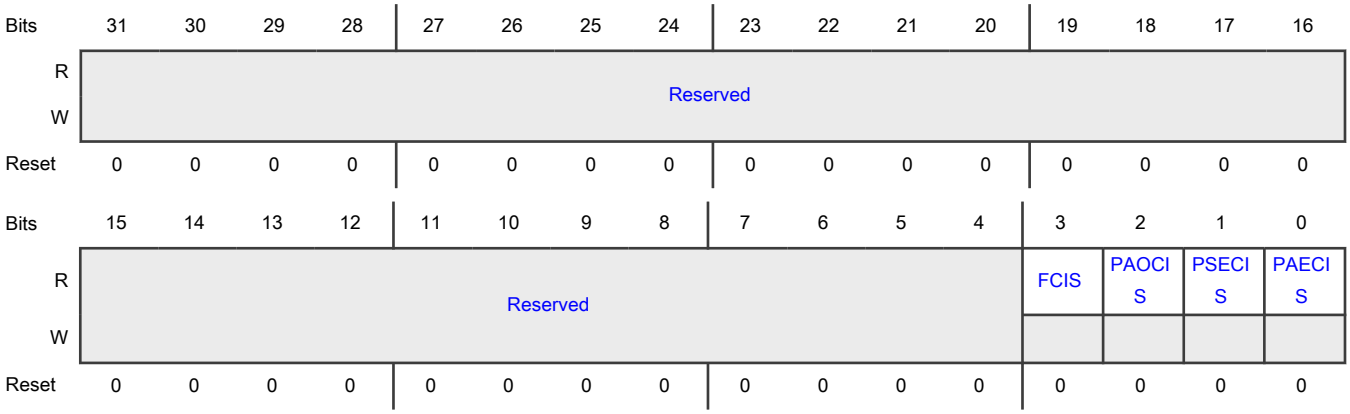
Offset

Register	Offset
MMC_FPE_Rx_Interrupt	8C0h

Function
Maintains the interrupts generated from all the FPE-related receive statistic counters, when these transmit statistic counters reach half of their maximum values (8000_0000h for a 32-bit counter and 8000h for a 16-bit counter), and when they cross their maximum values (FFFF_FFFFh for a 32-bit counter and FFFFh for 16-bit counter). When MMC_Control[CNTSTOPRO] = 1, the interrupts are set but the counter remains at all-ones. An interrupt bit of this register becomes 0 when the respective

MMC counter that caused the interrupt is read. The least significant byte lane (bits [7:0]) of the respective counter must be read to clear the interrupt bit.

Diagram



Fields

Field	Function
31-4 —	Reserved
3 FCIS	MMC Receive FPE Fragment Counter Interrupt Status Indicates whether the MMC receive FPE fragment counter interrupt status is detected. This field becomes 1 when the Rx_FPE_Fragment_Cntr counter reaches half of its maximum value or the maximum value. Exists when any one of the receive, transmit, or MMC counters is enabled during FPE-enabled configuration. Access restrictions apply to this field that clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
2 PAOCIS	MMC Receive Packet Assembly OK Counter Interrupt Status Indicates whether the MMC receive packet assembly OK counter interrupt status is detected. This field becomes 1 when the Rx_Packet_Assemble_Ok_Cntr counter reaches half of its maximum value or the maximum value. Exists when any one of the receive, transmit, or MMC counters is enabled during FPE-enabled configuration. Access restrictions apply to this field that clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 PSECIS	MMC Receive, Transmit, Packet SMD Error Counter Interrupt Status Indicates whether the status of the MMC receive packet SMD error counter interrupt is detected. This field becomes 1 when the Rx_Packet_SMD_Err_Cntr counter reaches half of its maximum value or the maximum value. Exists when any one of the receive, transmit, or MMC counters is enabled during FPE-enabled configuration. Access restrictions apply to this field that clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
0 PAECIS	MMC Receive, transmit, Packet Assembly Error Counter Interrupt Status Indicates whether the status of the MMC receive packet assembly error counter interrupt is detected. This field becomes 1 when the Rx_Packet_Assemble_Err_Cntr counter reaches half of its maximum value or the maximum value. Exists when any one of the receive, transmit, or MMC counters is enabled during the FPE-enabled configuration. Access restrictions apply to this field that clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected

75.17.119 MMC FPE Receive Interrupt Mask (MMC_FPE_Rx_Interrupt_Mask)

Offset

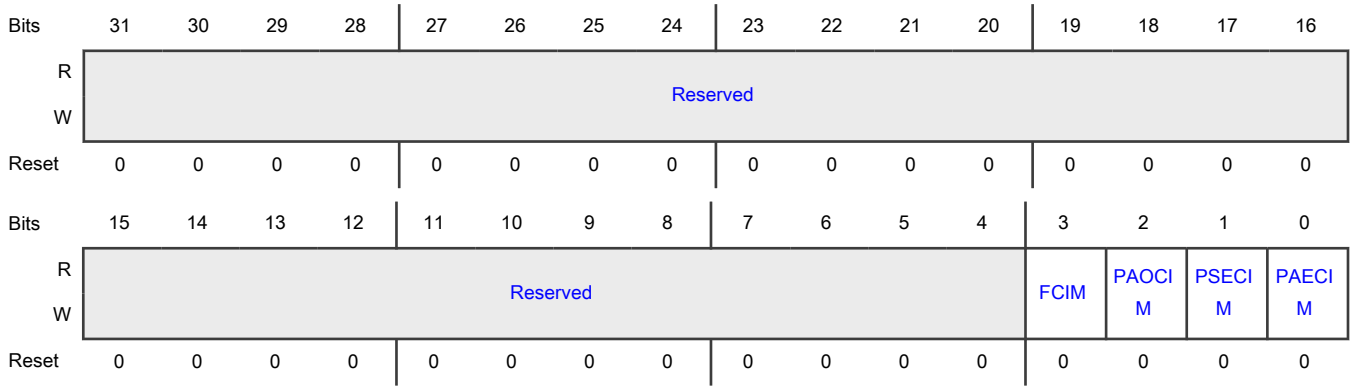
Register	Offset
MMC_FPE_Rx_Interrupt_Mask	8C4h

Function

Maintains the masks for interrupts generated from all FPE-related receive statistic counters.

This register maintains the masks for the interrupts generated when FPE-related receive statistic counters reach half of their maximum value or the maximum value.

Diagram



Fields

Field	Function
31-4 —	Reserved
3 FCIM	MMC Receive FPE Fragment Counter Interrupt Mask Indicates the status of the MMC receive FPE fragment counter interrupt mask. Writing 1 to this field masks the interrupt when the Tx_FPE_Fragment_Cntr counter reaches half of its maximum value or the maximum value. Exists when any one of the receive, transmit, or MMC counters is enabled during FPE-enabled configuration. 0b - Disabled 1b - Enabled
2 PAOCIM	MMC Receive Packet Assembly OK Counter Interrupt Mask Indicates the status of the MMC receive packet assembly OK counter interrupt mask. Writing 1 to this field masks the interrupt when the Rx_Packet_Assemble_Ok_Cntr counter reaches half of its maximum value or the maximum value. Exists when any one of the receive, transmit, or MMC counters is enabled during FPE-enabled configuration. 0b - Disabled 1b - Enabled
1 PSECIM	MMC Receive Packet SMD Error Counter Interrupt Mask Indicates the status of the MMC receive packet SMD error counter interrupt mask. Writing 1 to this field masks the interrupt when the Rx_Packet_SMD_Err_Cntr counter reaches half of its maximum value or the maximum value. Exists when any one of the receive, transmit, or MMC counters is enabled during FPE-enabled configuration.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disabled 1b - Enabled
0 PAECIM	MMC receive Packet Assembly Error Counter Interrupt Mask Indicates the status of the MMC receive packet assembly error counter interrupt mask. Writing 1 to this field masks the interrupt when the Rx_Packet_Assemble_Err_Cntr counter reaches half of its maximum value or the maximum value. Exists when any one of the receive, transmit, or MMC counters is enabled during FPE-enabled configuration. 0b - Disabled 1b - Enabled

75.17.120 MMC Receive Packet Assembly Error Counter (MMC_Rx_Packet_Assembly_Err_Cntr)

Offset

Register	Offset
MMC_Rx_Packet_Assembly_Err_Cntr	8C8h

Function

Provides the number of MAC frames having reassembly errors on the receiver arising out of mismatch in the fragment count value.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PAEC															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PAEC															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PAEC	Packet Assembly Error Counter Indicates the number of MAC frames having reassembly errors on the receiver arising out of mismatch in the fragment count value. Exists when any one of the receive, transmit, or MMC counters is enabled during FPE-enabled configuration.

75.17.121 MMC Receive Packet SMD Error Counter (MMC_Rx_Packet_SMD_Err_Cntr)

Offset

Register	Offset
MMC_Rx_Packet_SMD_Err_Cntr	8CCh

Function

Provides the number of received MAC frames rejected because of an unknown SMD value and MAC frame fragments rejected because of arriving with SMD-C when there was no preceding preempted frame.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PSEC															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PSEC															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PSEC	Packet SMD Error Counter Indicates the number of MAC frames rejected because of an unknown SMD value and MAC frame fragments rejected because of arriving with SMD-C when there was no preceding preempted frame. Exists when at least one of the receive, transmit, or MMC counters is enabled during FPE-enabled configuration.

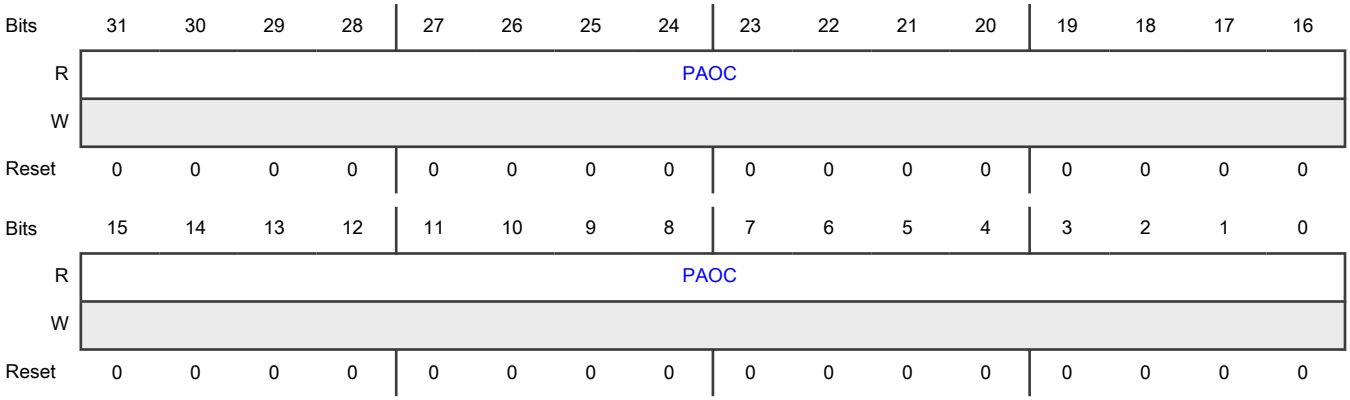
75.17.122 MMC Receive Packet Assembly OK Counter (MMC_Rx_Packet_Assembly_OK_Cntr)

Offset

Register	Offset
MMC_Rx_Packet_Assembly_OK_Cntr	8D0h

Function
Provides the number of MAC frames that were successfully reassembled and delivered to MAC.

Diagram



Fields

Field	Function
31-0 PAOC	Packet Assembly OK Counter Indicates the number of MAC frames that were successfully reassembled and delivered to MAC. Exists when at least one of the receive, transmit, or MMC counters is enabled during FPE-enabled configuration.

75.17.123 MMC Receive FPE Fragment Counter (MMC_Rx_FPE_Fragment_Cntr)

Offset

Register	Offset
MMC_Rx_FPE_Fragment_Cntr	8D4h

Function
Provides the number of additional mPackets received because of preemption.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	FFC															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FFC															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 FFC	FPE Fragment Counter Indicates the number of additional mPackets received because of preemption. Exists when at least one of the receive, transmit, or MMC counters is enabled during FPE-enabled configuration.

75.17.124 MAC Layer 3 Layer 4 Control 0 (MAC_L3_L4_Control0)

Offset

Register	Offset
MAC_L3_L4_Control0	900h

Function
Controls the filter 0 operations of the layer 3 and layer 4 protocols.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved			DMCHEN0	Reserved			DMCHN0	Reserved		L4DPIM0	L4DPM0	L4SPIM0	L4SPM0	Reserved	L4PEN0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	L3HDBM0				L3HSBM0						L3DAIM0	L3DAM0	L3SAIM0	L3SAM0	Reserved	L3PEN0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	Reserved
28 DMCHEN0	DMA Channel Select Enable Indicates the status of the DMA channel number. • If this field is 1, it enables the selection of the DMA channel number for the packet that the L3_L4 filter passes. The DMCHN fields indicate the DMA channel number. • If this field is 0, the filter does not decide the DMA channel number. 0b - Disabled 1b - Enabled
27-25 —	Reserved
24 DMCHN0	DMA Channel Number Indicates the DMA channel number. If the value of the DMCHEN fields is 1, this field selects the DMA channel number to which the packet that this filter passes is routed. The width of this field depends on the number of the DMA channels present in your configuration.
23-22 —	Reserved
21 L4DPIM0	Layer 4 Destination Port Inverse Match Enable Indicates the status of layer 4 destination port inverse matching. • If this field is 1, MAC_Layer4_Address0[L4DP0] is enabled for inverse matching. • If this field is 0, MAC_Layer4_Address0[L4DP0] is enabled for perfect matching. This field is valid and applicable only when the L4DPM0 field is 1. 0b - Disabled 1b - Enabled
20 L4DPM0	Layer 4 Destination Port Match Enable Indicates the status of layer 4 destination port matching. • If this field is 1, MAC_Layer4_Address0[L4DP0] is enabled for matching. • If this field is 0, MAC ignores MAC_Layer4_Address0[L4DP0] for matching. 0b - Disabled 1b - Enabled
19	Layer 4 Source Port Inverse Match Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
L4SPIM0	Indicates the status of layer 4 source port inverse matching. • If this field is 1, MAC_Layer4_Address0[L4SP0] is enabled for inverse matching. • If this field is 0, MAC_Layer4_Address0[L4SP0] is enabled for perfect matching. This field is valid and applicable only when MAC_Layer4_Address0[L4SP0] = 1. 0b - Disabled 1b - Enabled
18 L4SPM0	Layer 4 Source Port Match Enable Indicates the status of layer 4 source port matching. • If this field is 1, MAC_Layer4_Address0[L4SP0] is enabled for matching. • If this field is 0, MAC ignores MAC_Layer4_Address0[L4SP0] for matching. 0b - Disabled 1b - Enabled
17 —	Reserved
16 L4PEN0	Layer 4 Protocol Enable Indicates the status of layer 4 protocol. • If this field is 1, MAC_Layer4_Address0[L4SP0] and MAC_Layer4_Address0[L4DP0] are used for matching UDP packets. • If this field is 0, MAC_Layer4_Address0[L4SP0] and MAC_Layer4_Address0[L4DP0] are used for matching TCP packets. Layer 4 matching is performed only when the L4SPM0 field or the L4DPM0 field is 1. 0b - Disabled 1b - Enabled
15-11 L3HDBM0	Layer 3 IP DA Higher Bits Match IPv4 packets: This field contains the number of higher bits of IP DA that are matched in the IPv4 packets. The following list describes the values of this field: • 0: No bits are masked. • 1: LSb[0] is masked. • 2: Two LSbs [1:0] are masked. • - .. • 31: All bits except MSb are masked. IPv6 packets:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Bits [12:11] of this field correspond to bits [6:5] of the L3HSBM0 field, which indicates the number of lower bits of IP SA or DA that are masked in the IPv6 packets. The following list describes the concatenated values of the L3HDBM0[1:0] and L3HSBM0 fields: • 0: No bits are masked. • 1: LSb[0] is masked. • 2: Two LSbs [1:0] are masked. • .. • 127: All bits except MSb are masked. This field is valid and applicable only when the L3DAM0 field or the L3SAM0 field is 1.
10-6 L3HSBM0	Layer 3 IP SA Higher Bits Match IPv4 packets: This field contains the number of lower bits of IP SA that are masked for matching in the IPv4 packets. The following list describes the values of this field: • 0: No bits are masked. • 1: LSb[0] is masked. • 2: Two LSbs [1:0] are masked. • .. • 31: All bits except MSb are masked. IPv6 packets: This field contains bits [4:0] of L3HSBM0. These bits indicate the number of higher bits of IP SA or DA matched in the IPv6 packets. The field is valid and applicable only when the L3DAM0 field or the L3SAM0 field is 1.
5 L3DAIM0	Layer 3 IP DA Inverse Match Enable Indicates the status of layer 3 IP DA inverse matching. • If this field is 1, layer 3 IP DA is enabled for inverse matching. • If this field is 0, layer 3 IP DA is enabled for perfect matching. This field is valid and applicable only if L3DAM0 = 1. 0b - Disabled 1b - Enabled
4 L3DAM0	Layer 3 IP DA Match Enable Indicates the status of layer 3 IP DA matching. • If this field is 1, layer 3 IP DA is enabled for matching. • If this field is 0, MAC ignores layer 3 IP DA for matching.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE If L3PEN0 = 1, you must write 1 to either this field or the L3SAM0 field because either the IPv6 DA or SA can be checked for filtering. 0b - Disabled 1b - Enabled
3 L3SAIM0	Layer 3 IP SA Inverse Match Enable Indicates the status of layer 3 IP SA inverse matching. • If this field is 1, layer 3 IP SA is enabled for inverse matching. • If this field is 0, layer 3 IP SA is enabled for perfect matching. This field is valid and applicable only if the L3SAM0 field is 1. 0b - Disabled 1b - Enabled
2 L3SAM0	Layer 3 IP SA Match Enable Indicates the status of layer 3 IP SA matching. • If this field is 1, layer 3 IP SA matching is enabled. • If this field is 0, MAC ignores layer 3 IP SA matching. **NOTE** If L3PEN0 = 1, you must write 1 to either this field or the L3DAM0 field because either IPv6 SA or DA can be checked for filtering. 0b - Disabled 1b - Enabled
1 —	Reserved
0 L3PEN0	Layer 3 Protocol Enable Indicates the status of layer 3 protocol. • If this field is 1, the layer 3 IP SA or DA matching is enabled for IPv6 packets. • If this field is 0, the layer 3 IP SA or DA matching is enabled for IPv4 packets. The layer 3 matching is performed only when either the L3SAM0 field or the L3DAM0 field is 1. 0b - Disabled 1b - Enabled

75.17.125 MAC Layer 4 Address 0 (MAC_Layer4_Address0)

Offset

Register	Offset
MAC_Layer4_Address0	904h

Function

Provides the layer 4 source and destination port numbers.

MAC Layer 4 Address 1 (MAC_Layer4_Address1), MAC L3 L4 Control 1 (MAC_L3_L4_Control1), MAC Layer 3 Address 0 Reg 1 (MAC_Layer3_Addr0_Reg1), MAC Layer 3 Address 1 Reg 1 (MAC_Layer3_Addr1_Reg1), MAC Layer 3 Address 2 Reg 1 (MAC_Layer3_Addr2_Reg1), and MAC Layer 3 Address 3 Reg 1 (MAC_Layer3_Addr3_Reg1) are reserved registers (RO with a default value) if MAC_Packet_Filter[IPFE] is 0 when configuring the core.

You can configure the layer 3 and layer 4 address registers to be double-synchronized by selecting the synchronize layer 3 and layer 4 address registers to the receive clock domain option while configuring the core. When you select this option, the synchronization is triggered only when bits [31:24] (in Little-Endian mode) or bits [7:0] (in Big-Endian mode) of the layer 3 and layer 4 address registers are written to. For proper synchronization updates, you must perform consecutive writes to the same layer 3 and layer 4 address registers after a delay of at least four destination clock cycles.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	L4DP0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	L4SP0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 L4DP0	Layer 4 Destination Port Number Indicates layer 4 destination port number. If MAC_L3_L4_Control0[L4PEN0] = 0 and MAC_L3_L4_Control0[L4DPM0] = 1, this field contains the value to be matched with the TCP destination port number field in the IPv4 or IPv6 packets. If MAC_L3_L4_Control0[L4PEN0] and MAC_L3_L4_Control0[L4DPM0] are 1, this field contains the value to be matched with the UDP destination port number field in the IPv4 or IPv6 packets.
15-0 L4SP0	Layer 4 Source Port Number Indicates layer 4 source port number.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	If MAC_L3_L4_Control0[L4PEN0] = 0 and MAC_L3_L4_Control0[L4SPM0] = 1, this field contains the value to be matched with the TCP source port number field in the IPv4 or IPv6 packets. If MAC_L3_L4_Control0[L4PEN0] and MAC_L3_L4_Control0[L4SPM0] are 1, this field contains the value to be matched with the UDP source port number field in the IPv4 or IPv6 packets.

75.17.126 MAC Layer 3 Address 0 Reg 0 (MAC_Layer3_Addr0_Reg0)

Offset

Register	Offset
MAC_Layer3_Addr0_Reg0	910h

Function

Contains the 32-bit IP source address field for IPv4 packets. For IPv6 packets, the field contains bits [31:0] of the 128-bit IP source address or destination address field.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	L3A00															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	L3A00															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 L3A00	Layer 3 Address 0 Indicates layer 3 address 0. • If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3SAM0] are 1, this field contains the value to be matched with bits [31:0] of the IP source address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3DAM0] are 1, this field contains the value to be matched with bits [31:0] of the IP destination address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] = 0, and MAC_L3_L4_Control0[L3SAM0] = 1, this field contains the value to be matched with the IP source address field in the IPv4 packets.

75.17.127 MAC Layer 3 Address 1 Reg 0 (MAC_Layer3_Addr1_Reg0)

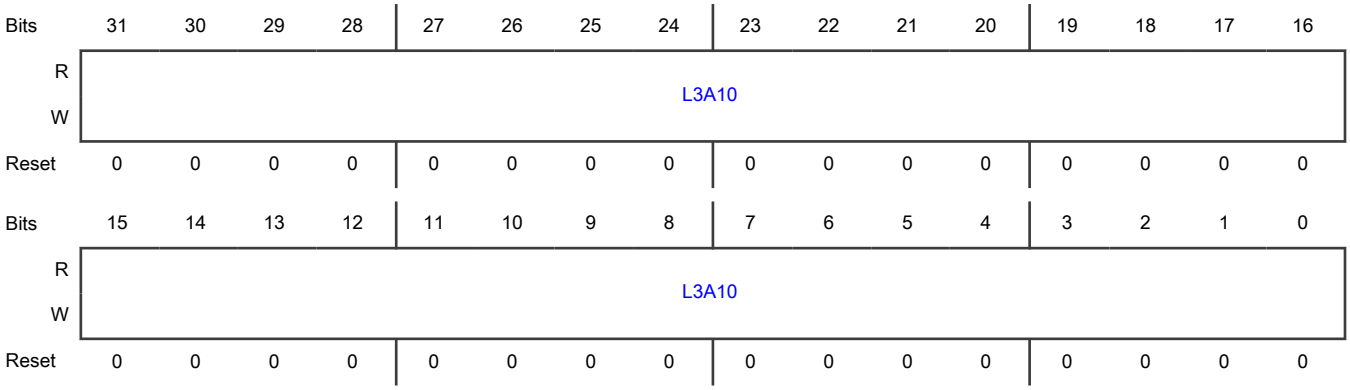
Offset

Register	Offset
MAC_Layer3_Addr1_Reg0	914h

Function

Contains the 32-bit IP destination address field for IPv4 packets. For IPv6 packets, the field contains bits [63:32] of the 128-bit IP source address or destination address field.

Diagram



Fields

Field	Function
31-0 L3A10	Layer 3 Address 1 Indicates layer 3 address 1. • If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3SAM0] are 1, this field contains the value to be matched with bits [63:32] of the IP source address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3DAM0] are 1, this field contains the value to be matched with bits [63:32] of the IP destination address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] = 0, and MAC_L3_L4_Control0[L3SAM0] = 1, this field contains the value to be matched with the IP source address field in the IPv4 packets.

75.17.128 MAC Layer 3 Address 2 Reg 0 (MAC_Layer3_Addr2_Reg0)

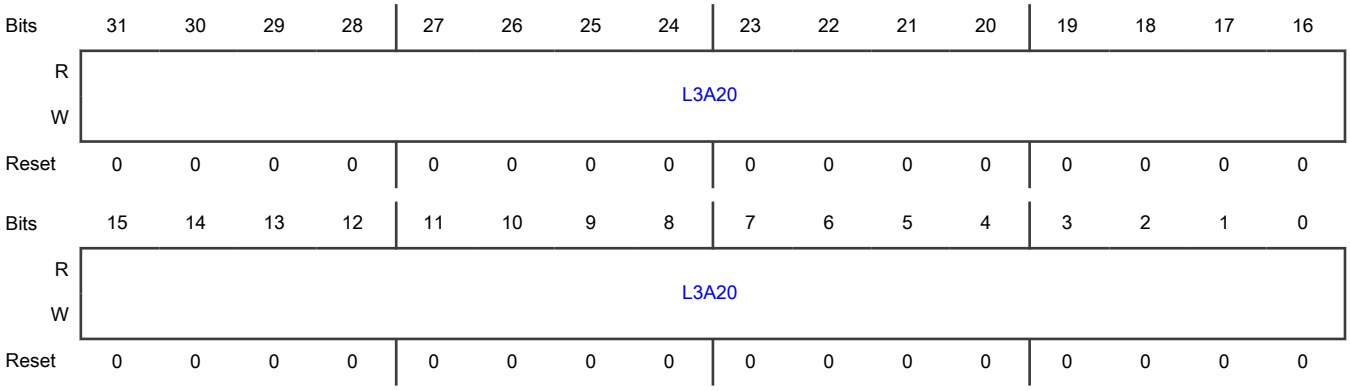
Offset

Register	Offset
MAC_Layer3_Addr2_Reg0	918h

Function

Contains the 32-bit IP destination address field for IPv4 packets. For IPv6 packets, the field contains bits [95:64] of the 128-bit IP source address or destination address field.

Diagram



Fields

Field	Function
31-0 L3A20	Layer 3 Address 2 Indicates layer 3 address 2. • If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3SAM0] are 1, this field contains the value to be matched with bits [95:64] of the IP source address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3DAM0] are 1, this field contains the value to be matched with bits [95:64] of the IP destination address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] = 0, this field is not used.

75.17.129 MAC Layer 3 Address 3 Reg 0 (MAC_Layer3_Addr3_Reg0)

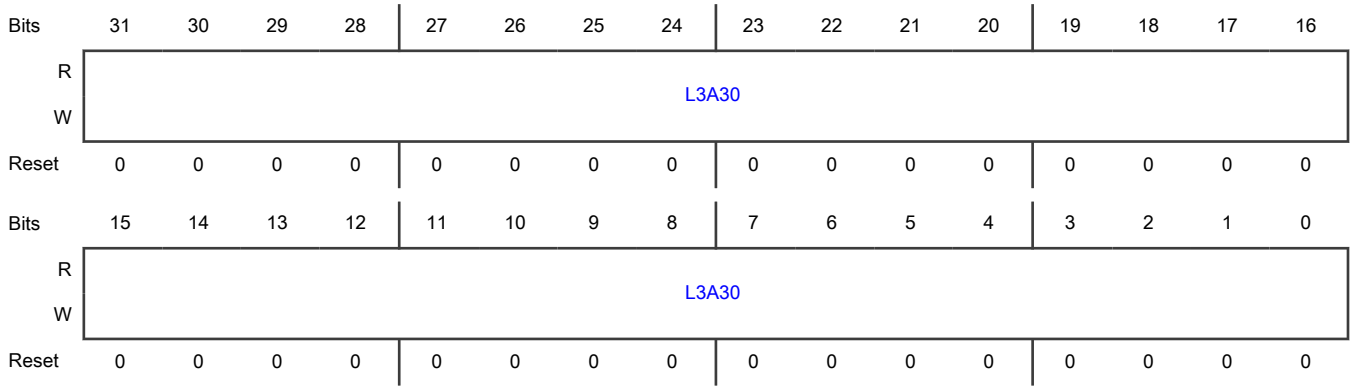
Offset

Register	Offset
MAC_Layer3_Addr3_Reg0	91Ch

Function

Contains the 32-bit IP destination address field for IPv4 packets. For IPv6 packets, the field contains bits [127:96] of the 128-bit IP source address or destination address field.

Diagram



Fields

Field	Function
31-0 L3A30	Layer 3 Address 3 Indicates layer 3 address 3. • If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3SAM0] are 1, this field contains the value to be matched with bits [127:96] of the IP source address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3DAM0] are 1, this field contains the value to be matched with bits [127:96] of the IP destination address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] = 0, this field is not used.

75.17.130 MAC L3 L4 Control 1 (MAC_L3_L4_Control1)

Offset

Register	Offset
MAC_L3_L4_Control1	930h

Function

Controls the filter 0 operations of the layer 3 and layer 4 protocols.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved			DMCHEN1	Reserved			DMCHN1	Reserved		L4DPIM1	L4DPM1	L4SPIM1	L4SPM1	Reserved	L4PEN1
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	L3HDBM1				L3HSBM1						L3DAIM1	L3DAM1	L3SAIM1	L3SAM1	Reserved	L3PEN1
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	Reserved
28 DMCHEN1	DMA Channel Select Enable 1 Indicates the status of the DMA channel number. • If this field is 1, it enables the selection of the DMA channel number for the packet that the L3_L4 filter passes. The DMCHN fields indicate the DMA channel number. • If this field is 0, the filter does not decide the DMA channel number. 0b - Disabled 1b - Enabled
27-25 —	Reserved
24 DMCHN1	DMA Channel Number 1 Indicates the DMA channel number. If the value of the DMCHEN fields is 1, this field selects the DMA channel number to which the packet that this filter passes is routed. The width of this field depends on the number of the DMA channels present in your configuration.
23-22 —	Reserved
21 L4DPIM1	Layer 4 Destination Port Inverse Match Enable 1 Indicates the status of layer 4 destination port inverse matching. • If this field is 1, MAC_Layer4_Address0[L4DP0] is enabled for inverse matching. • If this field is 0, MAC_Layer4_Address0[L4DP0] is enabled for perfect matching. This field is valid and applicable only when the L4DPM0 field is 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disabled 1b - Enabled
20 L4DPM1	Layer 4 Destination Port Match Enable 1 Indicates the status of layer 4 destination port matching. • If this field is 1, MAC_Layer4_Address0[L4DP0] is enabled for matching. • If this field is 0, MAC ignores MAC_Layer4_Address0[L4DP0] for matching. 0b - Disabled 1b - Enabled
19 L4SPIM1	Layer 4 Source Port Inverse Match Enable 1 Indicates the status of layer 4 source port inverse matching. • If this field is 1, MAC_Layer4_Address0[L4SP0] is enabled for inverse matching. • If this field is 0, MAC_Layer4_Address0[L4SP0] is enabled for perfect matching. This field is valid and applicable only when MAC_Layer4_Address0[L4SP0] = 1. 0b - Disabled 1b - Enabled
18 L4SPM1	Layer 4 Source Port Match Enable 1 Indicates the status of layer 4 source port matching. • If this field is 1, MAC_Layer4_Address0[L4SP0] is enabled for matching. • If this field is 0, MAC ignores MAC_Layer4_Address0[L4SP0] for matching. 0b - Disabled 1b - Enabled
17 —	Reserved
16 L4PEN1	Layer 4 Protocol Enable 1 Indicates the status of layer 4 protocol. • If this field is 1, MAC_Layer4_Address0[L4SP0] and MAC_Layer4_Address0[L4DP0] of UDP packets are used for matching. • If this field is 0, MAC_Layer4_Address0[L4SP0] and MAC_Layer4_Address0[L4DP0] of TCP packets are used for matching. Layer 4 matching is performed only when the L4SPM0 field or the L4DPM0 field is 1. 0b - Disabled 1b - Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-11 L3HDBM1	Layer 3 IP DA Higher Bits Match 1 IPv4 Packets: This field contains the number of higher bits of IP DA that are matched in the IPv4 packets. The following list describes the values of this field: • 0: No bits are masked. • 1: LSb[0] is masked. • 2: Two LSbs [1:0] are masked. • .. • 31: All bits except MSb are masked. IPv6 Packets: Bits [12:11] of this field correspond to bits [6:5] of MAC_L3_L4_Control0[L3HSBM0], which indicates the number of lower bits of IP SA or DA that are masked in the IPv6 packets. The following list describes the concatenated values of MAC_L3_L4_Control0[L3HDBM0][1:0] and MAC_L3_L4_Control0[L3HSBM0]: • 0: No bits are masked. • 1: LSb[0] is masked. • 2: Two LSbs [1:0] are masked • .. • 127: All bits except MSb are masked. This field is valid and applicable only when MAC_L3_L4_Control0[L3DAM0] or MAC_L3_L4_Control0[L3SAM0] is 1.
10-6 L3HSBM1	Layer 3 IP SA Higher Bits Match 1 IPv4 packets: This field contains the number of lower bits of IP SA that are masked for matching in the IPv4 packets. The following list describes the values of this field: • 0: No bits are masked. • 1: LSb[0] is masked. • 2: Two LSbs [1:0] are masked. • .. • 31: All bits except MSb are masked. IPv6 packets: This field contains bits [4:0] of MAC_L3_L4_Control0[L3HSBM0]. These bits indicate the number of higher bits of IP SA or DA matched in the IPv6 packets. The field is valid and applicable only when MAC_L3_L4_Control0[L3DAM0] or MAC_L3_L4_Control0[L3SAM0] is 1.
5 L3DAIM1	Layer 3 IP DA Inverse Match Enable 1 Indicates the status of layer 3 IP DA inverse matching.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• If this field is 1, layer 3 IP DA is enabled for inverse matching. • If this field is 0, layer 3 IP DA is enabled for perfect matching. This field is valid and applicable only if MAC_L3_L4_Control0[L3DAM0] = 1. 0b - Disabled 1b - Enabled
4 L3DAM1	Layer 3 IP DA Match Enable 1 Indicates the status of layer 3 IP DA matching. • If this field is 1, layer 3 IP DA is enabled for matching. • If this field is 0, MAC ignores layer 3 IP DA for matching. **NOTE** If MAC_L3_L4_Control0[L3PEN0] = 1, you must write 1 to either this field or to MAC_L3_L4_Control0[L3SAM0] because either the IPv6 DA or SA can be checked for filtering. 0b - Disabled 1b - Enabled
3 L3SAIM1	Layer 3 IP SA Inverse Match Enable 1 Indicates the status of layer 3 IP SA inverse matching. • If this field is 1, layer 3 IP SA is enabled for inverse matching. • If this field is 0, layer 3 IP SA is enabled for perfect matching. This field is valid and applicable only if MAC_L3_L4_Control0[L3SAM0] is 1. 0b - Disabled 1b - Enabled
2 L3SAM1	Layer 3 IP SA Match Enable 1 Indicates the status of layer 3 IP SA matching. • If this field is 1, layer 3 IP SA matching is enabled. • If this field is 0, MAC ignores layer 3 IP SA matching. **NOTE** If MAC_L3_L4_Control0[L3PEN0] = 1, you must write 1 to either this field or to MAC_L3_L4_Control0[L3DAM0] because either IPv6 SA or DA can be checked for filtering. 0b - Disabled 1b - Enabled
1 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 L3PEN1	Layer 3 Protocol Enable 1 Indicates the status of layer 3 protocol. • If this field is 1, the layer 3 IP SA or DA matching is enabled for IPv6 packets. • If this field is 0, the layer 3 IP SA or DA matching is enabled for IPv4 packets. The layer 3 matching is performed only when either MAC_L3_L4_Control0[L3SAM0] or MAC_L3_L4_Control0[L3DAM0] is 1. 0b - Disabled 1b - Enabled

75.17.131 MAC Layer 4 Address 1 (MAC_Layer4_Address1)

Offset

Register	Offset
MAC_Layer4_Address1	934h

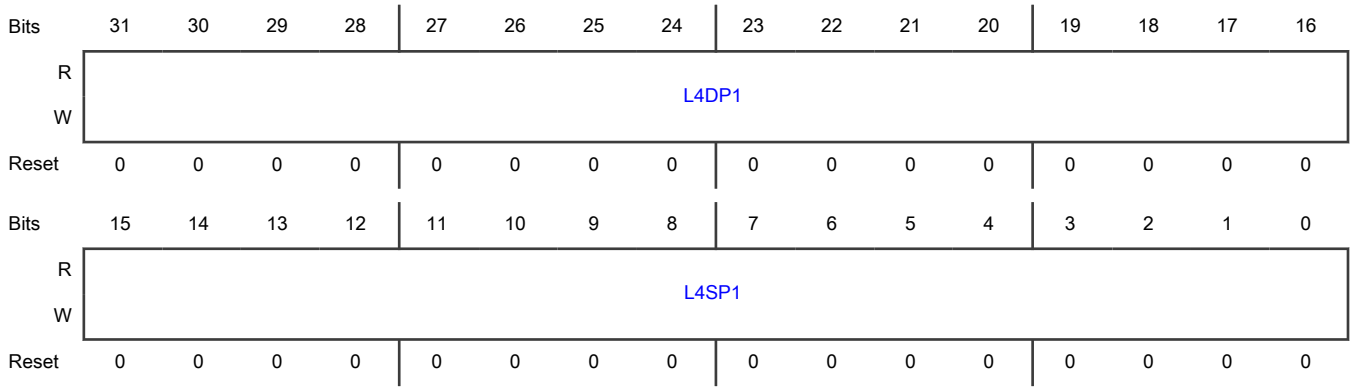
Function

Provides the layer 4 source and destination port numbers.

MAC Layer 4 Address 1 (MAC_Layer4_Address1), MAC L3 L4 Control 1 (MAC_L3_L4_Control1), MAC Layer 3 Address 0 Reg 1 (MAC_Layer3_Addr0_Reg1), MAC Layer 3 Address 1 Reg 1 (MAC_Layer3_Addr1_Reg1), MAC Layer 3 Address 2 Reg 1 (MAC_Layer3_Addr2_Reg1), and MAC Layer 3 Address 3 Reg 1 (MAC_Layer3_Addr3_Reg1) are reserved registers (RO with a default value) if the enable layer 3 and layer 4 packet filter option is not selected when configuring the core.

You can configure the layer 3 and layer 4 address registers to be double-synchronized by selecting the synchronize layer 3 and layer 4 address registers to the receive clock domain option while configuring the core. When you select this option, the synchronization is triggered only when bits [31:24] (in Little-Endian mode) or bits [7:0] (in Big-Endian mode) of the layer 3 and layer 4 address registers are written to. For proper synchronization updates, you must perform consecutive writes to the same layer 3 and layer 4 address registers after a delay of at least four destination clock cycles.

Diagram



Fields

Field	Function
31-16 L4DP1	Layer 4 Destination Port Number 1 Indicates layer 4 destination port number. If MAC_L3_L4_Control0[L4PEN0] = 0 and MAC_L3_L4_Control0[L4DPM0] = 1, this field contains the value to be matched with the TCP destination port number field in the IPv4 or IPv6 packets. If MAC_L3_L4_Control0[L4PEN0] and MAC_L3_L4_Control0[L4DPM0] are 1, this field contains the value to be matched with the UDP destination port number field in the IPv4 or IPv6 packets.
15-0 L4SP1	Layer 4 Source Port Number 1 Indicates layer 4 source port number. If MAC_L3_L4_Control0[L4PEN0] = 0 and MAC_L3_L4_Control0[L4SPM0] = 1, this field contains the value to be matched with the TCP source port number field in the IPv4 or IPv6 packets. If MAC_L3_L4_Control0[L4PEN0] and MAC_L3_L4_Control0[L4SPM0] are 1, this field contains the value to be matched with the UDP source port number field in the IPv4 or IPv6 packets.

75.17.132 MAC Layer 3 Address 0 Reg 1 (MAC_Layer3_Addr0_Reg1)

Offset

Register	Offset
MAC_Layer3_Addr0_Reg1	940h

Function

Contains the 32-bit IP source address field for IPv4 packets. For IPv6 packets, the field contains bits [31:0] of the 128-bit IP source address or destination address field.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	L3A01															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	L3A01															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 L3A01	Layer 3 Address 0 Indicates layer 3 address 0. • If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3SAM0] are 1, this field contains the value to be matched with bits [31:0] of the IP source address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3DAM0] are 1, this field contains the value to be matched with bits [31:0] of the IP destination address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] = 0, and MAC_L3_L4_Control0[L3SAM0] = 1, this field contains the value to be matched with the IP source address field in the IPv4 packets.

75.17.133 MAC Layer 3 Address 1 Reg 1 (MAC_Layer3_Addr1_Reg1)

Offset

Register	Offset
MAC_Layer3_Addr1_Reg1	944h

Function

Contains the 32-bit IP destination address field for IPv4 packets. For IPv6 packets, the field contains bits[63:32] of the 128-bit IP source address or destination address field.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	L3A11															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	L3A11															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 L3A11	Layer 3 Address 1 Indicates layer 3 address 1.

Table continues on the next page...

Field	Function
	• If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3SAM0] are 1, this field contains the value to be matched with bits [63:32] of the IP source address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3DAM0] are 1, this field contains the value to be matched with bits [63:32] of the IP destination address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] = 0, and MAC_L3_L4_Control0[L3SAM0] = 1, this field contains the value to be matched with the IP source address field in the IPv4 packets.

75.17.134 MAC Layer 3 Address 2 Reg 1 (MAC_Layer3_Addr2_Reg1)

Offset

Register	Offset
MAC_Layer3_Addr2_Reg1	948h

Function

Contains the 32-bit IP destination address field for IPv4 packets. For IPv6 packets, the field contains bits [95:64] of the 128-bit IP source address or destination address field.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	L3A21															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	L3A21															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 L3A21	Layer 3 Address 2 Indicates layer 3 address 2. • If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3SAM0] are 1, this field contains the value to be matched with bits [95:64] of the IP source address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3DAM0] are 1, this field contains the value to be matched with bits [95:64] of the IP destination address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] = 0, this field is not used.

75.17.135 MAC Layer 3 Address 3 Reg 1 (MAC_Layer3_Addr3_Reg1)

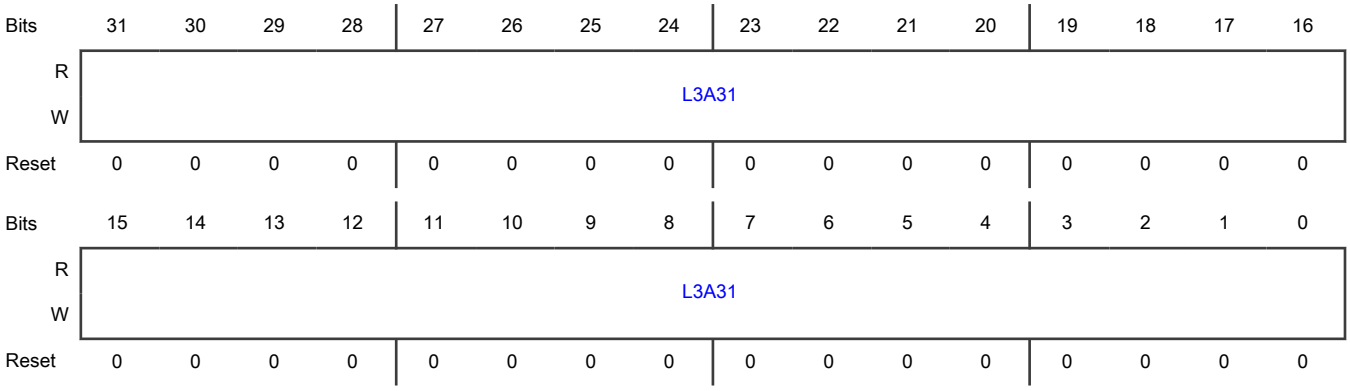
Offset

Register	Offset
MAC_Layer3_Addr3_Reg1	94Ch

Function

Contains the 32-bit IP destination address field for IPv4 packets. For IPv6 packets, the field contains bits [127:96] of the 128-bit IP source address or destination address field.

Diagram



Fields

Field	Function
31-0 L3A31	Layer 3 Address 3 Indicates layer 3 address 3. • If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3SAM0] are 1, this field contains the value to be matched with bits [127:96] of the IP source address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3DAM0] are 1, this field contains the value to be matched with bits [127:96] of the IP destination address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] = 0, this field is not used.

75.17.136 MAC L3 L4 Control 2 (MAC_L3_L4_Control2)

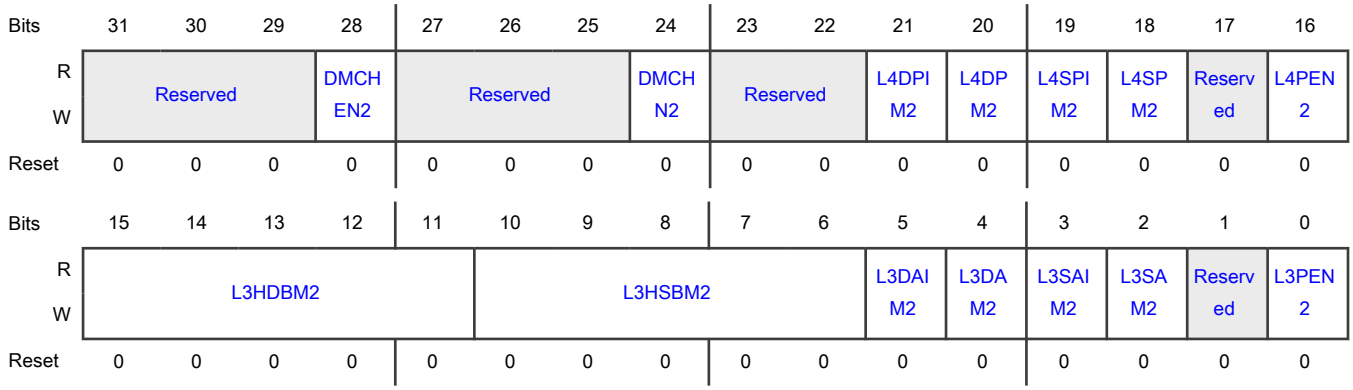
Offset

Register	Offset
MAC_L3_L4_Control2	960h

Function

Controls the filter 0 operations of the layer 3 and layer 4 protocols.

Diagram



Fields

Field	Function
31-29 —	Reserved
28 DMCHEN2	DMA Channel Select Enable 2 Indicates the status of the DMA channel number. • If this field is 1, it enables the selection of the DMA channel number for the packet that the L3_L4 filter passes. The DMCHN fields indicate the DMA channel number. • If this field is 0, the filter does not decide the DMA channel number. 0b - Disabled 1b - Enabled
27-25 —	Reserved
24 DMCHN2	DMA Channel Number 2 Indicates the DMA channel number. If the value of the DMCHEN fields is 1, this field selects the DMA channel number to which the packet that this filter passes is routed. The width of this field depends on the number of the DMA channels present in your configuration.
23-22 —	Reserved
21 L4DPIM2	Layer 4 Destination Port Inverse Match Enable 2 Indicates the status of layer 4 destination port inverse matching. • If this field is 1, MAC_Layer4_Address0[L4DP0] is enabled for inverse matching. • If this field is 0, MAC_Layer4_Address0[L4DP0] is enabled for perfect matching. This field is valid and applicable only when the L4DPM0 field is 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disabled 1b - Enabled
20 L4DPM2	Layer 4 Destination Port Match Enable 2 Indicates the status of layer 4 destination port matching. • If this field is 1, MAC_Layer4_Address0[L4DP0] is enabled for matching. • If this field is 0, MAC ignores MAC_Layer4_Address0[L4DP0] for matching. 0b - Disabled 1b - Enabled
19 L4SPIM2	Layer 4 Source Port Inverse Match Enable 2 Indicates the status of layer 4 source port inverse matching. • If this field is 1, MAC_Layer4_Address0[L4SP0] is enabled for inverse matching. • If this field is 0, MAC_Layer4_Address0[L4SP0] is enabled for perfect matching. This field is valid and applicable only when MAC_Layer4_Address0[L4SP0] = 1. 0b - Disabled 1b - Enabled
18 L4SPM2	Layer 4 Source Port Match Enable 2 Indicates the status of layer 4 source port matching. • If this field is 1, MAC_Layer4_Address0[L4SP0] is enabled for matching. • If this field is 0, MAC ignores MAC_Layer4_Address0[L4SP0] for matching. 0b - Disabled 1b - Enabled
17 —	Reserved
16 L4PEN2	Layer 4 Protocol Enable 2 Indicates the status of layer 4 protocol. • If this field is 1, MAC_Layer4_Address0[L4SP0] and MAC_Layer4_Address0[L4DP0] of UDP packets are used for matching. • If this field is 0, MAC_Layer4_Address0[L4SP0] and MAC_Layer4_Address0[L4DP0] of TCP packets are used for matching. Layer 4 matching is performed only when the L4SPM0 field or the L4DPM0 field is 1. 0b - Disabled 1b - Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-11 L3HDBM2	Layer 3 IP DA Higher Bits Match 2 IPv4 Packets: This field contains the number of higher bits of IP DA that are matched in the IPv4 packets. The following list describes the values of this field: • 0: No bits are masked. • 1: LSb[0] is masked. • 2: Two LSbs [1:0] are masked. • .. • 31: All bits except MSb are masked. IPv6 Packets: Bits [12:11] of this field correspond to bits [6:5] of MAC_L3_L4_Control0[L3HSBM0], which indicates the number of lower bits of IP SA or DA that are masked in the IPv6 packets. The following list describes the concatenated values of MAC_L3_L4_Control0[L3HDBM0][1:0] and MAC_L3_L4_Control0[L3HSBM0]: • 0: No bits are masked. • 1: LSb[0] is masked. • 2: Two LSbs [1:0] are masked • .. • 127: All bits except MSb are masked. This field is valid and applicable only when MAC_L3_L4_Control0[L3DAM0] or MAC_L3_L4_Control0[L3SAM0] is 1.
10-6 L3HSBM2	Layer 3 IP SA Higher Bits Match 2 IPv4 packets: This field contains the number of lower bits of IP SA that are masked for matching in the IPv4 packets. The following list describes the values of this field: • 0: No bits are masked. • 1: LSb[0] is masked. • 2: Two LSbs [1:0] are masked. • .. • 31: All bits except MSb are masked. IPv6 packets: This field contains bits [4:0] of MAC_L3_L4_Control0[L3HSBM0]. These bits indicate the number of higher bits of IP SA or DA matched in the IPv6 packets. The field is valid and applicable only when MAC_L3_L4_Control0[L3DAM0] or MAC_L3_L4_Control0[L3SAM0] is 1.
5 L3DAIM2	Layer 3 IP DA Inverse Match Enable 2 Indicates the status of layer 3 IP DA inverse matching.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• If this field is 1, layer 3 IP DA is enabled for inverse matching. • If this field is 0, layer 3 IP DA is enabled for perfect matching. This field is valid and applicable only if MAC_L3_L4_Control0[L3DAM0] = 1. 0b - Disabled 1b - Enabled
4 L3DAM2	Layer 3 IP DA Match Enable 2 Indicates the status of layer 3 IP DA matching. • If this field is 1, layer 3 IP DA is enabled for matching. • If this field is 0, MAC ignores layer 3 IP DA for matching. **NOTE** If MAC_L3_L4_Control0[L3PEN0] = 1, you must write 1 to either this field or to MAC_L3_L4_Control0[L3SAM0] because either the IPv6 DA or SA can be checked for filtering. 0b - Disabled 1b - Enabled
3 L3SAIM2	Layer 3 IP SA Inverse Match Enable 2 Indicates the status of layer 3 IP SA inverse matching. • If this field is 1, layer 3 IP SA is enabled for inverse matching. • If this field is 0, layer 3 IP SA is enabled for perfect matching. This field is valid and applicable only if MAC_L3_L4_Control0[L3SAM0] is 1. 0b - Disabled 1b - Enabled
2 L3SAM2	Layer 3 IP SA Match Enable 2 Indicates the status of layer 3 IP SA matching. • If this field is 1, layer 3 IP SA matching is enabled. • If this field is 0, MAC ignores layer 3 IP SA matching. **NOTE** If MAC_L3_L4_Control0[L3PEN0] = 1, you must write 1 to either this field or to MAC_L3_L4_Control0[L3DAM0] because either IPv6 SA or DA can be checked for filtering. 0b - Disabled 1b - Enabled
1 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 L3PEN2	Layer 3 Protocol Enable 2 Indicates the status of layer 3 protocol. • If this field is 1, the layer 3 IP SA or DA matching is enabled for IPv6 packets. • If this field is 0, the layer 3 IP SA or DA matching is enabled for IPv4 packets. The layer 3 matching is performed only when either MAC_L3_L4_Control0[L3SAM0] or MAC_L3_L4_Control0[L3DAM0] is 1. 0b - Disabled 1b - Enabled

75.17.137 MAC Layer 4 Address 2 (MAC_Layer4_Address2)

Offset

Register	Offset
MAC_Layer4_Address2	964h

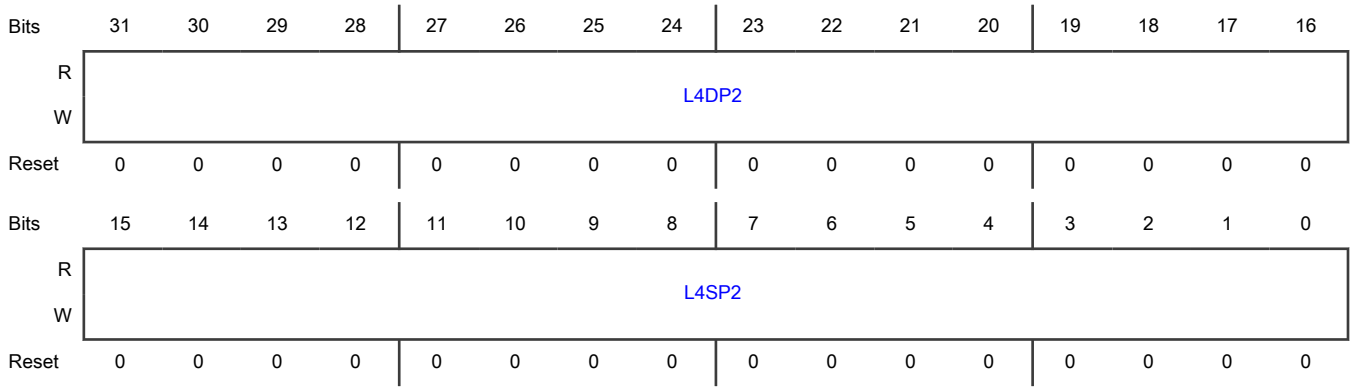
Function

Provides the layer 4 source and destination port numbers.

MAC Layer 4 Address 1 (MAC_Layer4_Address1), MAC L3 L4 Control 1 (MAC_L3_L4_Control1), MAC Layer 3 Address 0 Reg 1 (MAC_Layer3_Addr0_Reg1), MAC Layer 3 Address 1 Reg 1 (MAC_Layer3_Addr1_Reg1), MAC Layer 3 Address 2 Reg 1 (MAC_Layer3_Addr2_Reg1), and MAC Layer 3 Address 3 Reg 1 (MAC_Layer3_Addr3_Reg1) are reserved registers (RO with a default value) if the enable layer 3 and layer 4 packet filter option is not selected when configuring the core.

You can configure the layer 3 and layer 4 address registers to be double-synchronized by selecting the synchronize layer 3 and layer 4 address registers to the receive clock domain option while configuring the core. When you select this option, the synchronization is triggered only when bits [31:24] (in Little-Endian mode) or bits [7:0] (in Big-Endian mode) of the layer 3 and layer 4 address registers are written to. For proper synchronization updates, you must perform consecutive writes to the same layer 3 and layer 4 address registers after a delay of at least four destination clock cycles.

Diagram



Fields

Field	Function
31-16 L4DP2	Layer 4 Destination Port Number 2 Indicates layer 4 destination port number. If MAC_L3_L4_Control0[L4PEN0] = 0 and MAC_L3_L4_Control0[L4DPM0] = 1, this field contains the value to be matched with the TCP destination port number field in the IPv4 or IPv6 packets. If MAC_L3_L4_Control0[L4PEN0] and MAC_L3_L4_Control0[L4DPM0] are 1, this field contains the value to be matched with the UDP destination port number field in the IPv4 or IPv6 packets.
15-0 L4SP2	Layer 4 Source Port Number 2 Indicates layer 4 source port number. If MAC_L3_L4_Control0[L4PEN0] = 0 and MAC_L3_L4_Control0[L4SPM0] = 1, this field contains the value to be matched with the TCP source port number field in the IPv4 or IPv6 packets. If MAC_L3_L4_Control0[L4PEN0] and MAC_L3_L4_Control0[L4SPM0] are 1, this field contains the value to be matched with the UDP source port number field in the IPv4 or IPv6 packets.

75.17.138 MAC Layer 3 Address 0 Reg 2 (MAC_Layer3_Addr0_Reg2)

Offset

Register	Offset
MAC_Layer3_Addr0_Reg2	970h

Function

Contains the 32-bit IP source address field for IPv4 packets. For IPv6 packets, the field contains bits [31:0] of the 128-bit IP source address or destination address field.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	L3A02															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	L3A02															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 L3A02	Layer 3 Address 0 Indicates layer 3 address 0. • If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3SAM0] are 1, this field contains the value to be matched with bits [31:0] of the IP source address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3DAM0] are 1, this field contains the value to be matched with bits [31:0] of the IP destination address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] = 0, and MAC_L3_L4_Control0[L3SAM0] = 1, this field contains the value to be matched with the IP source address field in the IPv4 packets.

75.17.139 MAC Layer 3 Address 1 Reg 2 (MAC_Layer3_Addr1_Reg2)

Offset

Register	Offset
MAC_Layer3_Addr1_Reg2	974h

Function

Contains the 32-bit IP destination address field for IPv4 packets. For IPv6 packets, the field contains bits[63:32] of the 128-bit IP source address or destination address field.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	L3A12															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	L3A12															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 L3A12	Layer 3 Address 1 Indicates layer 3 address 1.

Table continues on the next page...

Field	Function
	• If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3SAM0] are 1, this field contains the value to be matched with bits [63:32] of the IP source address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3DAM0] are 1, this field contains the value to be matched with bits [63:32] of the IP destination address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] = 0, and MAC_L3_L4_Control0[L3SAM0] = 1, this field contains the value to be matched with the IP source address field in the IPv4 packets.

75.17.140 MAC Layer 3 Address 2 Reg 2 (MAC_Layer3_Addr2_Reg2)

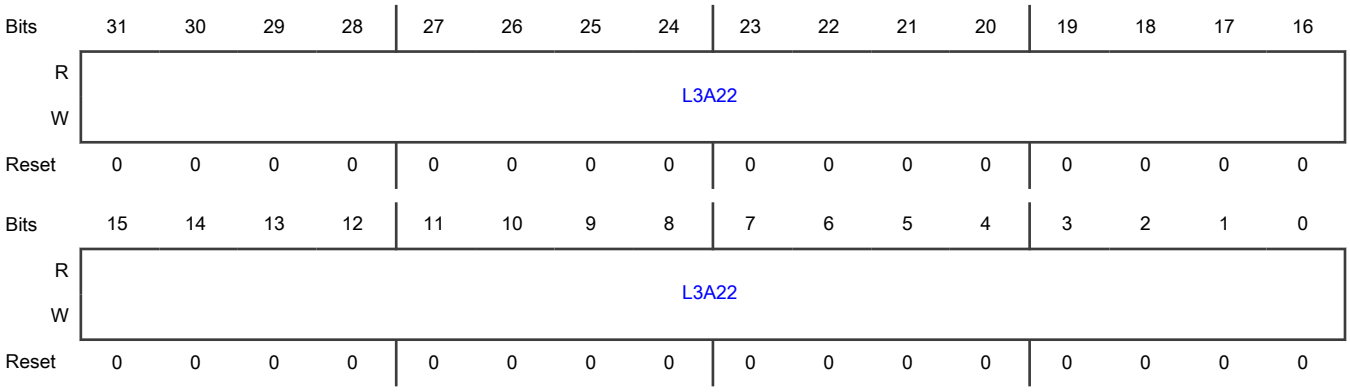
Offset

Register	Offset
MAC_Layer3_Addr2_Reg2	978h

Function

Contains the 32-bit IP destination address field for IPv4 packets. For IPv6 packets, the field contains bits [95:64] of the 128-bit IP source address or destination address field.

Diagram



Fields

Field	Function
31-0 L3A22	Layer 3 Address 2 Indicates layer 3 address 2. • If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3SAM0] are 1, this field contains the value to be matched with bits [95:64] of the IP source address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3DAM0] are 1, this field contains the value to be matched with bits [95:64] of the IP destination address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] = 0, this field is not used.

75.17.141 MAC Layer 3 Address 3 Reg 2 (MAC_Layer3_Addr3_Reg2)

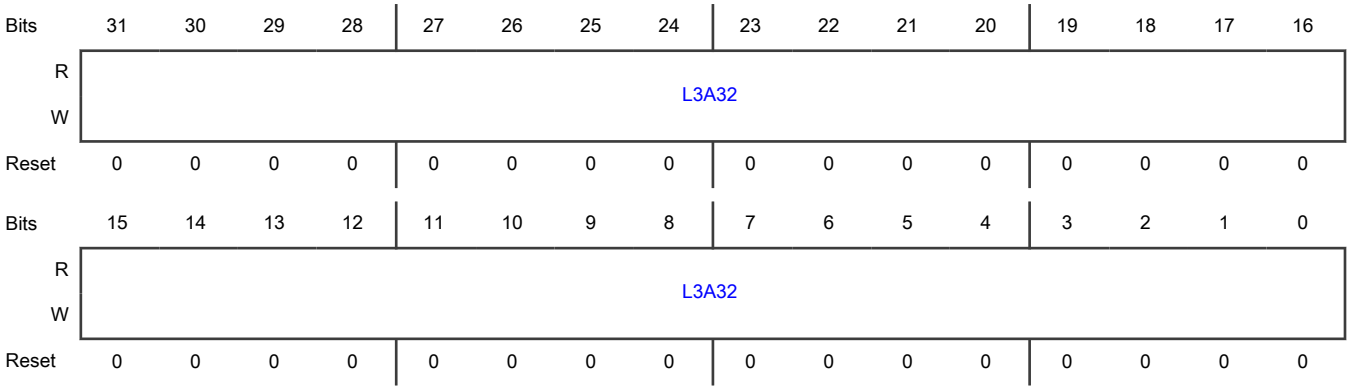
Offset

Register	Offset
MAC_Layer3_Addr3_Reg2	97Ch

Function

Contains the 32-bit IP destination address field for IPv4 packets. For IPv6 packets, the field contains bits [127:96] of the 128-bit IP source address or destination address field.

Diagram



Fields

Field	Function
31-0 L3A32	Layer 3 Address 3 Indicates layer 3 address 3. • If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3SAM0] are 1, this field contains the value to be matched with bits [127:96] of the IP source address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3DAM0] are 1, this field contains the value to be matched with bits [127:96] of the IP destination address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] = 0, this field is not used.

75.17.142 MAC L3 L4 Control 3 (MAC_L3_L4_Control3)

Offset

Register	Offset
MAC_L3_L4_Control3	990h

Function

Controls the filter 0 operations of the layer 3 and layer 4 protocols.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved			DMCHEN3	Reserved			DMCHN3	Reserved		L4DPIM3	L4DPM3	L4SPIM3	L4SPM3	Reserved	L4PEN3
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	L3HDBM3				L3HSBM3						L3DAIM3	L3DAM3	L3SAIM3	L3SAM3	Reserved	L3PEN3
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	Reserved
28 DMCHEN3	DMA Channel Select Enable 3 Indicates the status of the DMA channel number. • If this field is 1, it enables the selection of the DMA channel number for the packet that the L3_L4 filter passes. The DMCHN fields indicate the DMA channel number. • If this field is 0, the filter does not decide the DMA channel number. 0b - Disabled 1b - Enabled
27-25 —	Reserved
24 DMCHN3	DMA Channel Number 2 Indicates the DMA channel number. If the value of the DMCHEN fields is 1, this field selects the DMA channel number to which the packet that this filter passes is routed. The width of this field depends on the number of the DMA channels present in your configuration.
23-22 —	Reserved
21 L4DPIM3	Layer 4 Destination Port Inverse Match Enable 3 Indicates the status of layer 4 destination port inverse matching. • If this field is 1, MAC_Layer4_Address0[L4DP0] is enabled for inverse matching. • If this field is 0, MAC_Layer4_Address0[L4DP0] is enabled for perfect matching. This field is valid and applicable only when the L4DPM0 field is 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disabled 1b - Enabled
20 L4DPM3	Layer 4 Destination Port Match Enable 3 Indicates the status of layer 4 destination port matching. • If this field is 1, MAC_Layer4_Address0[L4DP0] is enabled for matching. • If this field is 0, MAC ignores MAC_Layer4_Address0[L4DP0] for matching. 0b - Disabled 1b - Enabled
19 L4SPIM3	Layer 4 Source Port Inverse Match Enable 3 Indicates the status of layer 4 source port inverse matching. • If this field is 1, MAC_Layer4_Address0[L4SP0] is enabled for inverse matching. • If this field is 0, MAC_Layer4_Address0[L4SP0] is enabled for perfect matching. This field is valid and applicable only when MAC_Layer4_Address0[L4SP0] = 1. 0b - Disabled 1b - Enabled
18 L4SPM3	Layer 4 Source Port Match Enable 3 Indicates the status of layer 4 source port matching. • If this field is 1, MAC_Layer4_Address0[L4SP0] is enabled for matching. • If this field is 0, MAC ignores MAC_Layer4_Address0[L4SP0] for matching. 0b - Disabled 1b - Enabled
17 —	Reserved
16 L4PEN3	Layer 4 Protocol Enable 3 Indicates the status of layer 4 protocol. • If this field is 1, MAC_Layer4_Address0[L4SP0] and MAC_Layer4_Address0[L4DP0] of UDP packets are used for matching. • If this field is 0, MAC_Layer4_Address0[L4SP0] and MAC_Layer4_Address0[L4DP0] of TCP packets are used for matching. Layer 4 matching is performed only when the L4SPM0 field or the L4DPM0 field is 1. 0b - Disabled 1b - Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-11 L3HDBM3	Layer 3 IP DA Higher Bits Match 3 IPv4 Packets: This field contains the number of higher bits of IP DA that are matched in the IPv4 packets. The following list describes the values of this field: • 0: No bits are masked. • 1: LSb[0] is masked. • 2: Two LSbs [1:0] are masked. • .. • 31: All bits except MSb are masked. IPv6 Packets: Bits [12:11] of this field correspond to bits [6:5] of MAC_L3_L4_Control0[L3HSBM0], which indicates the number of lower bits of IP SA or DA that are masked in the IPv6 packets. The following list describes the concatenated values of MAC_L3_L4_Control0[L3HDBM0][1:0] and MAC_L3_L4_Control0[L3HSBM0]: • 0: No bits are masked. • 1: LSb[0] is masked. • 2: Two LSbs [1:0] are masked • .. • 127: All bits except MSb are masked. This field is valid and applicable only when MAC_L3_L4_Control0[L3DAM0] or MAC_L3_L4_Control0[L3SAM0] is 1. .
10-6 L3HSBM3	Layer 3 IP SA Higher Bits Match 3 IPv4 packets: This field contains the number of lower bits of IP SA that are masked for matching in the IPv4 packets. The following list describes the values of this field: • 0: No bits are masked. • 1: LSb[0] is masked. • 2: Two LSbs [1:0] are masked. • .. • 31: All bits except MSb are masked. IPv6 packets: This field contains bits [4:0] of MAC_L3_L4_Control0[L3HSBM0]. These bits indicate the number of higher bits of IP SA or DA matched in the IPv6 packets. The field is valid and applicable only when MAC_L3_L4_Control0[L3DAM0] or MAC_L3_L4_Control0[L3SAM0] is 1.
5	Layer 3 IP DA Inverse Match Enable 3

Table continues on the next page...

Table continued from the previous page...

Field	Function
L3DAIM3	Indicates the status of layer 3 IP DA inverse matching. • If this field is 1, layer 3 IP DA is enabled for inverse matching. • If this field is 0, layer 3 IP DA is enabled for perfect matching. This field is valid and applicable only if MAC_L3_L4_Control0[L3DAM0] = 1. 0b - Disabled 1b - Enabled
4 L3DAM3	Layer 3 IP DA Match Enable 3 Indicates the status of layer 3 IP DA matching. • If this field is 1, layer 3 IP DA is enabled for matching. • If this field is 0, MAC ignores layer 3 IP DA for matching. **NOTE** If MAC_L3_L4_Control0[L3PEN0] = 1, you must write 1 to either this field or to MAC_L3_L4_Control0[L3SAM0] because either the IPv6 DA or SA can be checked for filtering. 0b - Disabled 1b - Enabled
3 L3SAIM3	Layer 3 IP SA Inverse Match Enable 3 Indicates the status of layer 3 IP SA inverse matching. • If this field is 1, layer 3 IP SA is enabled for inverse matching. • If this field is 0, layer 3 IP SA is enabled for perfect matching. This field is valid and applicable only if MAC_L3_L4_Control0[L3SAM0] is 1. 0b - Disabled 1b - Enabled
2 L3SAM3	Layer 3 IP SA Match Enable 3 Indicates the status of layer 3 IP SA matching. • If this field is 1, layer 3 IP SA matching is enabled. • If this field is 0, MAC ignores layer 3 IP SA matching. **NOTE** If MAC_L3_L4_Control0[L3PEN0] = 1, you must write 1 to either this field or to MAC_L3_L4_Control0[L3DAM0] because either IPv6 SA or DA can be checked for filtering. 0b - Disabled 1b - Enabled
1	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
0 L3PEN3	Layer 3 Protocol Enable 3 Indicates the status of layer 3 protocol. • If this field is 1, the layer 3 IP SA or DA matching is enabled for IPv6 packets. • If this field is 0, the layer 3 IP SA or DA matching is enabled for IPv4 packets. The layer 3 matching is performed only when either MAC_L3_L4_Control0[L3SAM0] or MAC_L3_L4_Control0[L3DAM0] is 1. 0b - Disabled 1b - Enabled

75.17.143 MAC Layer 4 Address 3 (MAC_Layer4_Address3)

Offset

Register	Offset
MAC_Layer4_Address3	994h

Function

Provides the layer 4 source and destination port numbers.

MAC Layer 4 Address 1 (MAC_Layer4_Address1), MAC L3 L4 Control 1 (MAC_L3_L4_Control1), MAC Layer 3 Address 0 Reg 1 (MAC_Layer3_Addr0_Reg1), MAC Layer 3 Address 1 Reg 1 (MAC_Layer3_Addr1_Reg1), MAC Layer 3 Address 2 Reg 1 (MAC_Layer3_Addr2_Reg1), and MAC Layer 3 Address 3 Reg 1 (MAC_Layer3_Addr3_Reg1) are reserved registers (RO with a default value) if the enable layer 3 and layer 4 packet filter option is not selected when configuring the core.

You can configure the layer 3 and layer 4 address registers to be double-synchronized by selecting the synchronize layer 3 and layer 4 address registers to the receive clock domain option while configuring the core. When you select this option, the synchronization is triggered only when bits [31:24] (in Little-Endian mode) or bits [7:0] (in Big-Endian mode) of the layer 3 and layer 4 address registers are written to. For proper synchronization updates, you must perform consecutive writes to the same layer 3 and layer 4 address registers after a delay of at least four destination clock cycles.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	L4DP3															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	L4SP3															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 L4DP3	Layer 4 Destination Port Number 3 Indicates layer 4 destination port number. If MAC_L3_L4_Control0[L4PEN0] = 0 and MAC_L3_L4_Control0[L4DPM0] = 1, this field contains the value to be matched with the TCP destination port number field in the IPv4 or IPv6 packets. If MAC_L3_L4_Control0[L4PEN0] and MAC_L3_L4_Control0[L4DPM0] are 1, this field contains the value to be matched with the UDP destination port number field in the IPv4 or IPv6 packets.
15-0 L4SP3	Layer 4 Source Port Number 3 Indicates layer 4 source port number. If MAC_L3_L4_Control0[L4PEN0] = 0 and MAC_L3_L4_Control0[L4SPM0] = 1, this field contains the value to be matched with the TCP source port number field in the IPv4 or IPv6 packets. If MAC_L3_L4_Control0[L4PEN0] and MAC_L3_L4_Control0[L4SPM0] are 1, this field contains the value to be matched with the UDP source port number field in the IPv4 or IPv6 packets.

75.17.144 MAC Layer 3 Address 0 Reg 3 (MAC_Layer3_Addr0_Reg3)

Offset

Register	Offset
MAC_Layer3_Addr0_Reg3	9A0h

Function

Contains the 32-bit IP source address field for IPv4 packets. For IPv6 packets, the field contains bits [31:0] of the 128-bit IP source address or destination address field.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	L3A03															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	L3A03															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 L3A03	Layer 3 Address 0 Indicates layer 3 address 0. • If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3SAM0] are 1, this field contains the value to be matched with bits [31:0] of the IP source address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3DAM0] are 1, this field contains the value to be matched with bits [31:0] of the IP destination address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] = 0, and MAC_L3_L4_Control0[L3SAM0] = 1, this field contains the value to be matched with the IP source address field in the IPv4 packets.

75.17.145 MAC Layer 3 Address 1 Reg 3 (MAC_Layer3_Addr1_Reg3)

Offset

Register	Offset
MAC_Layer3_Addr1_Reg3	9A4h

Function

Contains the 32-bit IP destination address field for IPv4 packets. For IPv6 packets, the field contains bits [63:32] of the 128-bit IP source address or destination address field.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	L3A13															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	L3A13															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 L3A13	Layer 3 Address 1 Indicates layer 3 address 1.

Table continues on the next page...

Field	Function
	• If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3SAM0] are 1, this field contains the value to be matched with bits [63:32] of the IP source address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3DAM0] are 1, this field contains the value to be matched with bits [63:32] of the IP destination address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] = 0, and MAC_L3_L4_Control0[L3SAM0] = 1, this field contains the value to be matched with the IP source address field in the IPv4 packets.

75.17.146 MAC Layer 3 Address 2 Reg 3 (MAC_Layer3_Addr2_Reg3)

Offset

Register	Offset
MAC_Layer3_Addr2_Reg3	9A8h

Function

Contains the 32-bit IP destination address field for IPv4 packets. For IPv6 packets, the field contains bits [95:64] of the 128-bit IP source address or destination address field.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	L3A23															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	L3A23															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 L3A23	Layer 3 Address 2 Indicates layer 3 address 2. • If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3SAM0] are 1, this field contains the value to be matched with bits [95:64] of the IP source address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3DAM0] are 1, this field contains the value to be matched with bits [95:64] of the IP destination address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] = 0, this field is not used.

75.17.147 MAC Layer 3 Address 3 Reg 3 (MAC_Layer3_Addr3_Reg3)

Offset

Register	Offset
MAC_Layer3_Addr3_Reg3	9ACh

Function

Contains the 32-bit IP destination address field for IPv4 packets. For IPv6 packets, the field contains bits [127:96] of the 128-bit IP source address or destination address field.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	L3A33															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	L3A33															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 L3A33	Layer 3 Address 3 Indicates layer 3 address 3. • If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3SAM0] are 1, this field contains the value to be matched with bits [127:96] of the IP source address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] and MAC_L3_L4_Control0[L3DAM0] are 1, this field contains the value to be matched with bits [127:96] of the IP destination address field in the IPv6 packets. • If MAC_L3_L4_Control0[L3PEN0] = 0, this field is not used.

75.17.148 MAC Timestamp Control (MAC_Timestamp_Control)

Offset

Register	Offset
MAC_Timestamp_Control	B00h

Function

Controls the operation of the system time generator and processing of PTP packets for timestamping in the receiver.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved			AV8021A...	Reserved			TXTSSTSM	Reserved			ESTI	Reserved	TSENMAC...	SNAPTYPSEL	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	TSMSTRE...	TSEVNTE...	TSIPV4E...	TSIPV6E...	TSIPENA	TSVER2E...	TSCTRLS...	TSENALL	Reserved	PTGE	TSADDREG	Reserved	TSUPDT	TSINIT	TSCFUPDT	TSENA
Reset	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	Reserved
28 AV8021ASMEN	AV 802.1AS Mode Enable Indicates the status of AV 802.1AS mode. If this field is 1, MAC processes only untagged PTP over Ethernet packets for providing PTP status and capturing timestamp snapshots, which means using the IEEE 802.1AS mode of operation. If MAC_HW_Feature1[PTOEN] = 1, for the purpose of PTP offload, the transport-specific field in the PTP header is generated and checked based on the value of the AV8021ASMEN field. 0b - Disabled 1b - Enabled
27-25 —	Reserved
24 TXTSSTSM	Transmit Timestamp Status Mode Indicates the status of transmit timestamp status mode. • If this field is 1, MAC overwrites the previous transmit timestamp status even if the software does not read it. MAC indicates this by writing 1 to MAC_Tx_Timestamp_Status_Nanoseconds[TXTSSMIS]. • If this field is 0, MAC ignores the timestamp status of the current packet if the software does not read the timestamp status of the previous packet. MAC indicates this by writing 1 to MAC_Tx_Timestamp_Status_Nanoseconds[TXTSSMIS]. 0b - Disabled 1b - Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
23-21 —	Reserved
20 ESTI	External System Time Input Indicates the status of external system time input. If this field is 1, MAC uses the external 64-bit reference system time input for these functions: • To consider the timestamp provided as status • To insert the timestamp in transmit PTP packets if the one-step timestamp or the timestamp offload feature is enabled If this field is 1, MAC uses the internal reference system time. 0b - Disabled 1b - Enabled
19 —	Reserved
18 TSENMACADDR	Enable MAC Address For PTP Packet Filtering Indicates whether the MAC address for PTP packet filtering is enabled. If this field is 1, the DA MAC address, which matches any MAC Address register, is used to filter the PTP packets when PTP is directly sent over Ethernet. If this field is 1, received PTP packets with DA containing a special multicast or unicast address that matches the one programmed in the MAC address registers are considered for processing when PTP is directly sent over Ethernet. • For a normal time stamping operation, MAC address registers 0 to 31 are considered for unicast destination address matching. • For PTP offload, only MAC address register 0 is considered for unicast destination address matching. 0b - Disabled 1b - Enabled
17-16 SNAPTYPSEL	Select PTP Packets For Taking Snapshots Determines the set of PTP packet types for which a snapshot needs to be taken. This field, along with the TSEVNTENA and TSMSTRENA fields, determines the set of PTP packet types for which a snapshot needs to be taken. See Receive path functions for encoding.
15 TSMSTRENA	Enable Snapshot For Messages Relevant To Master Indicates whether the snapshot for messages relevant to master is enabled. If this field is 1, the snapshot is taken only for the messages that are relevant to the master node. Otherwise, the snapshot is taken for the messages relevant to the slave node.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disabled 1b - Enabled
14 TSEVNTENA	Enable Timestamp Snapshot For Event Messages Enables or disables timestamp snapshot for event messages. If this field is 1, the timestamp snapshot is taken only for event messages (SYNC, Delay_Req, Pdelay_Req, or Pdelay_Resp). When the field becomes 0, the snapshot is taken for all the messages except Announce, Management, and Signaling. For more information on timestamp snapshot dependency on register bits, see Receive path functions. 0b - Disabled 1b - Enabled
13 TSIPV4ENA	Enable Processing Of PTP Packets Sent Over IPv4-UDP Indicates whether the processing of PTP packets sent over IPv4-UDP is enabled. If this field is 1, the MAC receiver processes the PTP packets encapsulated in the IPv4-UDP packets. When this field becomes 0, MAC ignores the PTP transported over the IPv4-UDP packets. The default value of this field is 1. 0b - Disabled 1b - Enabled
12 TSIPV6ENA	Enable Processing Of PTP Packets Sent Over IPv6-UDP Indicates whether the processing of PTP packets sent over IPv6-UDP is enabled. If this field is 1, the MAC receiver processes the PTP packets encapsulated in the IPv6-UDP packets. If the value is 0, MAC ignores the PTP transported over IPv6-UDP packets. 0b - Disabled 1b - Enabled
11 TSIPENA	Enable Processing Of PTP Over Ethernet Packets Indicates the status of PTP processing over Ethernet packets. If this field is 1, the MAC receiver processes the PTP packets encapsulated directly in the Ethernet packets. When the field becomes 0, MAC ignores the PTP over Ethernet packets. 0b - Disabled 1b - Enabled
10 TSVER2ENA	Enable PTP Packet Processing For Version 2 Format Indicates the status of PTP packet processing for version 2 format. If this field is 1, the IEEE 1588 version 2 format is used to process the PTP packets. When this field becomes 0, the IEEE 1588 version 1 format is used to process the PTP packets. See PTP processing and control for more information on the IEEE 1588 formats.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disabled 1b - Enabled
9 TSCTRLSSR	Timestamp Digital Or Binary Rollover Control Indicates the status of timestamp digital or binary rollover control. • If this field is 1, the Timestamp Low register rolls over after the 3B9A_C9FFh value (that is, 1 nanosecond accuracy) and increments the timestamp (high) seconds. • If this field is 0, the rollover value of the Subsecond register is 7FFF_FFFFh. The sub-second increment must be programmed correctly depending on the PTP reference clock frequency and the value of this field. 0b - Disabled 1b - Enabled
8 TSENALL	Enable Timestamp For All Packets Indicates the status of timestamp snapshot for all packets. If this field is 1, the timestamp snapshot is enabled for all packets that MAC receives. 0b - Disabled 1b - Enabled
7 —	Reserved
6 PTGE	Presentation Time Generation Enable Indicates the status of presentation time generation. If this field is 1, the presentation time generation is enabled. 0b - Disabled 1b - Enabled
5 TSADDREG	Update Addend Register Indicates whether MAC Timestamp Addend (MAC_Timestamp_Addend) is updated. If this field is 1, the contents of MAC Timestamp Addend (MAC_Timestamp_Addend) are updated in the PTP block for fine correction. The field becomes 0 after the update completes. The value of the field must also be 0 before you write 1 to it. Access restrictions apply to this field that clears automatically. Writing 0 has no effect. 0b - Not updated 1b - Updated
4 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
3 TSUPDT	Update Timestamp Indicates whether the timestamp is updated. If this field is 1, the system time is updated (added or subtracted) with the value specified in MAC System Time Seconds Update (MAC_System_Time_Seconds_Update) and MAC System Time Nanoseconds Update (MAC_System_Time_Nanoseconds_Update). The value of this field must be 0 before you update it. It resets after the update is complete in hardware. MAC_System_Time_Higher_Word_Seconds[TSHWR], if enabled during core configuration, is not updated. When media clock generation and recovery is configured (DWC_EQOS_FLEXI_PPS_OUT_EN) and enabled, MAC Presentation Time Update (MAC_Presn_Time_Updt) must also be updated before you write 1 to this field. Access restrictions apply to this field that clears automatically. Writing 0 has no effect. 0b - Not updated 1b - Updated
2 TSINIT	Initialize Timestamp Indicates whether the timestamp is initialized. If this field is 1, the system time is initialized (overwritten) with the value specified in MAC System Time Seconds Update (MAC_System_Time_Seconds_Update) and MAC System Time Nanoseconds Update (MAC_System_Time_Nanoseconds_Update). The value of this field must be 0 before you update it, and the field is reset after the initialization is complete. MAC_System_Time_Higher_Word_Seconds[TSHWR], if enabled during core configuration, can only be initialized. When media clock generation and recovery is configured (DWC_EQOS_FLEXI_PPS_OUT_EN) and enabled, MAC Presentation Time Update (MAC_Presn_Time_Updt) must also be updated before you write 1 to this field. Access restrictions apply to this field that clears automatically. Writing 0 has no effect. 0b - Not initialized 1b - Initialized
1 TSCFUPDT	Fine Or Coarse Timestamp Update Indicates the method used to update system timestamp. • If this field is 1, the fine method is used to update system timestamp. • If this field is 0, the coarse method is used to update the system timestamp. 0b - Coarse method 1b - Fine method
0 TSENA	Timestamp Enable Enables or disables timestamp for transmit and receive packets.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• If this field is 1, the timestamp is added for transmit and receive packets. • If this field is 0, timestamp is not added for transmit and receive packets and the timestamp generator is also suspended. You need to initialize the timestamp (system time) after enabling this mode. On the receive side, MAC processes the 1588 packets only if this field is 1. 0b - Disabled 1b - Enabled

75.17.149 MAC Sub Second Increment (MAC_Sub_Second_Increment)

Offset

Register	Offset
MAC_Sub_Second_Increment	B04h

Function

Specifies the value to be added to the internal system time register at every cycle of the CLK_PTP_REF_I clock.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved								SSINC							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SNSINC								Reserved							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-16 SSINC	Sub-Second Increment Value

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Contains the sub-second increment value, which accumulates every clock cycle (of clk_ptp_i) with the contents of the Subsecond register. For example, if the PTP clock is 50 MHz (period is 20 ns), you must program 20 (14h) when MAC System Time In Nanoseconds (MAC_System_Time_Nanoseconds) has an accuracy of 1 ns (MAC_Timestamp_Control[TSCTRLSSR] = 1). When TSCTRLSSR = 0, the Nanoseconds register has a resolution of ~0.465 ns. In this case, you must program a value of 43 (2Bh), which is derived by 20 ns/0.465.
15-8 SNSINC	Sub-Nanosecond Increment Value Contains the sub-nanosecond increment value, represented in nanoseconds multiplied by 2^8. This value is accumulated in the sub-nanoseconds field of the Subsecond register. For example, if MAC_Timestamp_Control[TSCTRLSSR] = 1, and if the required increment is 5.3 ns, then this field must be 4Ch and the SSINC field must be 05h.
7-0 —	Reserved

75.17.150 MAC System Time In Seconds (MAC_System_Time_Seconds)

Offset

Register	Offset
MAC_System_Time_Seconds	B08h

Function

Indicates, along with MAC System Time In Nanoseconds (MAC_System_Time_Nanoseconds), the current value of the system time that MAC maintains. Although it is updated on a continuous basis, there is some delay from the actual time because of clock domain transfer latencies (from CLK_PTP_REF_I to the CSR clock).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TSS															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TSS															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TSS	Timestamp Second Indicates the current value, in seconds, of the system time that MAC maintains.

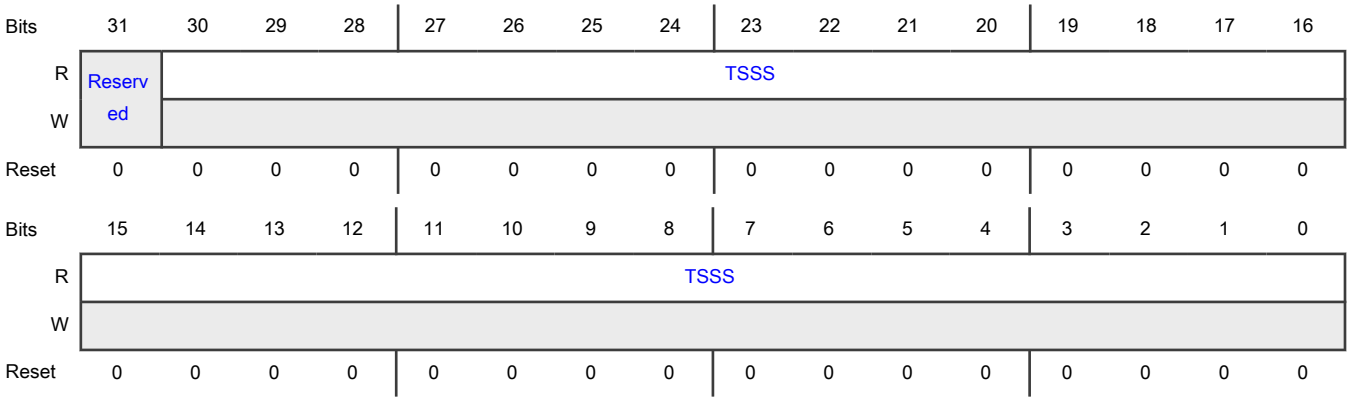
75.17.151 MAC System Time In Nanoseconds (MAC_System_Time_Nanoseconds)

Offset

Register	Offset
MAC_System_Time_Nanoseconds	B0Ch

Function
Indicates, along with MAC System Time In Seconds (MAC_System_Time_Seconds), the current value of the system time that MAC maintains.

Diagram



Fields

Field	Function
31 —	Reserved
30-0 TSSS	Timestamp Sub Seconds Indicates the sub-second representation of time, with an accuracy of 0.46 ns. When MAC_Timestamp_Control[TSCTRLSSR] = 1, each bit represents 1 ns. The maximum value is 3B9A_C9FFh after which it rolls over to 0.

75.17.152 MAC System Time Seconds Update (MAC_System_Time_Seconds_Update)

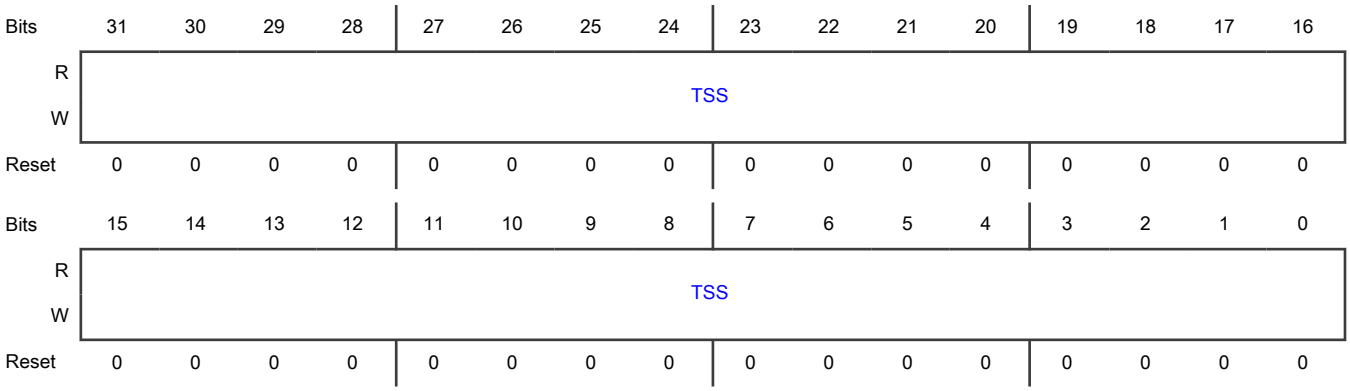
Offset

Register	Offset
MAC_System_Time_Seconds_Update	B10h

Function

Initializes or updates, along with MAC System Time Nanoseconds Update (MAC_System_Time_Nanoseconds_Update), the system time that MAC maintains. You must write both to this register and the MAC System Time Nanoseconds Update register before writing 1 to MAC_Timestamp_Control[TSINIT] and MAC_Timestamp_Control[TSUPDT].

Diagram



Fields

Field	Function
31-0 TSS	Timestamp Seconds Indicates the timestamp seconds value. The value in this field is the seconds part of the update. • If MAC_System_Time_Nanoseconds_Update[ADDSUB] = 0, this field must be programmed with the seconds part of the update value. • If MAC_System_Time_Nanoseconds_Update[ADDSUB] = 1, this field must be programmed with the complement of the seconds part of the update value. For example, if 2.000000001 seconds need to be subtracted from the system time, MAC_System_Time_Seconds[TSS] must be FFFF_FFFEh (that is, 2^32 - 2).

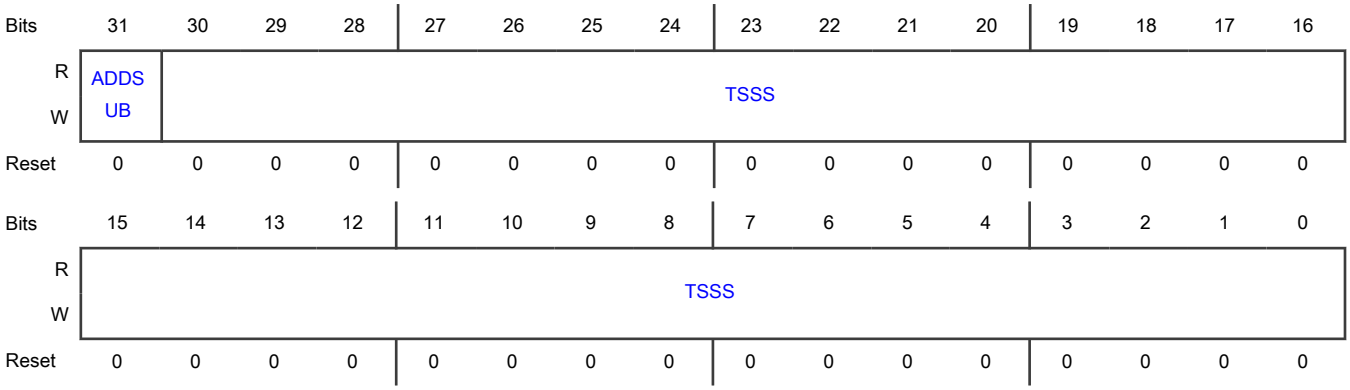
75.17.153 MAC System Time Nanoseconds Update (MAC_System_Time_Nanoseconds_Update)

Offset

Register	Offset
MAC_System_Time_Nanoseconds_Update	B14h

Function
Indicates system time nanoseconds update.

Diagram



Fields

Field	Function
31 ADDSUB	Add Or Subtract Time Indicates whether the time value is added or subtracted from the contents of the update register. • If this field is 1, the time value is subtracted from the contents of the update register. • If this field is 0, the time value is added to the contents of the update register. 0b - Add time 1b - Subtract time
30-0 TSSS	Timestamp Subseconds Indicates the sub-second part of the update. • If ADDSUB = 0, this field must be programmed with the sub-second part of the update value, with an accuracy based on MAC_Timestamp_Control[TSCTRLSSR]. • If ADDSUB = 0, this field must be programmed with the complement of the sub-second part of the update value as described below. • If MAC_Timestamp_Control[TSCTRLSSR] = 1, the programmed value must be 10^9 - <sub-second value>, and if MAC_Timestamp_Control[TSCTRLSSR] = 0, the programmed value must be 2^31 - <sub-second_value>.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• If MAC_Timestamp_Control[TSCTRLSSR] = 0, each bit of it represents an accuracy of 0.46 ns, and if MAC_Timestamp_Control[TSCTRLSSR] = 1, each bit represents 1 ns and the programmed value must not exceed 3B9A_C9FFh. For example, if 2.000000001 seconds need to be subtracted from the system time, this field must be 7FFF_FFFFh (that is, 2^31 - 1) if MAC_Timestamp_Control[TSCTRLSSR] = 0. This field must be 3B9A_C9FFh (that is, 10^9 - 1) if MAC_Timestamp_Control[TSCTRLSSR] = 1.

75.17.154 MAC Timestamp Addend (MAC_Timestamp_Addend)

Offset

Register	Offset
MAC_Timestamp_Addend	B18h

Function

Is used only when the system time is configured for Fine Update mode (MAC_Timestamp_Control[TSINIT] = 1). The contents of this register are added to a 32-bit accumulator in every clock cycle (of CLK_PTP_REF_I), and the system time is updated whenever the accumulator overflows.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	TSAR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	TSAR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TSAR	Timestamp Addend Register Indicates the 32-bit time value to be added to the Accumulator register to achieve time synchronization.

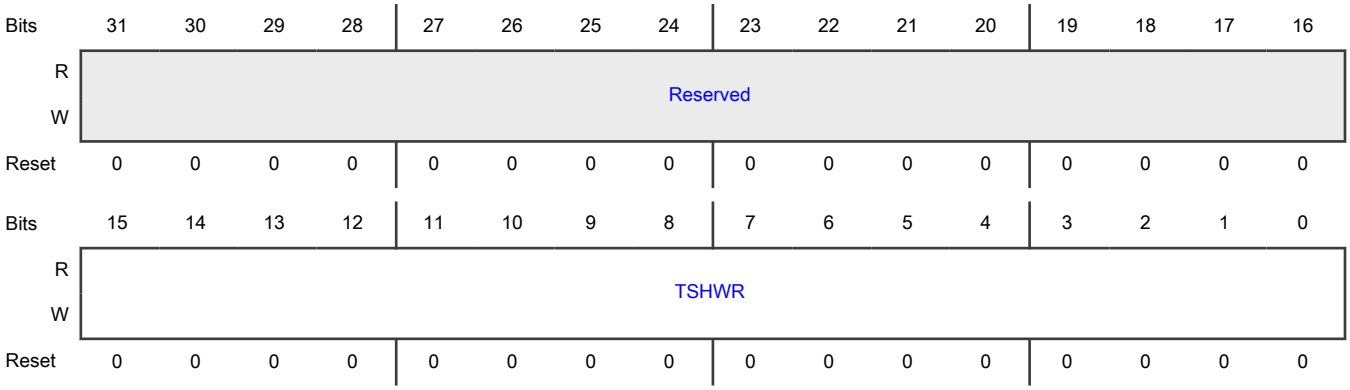
75.17.155 MAC System Time Higher Word In Seconds (MAC_System_Time_Higher_Word_Seconds)

Offset

Register	Offset
MAC_System_Time_Higher_Word_Seconds	B1Ch

Function
Indicates system time - higher word in seconds.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 TSHWR	Timestamp Higher Word Register Contains the most-significant 16 bits of the timestamp seconds value. This register is optional, and you can add it if you select the IEEE 1588 higher-word register option. You write to this register directly to initialize the value, which increments when there is an overflow from the 32 bits of MAC System Time In Seconds (MAC_System_Time_Seconds). Access restrictions apply to this field, which updates based on an event occurrence.

75.17.156 MAC Timestamp Status (MAC_Timestamp_Status)

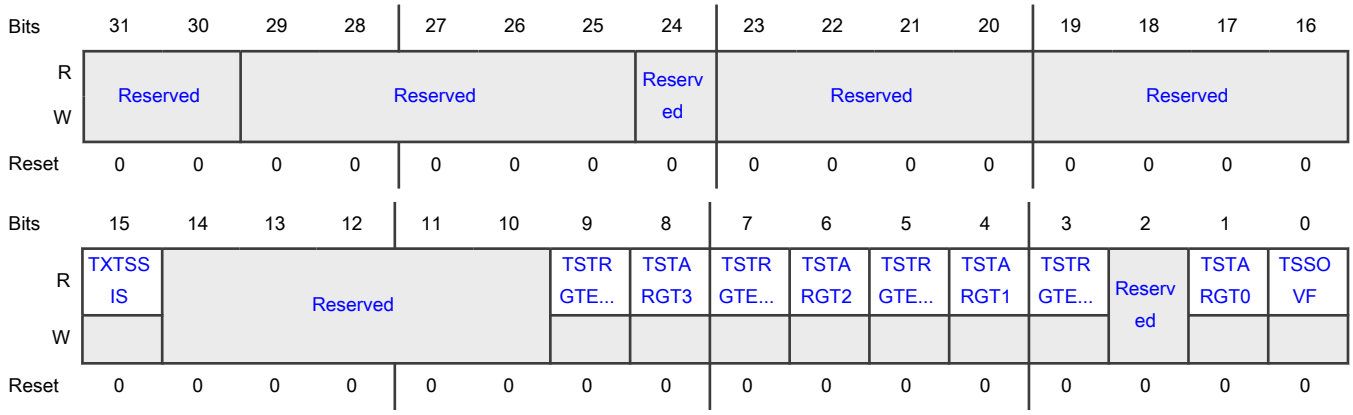
Offset

Register	Offset
MAC_Timestamp_Status	B20h

Function

Indicates timestamp status. All the bits except [27:25] are cleared when the application reads this register.

Diagram



Fields

Field	Function
31-30 —	Reserved
29-25 —	Reserved
24 —	Reserved
23-20 —	Reserved
19-16 —	Reserved
15 TXTSSIS	Transmit Timestamp Status Interrupt Status Indicates whether the transmit timestamp interrupt status is detected. In non-EQOS_CORE configurations, if the drop transmit status is enabled in MTL, this field becomes 1 when the captured transmit timestamp is updated in MAC Transmit Timestamp Status In Seconds (MAC_Tx_Timestamp_Status_Seconds) and MAC Transmit Timestamp Status In Nanoseconds (MAC_Tx_Timestamp_Status_Nanoseconds). If MAC_HW_Feature1[PTOEN] = 1, TXTSSIS becomes 1 when the captured transmit timestamp is updated in MAC Transmit Timestamp Status In Seconds (MAC_Tx_Timestamp_Status_Seconds) and MAC Transmit Timestamp Status In Nanoseconds (MAC_Tx_Timestamp_Status_Nanoseconds) for PTO-generated delay request and Pdelay request packets.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field becomes 0 if you read or write to MAC Transmit Timestamp Status In Seconds (MAC_Tx_Timestamp_Status_Seconds) when MAC_CSR_SW_Ctrl[RCWE] = 1. 0b - Not detected 1b - Detected
14-10 —	Reserved
9 TSTRGTERR3	Timestamp Target Time Error Indicates whether the timestamp target time error status is detected. This field becomes 1 if the latest target time programmed in MAC PPS3 Target Time In Seconds (MAC_PPS3_Target_Time_Seconds) and MAC PPS3 Target Time In Nanoseconds (MAC_PPS3_Target_Time_Nanoseconds) elapses. The field becomes 0 when the application reads it. Access restrictions apply to this field that clears when read (or when this field is written to 1 if MAC_CSR_SW_Ctrl[RCWE] = 1). Also, the field automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
8 TSTARGT3	Timestamp Target Time Reached For Target Time PPS3 Indicates whether the timestamp target time reached for target time PPS3 status is detected. If this field is 1, it indicates that the value of the system time is greater than or equal to the value specified in MAC PPS3 Target Time In Seconds (MAC_PPS3_Target_Time_Seconds) and MAC PPS3 Target Time In Nanoseconds (MAC_PPS3_Target_Time_Nanoseconds). Access restrictions apply to this field that clears when read (or when this field is written to 1 if MAC_CSR_SW_Ctrl[RCWE] = 1). Also, the field automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
7 TSTRGTERR2	Timestamp Target Time Error Indicates whether the timestamp target time error status is detected. This field becomes 1 when the latest target time programmed in MAC PPS2 Target Time In Seconds (MAC_PPS2_Target_Time_Seconds) and MAC PPS2 Target Time In Nanoseconds (MAC_PPS2_Target_Time_Nanoseconds) elapses. The field becomes 0 when the application reads it. Access restrictions apply to this field that clears when read (or when this field is written to 1 if MAC_CSR_SW_Ctrl[RCWE] = 1). Also, the field automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
6 TSTARGT2	Timestamp Target Time Reached For Target Time PPS2 Indicates whether the timestamp target time reached for the target time PPS2 status is detected.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	If this field is 1, it indicates that the value of system time is greater than or equal to the value specified in MAC PPS2 Target Time In Seconds (MAC_PPS2_Target_Time_Seconds) and MAC PPS2 Target Time In Nanoseconds (MAC_PPS2_Target_Time_Nanoseconds). Access restrictions apply to this field that clears when read (or when this field is written to 1 if MAC_CSR_SW_Ctrl[RCWE] = 1). Also, the field automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
5 TSTRGTERR1	Timestamp Target Time Error Indicates whether the timestamp target time error status is detected. This field becomes 1 when the latest target time programmed in MAC PPS1 Target Time In Seconds (MAC_PPS1_Target_Time_Seconds) and MAC PPS1 Target Time In Nanoseconds (MAC_PPS1_Target_Time_Nanoseconds) elapses. The field becomes 0 when the application reads it. Access restrictions apply to this field that clears when read (or when this field is written to 1 if MAC_CSR_SW_Ctrl[RCWE] = 1). Also, the field automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
4 TSTARGT1	Timestamp Target Time Reached For Target Time PPS1 Indicates whether the timestamp target time reached for target time PPS1 status is detected. If this field is 1, it indicates that the value of system time is greater than or equal to the value specified in MAC PPS1 Target Time In Seconds (MAC_PPS1_Target_Time_Seconds) and MAC PPS1 Target Time In Nanoseconds (MAC_PPS1_Target_Time_Nanoseconds) Access restrictions apply to this field that clears when read (or when this field is written to 1 if MAC_CSR_SW_Ctrl[RCWE] = 1). Also, the field automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
3 TSTRGTERR0	Timestamp Target Time Error Indicates whether the timestamp target time error status is detected. This field becomes 1 when the latest target time programmed in MAC PPS0 Target Time In Seconds (MAC_PPS0_Target_Time_Seconds) and MAC PPS0 Target Time In Nanoseconds (MAC_PPS0_Target_Time_Nanoseconds) elapses. The field becomes 0 when the application reads it. Access restrictions apply to this field that clears when read (or when this field is written to 1 if MAC_CSR_SW_Ctrl[RCWE] = 1). Also, the field automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
2 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 TSTARGT0	Timestamp Target Time Reached Indicates whether the timestamp target time reached status is detected. If this field is 1, it indicates that the value of the system time is greater than or equal to the value specified in MAC PPS0 Target Time In Seconds (MAC_PPS0_Target_Time_Seconds) and MAC PPS0 Target Time In Nanoseconds (MAC_PPS0_Target_Time_Nanoseconds) Access restrictions apply to this field that clears when read (or when this field is written to 1 if MAC_CSR_SW_Ctrl[RCWE] = 1). Also, the field automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
0 TSSOVF	Timestamp Seconds Overflow Indicates whether the timestamp seconds overflow status is detected. If this field is 1, it indicates that the seconds value of the timestamp (when supporting version 2 format) has overflowed beyond 32'hFFFF_FFFF. Access restrictions apply to this field that clears when read (or when this field is written to 1 if MAC_CSR_SW_Ctrl[RCWE] = 1). Also, the field automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected

75.17.157 MAC Transmit Timestamp Status In Nanoseconds (MAC_Tx_Timestamp_Status_Nanoseconds)

Offset

Register	Offset
MAC_Tx_Timestamp_Status_Nanoseconds	B30h

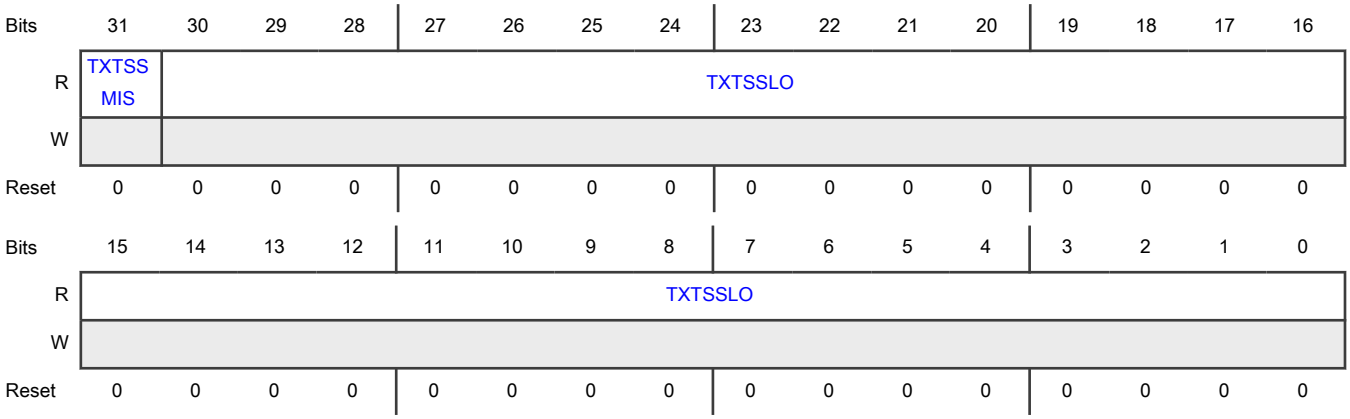
Function

Contains the nanosecond part of the timestamp captured for tansmit packets when the transmit status is disabled.

This register, along with MAC Transmit Timestamp Status In Seconds (MAC_Tx_Timestamp_Status_Seconds), gives the 64-bit timestamp captured for the PTP packet that MAC successfully transmits. The application considers reading this value after the last byte of this register is read. In Little-Endian mode, this means when bits [31:24] are read, and in Big-Endian mode, this means when bits [7:0] are read.

If the application does not read these registers and the timestamp of another packet is captured, then either the current timestamp is lost (overwritten) or the new timestamp is lost (dropped), depending on the settings specified in MAC_Timestamp_Control[TXTSSTSM]. The status bit TXTSC bit [15] in the MAC_Timestamp_Status register is set whenever the MAC transmitter captures the timestamp.

Diagram



Fields

Field	Function
31 TXTSSMIS	Transmit Timestamp Status Missed Indicates whether the transmit timestamp missed status is detected. If this field is 1, it indicates one of these conditions: • The timestamp of the current packet is ignored if MAC_Timestamp_Control[TXTSSTSM] = 0. • The timestamp of the previous packet is overwritten with the timestamp of the current packet if MAC_Timestamp_Control[TXTSSTSM] = 1. Access restrictions apply to this field that cleas on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
30-0 TXTSSLO	Transmit Timestamp Status Low Contains 31 bits of the Nanoseconds field of the transmit packet's captured timestamp.

75.17.158 MAC Transmit Timestamp Status In Seconds (MAC_Tx_Timestamp_Status_Seconds)

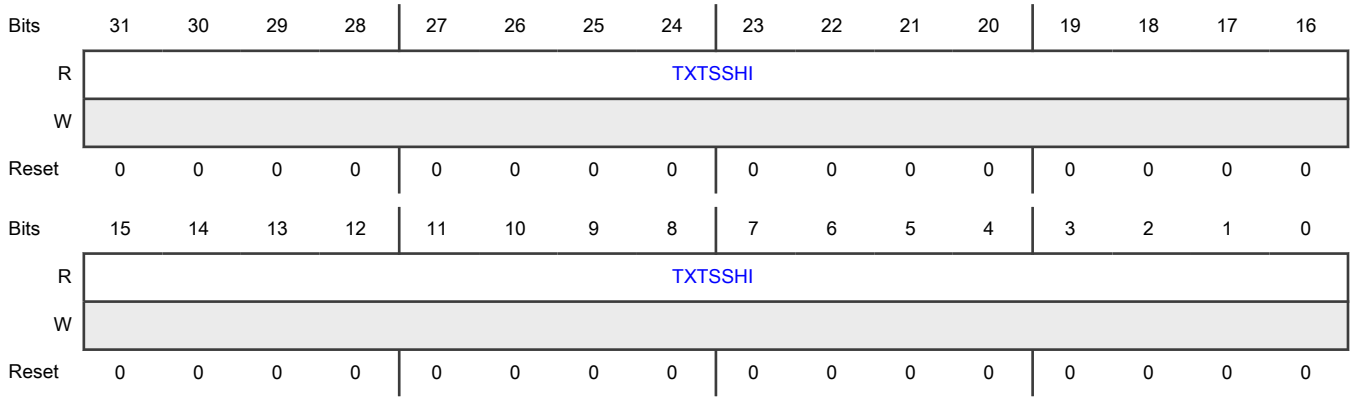
Offset

Register	Offset
MAC_Tx_Timestamp_Status_Seconds	B34h

Function

Contains the higher 32 bits of the timestamp (in seconds) captured when a PTP packet is transmitted.

Diagram



Fields

Field	Function
31-0 TXTSSHI	Transmit Timestamp Status High Contains the lower 32 bits of the Seconds field of the transmitted packet's captured timestamp.

75.17.159 MAC Timestamp Ingress Asymmetry Correction (MAC_Timestamp_Ingress_Asym_Corr)

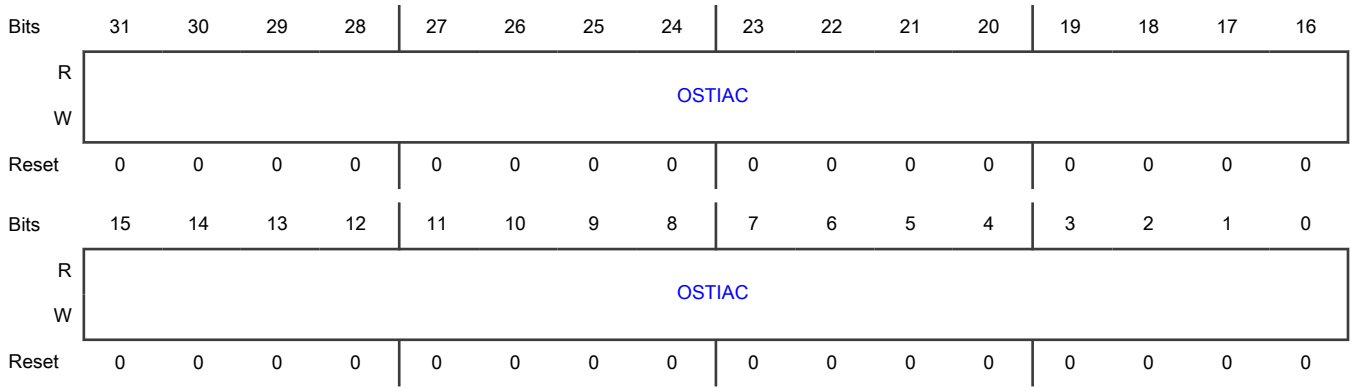
Offset

Register	Offset
MAC_Timestamp_Ingress_Asym_Corr	B50h

Function

Contains the ingress asymmetry correction value to be used when updating the correction field in the PDelay_Resp PTP messages.

Diagram



Fields

Field	Function
31-0 OSTIAC	One-Step Timestamp Ingress Asymmetry Correction Contains the ingress path asymmetry value to be added to the correctionField of the Pdelay_Resp PTP packet. The programmed value must be expressed in units of nanoseconds and multiplied by 2^16. For example, 2.5 ns is represented as 00028000h. This value can also be negative, which is represented in the two's-complement form with bit 31 representing the sign bit.

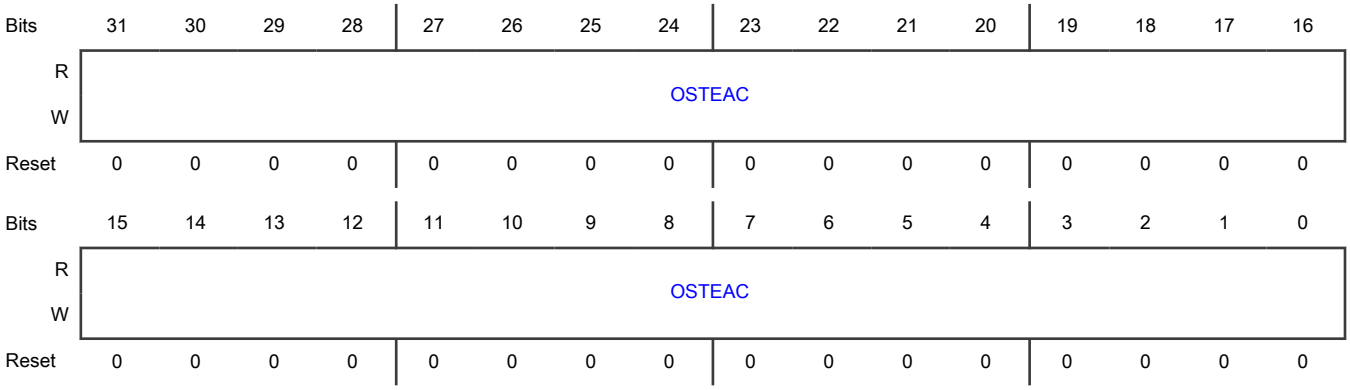
75.17.160 MAC Timestamp Egress Asymmetry Correction (MAC_Timestamp_Egress_Asym_Corr)

Offset

Register	Offset
MAC_Timestamp_Egress_Asym_Corr	B54h

Function
Contains the egress asymmetry correction value to be used when updating the correction field in the PDelay_Req PTP messages.

Diagram



Fields

Field	Function
31-0 OSTEAC	One-Step Timestamp Egress Asymmetry Correction Contains the egress path asymmetry value to be subtracted from the correctionField of the Pdelay_Resp PTP packet. The programmed value must be the negated value in units of nanoseconds multiplied by 2^16. For example, if the required correction is +2.5 ns, the programmed value must be FFFD_8000h, which is the two's-complement of 0002_8000h (2.5 * 216). Similarly, if the required correction is -3.3 ns, the programmed value is 0003_4CCCh (3.3 * 216).

75.17.161 MAC Timestamp Ingress Correction In Nanoseconds (MAC_Timestamp_Ingress_Corr_Nanosecond)

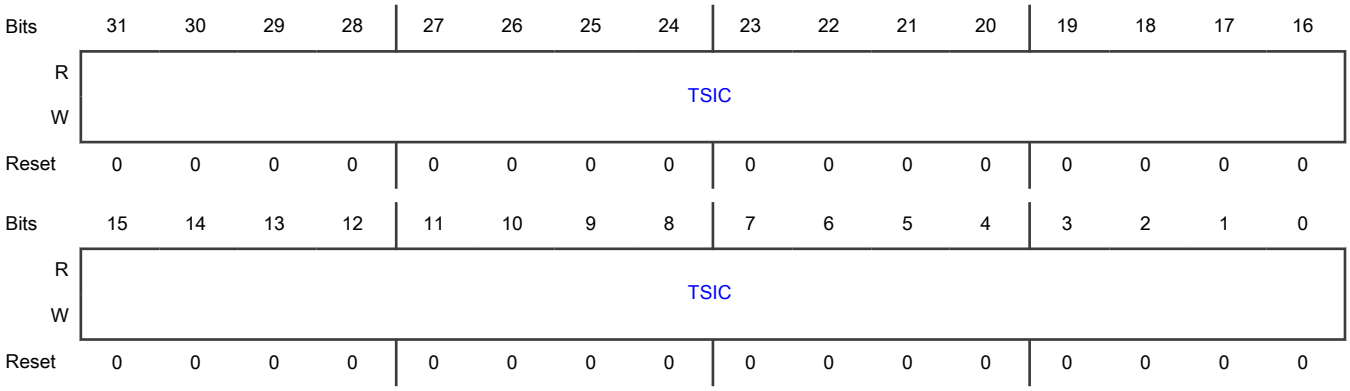
Offset

Register	Offset
MAC_Timestamp_Ingress_Corr_Nanosecond	B58h

Function

Contains the correction value, in nanoseconds, to be used with the captured timestamp value in the ingress path.

Diagram



Fields

Field	Function
31-0 TSIC	Timestamp Ingress Correction Contains the ingress path correction value as defined by the ingress correction expression.

75.17.162 MAC Timestamp Egress Correction In Nanoseconds (MAC_Timestamp_Egress_Corr_Nanosecond)

Offset

Register	Offset
MAC_Timestamp_Egress_Corr_Nanosecond	B5Ch

Function

Contains the correction value, in nanoseconds, to be used with the captured timestamp value in the egress path.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	TSEC															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	TSEC															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TSEC	Timestamp Egress Correction Contains the nanoseconds part of the egress path correction value as defined by the egress correction expression.

75.17.163 MAC Timestamp Ingress Correction In Subnanoseconds (MAC_Timestamp_Ingress_Corr_Subnanosec)

Offset

Register	Offset
MAC_Timestamp_Ingress_Corr_Subnanosec	B60h

Function

Contains the sub-nanosecond part of the correction value to be used with the captured timestamp value for ingress direction.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	TSICSNS								Reserved							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-8 TSICSNS	Timestamp Ingress Correction In Sub-Nanoseconds Contains the sub-nanoseconds part of the ingress path correction value as defined by the Ingress Correction expression.
7-0 —	Reserved

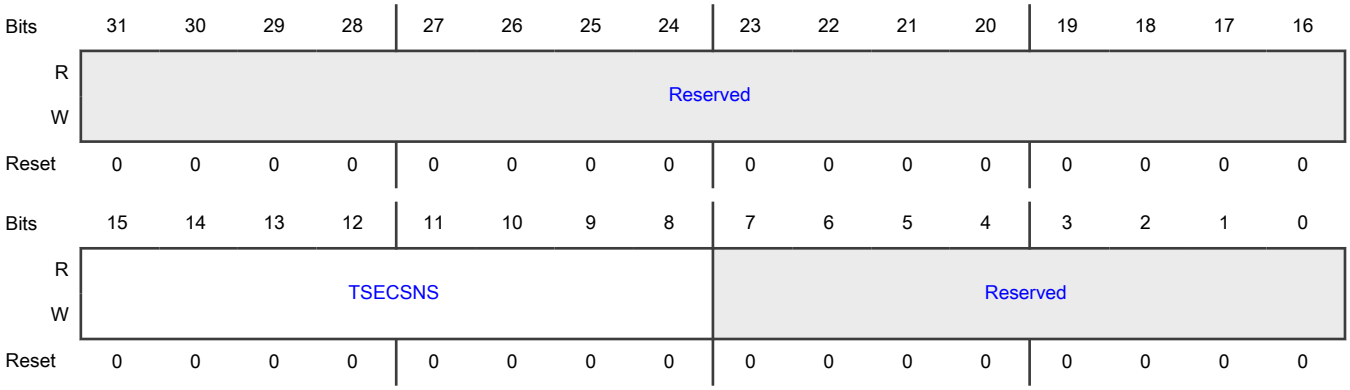
75.17.164 MAC Timestamp Engress Correction In Subnanoseconds (MAC_Timestamp_Egress_Corr_Subnanosec)

Offset

Register	Offset
MAC_Timestamp_Egress_Corr_Subnanosec	B64h

Function
Contains the sub-nanosecond part of the correction value to be used with the captured timestamp value for the egress direction.

Diagram



Fields

Field	Function
31-16	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
15-8 TSECSNS	Timestamp Egress Correction In Sub-Nanoseconds Contains the sub-nanoseconds part of the egress path correction value, as defined by the Egress Correction expression.
7-0 —	Reserved

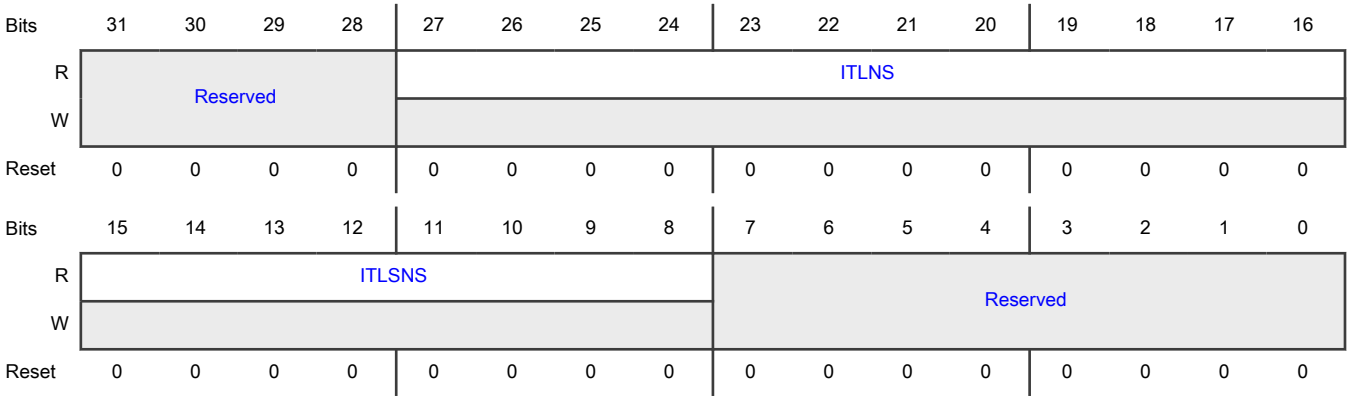
75.17.165 MAC Timestamp Ingress Latency (MAC_Timestamp_Ingress_Latency)

Offset

Register	Offset
MAC_Timestamp_Ingress_Latency	B68h

Function
Holds the ingress MAC latency.

Diagram



Fields

Field	Function
31-28 —	Reserved
27-16	Ingress Timestamp Latency In Sub-Nanoseconds

Table continues on the next page...

Table continued from the previous page...

Field	Function
ITLNS	Holds the average latency, in sub-nanoseconds, between the MAC input ports (PHY_RXD_I) and the actual point (GMII/MII), where the ingress timestamp is taken. The ingress correction value is computed as described in section 7.1.2.4.1 of QoS Databook.
15-8 ITLSNS	Ingress Timestamp Latency In Nanoseconds Holds the average latency, in nanoseconds, between the MAC input ports (PHY_RXD_I) and the actual point (GMII/MII), where the ingress timestamp is taken. The ingress correction value is computed as described in section 7.1.2.4.1 of QoS Databook.
7-0 —	Reserved

75.17.166 MAC Timestamp Egress Latency (MAC_Timestamp_Egress_Latency)

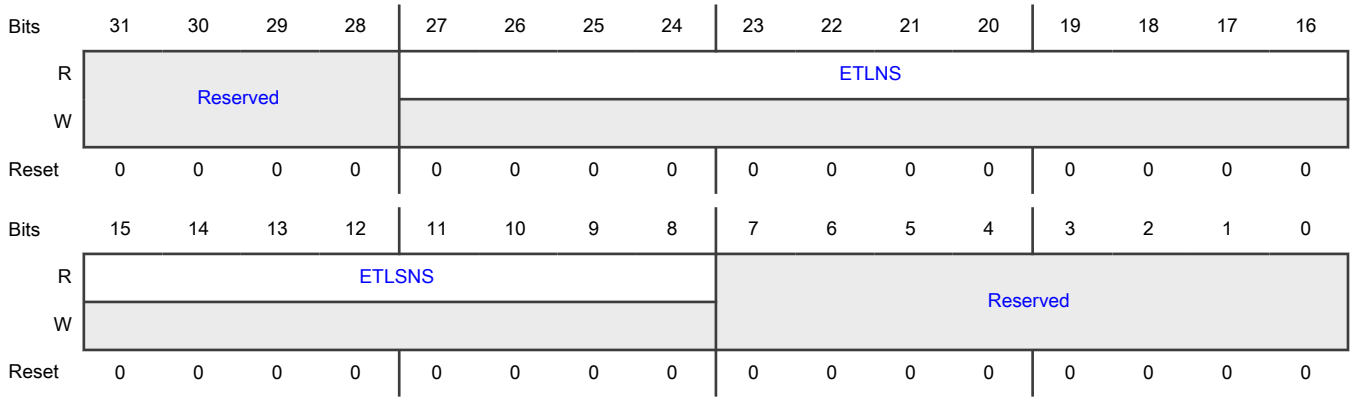
Offset

Register	Offset
MAC_Timestamp_Egress_Latency	B6Ch

Function

Holds the egress MAC latency.

Diagram



Fields

Field	Function
31-28 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
27-16 ETLNS	Egress Timestamp Latency In Nanoseconds Holds the average latency, in nanoseconds, between the actual point (GMII/MII) where the egress timestamp is taken and the MAC output ports (PHY_TXD_O). The ingress correction value is computed as described in section 7.1.2.4.2 of QoS Databook.
15-8 ETLSNS	Egress Timestamp Latency In Sub-Nanoseconds Holds the average latency, in sub-nanoseconds, between the actual point (GMII/MII) where the egress timestamp is taken and the MAC output ports (PHY_TXD_O). The ingress correction value is computed as described in section 7.1.2.4.2 of QoS Databook.
7-0 —	Reserved

75.17.167 MAC PPS Control (MAC_PPS_Control)

Offset

Register	Offset
MAC_PPS_Control	B70h

Function

Indicates PPS control.

In this register:

- Bits [30:24] are valid only when four flexible PPS outputs are selected.
- Bits [22:16] are valid only when three or more flexible PPS outputs are selected.
- Bits [14:8] are valid only when two or more flexible PPS outputs are selected.
- Bits [6:4] are valid only when the flexible PPS feature is selected.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MCGREN3	TRGTMODSEL3		Reserved	PPSCMD3				MCGREN2	TRGTMODSEL2		Reserved	PPSCMD2			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MCGREN1	TRGTMODSEL1		Reserved	PPSCMD1				MCGREN0	TRGTMODSEL0		PPSEN0	PPSCTRL_PPSCMD			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 MCGREN3	MCGR Mode Enable For PPS3 Output Enables the third PPS instance to operate in PPS or MCGR mode. If this field is 1, it operates in MCGR mode, and if the value is 0, it operates in PPS mode.
30-29 TRGTMODSEL3	Target Time Register Mode For PPS3 Output Indicates MAC PPS3 Target Time In Seconds (MAC_PPS3_Target_Time_Seconds) and MAC PPS3 Target Time In Nanoseconds (MAC_PPS3_Target_Time_Nanoseconds) mode for the PPS3 output signal. 00b - Target Time registers are programmed only for generating the interrupt event. The flexible PPS function must not be enabled in this mode, otherwise spurious transitions may be observed on the corresponding PTP_PPS_O output port. 01b - Reserved 10b - Target Time registers are programmed for generating the interrupt event and starting or stopping the PPS0 output signal generation. 11b - Target Time registers are programmed only for starting or stopping the PPS0 output signal generation. No interrupt is asserted.
28 —	Reserved
27-24 PPSCMD3	Flexible PPS3 Output Control Controls the flexible PPS3 output (PTP_PPS_O[3]) signal. The functioning of this field is similar to that of PPSCMD0[2:0] (presentation time control bits). If MCGREN3 = 1, PPSCMD3 that includes bits [27:24] are considered as presentation time control bits for media clock generation and recovery for comparator instance 3. If MCGREN3 is not 1, only three bits [26:24] are considered as PPSCMD3 and the fourth bit becomes 0. Access restrictions apply to this field that clears automatically. Writing 0 to it has no effect.
23 MCGREN2	MCGR Mode Enable For PPS2 Output Enables or disables the second PPS instance to operate in PPS or MCGR mode. If this field is 1, it operates in MCGR mode, and if the value is 0, it operates in PPS mode. 0b - Disabled 1b - Enabled
22-21 TRGTMODSEL2	Target Time Register Mode For PPS2 Output Indicates MAC PPS2 Target Time In Seconds (MAC_PPS2_Target_Time_Seconds) and MAC PPS2 Target Time In Nanoseconds (MAC_PPS2_Target_Time_Nanoseconds) mode for the PPS2 output signal. 00b - Target Time registers are programmed only for generating the interrupt event. The flexible PPS function must not be enabled in this mode, otherwise spurious transitions may be observed on the corresponding PTP_PPS_O output port. 01b - Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10b - Target Time registers are programmed for generating the interrupt event and starting or stopping the PPS0 output signal generation. 11b - Target Time registers are programmed only for starting or stopping the PPS0 output signal generation. No interrupt is asserted.
20 —	Reserved
19-16 PPSCMD2	Flexible PPS2 Output Control Controls the flexible PPS2 output (PTP_PPS_O[2]) signal. The functioning of this field is similar to that of the PPSCMD0 (presentation time control bits). If MCGREN2 = 1, the bits included in PPSCMD2 [19:16] are considered as presentation time control bits for media clock generation and recovery for comparator instance 2. If MCGREN2 is not 1, only three bits [18:16] are used as PPSCMD2 and the fourth bit becomes 0. Access restrictions apply to this field that clears automatically. Writing 0 to it has no effect.
15 MCGREN1	MCGR Mode Enable For PPS1 Output Enables or disables the first PPS instance to operate in PPS or MCGR mode. If this field is 1, it operates in MCGR mode, and if the value is 0, it operates in PPS mode. 0b - Disabled 1b - Enabled
14-13 TRGTMODSEL1	Target Time Register Mode For PPS1 Output Indicates MAC PPS1 Target Time In Seconds (MAC_PPS1_Target_Time_Seconds) and MAC PPS1 Target Time In Nanoseconds (MAC_PPS1_Target_Time_Nanoseconds) mode for the PPS1 output signal. 00b - Target Time registers are programmed only for generating the interrupt event. The flexible PPS function must not be 1 in this mode, otherwise spurious transitions may be observed on the corresponding PTP_PPS_O output port. 01b - Reserved 10b - Target Time registers are programmed for generating the interrupt event and starting or stopping the PPS0 output signal generation. 11b - Target Time registers are programmed only for starting or stopping the PPS0 output signal generation. No interrupt is asserted.
12 —	Reserved
11-8 PPSCMD1	Flexible PPS1 Output Control Controls the flexible PPS1 output (PTP_PPS_O[1]) signal. The functioning of this field is similar to that of the PPSCMD0 field.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	If MCGREN1 = 1, the bits included in PPSCMD1 [11:8] are considered as presentation time control bits for media clock generation and recovery for comparator instance 1. The functioning of PPSCMD1 is similar to that of the PPSCMD0 presentation time control bits. If MCGREN1 is not 1, only three bits [10:8] are used as PPSCMD1 and the fourth bit becomes 0. Access restrictions apply to this field that clears automatically. Writing 0 to it has no effect.
7 MCGREN0	MCGR Mode Enable For PPS0 Output Indicates whether the 0th PPS instance is enabled to operate in PPS or MCGR mode. If this field is 1, it operates in MCGR mode, and if the value is 0, it operates in PPS mode. 0b - PPS mode 1b - MCGR mode
6-5 TRGTMODSEL0	Target Time Register Mode For PPS0 Output Indicates MAC PPS0 Target Time In Seconds (MAC_PPS0_Target_Time_Seconds) and MAC PPS0 Target Time In Nanoseconds (MAC_PPS0_Target_Time_Nanoseconds) mode for the PPS0 output signal. 00b - Target Time registers are programmed only for generating the interrupt event. The flexible PPS function must not be 1 in this mode, otherwise spurious transitions may be observed on the corresponding PTP_PPS_O output port. 01b - Reserved 10b - Target Time registers are programmed for generating the interrupt event and starting or stopping the PPS0 output signal generation. 11b - Target Time registers are programmed only for starting or stopping the PPS0 output signal generation. No interrupt is asserted.
4 PPSEN0	Flexible PPS Output Mode Enable 0 Indicates the status of Flexible PPS Output mode. • If this field is 1, the PPSCTRL_PPSCMD field functions as PPSCMD. • If this field is 0, the PPSCTRL_PPSCMD field functions as PPSCTRL (fixed PPS mode). 0b - Disabled 1b - Enabled
3-0 PPSCTRL_PPSCMD	PPS Output Frequency Control Controls the frequency of the PPS0 output (PTP_PPS_O[0] signal). The default value of this field is 0, and the PPS output is 1 pulse (of width clk_ptp_i) every second. For the other values of PPSCTRL, the PPS output becomes a generated clock having these frequencies: • 0001: The binary rollover is 2 Hz, and the digital rollover is 1 Hz. • 0010: The binary rollover is 4 Hz, and the digital rollover is 2 Hz. • 0011: The binary rollover is 8 Hz, and the digital rollover is 4 Hz. • 0100: The binary rollover is 16 Hz, and the digital rollover is 8 Hz.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• .. • 1111: The binary rollover is 32.768 kHz and the digital rollover is 16.384 kHz. **NOTE** In Binary Rollover mode, the PPS output (PTP_PPS_O) has a duty cycle of 50 percent with these frequencies. In Digital Rollover mode, the PPS output frequency is an average number. The actual clock has a different frequency that is synchronized every second. For example, • If PPSCTRL = 0001, PPS (1 Hz) has a low period of 537 ms and a high period of 463 ms. • If PPSCTRL = 0010, PPS (2 Hz) is a sequence of one clock of 50 percent duty cycle and a 537 ms period. The second clock has a period of 463 ms (268 ms low and 195 ms high). • If PPSCTRL = 0011, PPS (4 Hz) is a sequence of three clocks of 50 percent duty cycle and a 268 ms period, and the fourth clock of 195 ms period (134 ms low and 61 ms high). This behavior is a result of the non-linear toggling of bits in Digital Rollover mode in MAC System Time In Nanoseconds (MAC_System_Time_Nanoseconds) or flexible PPS output (PTP_PPS_O[0]) control. Programming these bits with a non-zero value instructs MAC to initiate an event. When the command is transferred or synchronized with the PTP clock domain, these bits automatically become 0. You must ensure that these bits are programmed only when they are "all-zero." The following list describes the values of PPSCMD0. • 0000: No command • 0001: Start a single pulse. This command generates a single pulse rising at the start point defined in MAC PPS0 Target Time In Seconds (MAC_PPS0_Target_Time_Seconds) and MAC PPS0 Target Time In Nanoseconds (MAC_PPS0_Target_Time_Nanoseconds), for a duration defined in MAC PPS0 Width (MAC_PPS0_Width). • 0010: Start pulse train. This command generates the train of pulses rising at the start point defined in the Target Time registers and for a duration defined in MAC PPS0 Width (MAC_PPS0_Width), repeated at an interval defined in the PPS Interval registers. By default, the PPS pulse train runs freely unless the "stop pulse train at time" or "stop pulse train immediately" command stops it. • 0011: Cancel start. This command cancels the start single pulse and start pulse train commands if the system time has not crossed the programmed start time. • 0100: Stop pulse train at time. This command stops the train of pulses that the start pulse train command (PPSCMD = 0010) initiates after the time programmed in the Target Time registers elapses. • 0101: Stop pulse train immediately. This command immediately stops the train of pulses that the start pulse train command (PPSCMD = 0010) initiates. • 110: Cancel stop pulse train. This command cancels the stop pulse train at the time command if the programmed stop time has not elapsed. The PPS pulse train runs freely if the command executes successfully. • 0111-1111: Reserved or presentation time control. If MAC_PPS_Control[MCGREN0] = 0, then these are treated as presentation time control bits. This list describes the values of PPSCMD0:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• 0000: The MCGR operation is not carried out. If set to this value in the mid of clock recovery or generation, all the processing inputs are flushed. • 0001: Captures the presentation time at the rising edge of MCG_PST_TRIG_I[0] in MAC PPS0 Target Time In Seconds (MAC_PPS0_Target_Time_Seconds) • 0010: Captures the presentation time at the falling edge of MCG_PST_TRIG_I[0] in MAC PPS0 Target Time In Seconds (MAC_PPS0_Target_Time_Seconds) • 0011: Captures the presentation time at both the edges of MCG_PST_TRIG_I[0] in MAC PPS0 Target Time In Seconds (MAC_PPS0_Target_Time_Seconds) • 0100-1000: Reserved • 1001: Toggle output on compare • 1010: Pulse output low on compare for one PTP-clock cycle • 1011: Pulse output high on compare for one PTP-clock cycle • 1100-1111: Reserved

75.17.168 MAC PPS0 Target Time In Seconds (MAC_PPS0_Target_Time_Seconds)

Offset

Register	Offset
MAC_PPS0_Target_Time_Seconds	B80h

Function

Is used, along with the PPS Target Time Nanoseconds register, to schedule an interrupt event (MAC_Timestamp_Status[TSTARGT0]) when the system time exceeds the value programmed in these registers.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	TSTRH0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	TSTRH0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TSTRH0	PPS Target Time In Seconds Register Stores the target time in seconds. When the timestamp value matches or exceeds both the Target Timestamp registers, MAC starts or stops the PPS signal output and generates an interrupt (if enabled), based on Target Time mode selected for the corresponding PPS output in MAC PPS Control (MAC_PPS_Control). If DWC_EQOS_FLEXI_PPS_OUT_EN is enabled in the configuration and MAC_Timestamp_Control[PTGE] = 1, with the presentation time control set in Recovery mode, these fields indicate that the application programs the TPT. In Generation mode, it indicates that CPT is generated at the sampled trigger.

75.17.169 MAC PPS0 Target Time In Nanoseconds (MAC_PPS0_Target_Time_Nanoseconds)

Offset

Register	Offset
MAC_PPS0_Target_Time_Nanoseconds	B84h

Function
Indicates the target time in nanoseconds for PPS0.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TRGTBUS...				TTSL0											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TTSL0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 TRGTBUSY0	PPS Target Time Busy Status 0 Indicates whether the PPS target time busy status 0 is detected.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• MAC writes 1 to this field when MAC_PPS_Control[PPSCTRL_PPSCMD] is programmed to 010 or 011. This instructs MAC to synchronize the Target Time registers with the PTP clock domain. • MAC writes 0 to this field after synchronizing the Target Time registers with the PTP clock domain. The application must not update the Target Time registers when this field is read as 1. Otherwise, the synchronization of the previously programmed time is corrupted. 0b - Not detected 1b - Detected
30-0 TTSL0	Target Time Low For PPS0 Stores the time in (signed) nanoseconds. If the value of the timestamp matches the value in both the Target Timestamp registers, MAC starts or stops the PPS signal output and generates an interrupt, if enabled, based on MAC_PPS_Control[TRGTMODSEL0]. • If MAC_Timestamp_Control[TSCTRLSSR] = 0, this value must be calculated as (time in ns / 0.465). The actual start or stop time of the PPS signal output might have an error margin up to one unit of the sub-second increment value. • If MAC_Timestamp_Control[TSCTRLSSR] = 1, this value must not exceed 3B9A_C9FFh. The actual start or stop time of the PPS signal output might have an error margin up to one unit of the sub-second increment value. Access restriction apply to this field that clears automatically. Writing 0 to it has no effect.

75.17.170 MAC PPS0 Interval (MAC_PPS0_Interval)

Offset

Register	Offset
MAC_PPS0_Interval	B88h

Function

Contains the number of units of the sub-second increment value between the rising edges of the PPS0 signal output (PTP_PPS_O[0]).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PPSINT0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PPSINT0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PPSINT0	PPS Output Signal Interval 0 Stores the interval between the rising edges of the PPS0 signal output. The interval is stored in terms of the number of units of the sub-second increment value. You must program one value less than the required interval. For example, if the PTP reference clock is 50 MHz (period of 20 ns), and the desired interval between the rising edges of the PPS0 signal output is 100 ns (that is, five units of the sub-second increment value), you must program value 4 (5-1) in this register.

75.17.171 MAC PPS0 Width (MAC_PPS0_Width)

Offset

Register	Offset
MAC_PPS0_Width	B8Ch

Function

Contains the number of units of the sub-second increment value between the rising and corresponding falling edges of the PPS0 signal output (PTP_PPS_O[0]).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PPSWIDTH0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PPSWIDTH0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PPSWIDTH0	PPS Output Signal Width 0 Stores the width between the rising and corresponding falling edges of the PPS0 signal output. The width is stored in terms of the number of units of the sub-second increment value. You must program one value less than the required interval. For example, if the PTP reference clock has a frequency of 50 MHz (period of 20 ns), and a width between the rising and corresponding falling edges of the PPS0 signal output is 80 ns (that is, four units of the sub-second increment value), you should program value 3 (4-1) in this register. **NOTE** The value programmed in this register must be lesser than the value programmed in MAC PPS0 Interval (MAC_PPS0_Interval).

75.17.172 MAC PPS1 Target Time In Seconds (MAC_PPS1_Target_Time_Seconds)

Offset

Register	Offset
MAC_PPS1_Target_Time_Seconds	B90h

Function
Is used to schedule an interrupt event (MAC_Timestamp_Status[TSTARGT0]), along with the PPS Target Time Nanoseconds register, when the system time exceeds the value programmed in these registers.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	TSTRH1															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	TSTRH1															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0	PPS Target Time In Seconds 1

Table continues on the next page...

Field	Function
TSTRH1	Stores the target time in seconds. When the timestamp value matches or exceeds both the Target Timestamp registers, MAC starts or stops the PPS signal output and generates an interrupt, if enabled, based on Target Time mode selected for the corresponding PPS output in MAC PPS Control (MAC_PPS_Control). If DWC_EQOS_FLEXI_PPS_OUT_EN is enabled in the configuration and MAC_Timestamp_Control[PTGE] = 1, with the presentation time control set in Recovery mode, these fields indicate that the application programs the TPT. In Generation mode, it indicates that CPT is generated at the sampled trigger.

75.17.173 MAC PPS1 Target Time In Nanoseconds (MAC_PPS1_Target_Time_Nanoseconds)

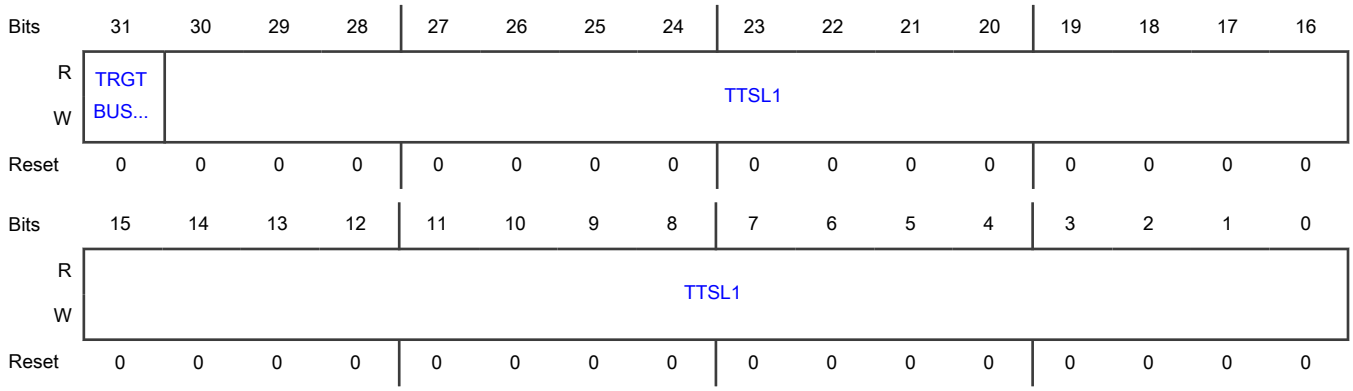
Offset

Register	Offset
MAC_PPS1_Target_Time_Nanoseconds	B94h

Function

Indicates the target time in nanoseconds for PPS1.

Diagram



Fields

Field	Function
31 TRGTBUSY1	PPS Target Time Busy Status 1 Indicates whether the PPS target time busy status 1 is detected. • MAC writes 1 to this field when MAC_PPS_Control[PPSCTRL_PPSCMD] is programmed to 010 or 011. Programming the PPSCMD0 field as 010 or 011 instructs MAC to synchronize the Target Time registers with the PTP clock domain.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• MAC writes 0 to this field after synchronizing the Target Time registers with the PTP clock domain. The application must not update the Target Time registers when this field is read as 1. Otherwise, the synchronization of the previously programmed time is corrupted. 0b - Not detected 1b - Detected
30-0 TTSL1	Target Time Low For PPS1 Stores the time in (signed) nanoseconds. If the value of the timestamp matches the value in both the Target Timestamp registers, MAC starts or stops the PPS signal output and generates an interrupt, if enabled, based on MAC_PPS_Control[TRGTMODSEL0]. • If MAC_Timestamp_Control[TSCTRLSSR] = 0, this value must be defined as (time in ns / 0.465). The actual start or stop time of the PPS signal output might have an error margin up to one unit of the sub-second increment value. • If MAC_Timestamp_Control[TSCTRLSSR] = 1, this value must not exceed 3B9A_C9FFh. The actual start or stop time of the PPS signal output might have an error margin up to one unit of the sub-second increment value. Access restriction apply to this field that clears automatically. Writing 0 to it has no effect.

75.17.174 MAC PPS1 Interval (MAC_PPS1_Interval)

Offset

Register	Offset
MAC_PPS1_Interval	B98h

Function

Contains the number of units of the sub-second increment value between the rising edges of the PPS0 signal output (PTP_PPS_O[0]).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PPSINT1															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PPSINT1															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PPSINT1	PPS Output Signal Interval 1 Stores the interval between the rising edges of the PPS0 signal output. The interval is stored in terms of the number of units of the sub-second increment value. You must program one value less than the required interval. For example, if the PTP reference clock is 50 MHz (period of 20 ns), and the desired interval between the rising edges of the PPS0 signal output is 100 ns (that is, five units of the sub-second increment value), you must program value 4 (5-1) in this register.

75.17.175 MAC PPS1 Width (MAC_PPS1_Width)

Offset

Register	Offset
MAC_PPS1_Width	B9Ch

Function

Contains the number of units of the sub-second increment value between the rising and corresponding falling edges of the PPS0 signal output (PTP_PPS_O[0]).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PPSWIDTH1															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PPSWIDTH1															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PPSWIDTH1	PPS Output Signal Width 1 Stores the width between the rising and corresponding falling edges of the PPS0 signal output. The width is stored in terms of the number of units of the sub-second increment value. You must program one value less than the required interval. For example, if the PTP reference clock has a frequency of 50 MHz (period of 20 ns), and a width between the rising and corresponding falling edges of the PPS0 signal output is 80 ns (that is, four units of the sub-second increment value), you should program value 3 (4-1) in this register.

Table continues on the next page...

Field	Function
	NOTE The value programmed in this register must be lesser than the value programmed in MAC PPS0 Interval (MAC_PPS0_Interval).

75.17.176 MAC PPS2 Target Time In Seconds (MAC_PPS2_Target_Time_Seconds)

Offset

Register	Offset
MAC_PPS2_Target_Time_Seconds	BA0h

Function

Is used to schedule an interrupt event (MAC_Timestamp_Status[TSTARGT0]), along with the PPS Target Time Nanoseconds register, when the system time exceeds the value programmed in these registers.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	TSTRH2															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	TSTRH2															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TSTRH2	PPS Target Time In Seconds 2 Stores the target time in seconds. When the timestamp value matches or exceeds both the Target Timestamp registers, MAC starts or stops the PPS signal output and generates an interrupt, if enabled, based on Target Time mode selected for the corresponding PPS output in MAC PPS Control (MAC_PPS_Control). If DWC_EQOS_FLEXI_PPS_OUT_EN is enabled in the configuration and MAC_Timestamp_Control[PTGE] = 1, with the presentation time control set in Recovery mode, these fields indicate that the application programs the TPT. In Generation mode, it indicates that CPT is generated at the sampled trigger.

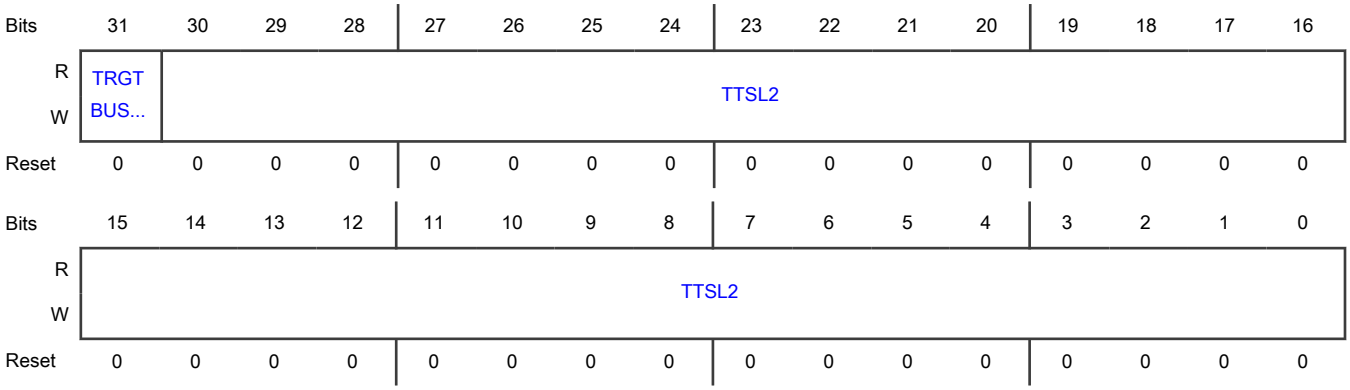
75.17.177 MAC PPS2 Target Time In Nanoseconds (MAC_PPS2_Target_Time_Nanoseconds)

Offset

Register	Offset
MAC_PPS2_Target_Time_Nanoseconds	BA4h

Function
Indicates the target time in nanoseconds for PPS2.

Diagram



Fields

Field	Function
31 TRGTBUSY2	PPS Target Time Busy Status 2 Indicates whether the PPS target time busy status 2 is detected. • MAC writes 1 to this field when MAC_PPS_Control[PPSCTRL_PPSCMD] is programmed to 010 or 011. This instructs MAC to synchronize the Target Time registers with the PTP clock domain. • MAC writes 0 to this field after synchronizing the Target Time registers with the PTP clock domain. The application must not update the Target Time registers when this field is read as 1. Otherwise, the synchronization of the previously programmed time is corrupted. 0b - Not detected 1b - Detected
30-0 TTSL2	Target Time Low For PPS2 Stores the time in (signed) nanoseconds. If the value of the timestamp matches the value in both the Target Timestamp registers, MAC starts or stops the PPS signal output and generates an interrupt, if enabled, based on MAC_PPS_Control[TRGTMODSEL0].

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• If MAC_Timestamp_Control[TSCTRLSSR] = 0, this value must be defined as (time in ns / 0.465). The actual start or stop time of the PPS signal output might have an error margin up to one unit of the sub-second increment value. • If MAC_Timestamp_Control[TSCTRLSSR] = 1, this value must not exceed 3B9A_C9FFh. The actual start or stop time of the PPS signal output might have an error margin up to one unit of the sub-second increment value. Access restriction apply to this field that clears automatically. Writing 0 to it has no effect.

75.17.178 MAC PPS2 Interval (MAC_PPS2_Interval)

Offset

Register	Offset
MAC_PPS2_Interval	BA8h

Function

Contains the number of units of the sub-second increment value between the rising edges of the PPS0 signal output (PTP_PPS_O[0]).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PPSINT2															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PPSINT2															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PPSINT2	PPS Output Signal Interval 2 Stores the interval between the rising edges of the PPS0 signal output. The interval is stored in terms of the number of units of the sub-second increment value. You must program one value less than the required interval. For example, if the PTP reference clock is 50 MHz (period of 20 ns), and the desired interval between the rising edges of the PPS0 signal output is 100 ns (that is, five units of the sub-second increment value), you must program value 4 (5-1) in this register.

75.17.179 MAC PPS2 Width (MAC_PPS2_Width)

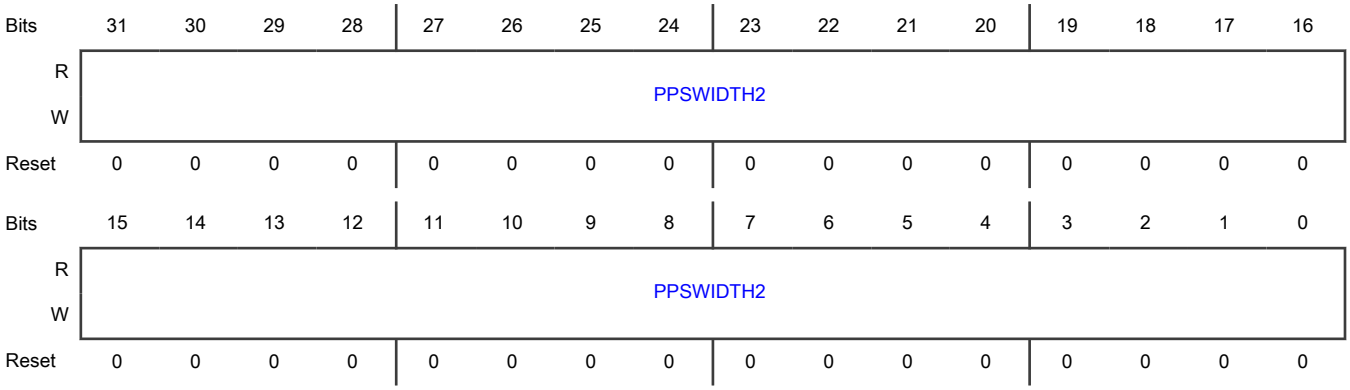
Offset

Register	Offset
MAC_PPS2_Width	BACh

Function

Contains the number of units of the sub-second increment value between the rising and corresponding falling edges of the PPS0 signal output (PTP_PPS_O[0]).

Diagram



Fields

Field	Function
31-0 PPSWIDTH2	PPS Output Signal Width 2 Stores the width between the rising and corresponding falling edges of the PPS0 signal output. The width is stored in terms of the number of units of the sub-second increment value. You must program one value less than the required interval. For example, if the PTP reference clock has a frequency of 50 MHz (period of 20 ns), and a width between the rising and corresponding falling edges of the PPS0 signal output is 80 ns (that is, four units of the sub-second increment value), you should program value 3 (4-1) in this register. **NOTE** The value programmed in this register must be lesser than the value programmed in MAC PPS0 Interval (MAC_PPS0_Interval).

75.17.180 MAC PPS3 Target Time In Seconds (MAC_PPS3_Target_Time_Seconds)

Offset

Register	Offset
MAC_PPS3_Target_Time_Seconds	BB0h

Function

Is used to schedule an interrupt event (MAC_Timestamp_Status[TSTARGT0]), along with the PPS Target Time Nanoseconds register, when the system time exceeds the value programmed in these registers.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	TSTRH3															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	TSTRH3															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TSTRH3	PPS Target Time In Seconds 3 Stores the target time in seconds. When the timestamp value matches or exceeds both the Target Timestamp registers, MAC starts or stops the PPS signal output and generates an interrupt, if enabled, based on Target Time mode selected for the corresponding PPS output in MAC PPS Control (MAC_PPS_Control). If DWC_EQOS_FLEXI_PPS_OUT_EN is enabled in the configuration and MAC_Timestamp_Control[PTGE] = 1, with the presentation time control set in Recovery mode, these fields indicate that the application programs the TPT. In Generation mode, it indicates that CPT is generated at the sampled trigger.

75.17.181 MAC PPS3 Target Time In Nanoseconds (MAC_PPS3_Target_Time_Nanoseconds)

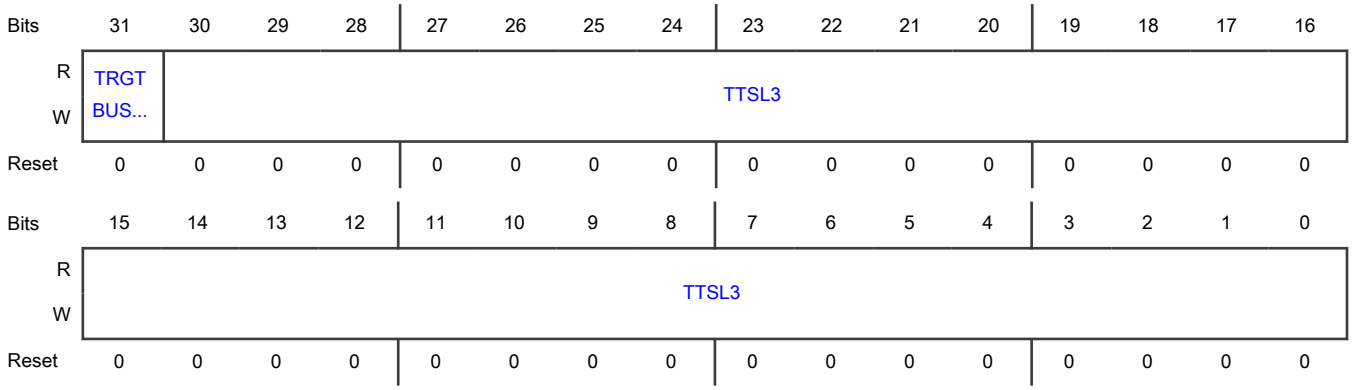
Offset

Register	Offset
MAC_PPS3_Target_Time_Nanoseconds	BB4h

Function

Indicates the target time in nanoseconds for PPS3.

Diagram



Fields

Field	Function
31 TRGTBUSY3	PPS Target Time Register Busy 3 Indicates whether the PPS target time register busy status is detected. MAC writes 1 to this field when MAC_PPS_Control[PPSCTRL_PPSCMD] is programmed to 010 or 011. This instructs MAC to synchronize the Target Time registers with the PTP clock domain. MAC writes 0 to this field after synchronizing the Target Time registers with the PTP clock domain. The application must not update the Target Time registers when this field is read as 1. Otherwise, the synchronization of the previously programmed time is corrupted. 0b - Not detected 1b - Detected
30-0 TTSL3	Target Time Low For PPS3 Stores the time in (signed) nanoseconds. If the value of the timestamp matches the value in both the Target Timestamp registers, MAC starts or stops the PPS signal output and generates an interrupt, if enabled, based on MAC_PPS_Control[TRGTMODSEL0]. • If MAC_Timestamp_Control[TSCTRLSSR] = 0, this value must be defined as (time in ns / 0.465). The actual start or stop time of the PPS signal output might have an error margin up to one unit of the sub-second increment value. • If MAC_Timestamp_Control[TSCTRLSSR] = 1, this value must not exceed 3B9A_C9FFh. The actual start or stop time of the PPS signal output might have an error margin up to one unit of the sub-second increment value. Access restriction apply to this field that clears automatically. Writing 0 to it has no effect.

75.17.182 MAC PPS3 Interval (MAC_PPS3_Interval)

Offset

Register	Offset
MAC_PPS3_Interval	BB8h

Function

Contains the number of units of the sub-second increment value between the rising edges of the PPS0 signal output (PTP_PPS_O[0]).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PPSINT3															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PPSINT3															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PPSINT3	PPS Output Signal Interval Stores the interval between the rising edges of the PPS0 signal output. The interval is stored in terms of the number of units of the sub-second increment value. You must program one value less than the required interval. For example, if the PTP reference clock is 50 MHz (period of 20 ns), and the desired interval between the rising edges of the PPS0 signal output is 100 ns (that is, five units of the sub-second increment value), you must program value 4 (5-1) in this register.

75.17.183 MAC PPS3 Width (MAC_PPS3_Width)

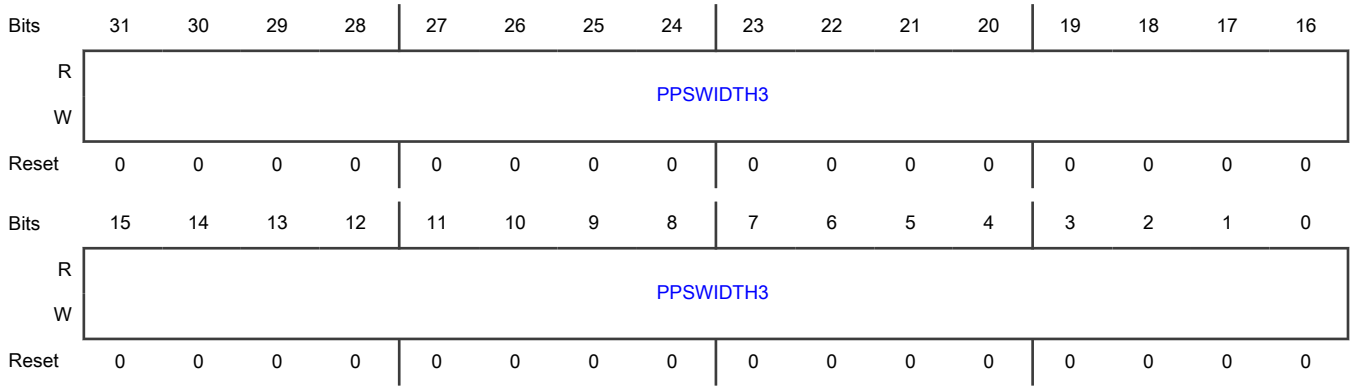
Offset

Register	Offset
MAC_PPS3_Width	BBCh

Function

Contains the number of units of the sub-second increment value between the rising and corresponding falling edges of the PPS0 signal output (PTP_PPS_O[0]).

Diagram



Fields

Field	Function
31-0 PPSWIDTH3	PPS Output Signal Width 3 Stores the width between the rising and corresponding falling edges of the PPS0 signal output. The width is stored in terms of the number of units of the sub-second increment value. You must program one value less than the required interval. For example, if the PTP reference clock has a frequency of 50 MHz (period of 20 ns), and a width between the rising and corresponding falling edges of the PPS0 signal output is 80 ns (that is, four units of the sub-second increment value), you should program value 3 (4-1) in this register. **NOTE** The value programmed in this register must be lesser than the value programmed in MAC PPS0 Interval (MAC_PPS0_Interval).

75.17.184 MTL Operation Mode (MTL_Operation_Mode)

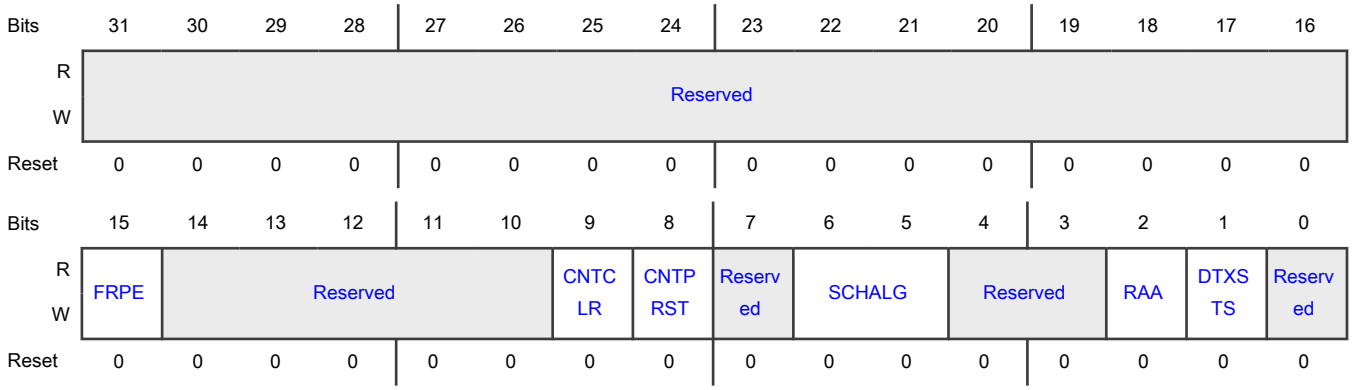
Offset

Register	Offset
MTL_Operation_Mode	C00h

Function

Establishes the transmit and receive operating modes and commands.

Diagram



Fields

Field	Function
31-16 —	Reserved
15 FRPE	Flexible Receive Parser Enable Indicates the status of the flexible receive parser. • If this field is 1, the programmable receive parser functionality is enabled. • If the receive parser is disabled and is in the middle of parsing, the receive parser functionality is disabled only after completing the current packet parsing. When the receive parser is enabled from the disabled state, the receive parser is activated for the next immediate packet. 0b - Disabled 1b - Enabled
14-10 —	Reserved
9 CNTCLR	Counters Reset Indicates whether the counters are reset. • If this field is 1, all the counters are reset. The field clears automatically after one clock cycle. • If you write 1 to this field along with the CNTPRST field, the CNT_PRESET field takes precedence. Access restrictions apply to this field that clears automatically. Writing 0 to it has no effect. 0b - Not reset 1b - Reset
8 CNTPRST	Counters Preset Indicates the status of preset counters. If this field is 1,

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• MTL Tx Queue 0 Underflow (MTL_TxQ0_Underflow)[bits 0-7] is initialized or preset to 12'h7F0. • Missed packet and overflow packet counters in MTL Rx Queue Missed Packet Overflow Count (MTL_RxQ0_Missed_Packet_Overflow_Cnt)[bits 0-7] are initialized or preset to 12'h7F0. Access restrictions apply to this field that clears automatically. Writing 0 to it has no effect. 0b - Disabled 1b - Enabled
7 —	Reserved
6-5 SCHALG	Transmit Scheduling Algorithm Indicates the algorithm for transmit scheduling. 00b - WRR algorithm 01b - WFQ algorithm when DCB feature selected; otherwise, reserved 10b - DWRR algorithm when DCB feature selected; otherwise, reserved 11b - Strict priority algorithm
4-3 —	Reserved
2 RAA	Receive Arbitration Algorithm Is used to select the arbitration algorithm for the receive side. Queue 0 has the lowest priority and the last queue has the highest priority. 0b - Strict priority (SP) 1b - Weighted strict priority (WSP)
1 DTXSTS	Drop Transmit Status Indicates whether the drop transmit status is enabled. • If this field is 1, the transmit packet status received from MAC is dropped in MTL. • If this field is 0, the transmit packet status received from MAC is forwarded to the application. 0b - Disabled 1b - Enabled
0 —	Reserved

75.17.185 MTL Debug Control (MTL_DBG_CTL)

Offset

Register	Offset
MTL_DBG_CTL	C08h

Function

Controls, along with MTL Debug Status (MTL_DBG_STS), the operation mode of FIFO debug access.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved													EIEC		EIEE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	STSIE	PKTIE	FIFOSEL		FIFOWREN	FIFORDEN	RSTSEL	RSTALL	Reserved	PKTSTATE		Reserved	BYTEEN		DBGMOD	FDBGEN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-19 —	Reserved
18-17 EIEC	ECC Inject Error Control Indicates the status of the ECC inject error control feature for transmit, receive, and TSO memories. If EIEE = 1, following are the errors inserted based on the value encoded in this field. 00b - 1-bit error 01b - 2-bit errors 10b - 3-bit errors 11b - 1-bit error in the address field
16 EIEE	ECC Inject Error Enable Indicates the status of the ECC error injection feature for transmit, receive, and TSO memories. • If the value if this field is 1, it enables the ECC error injection feature. • If the value if this field is 0, it disables the ECC error injection feature. 0b - Disabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Enabled
15 STSIE	Transmit Status Available Interrupt Status Enable Indicates whether the transmit packet available interrupt status is enabled. If this field is 1, an interrupt is generated when transmit status is available in Slave mode. 0b - Disabled 1b - Enabled
14 PKTIE	Receive Packet Available Interrupt Status Enable Indicates whether the receive packet available interrupt status is enabled. If this field is 1, an interrupt is generated when EOP of the received packet is written to the receive FIFO. 0b - Disabled 1b - Enabled
13-12 FIFOSEL	FIFO Selected for Access Indicates the FIFO selected for debug access. 00b - Transmit FIFO 01b - Transmit status FIFO (only read access when SLVMOD is set) 10b - TSO FIFO (cannot be accessed when SLVMOD is set) 11b - Receive FIFO
11 FIFOWREN	FIFO Write Enable Indicates the status of FIFO write. If this field is 1, it enables the write operation on the selected FIFO when FIFO debug access is enabled. This field must not be written to 1 when FIFO debug access is disabled, that is, when FDBGEN = 0. Access restrictions apply to this field that clears automatically. Also, writing 0 to the field clears it and writing 1 sets it. 0b - Disabled 1b - Enabled
10 FIFORDEN	FIFO Read Enable Indicates the status of FIFO read. If this field is 1, it enables the read operation on the selected FIFO when FIFO debug access is enabled. This field must not be written to 1 when FIFO debug access is disabled, that is, when FDBGEN = 0. Access restrictions apply to this field that clears automatically. Also, writing 0 to the field clears it and writing 1 sets it. 0b - Disabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Enabled
9 RSTSEL	Reset Pointers Of Selected FIFO Indicates whether the resetting of pointers for the selected FIFO is enabled. If this field is 1, the pointers of the currently selected FIFO are reset when FIFO debug access is enabled. This field must not be written to 1 when FIFO debug access is disabled, that is, when FDBGEN = 0. Access restrictions apply to this field that clears automatically. Also, writing 0 to the field clears it and writing 1 sets it. 0b - Disabled 1b - Enabled
8 RSTALL	Reset All Pointers Indicates whether the resetting of all the pointers is enabled. If this field is 1, the pointers of all the FIFOs are reset when the FIFO debug access is enabled. This field must not be written to 1 when FIFO debug access is disabled, that is, when FDBGEN = 0. Access restrictions apply to this field that clears automatically. Also, writing 0 to the field clears it and writing 1 sets it. 0b - Disabled 1b - Enabled
7 —	Reserved
6-5 PKTSTATE	Encoded Packet State Is used to write the control information to the transmit FIFO or receive FIFO. These are the values related to the transmit FIFO: • 00: Packet data • 01: Control word • 10: SOP data • 11: EOP data These are the values related to the receive FIFO: • 00: Packet data • 01: Normal status • 10: Last status • 11: EOP 00b - Packet data

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01b - Control word/normal status 10b - SOP data/last status 11b - EOP data/EOP
4 —	Reserved
3-2 BYTEEN	Byte Enables Indicates the number of data bytes valid in the data register during a write operation. This is valid only when MTL_DBG_STS[PKTSTATE] is 2'b10 (EOP), and the transmit FIFO or receive FIFO is selected. 00b - Byte 0 valid 01b - Byte 0 and byte 1 valid 10b - Byte 0, byte 1, and byte 2 valid 11b - All four bytes valid
1 DBGMOD	Debug Mode Access to FIFO Indicates the status of Debug mode access to FIFO. If this field is 1, it indicates that the current access to FIFO is read, write, and debug. In this mode, the following access types are allowed: • Read and write access to transmit FIFO, TSO FIFO, and receive FIFO • Read access to transmit status FIFO If this field is 0, it indicates that the current access to FIFO is slave access bypassing DMA. In this mode, the following access types are allowed: • Write access to transmit FIFO • Read access to receive FIFO and transmit status FIFO 0b - Disabled 1b - Enabled
0 FDBGEN	FIFO Debug Access Enable Indicates the status of FIFO debug access. • If this field is 1, it indicates that the FIFO debug mode access is enabled. • If the value of the field is 0, it indicates that FIFO can be accessed only through a master interface. 0b - Disabled 1b - Enabled

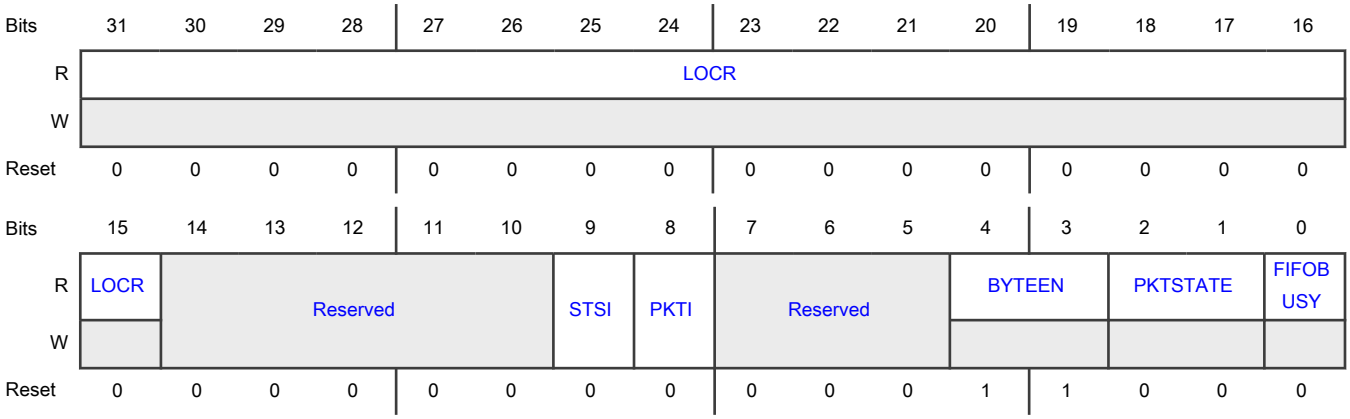
75.17.186 MTL Debug Status (MTL_DBG_STS)

Offset

Register	Offset
MTL_DBG_STS	C0Ch

Function
Contains the status of FIFO debug access.

Diagram



Fields

Field	Function
31-15 LOCR	Remaining Locations In FIFO Indicates the remaining locations in FIFO. For Slave Access mode, the field indicates the space available in the selected FIFO, and for Debug Access mode, this field contains the write or read pointer value of the selected FIFO during the write or read operation, respectively. Reset: In single transmit queue configurations, (DWC_EQOS_TXFIFO_SIZE/(DWC_EQOS_DATAWIDTH/8)), otherwise 0000h.
14-10 —	Reserved
9 STSI	Transmit Status Available Interrupt Status Indicates whether the transmit status available interrupt status is detected. If this field is 1, it indicates that Slave mode transmit packet is transmitted, and the status is available in transmit status FIFO. The field resets when you write 1 to it. 0b - Not detected 1b - Detected
8	Receive Packet Available Interrupt Status

Table continues on the next page...

Table continued from the previous page...

Field	Function
PKTI	Indicates whether the receive packet available interrupt status is detected. If this field is 1, it indicates that the MAC layer has written the EOP of the received packet to the receive FIFO. The field resets when you write 1 to it. 0b - Not detected 1b - Detected
7-5 —	Reserved
4-3 BYTEEN	Byte Enables Indicates the number of data bytes valid in the data register during a read operation. This is valid only when MTL_DBG_STS[PKTSTATE] is 2'b10 (EOP), and the transmit FIFO or receive FIFO is selected. 00b - Byte 0 valid 01b - Byte 0 and byte 1 valid 10b - Byte 0, byte 1, and byte 2 valid 11b - All four bytes valid
2-1 PKTSTATE	Encoded Packet State Is used to get the control or status information of the selected FIFO. These are the values related to the transmit FIFO. • 00: Packet data • 01: Control word • 10: SOP data • 11: EOP data These are the values related to the receive FIFO. • 00: Packet data • 01: Normal status • 10: Last status • 11: EOP This field is applicable only for transmit and receive FIFOs during a read operation. 00b - Packet data 01b - Control word/normal status 10b - SOP data/last status 11b - EOP data/EOP
0 FIFOBUSY	FIFO Busy Indicates whether the FIFO busy status is detected.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	If this field is 1, it indicates that a FIFO operation is in progress in MAC and the content of the following is invalid: • All other fields of this register • All the fields of MTL FIFO Debug Data (MTL_FIFO_Debug_Data) 0b - Not detected 1b - Detected

75.17.187 MTL FIFO Debug Data (MTL_FIFO_Debug_Data)

Offset

Register	Offset
MTL_FIFO_Debug_Data	C10h

Function
Contains the data to be written to or read from the FIFOs.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	FDBGDATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	FDBGDATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 FDBGDATA	FIFO Debug Data Contains the data to be written to the transmit FIFO, receive FIFO, or TSO FIFO during a debug or slave access write operations. The field contains the data read from the transmit FIFO, receive FIFO, TSO FIFO, or transmit status FIFO during debug or slave access read operations.

75.17.188 MTL Interrupt Status (MTL_Interrupt_Status)

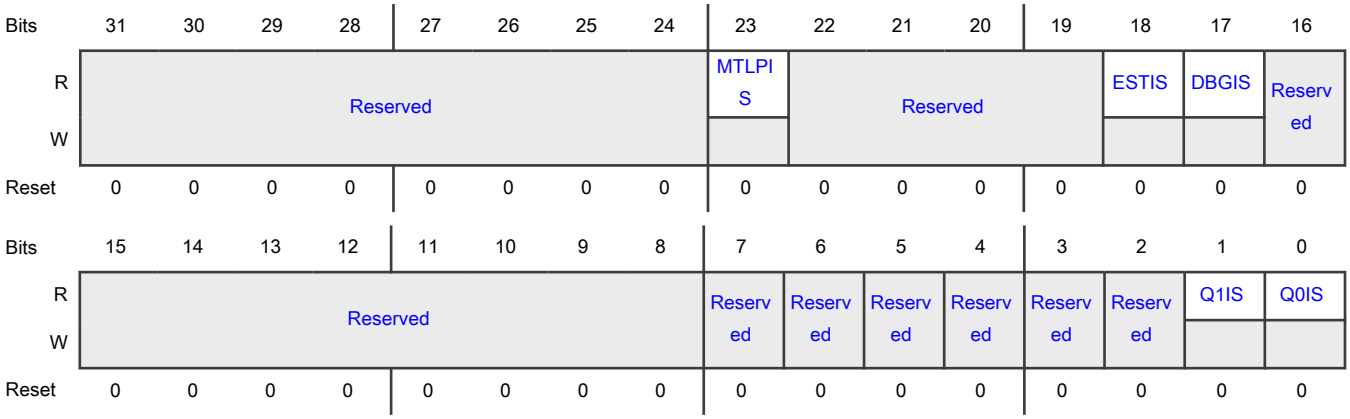
Offset

Register	Offset
MTL_Interrupt_Status	C20h

Function

Determines the interrupt status of MAC and MTL queues. The software driver (application) reads this register during an interrupt service routine or polling to perform this function.

Diagram



Fields

Field	Function
31-24 —	Reserved
23 MTLPIS	MTL Receive Parser Interrupt Status Indicates whether the MTL receive parser interrupt status is detected. This field indicates an interrupt from the receive parser block. To reset this field, the application must read MTL Rx Parser Interrupt Control Status (MTL_RXP_Interrupt_Control_Status) to get the exact cause of the interrupt and clear its source. 0b - Not detected 1b - Detected
22-19 —	Reserved
18 ESTIS	EST (TAS- 802.1Qbv) Interrupt Status Indicates whether the EST (TAS- 802.1Qbv) interrupt status is detected.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field indicates an interrupt event during the operation of 802.1Qbv. To reset the field, the application must clear the error or event that caused the interrupt. 0b - Not detected 1b - Detected
17 DBGIS	Debug Interrupt Status Indicates whether the debug interrupt status is detected. This field indicates an interrupt event during the slave access. To reset the field, the application must read MTL Debug Status (MTL_DBG_STS) to get the exact cause of the interrupt and clear its source. 0b - Not detected 1b - Detected
16 —	Reserved
15-8 —	Reserved
7 —	Reserved
6 —	Reserved
5 —	Reserved
4 —	Reserved
3 —	Reserved
2 —	Reserved
1 Q1IS	Queue 1 Interrupt Status Indicates whether the queue 1 interrupt status is detected. To reset this field, the application must read MTL Queue 1 Interrupt Control Status (MTL_Q1_Interrupt_Control_Status) to get the exact cause of the interrupt and clear its source. 0b - Not detected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Detected
0 Q0IS	Queue 0 Interrupt Status Indicates whether the queue 0 interrupt status is detected. To reset this field, the application must read MTL Queue 0 Interrupt Control Status (MTL_Q0_Interrupt_Control_Status) to get the exact cause of the interrupt and clear its source. 0b - Not detected 1b - Detected

75.17.189 MTL Receive Queue DMA Map 0 (MTL_RxQ_DMA_Map0)

Offset

Register	Offset
MTL_RxQ_DMA_Map0	C30h

Function
Reserves in EQOS-CORE and EQOS-MTL configurations.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved			Reserved	Reserved			Reserved	Reserved			Reserved	Reserved			Reserved
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved			Q1DDMACH	Reserved			Q1MDMACH	Reserved			Q0DDMACH	Reserved			Q0MDMACH
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	Reserved.
28 —	Reserved.

Table continues on the next page...

Table continued from the previous page...

Field	Function
27-25 —	Reserved.
24 —	Reserved.
23-21 —	Reserved.
20 —	Reserved.
19-17 —	Reserved.
16 —	Reserved.
15-13 —	Reserved.
12 Q1DDMACH	Queue 1 Enabled for DA-based DMA Channel Selection Enables or disables queue 1 for DA-based DMA channel selection. If this field is 1, it indicates that the packets received in queue 1 are routed to a particular DMA channel as decided in the MAC receiver based on the DMA channel number programmed in the L3-L4 filter registers, or the Ethernet DA address. When this field becomes 0, it indicates that the packets received in queue 1 are routed to the DMA Channel programmed in the MTL_RxQ_DMA_Map0[Q1MDMACH] (bits [10:8]). 0b - Disabled 1b - Enabled
11-9 —	Reserved.
8 Q1MDMACH	Queue 1 Mapped to DMA Channel Controls the routing of the received packet in Queue 1 to the DMA channel: 000b - DMA Channel 0 001b - DMA Channel 1 010b - DMA Channel 2 011b - DMA Channel 3

Table continues on the next page...

Table continued from the previous page...

Field	Function
	100b - DMA Channel 4 101b - DMA Channel 5 110b - DMA Channel 6 111b - DMA Channel 7 This field is valid when the MTL_RxQ_DMA_Map0[Q1DDMACH] resets. The field's width depends on the number of RX DMA channels and in some configurations all the values may not be valid . For example, if two RX DMA channels are selected, then only 000b and 001b are valid, the other bits are reserved.
7-5 —	Reserved.
4 Q0DDMACH	Queue 0 Enabled for DA-based DMA Channel Selection Enables or disables queue 0 for DA-based DMA channel selection. If this field is 1, it indicates that the packets received in queue 0 are routed to a particular DMA channel as decided in the MAC Receiver based on the DMA channel number programmed in the L3-L4 filter registers, or the Ethernet DA address. When this field becomes 0, it indicate that the packets received in queue 0 are routed to the DMA Channel programmed in the MTL_RxQ_DMA_Map0[Q0MDMACH]. 0b - Disable 1b - Enable
3-1 —	Reserved.
0 Q0MDMACH	Queue 0 Mapped to DMA Channel Controls the routing of the packet received in queue 0 to the DMA channel: 000b - DMA channel 0 001b - DMA channel 1 010b - DMA channel 2 011b - DMA channel 3 100b - DMA channel 4 101b - DMA channel 5 110b - DMA channel 6 111b - DMA channel 7 This field is valid when the MTL_RxQ_DMA_Map0[Q0DDMACH] resets.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	The field's width depends on the number of RX DMA channels in some configurations all the values may not be valid. For example, if the number of RX DMA channels selected is 2, only 000 and 001 bits are valid, the other bits are reserved.

75.17.190 MTL TBS Control (MTL_TBS_CTRL)

Offset

Register	Offset
MTL_TBS_CTRL	C40h

Function

Controls the operation of time based scheduling.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	LEOS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	LEOS								Reserved	LEGOS			Reserved		LEOV	ESTM
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-8 LEOS	Launch Expiry Offset Computes the launch expiry time. To compute the launch expiry time the value in units of 256 nanoseconds is added to the launch time. The field value is valid only if MTL_TBS_CTRL[LEOV] is 1. Max value: 999,999,999 nanosecond should be smaller than CTR-1 value when ESTM mode = 1, because this value is a modulo CTR value.
7 —	Reserved.
6-4	Launch Expiry GSN Offset

Table continues on the next page...

Table continued from the previous page...

Field	Function
LEGOS	Computes the Launch expiry time. To compute the launch expiry time the number of GSN slots is to be added to the launch GSN. The field value is valid only if MTL_TBS_CTRL[LEOV] is 1.
3-2 —	Reserved.
1 LEOV	Launch Expiry Offset Valid Indicates if MTL_TBS_CTRL[LEOS] is valid or invalid. When this field is 1, it indicates that MTL_TBS_CTRL[LEOS] is valid. When not 1, it indicates that the launch expiry Offset is not valid and the MTL must not check for launch expiry time. 0b - Invalid 1b - Valid
0 ESTM	EST offset Mode Enables or disables EST Offset mode. If this field is 1, the launch time value used in Time Based Scheduling is interpreted as an EST offset value and is added to the BTR of the current list. When this field becomes 0, the launch time value is used as an absolute value that must be compared with the system time [39:8]. 0b - Disabled 1b - Enabled

75.17.191 MTL EST Control (MTL_EST_Control)

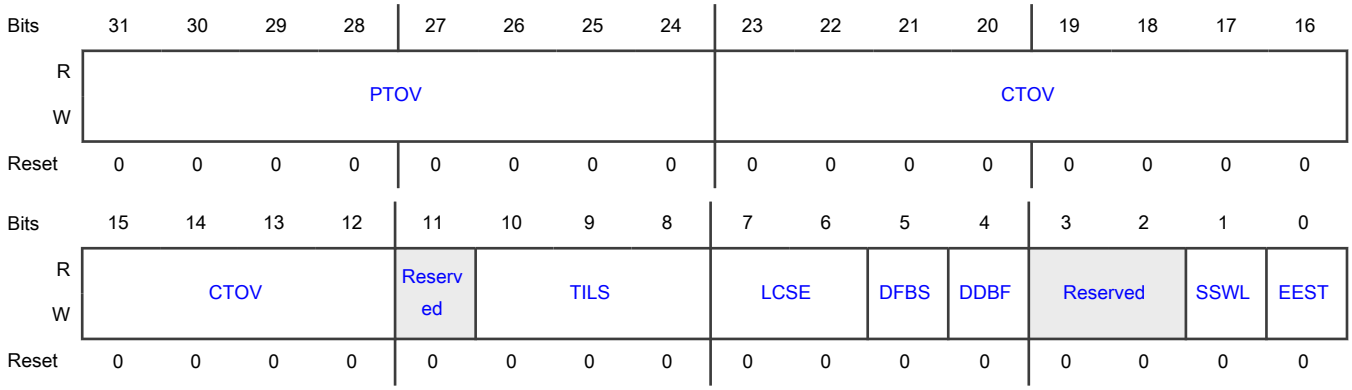
Offset

Register	Offset
MTL_EST_Control	C50h

Function

Controls the operation of enhancements to scheduled transmission (IEEE802.1Qbv).

Diagram



Fields

Field	Function
31-24 PTOV	PTP Time Offset Value Indicates the PTP time offset value. You must multiply 6 to the value of PTP clock period in nanoseconds. At the beginning of the installation of a new GCL this value avoid transmission overruns.
23-12 CTOV	Current Time Offset Value Provides a 12 bit time offset value in nano second which is added to the current time to compensate for all the implementation pipeline delays such as the CDC sync delay, buffering delays, data path delays and so on. This offset ensures that the impact of gate controls is visible on the line exactly at the pre-determined schedule (or as close to the schedule as possible).
11 —	Reserved.
10-8 TILS	Time Interval Left Shift Amount Provides the left shift amount for the programmed time interval values used in the gate control lists. 000b - No left shift needed (equal to x1ns) 001b - Left shift TI by 1 bit (equal to x2ns) 010b - Left shift TI by 2 bits (equal to x4ns) - - 100b - Left shift TI by 7 bits (equal to x128ns) On the basis of configuration you must consider one or more bits of this field as reserved or read-only.
7-6 LCSE	Loop Count to Report Scheduling Error Provides programmable number of GCL list iterations before reporting an HLBS error which is defined in MTL_EST_Status.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00b - 4 iterations 01b - 8 iterations 10b - 16 iterations 11b - 32 iterations
5 DFBS	Drop Frames Causing Scheduling Error Provides the status of the frames causing scheduling error. If this field is 1 then the frames report to cause an HOL blocking because the MTL_EST_Status[HLBS] is not scheduled after 4,8,16,and 32 (on the basis of MTL_EST_Control[LCSE]) GCL iterations are dropped. 0b - Do not drop frames 1b - Drop frames
4 DDBF	Do not Drop Frames during Frame Size Error Provides status of frames during Frame Size Error. If this field is 1, it indicates that the frames are not dropped during head-of-line blocking because of the frame size error (MTL_EST_Status[HLBF]). 0b - Drop frames during frame size error 1b - Do not Drop frames during frame size error
3-2 —	Reserved.
1 SSWL	Switch to Software Owned List If this field is 1, it indicates that the software has programmed that list that it currently owns (SWOL) and the hardware should switch to the new list based on the new BTR. Hardware clears this bit when the switch to the SWOL happens to indicate the completion of the switch or when an BTR error (BTRE in Status register) is set. When BTRE is set this bit is cleared but SWOL is not updated as the switch was not successful. Access restriction apply to this field that clears on read and automatically becomes 1 on an internal event occurrence. Enables or disables the switch to the software owned list. 0b - Disabled 1b - Enabled
0 EEST	Enable EST Enables or disables EST. If the field becomes 0, it indicate that the gate control list processing is halted and all gates are assumed to be in open state. The field must be 1 for the hardware to start processing the gate control lists. During the toggle from 0 to 1, when the MTL_EST_Control[SSWL] is 1, the gate control list processing starts. During the configuration when DWC_EQOS_ASP_ECC is selected, the hardware resets this field and disables the EST function if any uncorrectable error is detected in the EST memory .

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disabled 1b - Enabled

75.17.192 MTL EST Status (MTL_EST_Status)

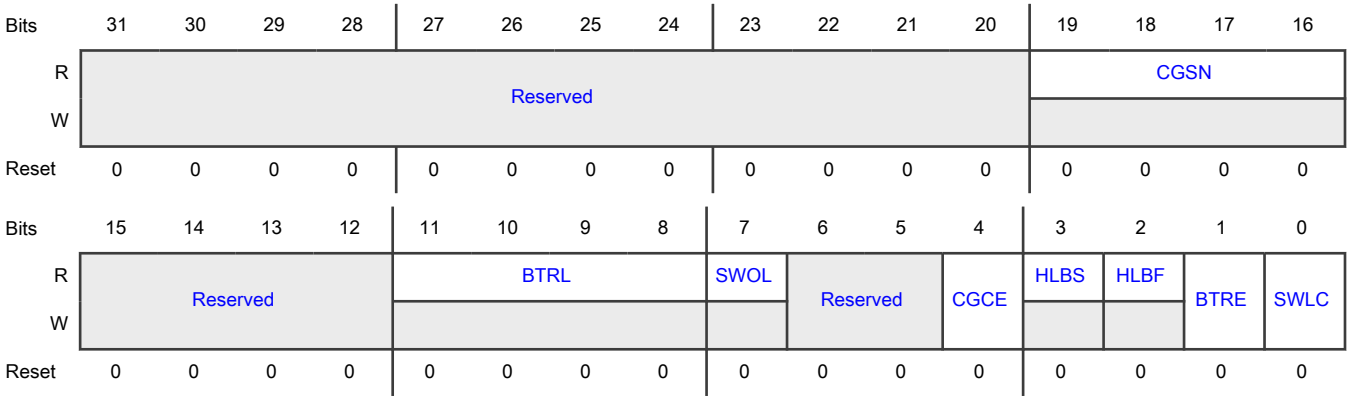
Offset

Register	Offset
MTL_EST_Status	C58h

Function

Provides status of enhancements to scheduled transmission (IEEE802.1Qbv).

Diagram



Fields

Field	Function
31-20 —	Reserved.
19-16 CGSN	Current GCL Slot Number Indicates the slot number of the GCL list. Slot number is a modulo 16 count of the GCL List loops executed so far. Even if a new GCL list is installed, the count is incremental.
15-12 —	Reserved.

Table continues on the next page...

Table continued from the previous page...

Field	Function
11-8 BTRL	BTR Error Loop Count Provides the minimum count (N) for which the equation Current Time =< New BTR + (N * New Cycle Time) becomes true. N = "1111" indicates the iterations exceeded the value of 8 and the hardware was not able to update New BTR to be equal to or greater than Current Time. Software intervention is needed to update the New BTR. Value cleared when BTRE field of this register is cleared.
7 SWOL	S/W owned list When '0' indicates Gate control list number "0" is owned by software and when "1" indicates the Gate Control list "1" is owned by the software. Any reads/writes by the software (using indirect access via GCL_Control) is directed to the list indicated by this value by default. The inverse of this value is treated as HWOL. R/W operations performed by hardware are directed to the list pointed by HWOL by default. 0b - Gate control list number "0" is owned by software 1b - Gate control list number "1" is owned by software
6-5 —	Reserved.
4 CGCE	Constant Gate Control Error This error occurs when the list length (LLR) is 1 and the Cycle Time (CTR) is less than or equal to the programmed Time Interval (TI) value after the optional Left Shifting. The above programming implies Gates are either always Closed or always Open based on the Gate Control values; the same effect can be achieved by other simpler (non TSN) programming mechanisms. Since the implementation does not support such a programming an error is reported. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. 0b - Constant Gate Control Error not detected 1b - Constant Gate Control Error detected
3 HLBS	Head-Of-Line Blocking due to Scheduling Set when the frame is not able to win arbitration and get scheduled even after 4 iterations of the GCL. Indicates to software a potential programming error. The one hot encoded values of the Queue Numbers that are not able to make progress are indicated in the MTL_EST_Sch_Error register. Bit cleared when MTL_EST_Sch_Error register is all zeros. 0b - Head-Of-Line Blocking due to Scheduling not detected 1b - Head-Of-Line Blocking due to Scheduling detected
2 HLBF	Head-Of-Line Blocking due to Frame Size Set when HOL Blocking is noticed on one or more Queues as a result of none of the Time Intervals of gate open in the GCL being greater than or equal to the duration needed for frame size (or frame fragment size when preemption is enabled) transmission. The one hot encoded Queue numbers that are experiencing HLBF are indicated in the MTL_EST_Frm_Size_Error register. Additionally, the first Queue number that experienced HLBF along with the frame size is captured in MTL_EST_Frm_Size_Capture register. Bit cleared when MTL_EST_Frame_Size_ Error register is all zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Head-Of-Line Blocking due to Frame Size not detected 1b - Head-Of-Line Blocking due to Frame Size detected
1 BTRE	BTR Error Indicates whether the BTR error is detected. When this field is 1, it indicates a programming error in the BTR of SWOL where the programmed value is less than the current time. If the BTRL = "1111", SWOL is not updated and software must reprogram the BTR to a value greater than current time and then set SSWL to reinitiate the switch to SWOL. Else if the value of BTRL < "1111", SWOL is updated and this field indicates the number of iterations (of + CycleTime) taken by hardware to update the BTR to a value greater than Current Time. Access restrictions apply to this field. It sets automatically to 1 on an internal event occurrence. Writing 1 clears this field and writing 0 has no effect. 0b - BTR Error not detected 1b - BTR Error detected
0 SWLC	Switch to Software Owned List Complete Indicates whether the switch to software owned list complete is detected. When the field is 1, it indicates that the hardware has successfully switched to the SWOL, and updated the MTL_EST_Status[SWOL] to that effect. This field is 0 when MTL_EST_Control[SSWL] transitions from 0 to 1, or on a software write. Access restrictions apply to this field. It clears automatically and writing 0 to it has no effect. 0b - Not detected 1b - Detected

75.17.193 MTL EST Scheduling Error (MTL_EST_Sch_Error)

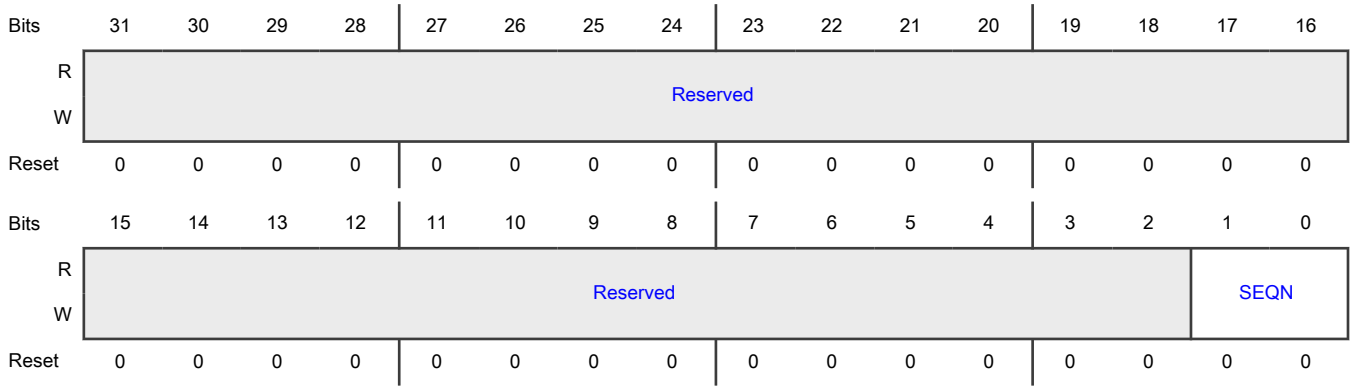
Offset

Register	Offset
MTL_EST_Sch_Error	C60h

Function

Provides the one hot encoded queue numbers that have the scheduling related error (timeout).

Diagram



Fields

Field	Function
31-2 —	Reserved.
1-0 SEQN	Schedule Error Queue Number Provides the one hot encoded queue numbers that have experienced error or timeout described in MTL_EST_Status[HLBS]. Access restriction apply to this field. It automatically becomes 1 on an internal event occurrence. Writing 1 clears this field and writing 0 to it has no effect.

75.17.194 MTL EST Frame Size Error (MTL_EST_Frm_Size_Error)

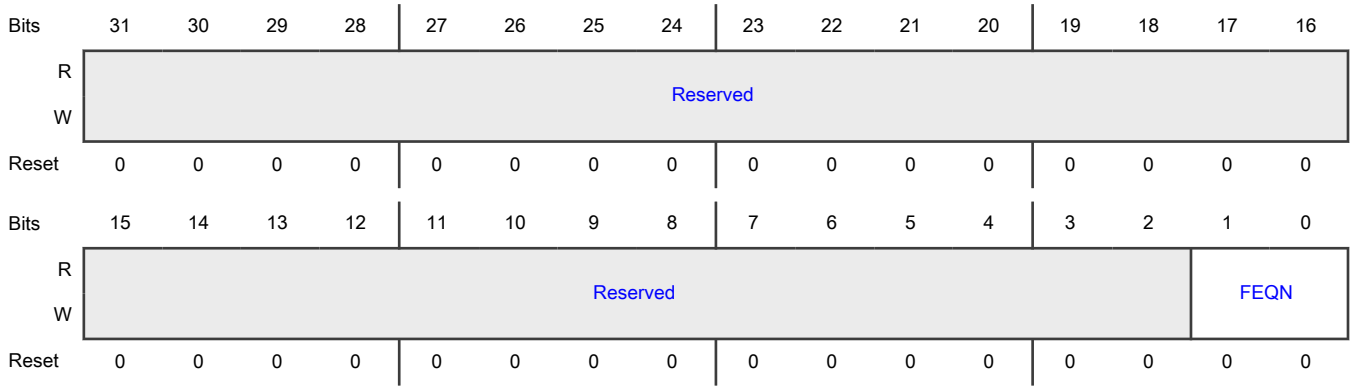
Offset

Register	Offset
MTL_EST_Frm_Size_Error	C64h

Function

Provides the one hot encoded queue numbers that have the frame Size related error.

Diagram



Fields

Field	Function
31-2 —	Reserved.
1-0 FEQN	Frame Size Error Queue Number Provides the one hot encoded queue Nnumbers that have experienced error described in MTL_EST_Status[HLBF]. Access restriction apply to this field. It automatically becomes 1 on an internal event occurrence. Writing 1 clears this field and writing 0 to it has no effect.

75.17.195 MTL EST Frame Size Capture (MTL_EST_Frm_Size_Capture)

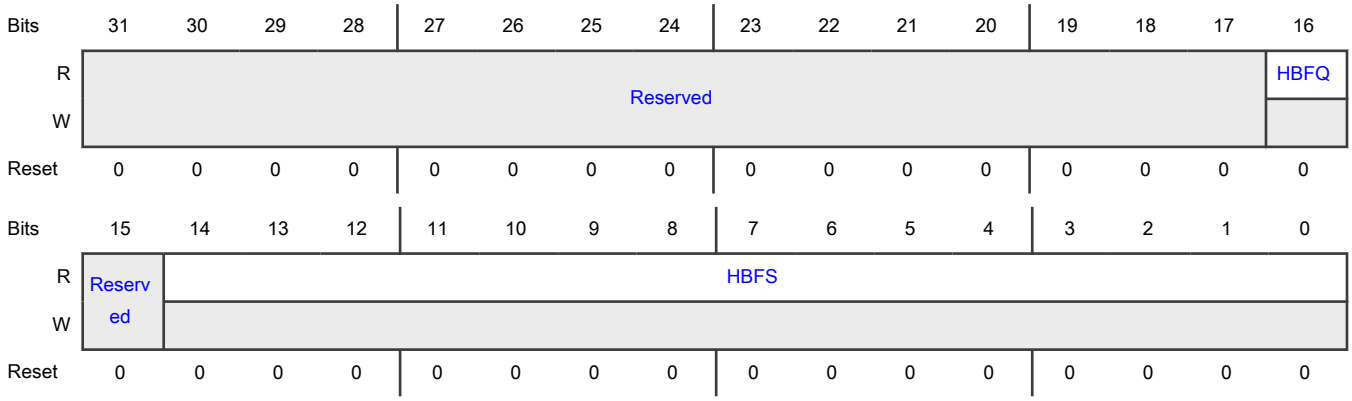
Offset

Register	Offset
MTL_EST_Frm_Size_Capture	C68h

Function

Captures the frame Size and queue number of the first occurrence of the frame Size related error. When you write 0, it captures the data of immediate next occurrence of a similar error.

Diagram



Fields

Field	Function
31-17 —	Reserved.
16 HBFQ	Queue Number of HLBF Captures the binary value of the first queue (number) which experiences the HLBF error (see MTL_EST_Status[HLBF]). Any subsequent queue errors similar to HLBF errors does not alter the value after it is written. After this field is 0, it captures the queue number of the next occurring HLBF error. Field's width is based on the number of transit queues configured; remaining fields are read-only. This field clears when MTL_EST_Frm_Size_Error is all zeros.
15 —	Reserved.
14-0 HBFS	Frame Size of HLBF Captures the frame Size of the dropped frame related to queue number indicated in MTL_EST_Frm_Size_Capture[HBFQ]. If this field is zero, then this register must be considered invalid. This field clears when MTL_EST_Frm_Size_Error is all zeros.

75.17.196 MTL EST Interrupt Enable (MTL_EST_Intr_Enable)

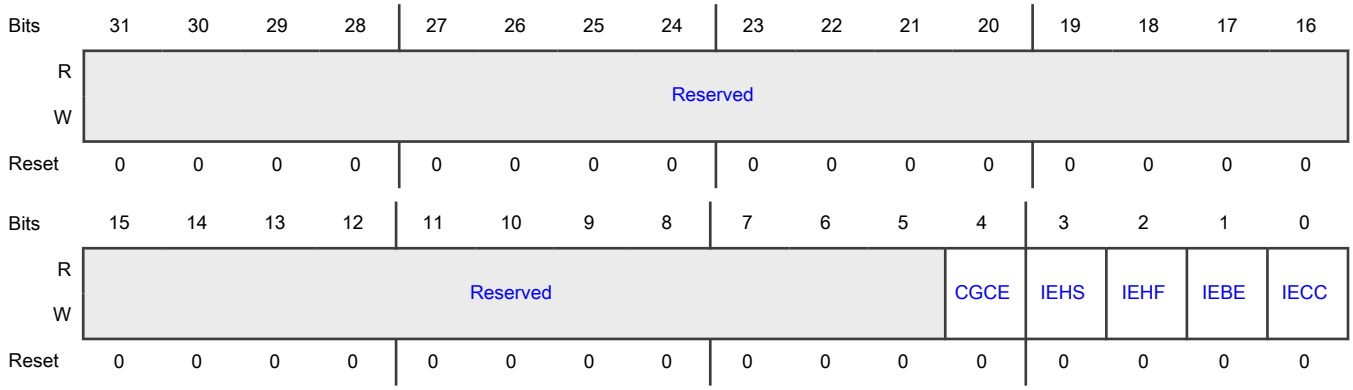
Offset

Register	Offset
MTL_EST_Intr_Enable	C70h

Function

Implements the interrupt enable fields for the various events that generate an interrupt. Bit positions have a 1 to 1 correlation with the status bit positions in MTL_ETS_Status register.

Diagram



Fields

Field	Function
31-5 —	Reserved.
4 CGCE	Interrupt Enable for CGCE Indicates whether an interrupt for MTL_EST_Intr_Enable[CGCE] is enabled. When this field is 1, it generates an interrupt when the constant gate control error occurs and is indicated in the status. When this field becomes 0, this event does not generate an interrupt 0b - Disabled 1b - Enabled
3 IEHS	Interrupt Enable for HLBS Indicates whether an interrupt for MTL_EST_Status[HLBS] is enabled. When this field is 1, it generates an interrupt when the head-of-line blocking due to scheduling issue occurs and is indicated in the status. When this field becomes 0, this event does not generate an interrupt. 0b - Disabled 1b - Enabled
2 IEHF	Interrupt Enable for HLBF Indicates whether an interrupt for MTL_EST_Status[HLBF] is enabled. When this field is 1, it generates an interrupt when the head-of-line blocking due to frame size error occurs and is indicated in the status. When this field becomes 0, this event does not generate an interrupt. 0b - Disabled 1b - Enabled
1 IEBE	Interrupt Enable for BTR Error Indicates whether an interrupt for BTR error is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When this field is 1, it generates an interrupt when the BTR error occurs and is indicated in the status. When this field becomes 0, this event does not generate an interrupt. 0b - Disabled 1b - Enabled
0 IECC	Interrupt Enable for Switch List Indicates whether the interrupt for switch list is enabled. When this field is 1, it generates an interrupt when the configuration change is successful and the hardware switches to the new list. When this field becomes 0, this event does not generate an interrupt. 0b - Disabled 1b - Enabled

75.17.197 MTL EST GCL Control (MTL_EST_GCL_Control)

Offset

Register	Offset
MTL_EST_GCL_Control	C80h

Function

Provides the control information for reading and writing to the gate control lists.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved								ESTEIEC		ESTEIEE	ERR0	Reserved			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ADDR								Reserved		DBGB	DBGM	Reserved	GCRR	R1W0	SRWO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24	Reserved.

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
23-22 ESTEIEC	ECC Inject Error Control for EST Memory Provides the status of inject error control for EST memory. When MTL_EST_GCL_Control[ESTEIEE] = 1, it indicates that on the basis of the value encoded in this field following errors are inserted. This field is valid only if you select DWC_EQOS_ASP_ECC feature during the configuration, otherwise it is reserved. 00b - Insert 1 bit error 01b - Insert 2 bit errors 10b - Insert 3 bit errors 11b - Insert 1 bit error in address field
21 ESTEIEE	EST ECC Inject Error Enable Indicates whether the EST ECC inject error is enabled. If this field along with MTL_EST_Control[EEST] = 1, it enables the ECC error injection feature. When this field becomes 0, it disables the ECC error injection feature. 0b - Disabled 1b - Enabled
20 ERR0	If this field is 1, it indicates that when the software writes to GCL the last write operation was aborted and when MTL_EST_Control[SSWL] is 1, GCL registers are prohibited. Access restriction apply to this field and automatically becomes 1 on an internal event occurrence. Writing 1 to this field clears it and writing 0 to this field has no effect. 0b - ERR0 is disabled 1b - ERR1 is enabled
19-16 —	Reserved.
15-8 ADDR	Gate Control List Address: (GCLA when GCRR is "0"). Provides the address (row number) of the gate control list at which the R/W operation has to be performed. By default the gate control list to which MTL_EST_Status[SWOL] points is selected for read and write. If MTL_EST_GCL_Control[DBGM] is 1, it indicates that MTL_EST_GCL_Control[DBGB] gives Debug mode access to read and write. The field's width depends on DWC_EQOS_EST_DEP and the unused bits must be treated as read only. Gate Control list Related Registers Address: (GCRA when GCRR is "1"). By default the GCL related register set pointed by MTL_EST_Status[SWOL] is selected for read or write. If MTL_EST_GCL_Control[DBGM], it indicates that MTL_EST_GCL_Control[DBGB] gives Debug mode access to read write. Lower 3 bits are only used in this mode, higher order bits are treated as don't cares.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	000b - BTR Low (31:0) 001b - BTR High (63:31) 010b - CTR Low (31:0) 011b - CTR High (39:32) 100b - TER (31:0) 101b - LLR (n:0) (where n is (log{DWC_EQOS_EST_DEP} / log2)) Others - Reserved
7-6 —	Reserved.
5 DBGB	Debug Mode Bank Select Indicates whether the read and write in Debug mode is directed to bank 0 or bank 1. When this field is 0, it indicates that read or write in Debug mode must be directed to bank 0 (GCL0 and corresponding Time related registers). When this field is 1, it indicates that read or write in Debug mode should be directed to bank 1 (GCL1 and corresponding Time related registers). When MTL_EST_GCL_Control[DBGM] is 1 you can use this value and overrides with the MTL_EST_Status[SWOL] value. 0b - Directed to bank 0 1b - Directed to bank 1
4 DBGM	Debug Mode Enables or disables Debug mode. When this field is 1, it indicates that the read and write is in Debug mode where the MTL_EST_GCL_Control[DBGB] value explicitly provides the memory bank (for GCL and time related registers). When this field is 0, it indicates that MTL_EST_Status[SWOL] determines which bank to use. 0b - Disabled 1b - Enabled
3 —	Reserved.
2 GCRR	Gate Control Related Registers Enables or disables gate control related registers. When this field is 1, it indicates that the read or write access is for the GCL related registers (BTR, CTR, TER, LLR) for which GCRA provides the address. When this field is 0, it indicates that read and write must be directed to GCL from the address which GCLA provides. 0b - Disabled 1b - Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 R1W0	Read '1', Write '0' Indicates whether this field performs read operation or write operation. When this field is 1 it indicates that it performs a read operation. When this field is 0 it indicates that it performs a write operation. 0b - Write operation 1b - Read operation
0 SRWO	Start Read/Write Operation Indicates whether the read/write operation is enabled. When this field is 1, it indicates that a read/write operation has started and is in progress. When hardware resets this field and MTL_EST_GCL_Control[ERR0] = 1 , it indicates that the read/write operation has completed or an error has occurred. Reads: When this field becomes 0, it indicates that MTL_EST_GCL_Data reads data. Writes: Before writing 1 to this field, you must program MTL_EST_GCL_Data with write data. Access restrictions apply to this field. It clears automatically. Writing 1 sets this field and writing 0 to it has no effect. 0b - Disabled 1b - Enabled

75.17.198 MTL EST GCL Data (MTL_EST_GCL_Data)

Offset

Register	Offset
MTL_EST_GCL_Data	C84h

Function

Holds the read data or write data in case of reads and writes respectively.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	GCD															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	GCD															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 GCD	Gate Control Data Provides the data corresponding to the address selected in the MTL_EST_GCL_Control. This field is used for both read and write operations.

75.17.199 MTL FPE Control Status (MTL_FPE_CTRL_STS)

Offset

Register	Offset
MTL_FPE_CTRL_STS	C90h

Function

Controls the operation of, and provides status for frame preemption (IEEE802.1Qbu/802.3br).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved			HRS	Reserved											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved						PEC		Reserved						AFSZ	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	Reserved.
28 HRS	Hold/Release Status Indicates whether a set-and-release-MAC operation which was last executed and the pMAC is in release state or in hold state. 0b - Release state 1b - Hold state
27-16 —	Reserved.
15-10 —	Reserved.
9-8 PEC	Preemption Classification When this field is 1, it indicates that you must classify the corresponding queue as preemptable. When this field is 0, it indicates that you must classify the queue as express. When set indicates the corresponding Queue must be classified as preemptable, when '0' Queue is classified as express. When both EST (Qbv) and Preemption are enabled, Queue-0 is always assumed to be preemptable. When you enables EST (Qbv), the preemptable queues are always assumed to be in open state in gate control list.
7-2 —	Reserved.
1-0 AFSZ	Additional Fragment Size Indicates that, in units of 64 bytes, the minimum number of bytes over 64 bytes required in non-final fragments of preempted frames. The minimum non-final fragment size is (AFSZ +1) * 64 bytes.

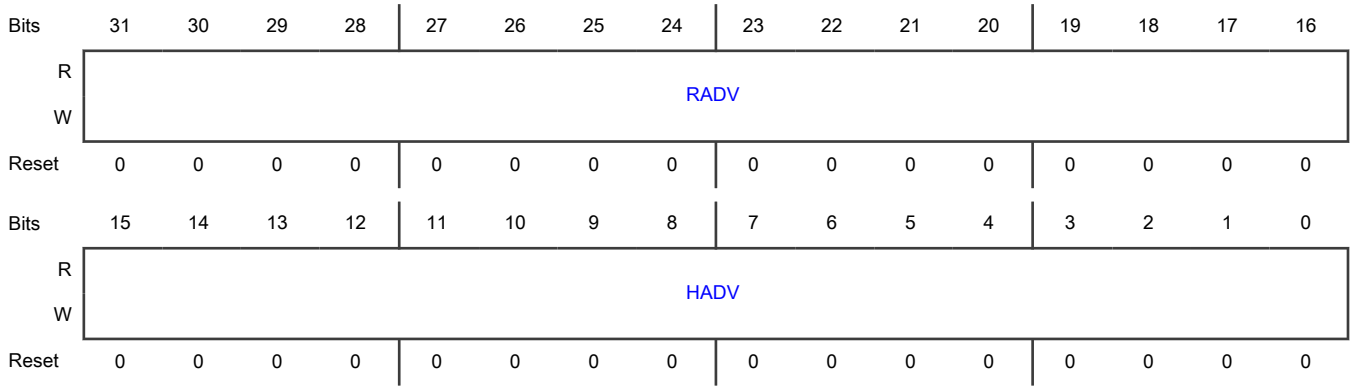
75.17.200 MTL FPE Advance (MTL_FPE_Advance)

Offset

Register	Offset
MTL_FPE_Advance	C94h

Function
Holds the hold and release advance time.

Diagram



Fields

Field	Function
31-16 RADV	Release Advance Provides the maximum time in nanoseconds that can elapse between issuing a release to the MAC and the MAC being ready to resume transmission of preemptable frames, in the absence of there being any express frames available for transmission.
15-0 HADV	Hold Advance Provides the maximum time in nanoseconds that can elapse between issuing a hold to the MAC and the MAC ceasing to transmit any preemptable frame that is in the process of transmission or any preemptable frames that are queued for transmission.

75.17.201 MTL Rx Parser Control Status (MTL_RXP_Control_Status)

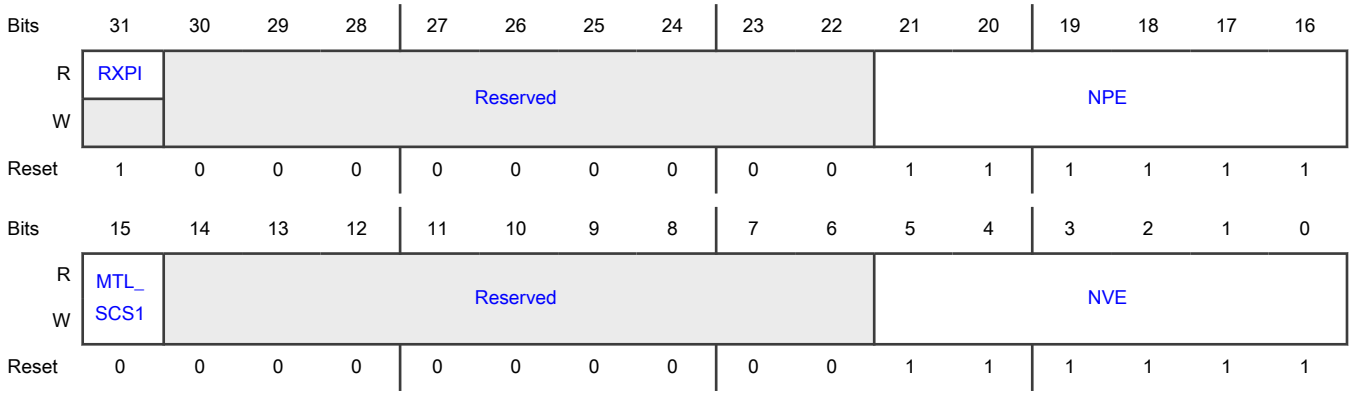
Offset

Register	Offset
MTL_RXP_Control_Status	CA0h

Function

Establishes the operating mode of receive parser and provides some status.

Diagram



Fields

Field	Function
31 RXPI	RX Parser in Idle State Indicates whether the receive parser is in idle state. When this field is 1, it indicates that the receive parser is in idle state and waiting for the processing of a new packet. When parser disables, you can use this field as a handshake with software and also the software can update the receive parser instruction table when you write 1 to this field. 0b - Not in Idle state 1b - Idle state
30-22 —	Reserved.
21-16 NPE	Number of parsable entries in the Instruction table Indicates the number of parsable entries in the instruction memory. This is used in receive parser logic to detect programming error. MTL_RXP_Interrupt_Control_Status[NPEOVIS] = 1, if the number of parsed entries for a packet is more than this entry.
15 MTL_SCS1	MTL_SCS1 Is reserved for NXP internal use. The value of this field must always be 0 unless NXP instructs you otherwise. Writing 1 to any of the bits of this field might cause unexpected behavior in th IP.
14-6 —	Reserved.
5-0 NVE	Number Of Valid Entry Address Or Index In The Instruction Table Indicates the number of valid entry address or index in the instruction memory.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	The maximum valid entry address is = NVE + 1 and the addresses or indices range from 0 to 32. Therefore, if the value of this field is 32, the number of valid entry address is 33. This value is used in receive parser logic to detect if any programming errors exist. When parsing, the module writes 1 to NVEOVIS if the number of the memory address is found to be more than the defined maximum valid entry address number. **NOTE** The minimum value of this field must be two.

75.17.202 MTL Rx Parser Interrupt Control Status (MTL_RXP_Interrupt_Control_Status)

Offset

Register	Offset
MTL_RXP_Interrupt_Control_Status	CA4h

Function
Provides enable control for the interrupts and also provides interrupt status.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved												PDRFIE	FOOVIE	NPEOVIE	NVEOVIE
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved												PDRFIS	FOOVIS	NPEOVIS	NVEOVIS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-20 —	Reserved.
19 PDRFIE	Packet Drop due to RF Interrupt Enable Indicates whether the packet drop due to RF interrupt is enabled. If this field is 1, it enables the PDRFIS interrupt. When this field becomes 0, it disables the PDRFIS interrupt.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disabled 1b - Enabled
18 FOOVIE	Frame Offset Overflow Interrupt Enable Indicates whether the frame offset overflow interrupt is enabled. If this field is 1, it enables the FOOVIS interrupt. When this field becomes 0, it disables the FOOVIS interrupt. 0b - Disabled 1b - Enabled
17 NPEOVIE	Number of Parsable Entries Overflow Interrupt Enable Enables or disables the number of parsable entries overflow interrupt. If this field is 1, it enables the NPEOVIS interrupt. When this field becomes 0, it disables the NPEOVIS interrupt. 0b - Disable 1b - Enable
16 NVEOVIE	Number of Valid Entries Overflow Interrupt Enable Enables or disables the number of valid entries overflow interrupt. If this field is 1, it enables the NVEOVIS interrupt. When this field becomes 0, it disables the NVEOVIS interrupt. 0b - Disable 1b - Enable
15-4 —	Reserved.
3 PDRFIS	Packet Dropped due to RF Interrupt Status Indicates whether the packet dropped due to RF interrupt status is detected. If the Rx Parser result says to drop the packet by setting RF=1 in the instruction memory, then this bit is set to 1. This field clears when the application writes 1 to this field. Access apply to this field. It automatically sets to 1 on an internal event occurrence. Writing 1 clears this field and writing 0 to it has no effect. 0b - Not detected 1b - Detected
2	Frame Offset Overflow Interrupt Status

Table continues on the next page...

Table continued from the previous page...

Field	Function
FOOVIS	Indicates whether the frame offset overflow interrupt status is detected. If this field is 1, it indicates that when parsing, the instruction table entry's frame offset is more than the EOF offset. This field clears when the application writes 1 to this field. Access apply to this field. It automatically sets to 1 on an internal event occurrence. Writing 1 clears this field and writing 0 to it has no effect. 0b - Not detected 1b - Detected
1 NPEOVIS	Number of Parsable Entries Overflow Interrupt Status Indicates whether the number of parsable entries overflow interrupt status is detected. If this field is 1, it indicates that when parsing a packet the number of parsed entries are more than NPE[] (MTL_RXP_Control_Status[NPE]). This field clears when the application writes 1 to this field. Access apply to this field. It automatically sets to 1 on an internal event occurrence. Writing 1 clears this field and writing 0 to it has no effect. 0b - Not detected 1b - Detected
0 NVEOVIS	Number of Valid Entry Address/Index Overflow Interrupt Status Indicates whether the number of valid entry address or index overflow interrupt status is detected. If this field is 1, it indicates that when parsing, the instruction address is more than MTL_RXP_Control_Status[NVE]. For example, when MTL_RXP_Control_Status[NVE] = 31, the maximum valid entry address/index is NVE+1 that is 32 (addresses/indices = 0 to 32, or 33 entries). MTL_RXP_Interrupt_Control_Status[NVEOVIS] = 1 when currently processed entry indicates that the next address is 33 or more, that is 34th or later entries. This field clears when the application writes 1 to this field. Access apply to this field. It automatically sets to 1 on an internal event occurrence. Writing 1 clears this field and writing 0 to it has no effect. 0b - Not detected 1b - Detected

75.17.203 MTL Rx Parser Drop Count (MTL_RXP_Drop_Cnt)

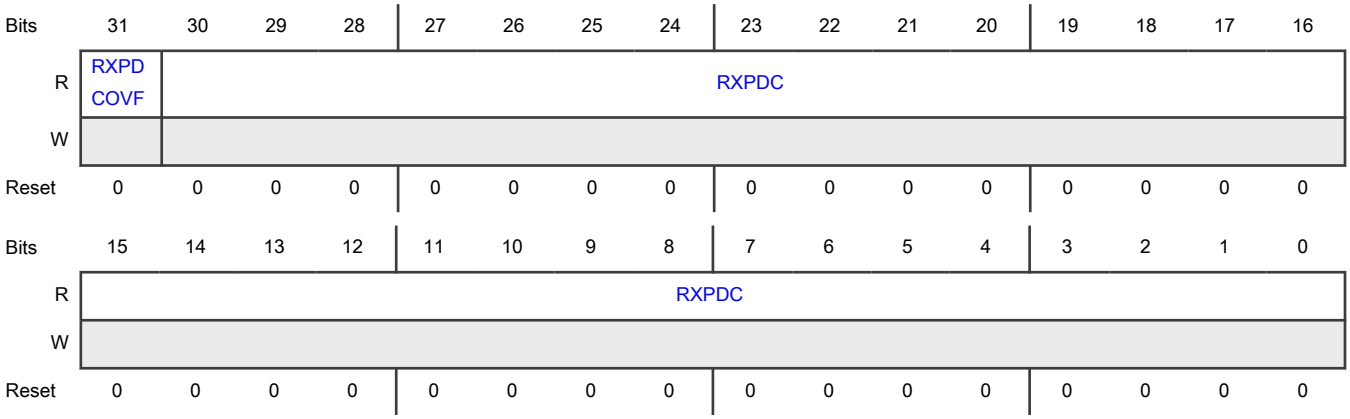
Offset

Register	Offset
MTL_RXP_Drop_Cnt	CA8h

Function

Provides the drop count of receive parser initiated drops.

Diagram



Fields

Field	Function
31 RXPDCOVF	Rx Parser Drop Counter Overflow Bit Indicates whether the receive parser drop count overflow has occurred. When this field is 1, it indicates that the MTL_RXP_Drop_Cnt[RXPDC] counter field has crossed the maximum limit. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not occurred 1b - Occurred
30-0 RXPDC	Rx Parser Drop Count Provides the receive parser drop count. This 31-bit counter is implemented whenever a receive parser drops a packet due to RF = 1. The counter clears when you read the register.

75.17.204 MTL Rx Parser Error Count (MTL_RXP_Error_Cnt)

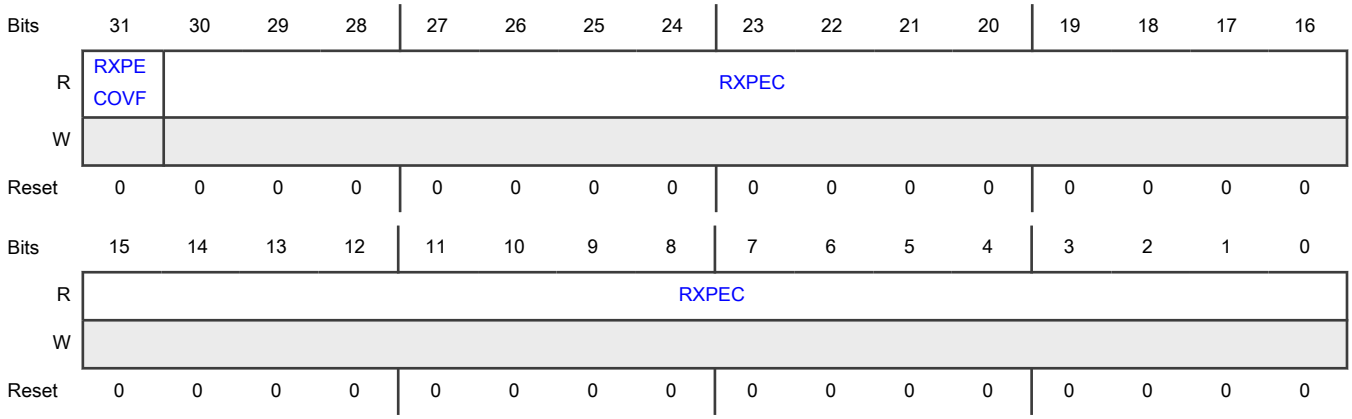
Offset

Register	Offset
MTL_RXP_Error_Cnt	CACh

Function

Provides the receive parser related error occurrence count.

Diagram



Fields

Field	Function
31 RXPECOVF	Rx Parser Error Counter Overflow Bit Indicates whether the receive parser error counter overflow has occurred. When this field is 1, it indicates that MTL_RXP_Error_Cnt[RXPEC] counter field has crossed the maximum limit. Access restrictions apply to this field, which clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not occurred 1b - Occurred
30-0 RXPEC	Rx Parser Error Count Provides the receive parser error count. This 31-bit counter is implemented whenever a receive parser encounters following error scenarios: Entry address >= NVE[] Number parsed entries >= NPE[] Entry address > EOF data entry address The counter clears when you read the register.

75.17.205 MTL Rx Parser Indirect Access Control Status (MTL_RXP_Indirect_Acc_Control_Status)

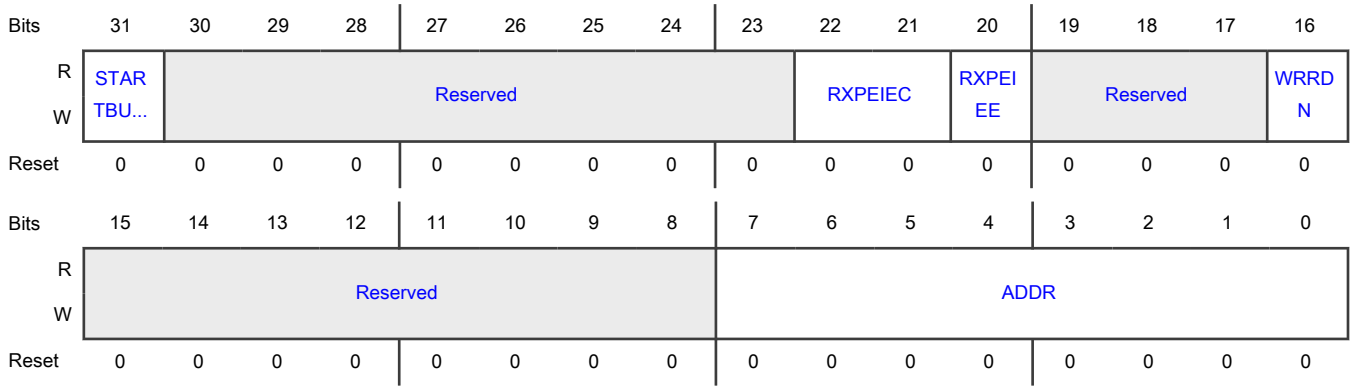
Offset

Register	Offset
MTL_RXP_Indirect_Acc_Control_Status	CB0h

Function

Provides the indirect access control and status for receive parser memory.

Diagram



Fields

Field	Function
31 STARTBUSY	FRP Instruction Table Access Busy When this bit is set to 1 by the software then it indicates to start the Read/Write operation from/to the Rx Parser Memory. Software should read this bit as 0 before issuing read or write request to access the Parser Memory Instructions. This bit when set to 1 indicates that hardware is busy until its gets cleared by hardware and software should not issue any read or write operation. 0b - hardware not busy 1b - hardware is busy (Read/Write operation from/to the Rx Parser Memory)
30-23 —	Reserved.
22-21 RXPEIEC	ECC Inject Error Control for Rx Parser Memory Controls the ECC inject error for receive parser memory. When MTL_RXP_Indirect_Acc_Control_Status[RXPEIEE] = 1, it indicates that on the basis of the value encoded in this field following errors are inserted: 00b - Insert 1 bit error 01b - Insert 2 bit errors 10b - Insert 3 bit errors 11b - Insert 1 bit error in address field
20 RXPEIEE	ECC Inject Error Enable for Rx Parser Memory Indicates whether the ECC inject error for receive parser memory is enabled. If this field is 1, it enables the ECC error injection feature. When this field becomes 0, it disables the ECC error injection feature.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disabled 1b - Enabled
19-17 —	Reserved.
16 WRRDN	Read Write Control Controls the read and write operation to the receive parser memory. If this field is 1, it indicates the write operation to the receive parser memory. When this field is 0, it indicates the read operation to the receive parser memory. 0b - Read operation to the receive parser memory 1b - Write operation to the receive parser memory
15-8 —	Reserved.
7-0 ADDR	FRP Instruction Table Offset Address Indicates the ADDR of the 32-bit entry in receive parser instruction table. Each entry has 128-bit (4x32-bit words). The X depends on the configurable DWC_EQOS_FRP_ENTRIES. If DWC_EQOS_FRP_ENTRIES = 256 , then X = 9 If DWC_EQOS_FRP_ENTRIES = 128 , then X = 8 If DWC_EQOS_FRP_ENTRIES = 64, then X = 7

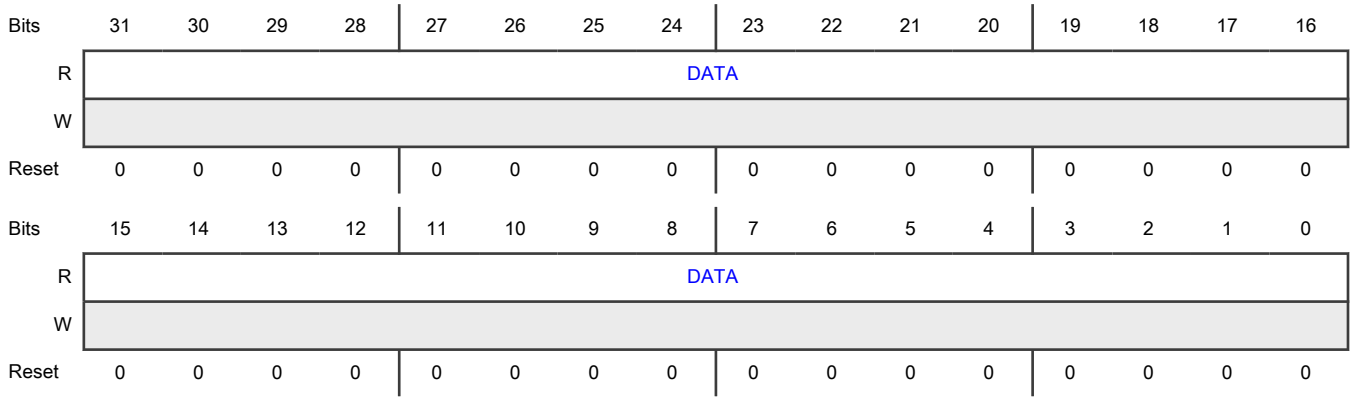
75.17.206 MTL Rx Parser Indirect Access Data (MTL_RXP_Indirect_Acc_Data)

Offset

Register	Offset
MTL_RXP_Indirect_Acc_Data	CB4h

Function
Provides the data associated to indirect access to receive parser memory.

Diagram



Fields

Field	Function
31-0 DATA	FRP Instruction Table Write/Read Data You must write this register before issuing any write command. After read command, when MTL_RXP_Indirect_Acc_Control_Status[STARTBUSY] = 0, the hardware provides the read data from the receive parser memory for read operation.

75.17.207 MTL ECC Control (MTL_ECC_Control)

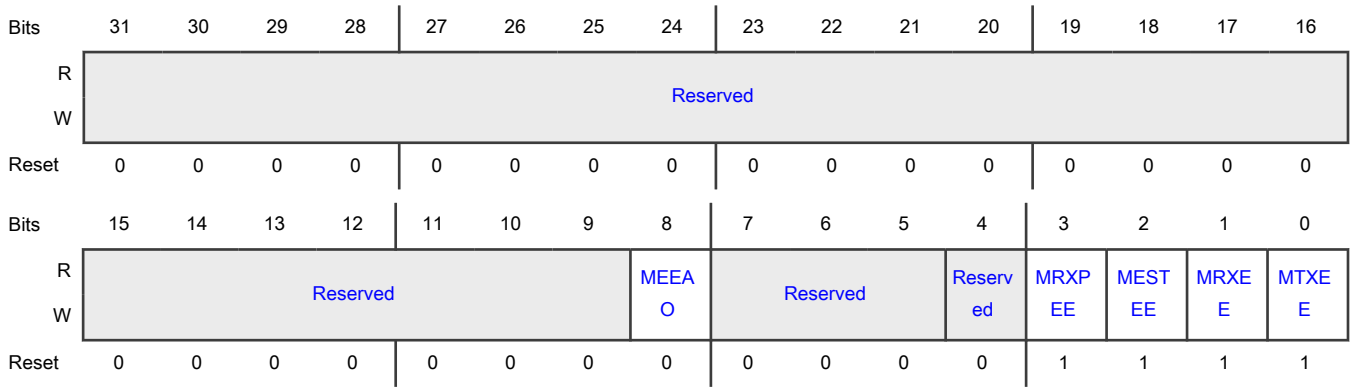
Offset

Register	Offset
MTL_ECC_Control	CC0h

Function

Establishes the operating mode of ECC related to MTL memories.

Diagram



Fields

Field	Function
31-9 —	Reserved.
8 MEEAO	MTL ECC Error Address Status Over-ride Enables or disables the MTL ECC error address status over-ride. When this field is 1, it indicates that MTL_ECC_Err_Addr_Status[EUEAS] and MTL_ECC_Err_Addr_Status[ECEAS] hold the last valid address where the error is detected. When this field becomes 0, it indicates that MTL_ECC_Err_Addr_Status[EUEAS] and MTL_ECC_Err_Addr_Status[ECEAS] hold the first address where the error is detected. 0b - Disable 1b - Enable
7-5 —	Reserved.
4 —	Reserved.
3 MRXPEE	MTL Rx Parser ECC Enable Indicates whether the MTL receive parser ECC is enabled. If this field is 1, it enables the ECC feature for receive parser memory. When this field is 0, it disables the ECC feature for receive parser memory. 0b - Disabled 1b - Enabled
2 MESTEE	MTL EST ECC Enable Indicates whether the MTL EST ECC is enabled. If this field is 1, it enables the ECC feature for EST memory. When this field is 0, it disables the ECC feature for EST memory. 0b - Disabled 1b - Enabled
1 MRXEE	MTL Rx FIFO ECC Enable Indicates whether the MTL Rx FIFO ECC is enabled. If this field is 1, it enables the ECC feature for MTL receive FIFO memory. When this field is 0, it disables the ECC feature for MTL receive FIFO memory. 0b - Disabled 1b - Enabled
0	MTL Tx FIFO ECC Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
MTXEE	Indicates whether the MTL Tx FIFO ECC is enabled. If this field is 1, it enables the ECC feature for MTL Tx FIFO memory. When field is 0, it disables the ECC feature for MTL Tx FIFO memory. 0b - Disabled 1b - Enabled

75.17.208 MTL Safety Interrupt Status (MTL_Safety_Interrupt_Status)

Offset

Register	Offset
MTL_Safety_Interrupt_Status	CC4h

Function

Provides safety interrupt status.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved	Reserved														
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved														MEUIS	MECIS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved Indicates whether the MAC safety uncorrectable interrupt status is detected. 0b - Not detected 1b - Detected
30-2	Reserved.

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
1 MEUIS	MTL ECC Uncorrectable Error Interrupt Status Indicates whether the MTL ECC uncorrectable error interrupt status is detected. Indicates that an uncorrectable error interrupt event in the MTL ECC safety feature. The application must read MTL_ECC_Interrupt_Status to get the exact cause of the interrupt. 0b - Not detected 1b - Detected
0 MECIS	MTL ECC Correctable Error Interrupt Status Indicates whether the MTL ECC correctable error interrupt status is detected. Indicates that a correctable error interrupt event in the MTL ECC safety feature. The application must read MTL_ECC_Interrupt_Status to get the exact cause of the interrupt . 0b - MTL ECC Correctable error Interrupt Status not detected 1b - MTL ECC Correctable error Interrupt Status detected

75.17.209 MTL ECC Interrupt Enable (MTL_ECC_Interrupt_Enable)

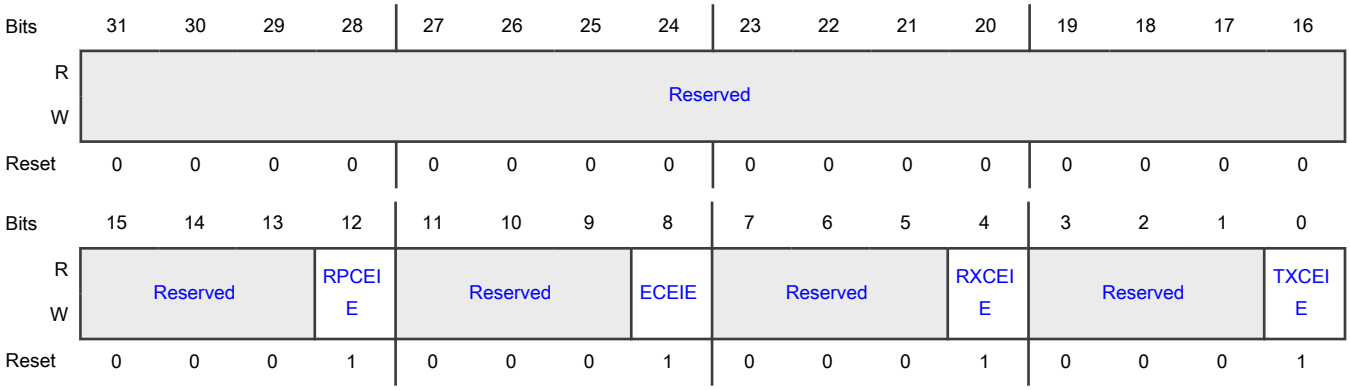
Offset

Register	Offset
MTL_ECC_Interrupt_Enable	CC8h

Function

Provides enable bits for the ECC interrupts.

Diagram



Fields

Field	Function
31-13 —	Reserved.
12 RPCEIE	Rx Parser Memory Correctable Error Interrupt Enable Indicates whether the Rx parser memory correctable error interrupt is enabled. When this field is 1, it generates an interrupt when an uncorrectable error is detected at the Rx parser memory interface. It is indicated in MTL_ECC_Interrupt_Status[RPCES]. When this field becomes 0, it indicates that this event does not generates an interrupt. 0b - Disabled 1b - Enabled
11-9 —	Reserved.
8 ECEIE	EST Memory Correctable Error Interrupt Enable Indicates whether the EST memory correctable error interrupt is enabled. When this field is 1, it generates an interrupt when a correctable error is detected at the MTL EST memory interface. It is indicated in MTL_ECC_Interrupt_Status[ECES]. When this field becomes 0, it indicates that this event does not generates an interrupt. 0b - Disabled 1b - Enabled
7-5 —	Reserved.
4 RXCEIE	Rx Memory Correctable Error Interrupt Enable Indicates whether the Rx memory correctable error interrupt is enabled. When this field is 1, it generates an interrupt when a correctable error is detected at the MTL Rx memory interface. It is indicated in MTL_ECC_Interrupt_Status[RXCES]. When this field becomes 0, it indicates that this event does not generates an interrupt. 0b - Disabled 1b - Enabled
3-1 —	Reserved.
0 TXCEIE	Tx Memory Correctable Error Interrupt Enable Indicates whether the Tx memory correctable error interrupt is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When this field is 1, it generates an interrupt when a correctable error is detected at the MTL Tx memory interface. It is indicated in MTL_ECC_Interrupt_Status[TXCES]. When this field becomes 0, it indicates that this event does not generates an interrupt. 0b - Disabled 1b - Enabled

75.17.210 MTL ECC Interrupt Status (MTL_ECC_Interrupt_Status)

Offset

Register	Offset
MTL_ECC_Interrupt_Status	CCCh

Function
Provides MTL ECC Interrupt Status.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved	RPUES	RPAMS	RPCES	Reserved	EUES	EAMS	ECES	Reserved	RXUES	RXAMS	RXCES	Reserved	TXUES	TXAMS	TXCES
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-15 —	Reserved.
14 RPUES	Rx Parser Memory Uncorrectable Error Status Indicates whether the Rx parser memory uncorrectable error status is detected. When this field is 1, it indicates that an uncorrectable error is detected at Rx parser memory interface.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Not detected 1b - Detected
13 RPAMS	MTL Rx Parser Memory Address Mismatch Status Indicates whether the MTL Rx parser memory address mismatch status is detected. When this field is 1, it indicates that an address mismatch is found for Rx parser memory's address bus. 0b - Not detected 1b - Detected
12 RPCES	MTL Rx Parser Memory Correctable Error Status Indicates whether the MTL Rx parser memory correctable error status is detected. When this field is 1, it indicates that a correctable error is detected at RX parser memory interface. 0b - Not detected 1b - Detected
11 —	Reserved.
10 EUES	MTL EST Memory Uncorrectable Error Status Indicates whether the MTL EST memory uncorrectable error status is detected. When this field is 1, it indicates that an uncorrectable error is detected at MTL EST memory interface. 0b - Not detected 1b - Detected
9 EAMS	MTL EST Memory Address Mismatch Status Indicates whether the MTL EST memory address mismatch status is detected. When this field is 1, it indicates that an address mismatch is found for MTL EST memory's address bus. 0b - Not detected 1b - Detected
8 ECES	MTL EST Memory Correctable Error Status Indicates whether the MTL EST memory correctable error status is detected. When this field is 1, it indicates that a correctable error is detected at the MTL EST memory. 0b - Not detected 1b - Detected
7 —	Reserved.

Table continues on the next page...

Table continued from the previous page...

Field	Function
6 RXUES	MTL Rx Memory Uncorrectable Error Status Indicates whether the MTL Rx memory uncorrectable error status is detected. When this field is 1, it indicates that an uncorrectable error is detected at the MTL Rx memory interface. 0b - Not detected 1b - Detected
5 RXAMS	MTL Rx Memory Address Mismatch Status Indicates whether the MTL Rx memory address mismatch status is detected. When this field is 1, it indicates that an address mismatch is found for MTL Rx memory's address bus. 0b - Not detected 1b - Detected
4 RXCES	MTL Rx memory Correctable Error Status Indicates whether the MTL Rx memory correctable error status is detected. When this field is 1, it indicates that a correctable error is detected at the MTL Rx memory. 0b - Not detected 1b - Detected
3 —	Reserved.
2 TXUES	MTL Tx Memory Uncorrectable Error Status Indicates whether the MTL Tx memory uncorrectable error status is detected. When this field is 1, it indicates that an uncorrectable error is detected at the MTL TX memory interface. 0b - Not detected 1b - Detected
1 TXAMS	MTL Tx Memory Address Mismatch Status Indicates whether the MTL Tx memory address mismatch status is detected. When this field is 1, it indicates that an address mismatch is found for MTL Tx memory's address bus. 0b - Not detected 1b - Detected
0 TXCES	MTL Tx Memory Correctable Error Status Indicates whether the MTL Tx memory correctable error status is detected. When this field is 1, it indicates that a correctable error is detected at the MTL Tx memory. 0b - Not detected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Detected

75.17.211 MTL ECC Error Status (MTL_ECC_Err_Sts_Rctl)

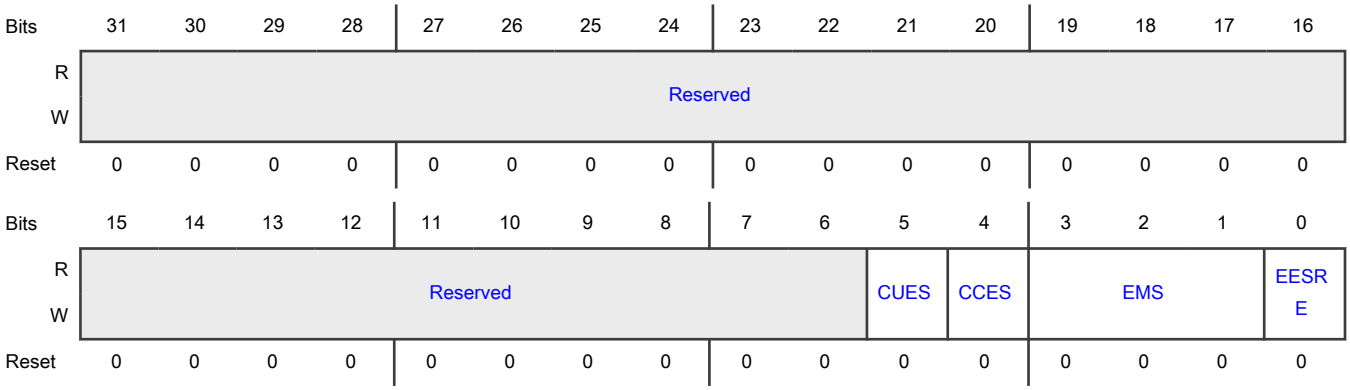
Offset

Register	Offset
MTL_ECC_Err_Sts_Rctl	CD0h

Function

Establishes the control for ECC error status capture.

Diagram



Fields

Field	Function
31-6 —	Reserved.
5 CUES	Clear Uncorrectable Error Status Indicates whether the clear uncorrectable error status is detected. When this field and MTL_ECC_Err_Sts_Rctl[EESRE] is 1, it indicates that based on the MTL_ECC_Err_Sts_Rctl[EMS], the respective memory's uncorrectable error address and uncorrectable error count values are cleared upon reading. When all the error status values are cleared hardware resets this field. 0b - Not detected 1b - Detected

Table continues on the next page...

Table continued from the previous page...

Field	Function
4 CCES	Clear Correctable Error Status Indicates whether the clear correctable error status is detected. When this field and MTL_ECC_Err_Sts_Rctl[EESRE] is 1, it indicates that based on the MTL_ECC_Err_Sts_Rctl[EMS],, the respective memory's correctable error address and correctable error count values are cleared upon reading. When all the error status values are cleared, hardware resets this field. 0b - Not detected 1b - Detected
3-1 EMS	MTL ECC Memory Selection Provides the memory selection encoding. When MTL_ECC_Err_Sts_Rctl[EESRE] is 1, this field indicates which memory's error status value to be read. The memory selection encoding is as described below. 000b - MTL Tx memory 001b - MTL Rx memory 010b - MTL EST memory 011b - MTL Rx Parser memory 100b - DMA TSO memory
0 EESRE	MTL ECC Error Status Read Enable Indicates whether the MTL ECC error status read is enabled. When this field is 1, it indicates that based on MTL_ECC_Err_Sts_Rctl[EMS], the respective memory's error status values are captured as described below: • The correctable and uncorrectable error count values are captured into MTL_ECC_Err_Cnt_Status. • The address location of correctable and uncorrectable errors are captured into MTL_ECC_Err_Addr_Status. When all the status values are captured into MTL_ECC_Err_Cnt_Status and MTL_ECC_Err_Addr_Status, hardware resets this field. 0b - Disabled 1b - Enabled

75.17.212 MTL ECC Error Address Status (MTL_ECC_Err_Addr_Status)

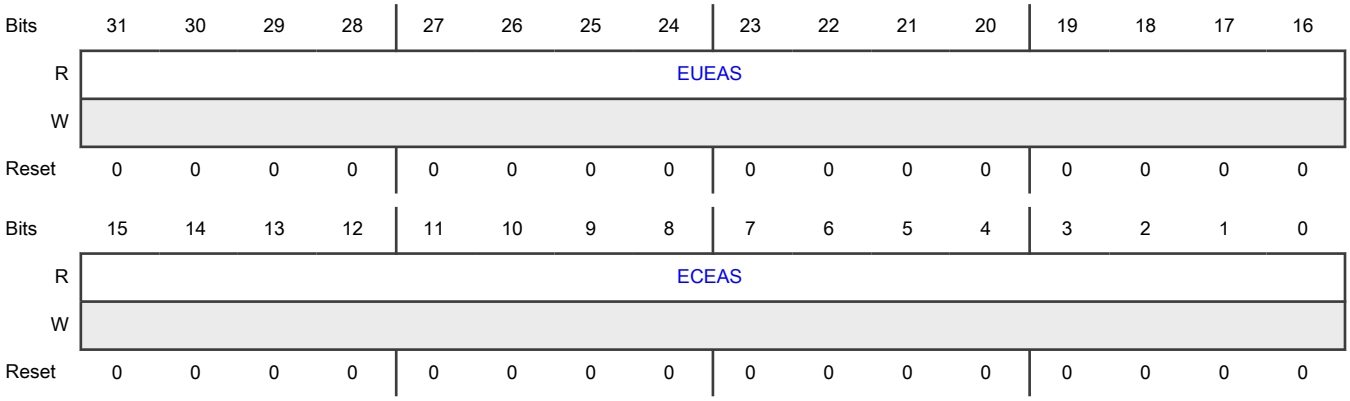
Offset

Register	Offset
MTL_ECC_Err_Addr_Status	CD4h

Function

Provides the memory addresses for the correctable and uncorrectable errors.

Diagram



Fields

Field	Function
31-16 EUEAS	MTL ECC Uncorrectable Error Address Status Based on the MTL_ECC_Err_Sts_Rctl[EMS], this field holds the respective memory's address locations for which an uncorrectable error or address mismatch is detected. When MTL_ECC_Control[MEEAO] = 1, this field holds the last valid address of memory for which either an uncorrectable error or an address mismatch is detected. When MTL_ECC_Control[MEEAO] = 0, this field holds the first address of the memory for which either an uncorrectable error or address mismatch is detected.
15-0 ECEAS	MTL ECC Correctable Error Address Status Based on the MTL_ECC_Err_Sts_Rctl[EMS], this field holds the respective memory's address locations for which a correctable error is detected. When MTL_ECC_Control[MEEAO] = 1, this field holds the last valid address of memory for which the correctable error or address mismatch is detected. When MTL_ECC_Control[MEEAO] = 0, this field holds the first address of the memory for which correctable error is detected.

75.17.213 MTL ECC Error Control Status (MTL_ECC_Err_Cntr_Status)

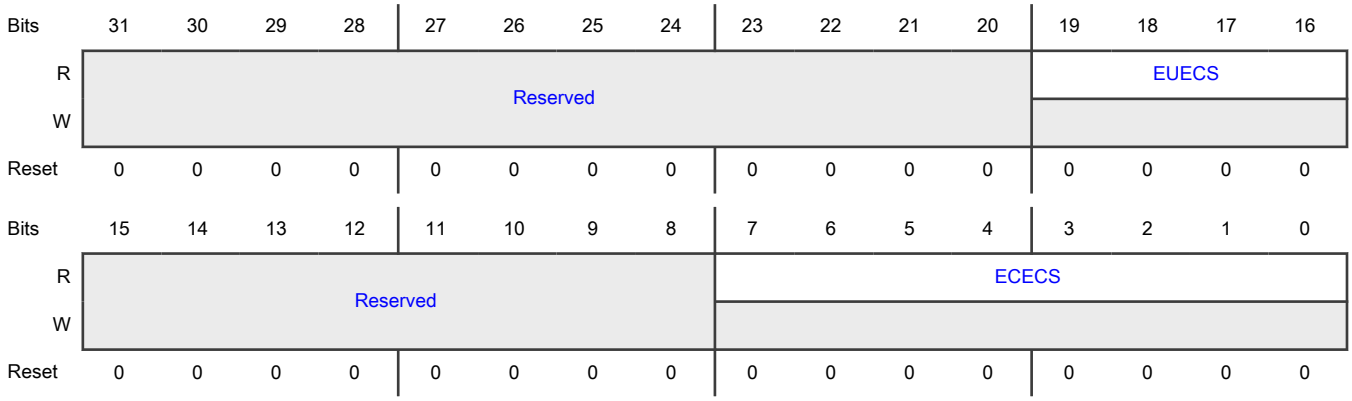
Offset

Register	Offset
MTL_ECC_Err_Cntr_Status	CD8h

Function

Provides ECC Error count for correctable and uncorrectable errors.

Diagram



Fields

Field	Function
31-20 —	Reserved.
19-16 EUECS	MTL ECC Uncorrectable Error Counter Status Based on the EMS field of MTL_ECC_Err_Cntr_Rctl register, this field holds the respective memory's uncorrectable error count value.
15-8 —	Reserved.
7-0 ECECS	MTL ECC Correctable Error Counter Status Based on the EMS field of MTL_ECC_Err_Cntr_Rctl register, this field holds the respective memory's correctable error count value.

75.17.214 MTL DPP Control (MTL_DPP_Control)

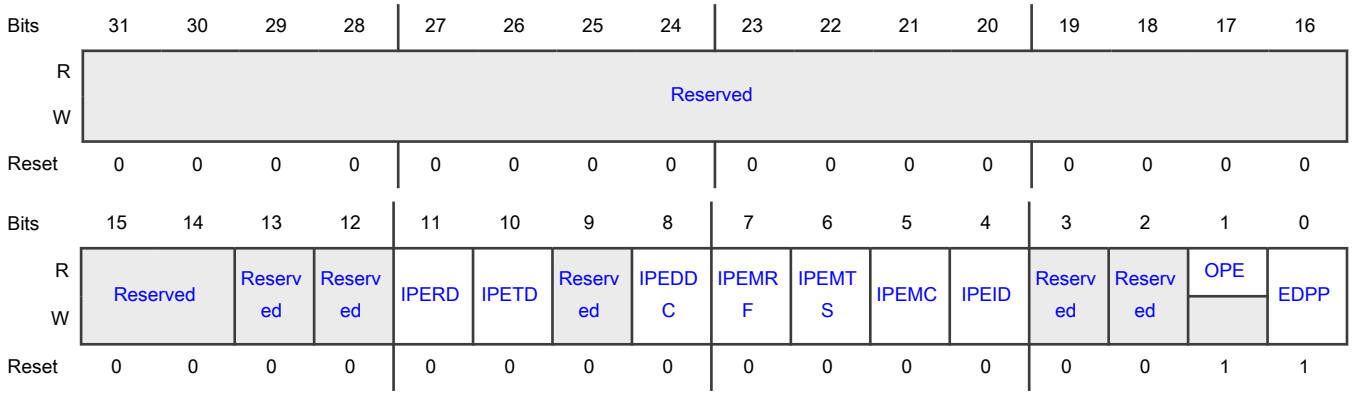
Offset

Register	Offset
MTL_DPP_Control	CE0h

Function

Establishes the operating mode of data parity protection and error injection.

Diagram



Fields

Field	Function
31-14 —	Reserved.
13 —	Reserved.
12 —	Reserved.
11 IPERD	Insert Parity error in Rx write-back Descriptor parity generator Indicates whether the insert parity error in receive write-back descriptor parity generator is enabled. When this field is 1, it flips the parity bit of first valid data to which the DMA Rx write-back descriptor parity generator(or at PG8 as shown in Receive data path parity protection diagram) generates. 0b - Disabled 1b - Enabled
10 IPETD	Insert Parity error in Tx write-back Descriptor parity generator Indicates whether the insert parity error in transit write-back descriptor parity generator is enabled. When this field is 1, it flips the parity bit of first valid data to which the DMA Tx write-back descriptor parity generator(or at PG4 as shown in Transmit data path parity protection diagram) generates. Hardware clears this field, when the respective parity bit flips. 0b - Disabled 1b - Enabled
9 —	Reserved.
8	Insert Parity Error in DMA DTX Control Word Parity Generator

Table continues on the next page...

Table continued from the previous page...

Field	Function
IPEDDC	Indicates whether the insert parity error in DMA DTX control word parity generator is enabled. When this field is 1, it flips the parity bit of first valid data to which the DMA DTX Control word parity generator (or at PG2 as shown in Transmit data path parity protection diagram) generates. Hardware clears this field, when the respective parity bit flips. 0b - Disabled 1b - Enabled
7 IPEMRF	Insert Parity Error in MTL Rx FIFO Read Control Parity Generator Inserts parity Error in MTL receive FIFO read control parity generator. When this field is 1, it flips the parity bit of first valid data to which the MTL Rx FIFO read control parity generator (or at PG7 as shown in Receive data path parity protection diagram) generates. Hardware clears this field, when the respective parity bit flips. 0b - Disabled 1b - Enabled
6 IPEMTS	Insert Parity Error in MTL Tx Status Parity Generator Indicates whether the insert parity error in MTL transit status parity generator is enabled. When this field is 1, it flips the parity bit of first valid data to which the MTL Tx Status parity generator (or at PG6 as shown in Transmit data path parity protection diagram) generates. Hardware clears this field, when the respective parity bit flips. 0b - Disabled 1b - Enabled
5 IPEMC	Insert Parity Error in MTL Checksum Parity Generator Indicates whether the insert parity error in MTL checksum parity generator is enabled. When this field is 1, it flips the parity bit of first valid data to which the MTL checksum parity generator (or at PG5 as shown in Transmit data path parity protection diagram) generates. Hardware clears this field, when the respective parity bit flips. 0b - Disabled 1b - Enabled
4 IPEID	Insert Parity Error in Interface Data Parity Generator Indicates whether the insert parity error in interface data parity generator is enabled. When this field is 1, it flips the parity bit of first valid input data to which the interface data parity generator (or at PG1 as shown in Transmit data path parity protection diagram) generates. Following are the input data bus on which parity bits are generated on the basis of configuration selected. In AHB Config, hrdata_i In AXI config, rdata_m_i

Table continues on the next page...

Table continued from the previous page...

Field	Function
	In DMA Config, mdc_rdata_i In MTL Config, ati_data_i Hardware clears this field, when the respective parity bit flips. 0b - Disabled 1b - Enabled
3 —	Reserved.
2 —	Reserved.
1 OPE	Odd Parity Enable Indicates whether an odd parity protection is enabled. When this field is 1, it enables an odd parity protection on all the external interfaces and when this field is 0, it enables an even parity protection on all the external interfaces. 0b - Disabled 1b - Enabled
0 EDPP	Enable Data path Parity Protection Enables or disables the data path parity protection. When this field is 1, it indicates that by generating and checking the parity on EQOS datapath it enables the parity protection for EQOS datapath . When this field is 0, it disables the parity protection for EQOS datapath. 0b - Disable 1b - Enable

75.17.215 MTL Tx Queue 0 Operation Mode (MTL_TxQ0_Operation_Mode)

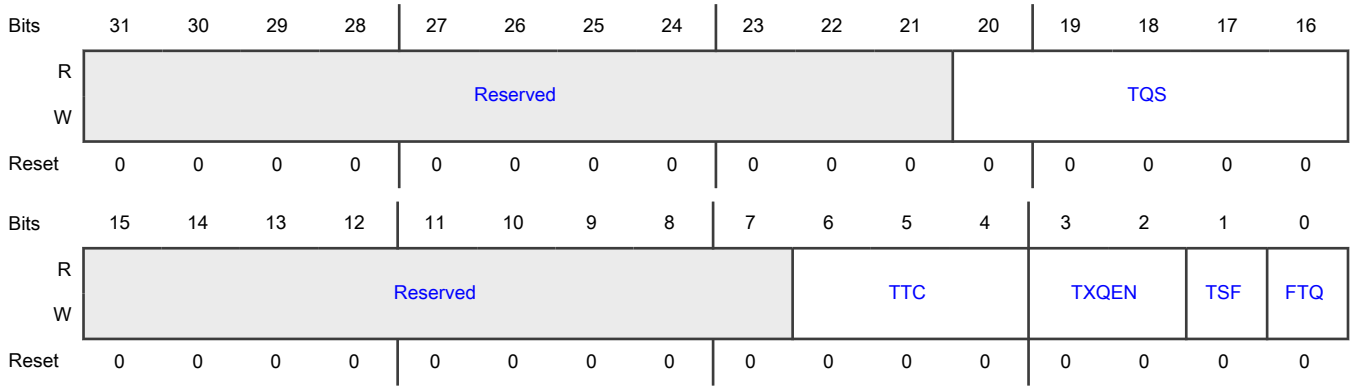
Offset

Register	Offset
MTL_TxQ0_Operation_Mode	D00h

Function

Establishes the transmit queue operating modes and commands.

Diagram



Fields

Field	Function
31-21 —	Reserved.
20-16 TQS	Transmit Queue Size Indicates the size of the allocated transmit queues in blocks of 256 bytes. This field is read-write only if the number of transit queues are more than one, the reset value is 0x0 and indicates 256 bytes size. This means that value of 0x0 = 256 bytes, 0x1 = 512 bytes and so on. You must program TQS [5:0] = 6'b001111 to allocate queue size of 4096 (4K) bytes . In general, the size of the Queue = (TQS+1)*256 bytes. When the number of transit queue is one, the field is read-only and the reset value reflects the configured TX FIFO size in blocks of 256 bytes. The field's width depends on the transit memory size selected in your configuration. For example, if the memory size is 2048, the field's width is 3 bits: LOG2(2048/256) = LOG2(8) = 3 bits
15-7 —	Reserved.
6-4 TTC	Transmit Threshold Control Controls the threshold level of the MTL transit queue. The transmission starts when the packet size within the MTL transit queue is larger than the threshold, also transmits full packets with length less than the threshold. These fields are used only when MTL_TxQ0_Operation_Mode[TSF] resets. 000b - 32 001b - 64 010b - 96 011b - 128 100b - 192 101b - 256

Table continues on the next page...

Table continued from the previous page...

Field	Function
	110b - 384 111b - 512
3-2 TXQEN	Transmit Queue Enable Enables or disables the transmit queue 0. 2'b00 - Not enabled 2'b01 - Reserved 2'b10 - Enabled 2'b11 - Reserved This field is read only in single queue configurations and read write in multiple queue configurations. **NOTE** In multiple transit queue configuration, all the queues disables by default and programming this field enables the transit queue. 00b - Not enabled 01b - Enable in AV mode (Reserved in non-AV) 10b - Enabled 11b - Reserved
1 TSF	Transmit Store and Forward Indicates whether the transmit store and forward is enabled. When this field is 1, it indicates that the transmission starts when a full packet resides in the MTL transit queue and the TTC values specified in Bits[6:4] of this register are ignored. This field must be changed only when you stop the transmission. 0b - Disabled 1b - Enabled
0 FTQ	Flush Transmit Queue Indicates whether the flush transmit queue is enabled. When this field is 1, it indicates that the transit queue controller logic resets to its default values. Therefore, all the data in the transit queue is lost or flushed. After the flushing operation completes this field resets internally. You must not write to the MTL_TxQ1_Operation_Mode until this field resets. The data to which the MAC transmitter has already accepted is not flushed. It is scheduled for transmission and results in underflow and runt packet transmission. **NOTE** The flush operation completes only when the transit queue is empty and the application accepts the pending transit status of all the transmitted packets. To complete the flush operation, the PHY Tx clock (CLK_TX_I) must be active.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Access restriction apply to this field. It clears automatically. Writing 1 sets this field and writing 0 has no effect. 0b - Disabled 1b - Enabled

75.17.216 MTL Tx Queue 0 Underflow (MTL_TxQ0_Underflow)

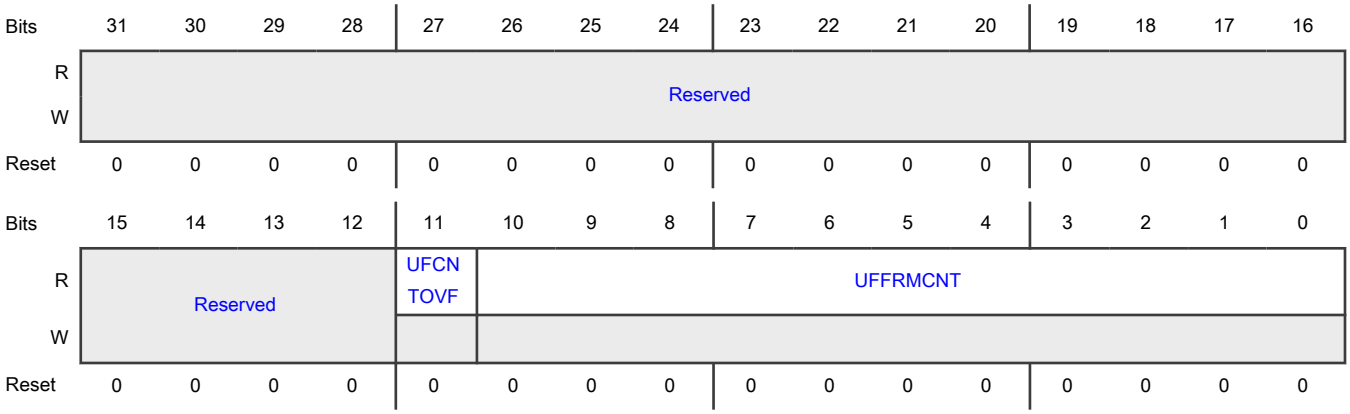
Offset

Register	Offset
MTL_TxQ0_Underflow	D04h

Function

Contains the counter for packets aborted because of transmit queue underflow and packets missed because of receive queue packet flush.

Diagram



Fields

Field	Function
31-12 —	Reserved.
11 UFCNTOVF	Overflow Bit for Underflow Packet Counter Indicates whether the overflow is detected for underflow packet counter.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is 1, when the transit queue underflow packet counter field overflows, that is, it has crossed the maximum count. In such a scenario, the overflow packet counter resets to all-zeros and this field indicates that the rollover occurred. Access restriction apply to this field. It clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
10-0 UFFRMCNT	Underflow Packet Counter Indicates the number of packets aborted by the controller because of transit queue underflow. This counter increments each time the MAC aborts outgoing packet because of underflow. The counter clears when this register is read with mci_be_i[0] at 1'b1. Access restriction apply to this field. It clears on read and automatically becomes 1 on an internal event occurrence.

75.17.217 MTL Tx Queue 0 Debug (MTL_TxQ0_Debug)

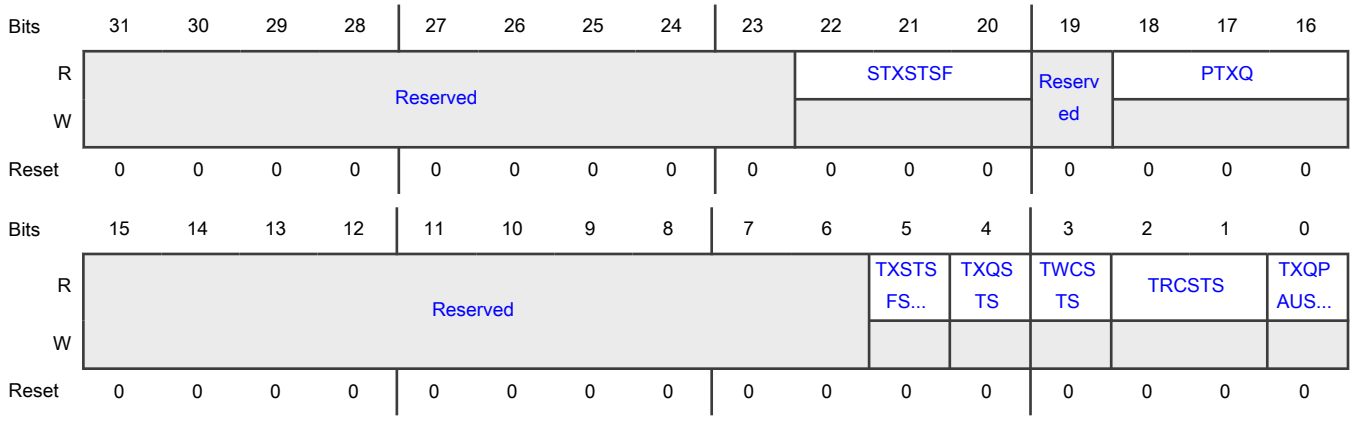
Offset

Register	Offset
MTL_TxQ0_Debug	D08h

Function

Provides the debug status of various blocks related to the transmit queue.

Diagram



Fields

Field	Function
31-23 —	Reserved.
22-20 STXSTSF	Number of Status Words in Tx Status FIFO of Queue Indicates the current number of status in the transit status FIFO of this queue. This field does not reflect the number of status words in transit status FIFO, when MTL_Operation_Mode[DTXSTS] = 1.
19 —	Reserved.
18-16 PTXQ	Number of Packets in the Transmit Queue Indicates the current number of packets in the transit queue. This field does not reflect the number of packets in the transmit queue, when MTL_Operation_Mode[DTXSTS] = 1.
15-6 —	Reserved.
5 TXSTSFSTS	MTL Tx Status FIFO Full Status Indicates whether the MTL transit status and FIFO full status is detected. When this field is 1, it indicates that the MTL transit status FIFO is full. Therefore, MTL cannot accept any more packets for transmission. 0b - Not detected 1b - Detected
4 TXQSTS	MTL Tx Queue Not Empty Status Indicates whether the MTL transit queue not empty status is detected. When this field is high, it indicates that the MTL transit queue is not empty and some data is left for transmission. 0b - Not detected 1b - Detected
3 TWCSTS	MTL Tx Queue Write Controller Status Indicates whether the MTL transit queue write controller status is detected. When high, this field indicates that the MTL transit queue write controller is active, and transfers the data to the transit queue. 0b - Not detected 1b - Detected

Table continues on the next page...

Table continued from the previous page...

Field	Function
2-1 TRCSTS	MTL Tx Queue Read Controller Status Indicates the state of the transit queue read controller. 00b - Idle state 01b - Read state (transferring data to the MAC transmitter) 10b - Waiting for pending transit status from the MAC transmitter 11b - Flushing the transit queue because of the packet abort request from the MAC
0 TXQPAUSED	Transmit Queue in Pause Indicates whether the transmit queue in pause is detected. When this field is 1 and the receive flow control is enabled, it indicates that the transit queue is in the pause condition (in the full-duplex only mode) because of the following scenario: • Receives PFC packet with the priorities assigned to the transit queue when PFC is enabled. • Receives 802.3x pause packet when PFC is disabled. 0b - Not detected 1b - Detected

75.17.218 MTL Tx Queue 0 ETS Status (MTL_TxQ0_ETS_Status)

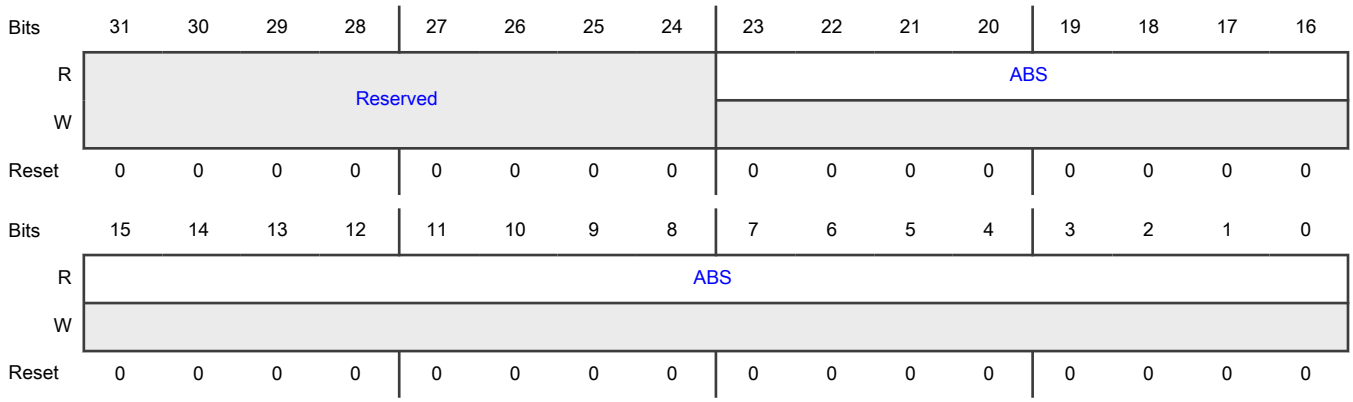
Offset

Register	Offset
MTL_TxQ0_ETS_Status	D14h

Function

Provides the average traffic transmitted in queue 0.

Diagram



Fields

Field	Function
31-24 —	Reserved.
23-0 ABS	Average Bits per Slot Contains the average transmitted bits per slot. Computes over every 10 million bit times slot (4 ms in 2500 Mbit/s; 10 ms in 1000 Mbit/s; 100 ms in 100 Mbit/s), when the DCB operation enables for queue 0. The maximum value is 0x989680.

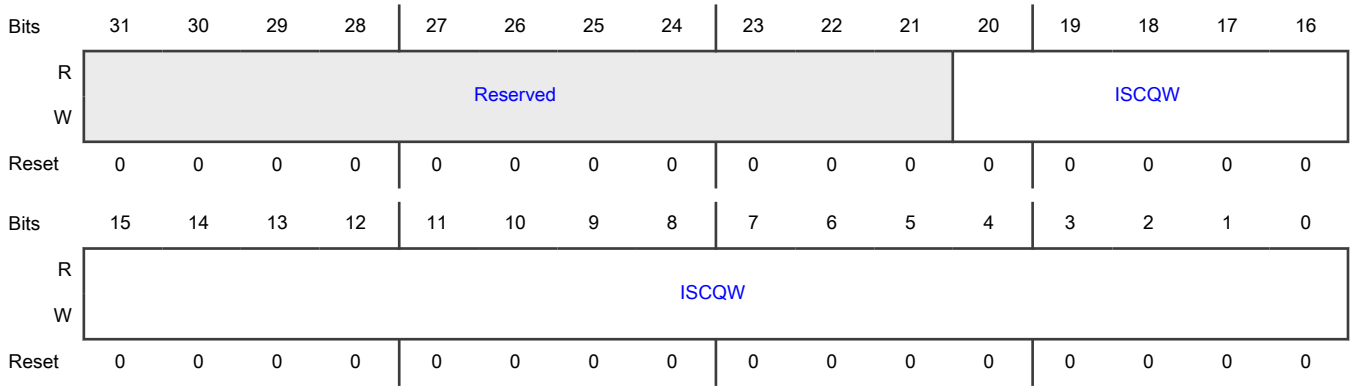
75.17.219 MTL Tx Queue Quantum Weight (MTL_TxQ0_Quantum_Weight)

Offset

Register	Offset
MTL_TxQ0_Quantum_Weight	D18h

Function
Contains the quantum value for Deficit Weighted Round Robin (DWRR), weights for the Weighted Round Robin (WRR), and Weighted Fair Queuing (WFQ) for queue 0.

Diagram



Fields

Field	Function
31-21 —	Reserved.
20-0	Quantum or Weights

Table continues on the next page...

Table continued from the previous page...

Field	Function
ISCQW	Contains the quantum value in bytes which is added to credit during every queue scanning cycle, when DCB operation enables with DWRR algorithm for queue 0 traffic. The maximum value is 0x1312D0 bytes. Contains the weight for this queue, when DCB operation enables with WFQ algorithm for queue 0 traffic. The maximum value is 0x3FFF where weight of 0 indicates 100% bandwidth. Write 0 to bits[20:14]. The higher the programmed weights, the lesser is the bandwidth that is allocated for the particular transmit queue. This is because the weights compute the packet finish time (weights*packet_size). The lesser the finish time, the higher is the probability of the packet getting scheduled first and consuming more bandwidth. Contains the weight for this queue, when DCB operation or generic queuing operation enables with WRR algorithm for queue 0 traffic. The maximum value is 0x64. Write 0 to bits [20:7].

75.17.220 MTL Queue 0 Interrupt Control Status (MTL_Q0_Interrupt_Control_Status)

Offset

Register	Offset
MTL_Q0_Interrupt_Control_Status	D2Ch

Function
Contains the interrupt enable and status bits for the queue 0 interrupts.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved							RXOIE	Reserved							RXOVFIS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved						ABPSIE	TXUIE	Reserved						ABPSIS	TXUNFIS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-25 —	Reserved.

Table continues on the next page...

Table continued from the previous page...

Field	Function
24 RXOIE	Receive Queue Overflow Interrupt Enable Enables or disables the receive queue overflow interrupt. When this field is 1, it enables the receive queue overflow interrupt. When this field becomes 0, it disables the receive queue overflow interrupt. 0b - Disabled 1b - Enabled
23-17 —	Reserved.
16 RXOVFIS	Receive Queue Overflow Interrupt Status Indicates whether the status of receive queue overflow interrupt is detected. This bit indicates that the receive queue had an overflow when you receive the packet. If you transfer a partial packet to the application, the overflow status sets in RDES3[21]. This field is 0 when the application writes 1 to it. Access restriction apply to this field. It becomes 0 automatically on an internal event occurrence. Writing 1 clears this field and writing 0 has no effect. 0b - Not detected 1b - Detected
15-10 —	Reserved.
9 ABPSIE	Average Bits Per Slot Interrupt Enable Enables or disables the average bits per slot interrupt. When this field is 1, it indicates that when you update the average bits per slot status, the MAC asserts the sbd_intr_o or mci_intr_o interrupt. When this field is 0, it indicates that the interrupt do not assert for such an event. 0b - Disabled 1b - Enabled
8 TXUIE	Transmit Queue Underflow Interrupt Enable Enables or disables transmit queue underflow interrupt. When this field is 1, it enables the transmit queue underflow interrupt. When this field becomes 0, it disables the transmit queue underflow interrupt. 0b - Disabled 1b - Enabled
7-2	Reserved.

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
1 ABPSIS	Average Bits Per Slot Interrupt Status Indicates whether the status of average bits per slot interrupt is detected. When this field is 1, it indicates that the MAC has updated the ABS value. This field is 0 when the application writes 1 it. Access restriction apply to this field. It becomes 0 automatically on an internal event occurrence. Writing 1 sets this field and writing 0 has no effect. 0b - Not detected 1b - Detected
0 TXUNFIS	Transmit Queue Underflow Interrupt Status Indicates whether the status of transmit queue underflow interrupt is detected. This field indicates that, when you transmit the packet,the transmit queue had an underflow. Suspend the transmission and write 1 to an underflow error TDES3[2]. This field is 0, when the application writes 1 to it. Access restriction apply to this field. It becomes 0 automatically on an internal event occurrence. Writing 1 sets this field and writing 0 has no effect. 0b - Not detected 1b - Detected

75.17.221 MTL Rx Queue 0 Operation Mode (MTL_RxQ0_Operation_Mode)

Offset

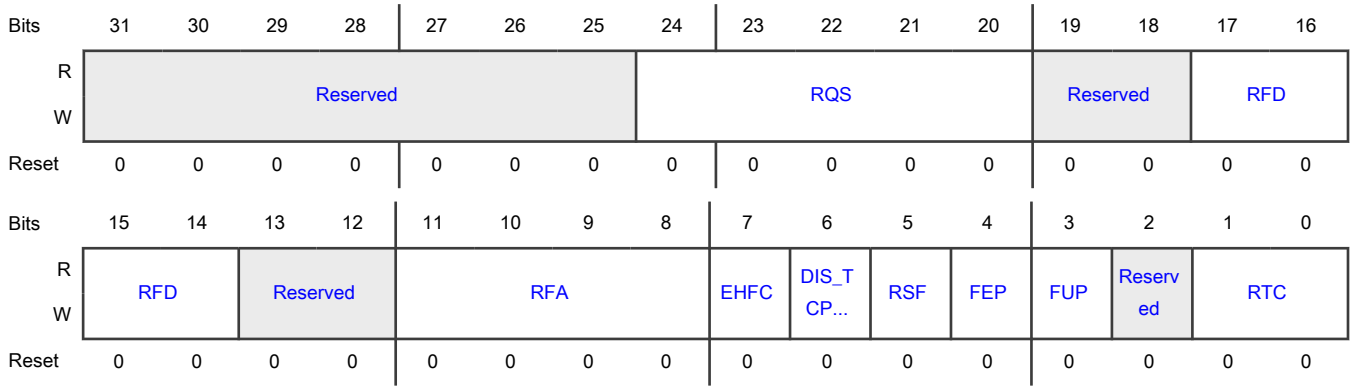
Register	Offset
MTL_RxQ0_Operation_Mode	D30h

Function

Establishes the receive queue operating modes and command.

The RFA and RFD fields are not backward compatible with the RFA and RFD fields of 4.00a release

Diagram



Fields

Field	Function
31-25 —	Reserved.
24-20 RQS	Receive Queue Size Indicates the size of the allocated receive queues in blocks of 256 bytes. The MTL_RxQ0_Operation_Mode[RQS] is read-write only if the number of receive queues are more than one, the reset value is 0x0 and indicates size of 256 bytes. This means that value of 0x0 = 256 bytes, 0x1 = 512 bytes and so on. You must program RQS [5:0] = 6'b001111 to allocate queue size of 4096 (4K) bytes. In general, the size of the Queue = (RQS+1)*256 bytes. When the number of receive queues is one, the field is read-only and the configured receive FIFO size in blocks of 256 bytes is reflected in the reset value. The field width depends on the receive memory size selected in your configuration. For example, if the memory size is 2048, the field width is 3 bits: LOG2(2048/256) = LOG2(8) = 3 bits
19-18 —	Reserved.
17-14 RFD	Threshold for Deactivating Flow Control (in half-duplex and full-duplex modes) Controls the threshold (fill-level of Rx queue) at which the flow control is de-asserts after activation. 0 - Full minus 1 KB, that is, FULL 1 KB 1 - Full minus 1.5 KB, that is, FULL 1.5 KB 2 - Full minus 2 KB, that is, FULL 2 KB 3 - Full minus 2.5 KB, that is, FULL 2.5 KB ... 14 - Full minus 8 KB, that is, FULL 8 KB 15 - Full minus 8.5 KB, that is, FULL 8.5 KB

Table continues on the next page...

Table continued from the previous page...

Field	Function
	De-assertion is effective only after flow control is asserted. **NOTE** You must program the value in such a way that the threshold is a positive number. When MTL_RxQ0_Operation_Mode[EHFC] = 1, these values are applicable only when the receive queue size which the MTL_RxQ0_Operation_Mode[RQS] determines is equal to or greater than 4 KB. For a given queue size, the values ranges between 0 and the encoding for FULL minus (QSIZE - 0.5 KB) and all other values are illegal. Here the term FULL and QSIZE refers to the queue size which the MTL_RxQ0_Operation_Mode[RQS] determines. The field width depends on RX FIFO size selected during the configuration. Remaining bits are reserved and read only.
13-12 —	Reserved.
11-8 RFA	Threshold for Activating Flow Control (in half-duplex and full-duplex) Controls the threshold (fill-level of receive queue) at which the flow control is activated. See MTL_RxQ0_Operation_Mode[RFD] for more information on encoding for this field.
7 EHFC	Enable Hardware Flow Control Enables or disables hardware flow control. If this field is 1, it enables the flow control signal operation, on the basis of the fill-level of receive queue. When this field becomes 0, it disables the flow control operation. 0b - Disable 1b - Enable
6 DIS_TCP_EF	Disable Dropping of TCP/IP Checksum Error Packets Enables or disable dropping of TCP or IP checksum error packets. When this field is 1, it indicates that the MAC does not drop the packets which only have the errors which the receive checksum offload engine detects. Such packets have errors only in the encapsulated payload. There are no errors (including FCS error) in the Ethernet packet received by the MAC. When this field becomes 0, it indicates that all error packets are dropped if MTL_RxQ0_Operation_Mode[FEP] resets. 0b - Enable 1b - Disable
5 RSF	Receive Queue Store and Forward Indicates whether the receive queue store and forward is enabled. When this field is 1, it indicates that the module reads a packet from the receive queue only after the complete packet has been written to it, ignores the MTL_RxQ0_Operation_Mode[RTC].

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When this field becomes 0, it indicates that the receive queue operates in the Threshold (cut-through) mode, subject to the threshold which the MTL_RxQ0_Operation_Mode[RTC] specifies. 0b - Disabled 1b - Enabled
4 FEP	Forward Error Packets Indicates whether the forward error packets are enabled. When this field becomes 0, it indicates that the receive queue drop packets with an error status (CRC error, GMII_ER, watchdog timeout, or overflow). However, the packet does not drop if the start byte (write) pointer of a packet is already transferred to the read controller side (in Threshold mode). When this field is 1, it indicates that all packets except the runt error packets are forwarded to the application or DMA. The packet is dropped irrespective of the setting of this field, if MTL_RxQ0_Operation_Mode[RSF] is 1 and the receive queue overflows when a partial packet is written. However, if MTL_RxQ0_Operation_Mode[RSF] becomes 0 and the receive queue overflows when a partial packet is written, a partial packet might be forwarded to the application or DMA. 0b - Disabled 1b - Enabled
3 FUP	Forward Undersized Good Packets Indicates whether the forward undersized good packets are enabled. When this field is 1, it indicates that the receive queue forwards the undersized good packets (packets with no error and length less than 64 bytes), including pad-bytes and CRC. When this field becomes 0, it indicates that the receive queue drops all packets of less than 64 bytes, unless a packet is already transferred because of the lower value of receive threshold, for example, MTL_RxQ0_Operation_Mode[RTC] = 01. 0b - Disabled 1b - Enabled
2 —	Reserved.
1-0 RTC	Receive Queue Threshold Control Controls the threshold level of the MTL receive queue (in bytes). The received packet is transferred to the application or DMA when the packet size within the MTL receive queue is larger than the threshold. In addition, automatically transfer full packets with length less than the threshold. This field is valid only when MTL_RxQ0_Operation_Mode[RSF] = 0. This field is ignored when MTL_RxQ0_Operation_Mode[RSF] = 1. 00b - 64

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01b - 32 10b - 96 11b - 128

75.17.222 MTL Rx Queue Missed Packet Overflow Count (MTL_RxQ0_Missed_Packet_Overflow_Cnt)

Offset

Register	Offset
MTL_RxQ0_Missed_Packet_Overflow_Cnt	D34h

Function

Contains the counter for packets missed because of receive queue packet flush and packets discarded because of receive queue overflow.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved				MISCNTO...	MISPKTCNT										
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved				OVFCNTO...	OVFPKTCNT										
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved.
27 MISCNTOVF	Missed Packet Counter Overflow Bit Indicates whether the missed packet counter overflow is detected. When this field is 1, it indicates that the receive queue missed packet counter has crossed the maximum limit.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Access restriction apply to this field. It clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
26-16 MISPKTCNT	Missed Packet Counter Indicates the number of packets this module has missed because the application asserts ari_pkt_flush_i[] for this queue. This counter resets when this register reads with mci_be_i[0] at 1b1. This counter increments by 1 when the DMA discards the packet because of buffer unavailability. Access restriction apply to this field. It clears on read and automatically becomes 1 on an internal event occurrence.
15-12 —	Reserved.
11 OVFCNTOVF	Overflow Counter Overflow Bit Indicates whether the counter overflow is detected. When this field is 1, it indicates that the receive queue overflow packet counter field has crossed the maximum limit. Access restriction apply to this field. It clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
10-0 OVFPKTCNT	Overflow Packet Counter Indicates the number of packets which this module discards because of receive queue overflow. This counter increments each time the module discards an incoming packet because of overflow. This counter resets when this register reads with mci_be_i[0] at 1'b1. Access restriction apply to this field. It clears on read and automatically becomes 1 on an internal event occurrence.

75.17.223 MTL Rx Queue 0 Debug (MTL_RxQ0_Debug)

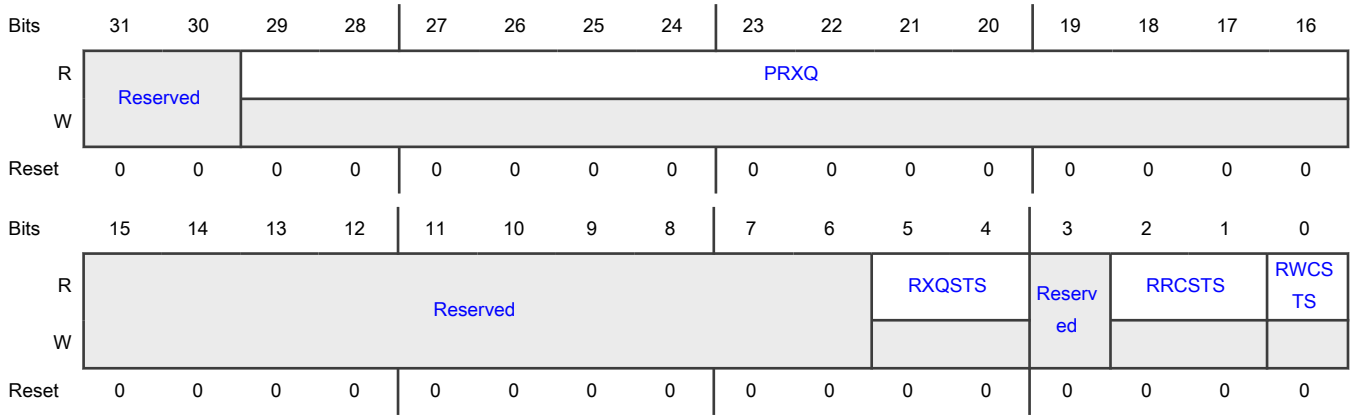
Offset

Register	Offset
MTL_RxQ0_Debug	D38h

Function

Provides the debug status of various blocks related to the receive queue.

Diagram



Fields

Field	Function
31-30 —	Reserved.
29-16 PRXQ	Number of Packets in Receive Queue Indicates the current number of packets in the receive queue. The theoretical maximum value for this field is 256KB/16B = 16K packets, that is, Max_Queue_Size/Min_Packet_Size.
15-6 —	Reserved.
5-4 RXQSTS	MTL Rx Queue Fill-Level Status Provides the status of the receive queue fill-level. 00b - Rx Queue empty 01b - Rx Queue fill-level below flow-control deactivate threshold 10b - Rx Queue fill-level above flow-control activate threshold 11b - Rx Queue full
3 —	Reserved.
2-1 RRCSTS	MTL Rx Queue Read Controller State Provides the state of the receive queue read controller. 00b - Idle state 01b - Reading packet data 10b - Reading packet status (or timestamp) 11b - Flushing the packet data and status

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 RWCSTS	MTL Rx Queue Write Controller Active Status Indicates whether the MTL receive queue write controller active status is detected. When high, this bit indicates that the MTL receive queue write controller is active, and it is transferring a received packet to the receive queue. 0b - Not detected 1b - Detected

75.17.224 MTL Rx Queue 0 Control 0 (MTL_RxQ0_Control)

Offset

Register	Offset
MTL_RxQ0_Control	D3Ch

Function

Controls the receive arbitration and passing of received packets to the application.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved												RXQ_FRM...	RXQ_WEGT		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-4 —	Reserved.
3 RXQ_FRM_ARBIT	Receive Queue Packet Arbitration Indicates whether the receive queue packet arbitration is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When this field is 1, it indicates that the module drives the packet data to the ARI interface such that the entire packet data of currently-selected queue transmits before switching to other queue. When this field becomes 0, it indicates that the module drives the packet data to the ARI interface such that the following amount of data of currently-selected queue transmits before switching to other queue: • PBL amount of data (indicated by ari_qN_pbl_i[]), or • Complete data of a packet The status and the timestamp are not a part of the PBL data. Therefore, the module drives the complete status (including timestamp status) during first PBL request for the packet (in store-and-forward mode) or the last PBL request for the packet (in Threshold mode). 0b - Disabled 1b - Enabled
2-0 RXQ_WEGT	Receive Queue Weight Indicates the weight assigned to receive queue 0. You must write a value to this field that is 1 less than the required queue weight. Therefore, if this field is 0, it indicates that the queue weight is 1, if this field is 1, it indicates that the queue weight is 2, and so on. You must use this weight to calculate the number of continuous PBL or packet requests, depending on the value of RXQ_FRM_ARBIT, allocated to the queue in an arbitration cycle. The field's value changes when the current service round completes or when there is a change from RAA=SP to RAA=WSP algorithm. This approach is required for a smooth transition. You must configure MTL Rx Queue 0 Control 0 (MTL_RxQ0_Control) before MTL Operation Mode (MTL_Operation_Mode) if you want to change the field's value before either of these two processes completes.

75.17.225 MTL Tx Queue 1 Operation Mode (MTL_TxQ1_Operation_Mode)

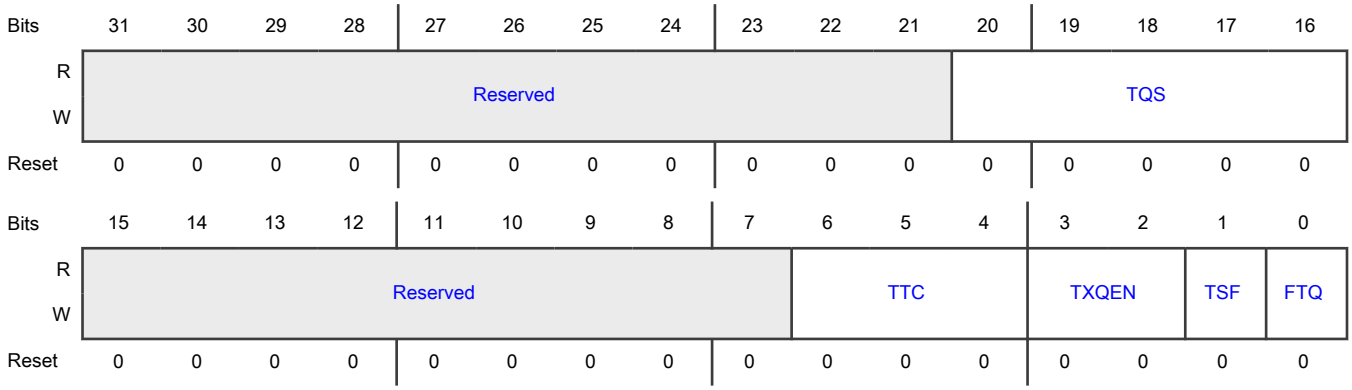
Offset

Register	Offset
MTL_TxQ1_Operation_Mode	D40h

Function

Establishes the transmit queue operating modes and commands.

Diagram



Fields

Field	Function
31-21 —	Reserved.
20-16 TQS	Transmit Queue Size Indicates the size of the allocated transmit queues in blocks of 256 bytes. This field is read-write only if the number of transit queues is more than one, the reset value is 0x0 and indicates size of 256 bytes. This means that value of 0x0 = 256 bytes, 0x1 = 512 bytes and so on. You must program TQS [5:0] = 6'b001111 to allocate queue size of 4096 (4K) bytes. In general, the size of the queue = (TQS+1)*256 bytes. When the number of transit queues is one, the field is read-only and the reset value reflects the configured TX FIFO size in blocks of 256 bytes. The field width depends on the transit memory size selected in your configuration. For example, if the memory size is 2048, the field width is 3 bits: LOG2(2048/256) = LOG2(8) = 3 bits
15-7 —	Reserved.
6-4 TTC	Transmit Threshold Control Controls the threshold level of the MTL Tx Queue. The transmission starts when the packet size within the MTL Tx Queue is larger than the threshold. In addition, it also transmit full packets with length less than the threshold. These fields are used only when the MTL_TxQ1_Operation_Mode[TSF] resets. 000b - 32 001b - 64 010b - 96 011b - 128 100b - 192 101b - 256

Table continues on the next page...

Table continued from the previous page...

Field	Function
	110b - 384 111b - 512
3-2 TXQEN	Transmit Queue Enable Enables or disables the transmit queue 0. 2'b00 - Not enabled 2'b01 - Enable in AV mode 2'b10 - Enabled 2'b11 - Reserved **NOTE** All the queues disables by default, in multiple transit queues configuration. You must program this field to enable the transit queue. 00b - Not enabled 01b - Enable in AV mode (Reserved in non-AV) 10b - Enabled 11b - Reserved
1 TSF	Transmit Store and Forward Indicates whether the transmit store and forward is enabled. When this field is 1, it indicates that the transmission starts when full packet resides in the MTL transit queue and the TTC values specified in MTL_TxQ1_Operation_Mode[TTC] are ignored. This field must change only when the transmission stops. 0b - Transmit Store and Forward is disabled 1b - Transmit Store and Forward is enabled
0 FTQ	Flush Transmit Queue Indicates whether the flush transmit queue is enabled. When this field is 1, it indicates that the transmit queue controller logic resets to its default values. Therefore, all the data in the transit queue is lost or flushed. This field resets internally when the flushing operation completes. Until this field becomes 0, you must not write to MTL_TxQ1_Operation_Mode. The data which the MAC transmitter has already accepted is not flushed. It is scheduled for transmission and results in an underflow and runt packet transmission. **NOTE** The flush operation completes only when the transit queue is empty and the application has accepted the pending transit status of all transmitted packets. To complete this flush operation, the PHY Tx clock (CLK_TX_I) must be active. Access restriction apply to this field. Writing 1 sets this field and it clears automatically. Writing 0 has no effect.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disabled 1b - Enabled

75.17.226 MTL Tx Queue 1 Underflow (MTL_TxQ1_Underflow)

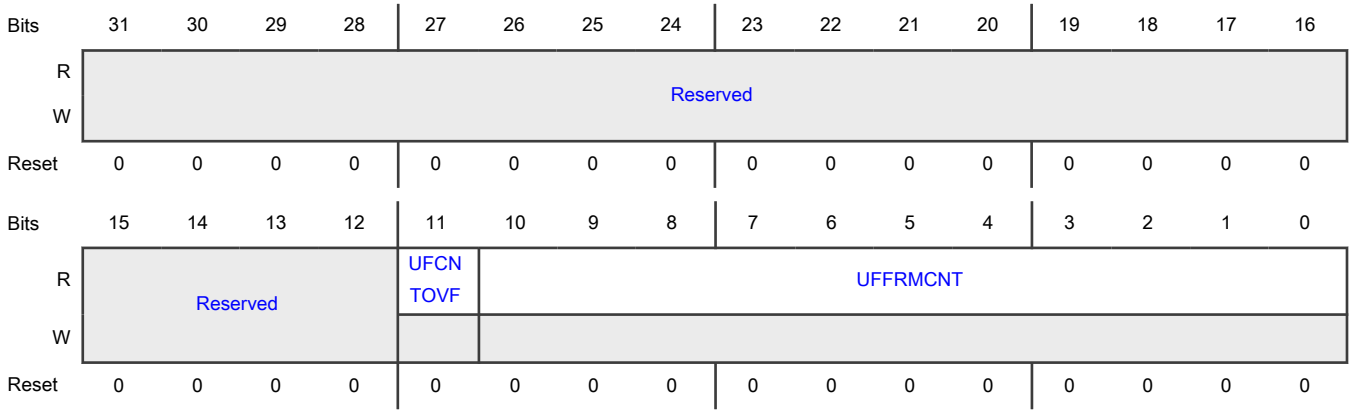
Offset

Register	Offset
MTL_TxQ1_Underflow	D44h

Function

Contains the counter for packets aborted because of transmit queue underflow and packets missed because of receive queue packet flush.

Diagram



Fields

Field	Function
31-12 —	Reserved.
11 UFCNTOVF	Overflow Bit for Underflow Packet Counter Indicates whether the overflow is detected for underflow packet counter. When this field is 1, it indicates that every time the transit queue underflow packet counter field overflows, that is, it has crossed the maximum count. In such case, the overflow packet counter resets to all-zeros and this field indicates that the rollover have occurred.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Access restriction apply to this field. It clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
10-0 UFFRMCNT	Underflow Packet Counter Indicates the number of packets which the controller abort because of transit queue underflow. This counter increments each time the MAC aborts outgoing packet because of underflow. The counter clears when this register is read with mci_be_i[0] at 1'b1. Access restriction apply to this field. It clears on read and automatically becomes 1 on an internal event occurrence.

75.17.227 MTL Tx Queue 1 Debug (MTL_TxQ1_Debug)

Offset

Register	Offset
MTL_TxQ1_Debug	D48h

Function

Provides the debug status of various blocks related to the transmit queue.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved									STXSTSF			Reserved	PTXQ		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved										TXSTSFS...	TXQSTS	TWCSTS	TRCSTS		TXQPAUS...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-23	Reserved.

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
22-20 STXSTSF	Number of Status Words in Tx Status FIFO of Queue Indicates the current number of status in the transit status FIFO of this queue. When MTL_Operation_Mode[DTXSTS] = 1, this field does not reflect the number of status words in transit status FIFO.
19 —	Reserved.
18-16 PTXQ	Number of Packets in the Transmit Queue Indicates the current number of packets in the transit queue. When MTL_Operation_Mode[DTXSTS] = 1, this field does not reflect the number of packets in the transmit queue.
15-6 —	Reserved.
5 TXSTSFSTS	MTL Tx Status FIFO Full Status Indicates whether the MTL transit status FIFO full status is detected. When this field is 1, it indicates that the MTL transit status FIFO is full. Therefore, the MTL cannot accept any more packets for transmission. 0b - Not detected 1b - Detected
4 TXQSTS	MTL Tx Queue Not Empty Status Indicates whether the MTL transit queue not empty status is detected. When this field is 1, it indicates that the MTL transit queue is not empty and some data is left for transmission. 0b - Not detected 1b - Detected
3 TWCSTS	MTL Tx Queue Write Controller Status Indicates whether the MTL transit queue write controller status is detected. When this field is 1, it indicates that the MTL transit queue write controller is active, and it is transferring the data to the transit queue. 0b - Not detected 1b - Detected
2-1 TRCSTS	MTL Tx Queue Read Controller Status Indicates the state of the transit queue read controller.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00b - Idle state 01b - Read state (transferring data to the MAC transmitter) 10b - Waiting for pending transit status from the MAC transmitter 11b - Flushing the transit queue because of the packet abort request from the MAC
0 TXQPAUSED	Transmit Queue in Pause Indicates whether the transmit queue in pause status is detected. When this field is 1 and the receive flow control is enabled, it indicates that the transit queue is in the pause condition (in the full-duplex only mode) because of these scenarios: • Reception of the PFC packet for the priorities assigned to the transit queue when PFC is enabled. • Reception of 802.3x pause packet when PFC is disabled. 0b - Not detected 1b - Detected

75.17.228 MTL Tx Queue 1 ETS Control (MTL_TxQ1_ETS_Control)

Offset

Register	Offset
MTL_TxQ1_ETS_Control	D50h

Function

Controls the enhanced transmission selection operation.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								SLC				CC	AVALG	Reserved	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-7 —	Reserved.
6-4 SLC	Slot Count If the credit-based shaper algorithm is enabled, then you can program the number of slots (of duration programmed in DMA_CH(#i)_Slot_Interval register) over which the average transmitted bits per slot, provided in the MTL_TxQ(#i)_ETS_Status register, is computed for queue. The encoding is as follows: 000b - 1 slot 001b - 2 slots 010b - 4 slots 011b - 8 slots 100b - 16 slots 101b - Reserved
3 CC	Credit Control Indicates whether the credit control is enabled. When this field is 1, it indicates that the accumulated credit parameter in the credit-based shaper algorithm logic is not reset to zero, when there is positive credit and no packet to transmit in channel 1. The credit accumulates even when no packet is waiting in channel 1 and another channel is transmitting. When this field becomes 0, it indicates that the accumulated credit parameter in the credit-based shaper algorithm logic sets to zero, when there is positive credit and no packet to transmit in channel 1. No credit accumulates when no packet is waiting in channel 1 and other channel is transmitting. 0b - Disabled 1b - Enabled
2 AVALG	AV Algorithm Indicates whether the CBS algorithm is enabled. If you program queue 1 for AV, this field configures the scheduling algorithm for this queue. When this field is 1, it indicates that the credit based shaper algorithm (CBS) is selected for queue 1 traffic. If this field becomes 0, strict priority is selected. 0b - Disabled 1b - Enabled
1-0 —	Reserved.

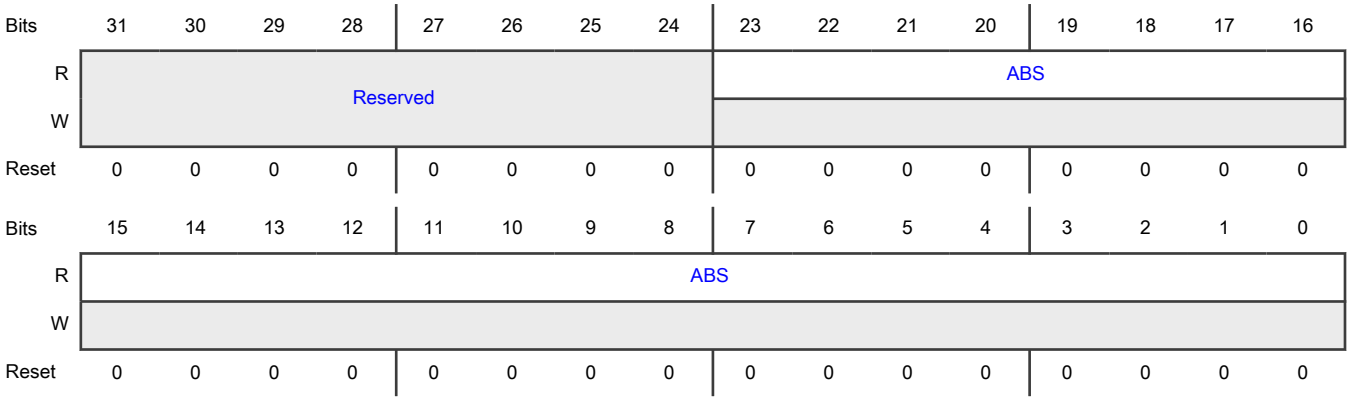
75.17.229 MTL Tx Queue 1 ETS Status (MTL_TxQ1_ETS_Status)

Offset

Register	Offset
MTL_TxQ1_ETS_Status	D54h

Function
Provides the average traffic transmitted in queue 1.

Diagram



Fields

Field	Function
31-24 —	Reserved.
23-0 ABS	Average Bits per Slot Contains the average transmitted bits per slot. Computes this field over number of slots programmed in the MTL_TxQ(#i)_ETS_CONTROL register's SLC field of, if you enables AV operation. This field's maximum value is 0x6_4000 in 100 Mbit/s, 0x3E_8000 in 1000 Mbit/s and 9C_4000 in 2500 Mbit/s mode respectively. Computes this field over every 10 million bit times slot (4 ms in 2500 Mbit/s; 10 ms in 1000 Mbit/s; 100 ms in 100 Mbit/s) when you enables DCB operation for queue. The maximum value is 0x989680.

75.17.230 MTL Tx Queue 1 Quantum Weight (MTL_TxQ1_Quantum_Weight)

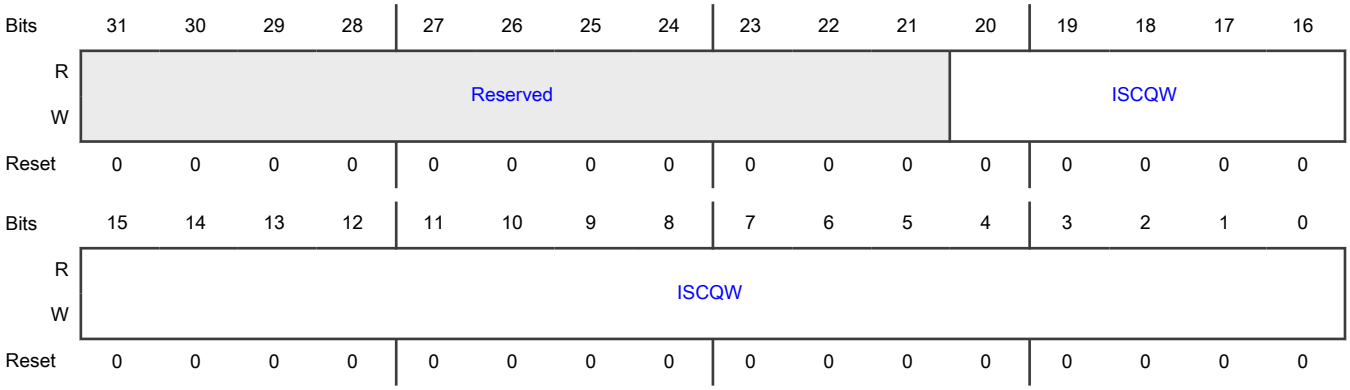
Offset

Register	Offset
MTL_TxQ1_Quantum_Weight	D58h

Function

Provides the average traffic transmitted in queue 1.

Diagram



Fields

Field	Function
31-21 —	Reserved.
20-0 ISCQW	idleSlopeCredit, Quantum or Weights idleSlopeCredit Contains the idleSlopeCredit value required for the credit-based shaper algorithm for queue 1, when you enables the AV feature. This is the rate of change of credit in bits per cycle (40 ns for 100 Mbit/s; 8 ns for 1000 Mbit/s; 3.2 ns for 2500 Mbit/s) when the credit is increasing. You must program this field with computed credit in bits per cycle scaled by 1,024. The maximum value is portTransmitRate, that is, 0x2000 in 1000/2500 Mbit/s mode and 0x1000 in 100 Mbit/s mode. Bits[20:14] must be written to zero. Quantum Contains the quantum value in bytes to be added to credit during every queue scanning cycle, when you enables the DCB operation with DWRR algorithm for queue 1 traffic. The maximum value is 0x1312D0 bytes. Weights Contains the weight for this queue, when you enables the DCB operation with WFQ algorithm for queue 1 traffic. The maximum value is 0x3FFF where weight of 0 indicates 100% bandwidth. Bits[20:14] must be written to zero. Contains the weight for this queue, when you enables the DCB operation or generic queuing operation with WRR algorithm for queue 1 traffic. The maximum value is 0x64. Bits [20:7] must be written to zero.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE In multiple Queue configuration you must program this field in respective per queue register to some non-zero value when multiple queues are enabled or single queue other than Q0 is enabled. You must not program this field when only Q0 is enabled. In general, you must program a non-zero value on both receive and transmit when WRR algorithm is selected. In receive, the register is MTL_Operation_Mode register. **NOTE** For WFQ algorithm, higher the programmed weights lesser the bandwidth allocated for that transmit queue. The finish time is not a function of particular packet alone but it is according to the formula: (previous_finish_time of particular Transmit Queue + (weights*packet_size)) **NOTE** The programmed weights do not correspond to the number of packets but the fraction of bandwidth or time allocated for particular queue w.r.t. total BW or time.

75.17.231 MTL Tx Queue 1 Sendslope Credit (MTL_TxQ1_SendSlopeCredit)

Offset

Register	Offset
MTL_TxQ1_SendSlopeCredit	D5Ch

Function

Contains the sendSlope credit value required for the credit-based shaper algorithm for the queue.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		SSC													
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-14 —	Reserved.
13-0 SSC	sendSlopeCredit Value Contains the sendSlopeCredit value required for credit-based shaper algorithm for Queue 1, when you enables the AV operation. This is the rate of change of credit in bits per cycle (40 ns, 8 ns and 3.2 ns for 100 Mbit/s, 1000 Mbit/s and 2500 Mbit/s respectively) when the credit is decreasing. You must program this field with computed credit in bits per cycle scaled by 1,024. The maximum value is portTransmitRate, that is, 0x2000 in 1000/2500 Mbit/s mode and 0x1000 in 100 Mbit/s mode. You must program this field with absolute sendSlopeCredit value. The credit-based shaper logic subtracts it from the accumulated credit when channel 1 is selected for transmission.

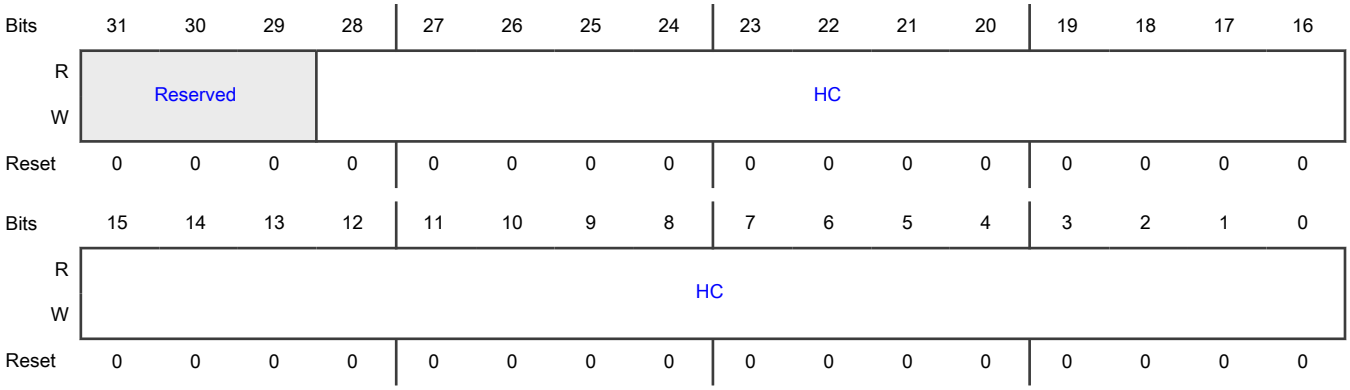
75.17.232 MTL Tx Queue 1 HiCredit (MTL_TxQ1_HiCredit)

Offset

Register	Offset
MTL_TxQ1_HiCredit	D60h

Function
Contains the hiCredit value required for the credit-based shaper algorithm for the queue.

Diagram



Fields

Field	Function
31-29 —	Reserved.

Table continues on the next page...

Table continued from the previous page...

Field	Function
28-0 HC	hiCredit Value Contains the hiCredit value required for the credit-based shaper algorithm, when you enables the AV feature. This is the maximum value that you can accumulate in the credit parameter. This is specified in bits scaled by 1,024. The maximum value is maxInterferenceSize, that is, best-effort maximum packet size (16,384 bytes or 131,072 bits). The specified value is 131,072 * 1,024 = 134,217,728 or 0x0800_0000.

75.17.233 MTL Tx Queue 1 LoCredit (MTL_TxQ1_LoCredit)

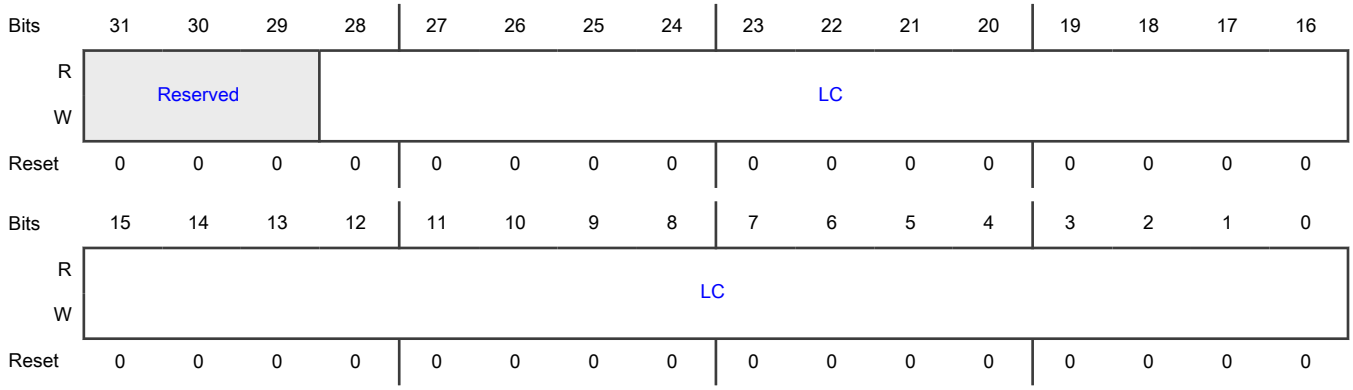
Offset

Register	Offset
MTL_TxQ1_LoCredit	D64h

Function

Contains the loCredit value required for the credit-based shaper algorithm for the queue.

Diagram



Fields

Field	Function
31-29 —	Reserved.
28-0 LC	loCredit Value Contains the loCredit value required for the credit-based shaper algorithm, when you enables the AV operation. This is the minimum value that you can accumulate in the credit parameter. This is specified in bits scaled by 1,024. The maximum value you can program corresponds to twice the maxFrameSize this queue transmits. If the maxFrameSize is 8192 bytes, then (8192*2) * 8 * 1024 = 134,217,728 or 0x0800_0000. The programmed value is 2's complement of the value, that is, 0x1800_0000 because it is a negative value.

75.17.234 MTL Queue 1 Interrupt Control Status (MTL_Q1_Interrupt_Control_Status)

Offset

Register	Offset
MTL_Q1_Interrupt_Control_Status	D6Ch

Function

Contains the interrupt enable and status bits for the queue 1 interrupts.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved							RXOIE	Reserved							RXOVFIS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved						ABPSIE	TXUIE	Reserved						ABPSIS	TXUNFIS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-25 —	Reserved.
24 RXOIE	Receive Queue Overflow Interrupt Enable Enables or disables the receive queue overflow interrupt. When this field is 1, it enables the receive queue overflow interrupt. When this field becomes 0, it disables the receive queue overflow interrupt. 0b - Disable 1b - Enable
23-17 —	Reserved.
16 RXOVFIS	Receive Queue Overflow Interrupt Status Indicates whether the status of receive queue overflow interrupt is detected. Indicates that the receive queue had an overflow when it receives the packet. If you transfers a partial packet to the application, the overflow status sets in RDES3[21]. This field is 0 when the application writes 1 to it.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Access restriction apply to this field. It automatically becomes 1 on an internal event occurrence. Writing 1 clears this field and writing 0 has no effect. 0b - Not detected 1b - Detected
15-10 —	Reserved.
9 ABPSIE	Average Bits Per Slot Interrupt Enable Indicates whether an average bits per slot interrupt is enabled. When this field is 1, it indicates that the MAC asserts the sbd_intr_o or mci_intr_o interrupt when you update the average bits per slot status. When this field is 0, it indicates that the interrupt is not asserted for such an event. 0b - Disabled 1b - Enabled
8 TXUIE	Transmit Queue Underflow Interrupt Enable Enables or disables the transmit queue underflow interrupt. When this field is 1, it enables the transmit queue underflow interrupt. When this field becomes 0, it disables the transmit queue underflow interrupt. 0b - Disabled 1b - Enabled
7-2 —	Reserved.
1 ABPSIS	Average Bits Per Slot Interrupt Status Indicates whether the average bits per slot interrupt status is detected. When this field is 1, it indicates that the MAC has updated the ABS value. This field is 0 when the application writes 1 to it. Access restriction apply to this field. It automatically becomes 1 on an internal event occurrence. Writing 1 clears this field and writing 0 has no effect. 0b - Not detected 1b - Detected
0 TXUNFIS	Transmit Queue Underflow Interrupt Status Indicates whether the transmit queue underflow interrupt status is detected. Indicates that the transmit queue had an underflow when you transmits the packet. Suspends the transmission and sets an underflow error TDES3[2]. This field is 0 when the application writes 1 to this field.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Access restriction apply to this field. It automatically becomes 1 on an internal event occurrence. Writing 1 clears this field and writing 0 has no effect. 0b - Not detected 1b - Detected

75.17.235 MTL Rx Queue 1 Operation Mode (MTL_RxQ1_Operation_Mode)

Offset

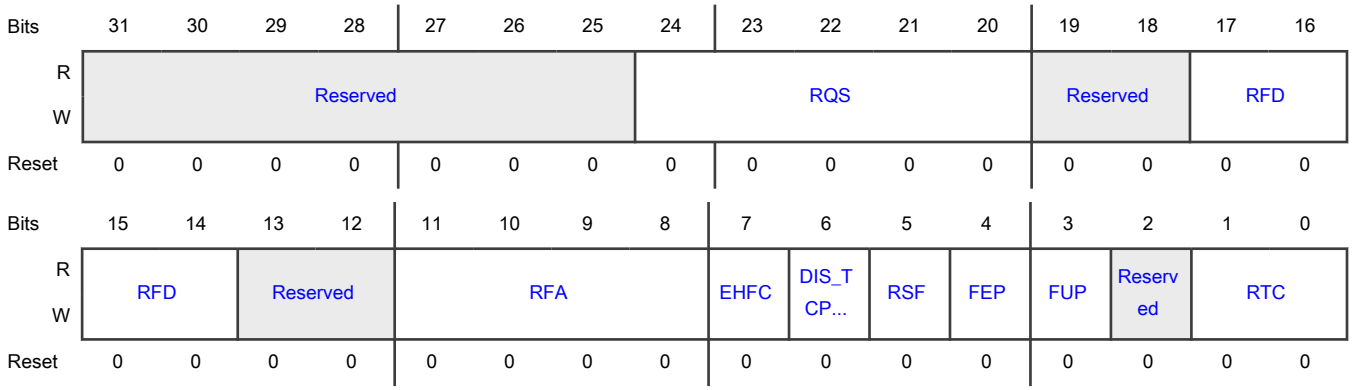
Register	Offset
MTL_RxQ1_Operation_Mode	D70h

Function

Establishes the receive queue operating modes and command.

The MTL_RxQ1_Operation_Mode[RFA] and MTL_RxQ1_Operation_Mode[RFD] are not backward compatible with the RFA and RFD fields of 4.00a release.

Diagram



Fields

Field	Function
31-25 —	Reserved.
24-20 RQS	Receive Queue Size Indicates the size of the allocated receive queues in blocks of 256 bytes. MTL_RxQ1_Operation_Mode[RQS] is read-write only if the number of receive queues is more than one, the

Table continues on the next page...

Table continued from the previous page...

Field	Function
	reset value is 0x0 and indicates 256 bytes size. This means that value of 0x0 = 256 bytes, 0x1 = 512 bytes and so on. You must program RQS [5:0] = 6'b001111 to allocate queue size of 4096 (4K) bytes. In general, the size of the Queue = (RQS+1)*256 bytes. When the number of receive queues is one, the field is read-only and the reset value reflects the configured RX FIFO size in blocks of 256 bytes. The field width depends on the receive memory size selected in your configuration. For example, if the memory size is 2048, the field width is 3 bits: LOG2(2048/256) = LOG2(8) = 3 bits
19-18 —	Reserved.
17-14 RFD	Threshold for Deactivating Flow Control (in half-duplex and full-duplex modes) Controls the threshold (fill-level of Rx queue) at which the flow control de-asserts after activation 0 - Full minus 1 KB, that is, FULL 1 KB 1 - Full minus 1.5 KB, that is, FULL 1.5 KB 2 - Full minus 2 KB, that is, FULL 2 KB 3 -Full minus 2.5 KB, that is, FULL 2.5 KB ... 14 - Full minus 8 KB, that is, FULL 8 KB 15 - Full minus 8.5 KB, that is, FULL 8.5 KB The de-assertion is effective only when flow control asserts. **NOTE** Program the value in such a way that the threshold is a positive number. When MTL_RxQ1_Operation_Mode[EHFC] = 1, these values are applicable only when the receive queue size which MTL_RxQ1_Operation_Mode[RQS] determines, is equal to or greater than 4 KB. For a given queue size, the values ranges between 0 and the encoding for FULL minus (QSIZE - 0.5 KB) and all other values are illegal. Here the term FULL and QSIZE refers to the queue size which MTL_RxQ1_Operation_Mode[RQS] determines. The field width depends on RX FIFO size selected during the configuration. Remaining bits are reserved and read only.
13-12 —	Reserved.
11-8 RFA	Threshold for Activating Flow Control (in half-duplex and full-duplex Controls the threshold (fill-level of Rx queue) at which the flow control activates. See MTL_RxQ1_Operation_Mode[RFD], for more information on encoding for this field.

Table continues on the next page...

Table continued from the previous page...

Field	Function
7 EHFC	Enable Hardware Flow Control Enables or disables hardware flow control. If this field is 1, it enables the flow control signal operation, based on the receive queue's fill-level. When this field becomes 0, it disables the flow control operation. 0b - Disable 1b - Enable
6 DIS_TCP_EF	Disable Dropping of TCP or IP Checksum Error Packets Indicates whether the dropping of TCP or IP checksum error packets is enabled. When this field is 1, it indicates that the MAC does not drop the packets which only have the errors that the receive checksum offload engine detects. Such packets have errors only in the encapsulated payload. MAC receives an Ethernet packet in which there are no errors (including FCS error). When this field becomes 0, it indicates that all the error packets drop if MTL_RxQ1_Operation_Mode[FEP] resets. 0b - Enabled 1b - Disabled
5 RSF	Receive Queue Store and Forward Indicates whether the receive queue store and forward is enabled. When this field is 1, it indicates that the module reads a packet from the receive queue only after the complete packet is written to it and ignores MTL_RxQ1_Operation_Mode[RTC]. When this field becomes 0, it indicates that the receive queue operates in the Threshold (cut-through) mode, subject to the threshold which MTL_RxQ1_Operation_Mode[RTC] specifies. 0b - Disabled 1b - Enabled
4 FEP	Forward Error Packets Indicates whether the forward error packets are enabled. If this field becomes 0, it indicates that the receive queue drops the packets with an error status (CRC error, GMII_ER, watchdog timeout, or overflow). However, the packet does not drops, if the start byte (write) pointer of a packet is already transferred to the read controller side in Threshold mode. When this field is 1, it indicates that all the packets except the runt error packets forward to the application or DMA. If MTL_RxQ1_Operation_Mode[RSF] is 1 and the receive queue overflows when a partial packet is written, the packet drops irrespective of the setting of this field. However, if MTL_RxQ1_Operation_Mode[RSF] resets and the receive queue overflows when a partial packet is written, you may forward a partial packet to the application or DMA. 0b - Disabled 1b - Enabled
3	Forward Undersized Good Packets

Table continues on the next page...

Table continued from the previous page...

Field	Function
FUP	Indicates whether the forward undersized good packets are enabled. When this field is 1, it indicates that the receive queue forwards the undersized good packets (packets with no error and length less than 64 bytes), including pad-bytes and CRC. When this field becomes 0, it indicates that the receive queue drops all the packets of less than 64 bytes, unless a packet is already transferred because of the lower value of Rx Threshold, for example, RTC = 01. 0b - Disabled 1b - Enabled
2 —	Reserved.
1-0 RTC	Receive Queue Threshold Control Controls the threshold level of the MTL Rx queue (in bytes): The received packet is transferred to the application or DMA when the packet size within the MTL receive queue is larger than the threshold. Also, the full packets with length less than the threshold are automatically transferred. This field is valid only when MTL_RxQ1_Operation_Mode[RSF] = 0. This field is ignored when MTL_RxQ1_Operation_Mode[RSF] = 1. 00b - 64 01b - 32 10b - 96 11b - 128

75.17.236 MTL Rx Queue 1 Missed Packet Overflow Counter (MTL_RxQ1_Missed_Packet_Overflow_Cnt)

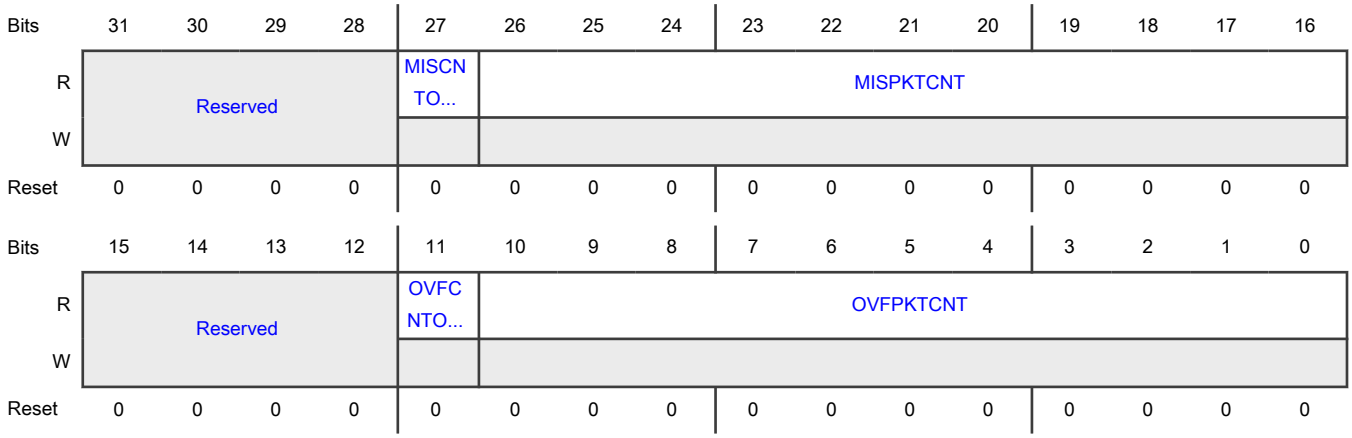
Offset

Register	Offset
MTL_RxQ1_Missed_Packet_Overflow_Cnt	D74h

Function

Contains the counter for packets missed because of receive queue packet flush and packets discarded because of receive queue overflow.

Diagram



Fields

Field	Function
31-28 —	Reserved.
27 MISCNTOVF	Missed Packet Counter Overflow Bit Indicates whether the missed packet counter overflow is detected. When this field is 1, it indicates that the receive queue missed packet counter has crossed the maximum limit. Access restriction apply to this field. It clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
26-16 MISPKTCNT	Missed Packet Counter Indicates the number of packets the module has missed because the application asserted ari_pkt_flush_i[] for this queue. This counter resets when this register is read with mci_be_i[0] at 1b1. This counter increases by 1 when the DMA discards the packet because of buffer unavailability. Access restriction apply to this field. It clears on read and automatically becomes 1 on an internal event occurrence.
15-12 —	Reserved.
11 OVFCNTOVF	Overflow Counter Overflow Bit Indicates whether the counter overflow status is detected. When this field is 1, it indicates that the receive queue overflow packet counter field has crossed the maximum limit.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Access restriction apply to this field. It clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not detected 1b - Detected
10-0 OVFPKTCNT	Overflow Packet Counter Indicates the number of packets which the module discards because of receive queue overflow. This counter increments each time the module discards an incoming packet because of overflow. It resets when this register is read with mci_be_i[0] at 1'b1. Access restriction apply to this field. It clears on read and automatically becomes 1 on an internal event occurrence.

75.17.237 MTL Rx Queue 1 Debug (MTL_RxQ1_Debug)

Offset

Register	Offset
MTL_RxQ1_Debug	D78h

Function
Provides the debug status of various blocks related to the receive queue.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved		PRXQ													
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved										RXQSTS		Reserved	RRCSTS		RWCSTS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30	Reserved.

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
29-16 PRXQ	Number of Packets in Receive Queue Indicates the current number of packets in the receive queue. The theoretical maximum field value is 256KB/16B = 16K packets, that is, Max_Queue_Size/Min_Packet_Size.
15-6 —	Reserved.
5-4 RXQSTS	MTL Rx Queue Fill-Level Status Provides the status of the receive queue fill-level. 00b - Rx Queue empty 01b - Rx Queue fill-level below flow-control deactivate threshold 10b - Rx Queue fill-level above flow-control activate threshold 11b - Rx Queue full
3 —	Reserved.
2-1 RRCSTS	MTL Rx Queue Read Controller State Provides the state of the receive queue read controller. 00b - Idle state 01b - Reading packet data 10b - Reading packet status (or timestamp) 11b - Flushing the packet data and status
0 RWCSTS	MTL Rx Queue Write Controller Active Status Indicates whether the MTL receive queue write controller active status is detected. When this field is 1, it indicates that the MTL receive queue write controller is active, and it is transferring a received packet to the receive queue. 0b - Not detected 1b - Detected

75.17.238 MTL Rx Queue 1 Control (MTL_RxQ1_Control)

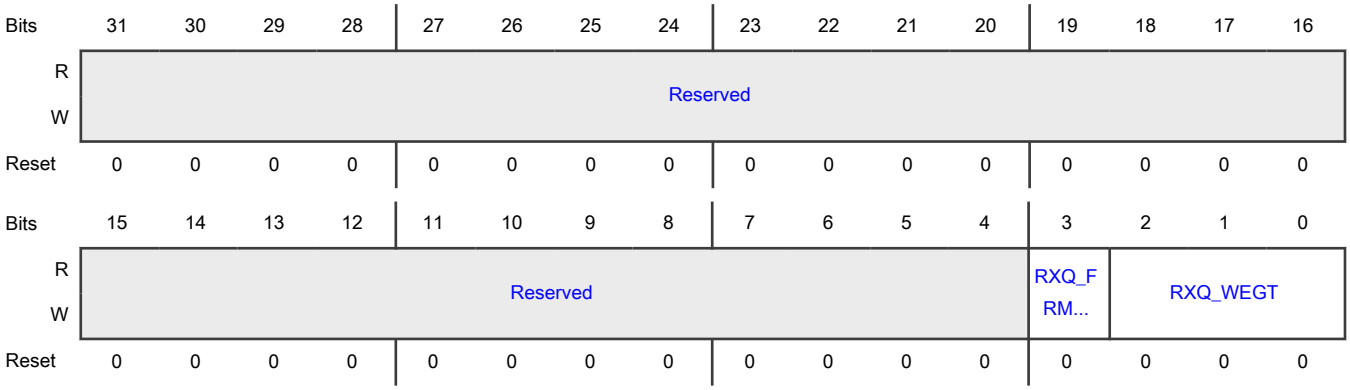
Offset

Register	Offset
MTL_RxQ1_Control	D7Ch

Function

Controls the receive arbitration and passing of received packets to the application.

Diagram



Fields

Field	Function
31-4 —	Reserved.
3 RXQ_FRM_ARBIT	Receive Queue Packet Arbitration Indicates whether the receive queue packet arbitration is enabled. When this field is 1, it indicates that the module drives the packet data to the ARI interface such that the entire packet data of currently-selected queue transmits before switching to other queue. When this field becomes 0, it indicates that the module drives the packet data to the ARI interface such that the following amount of data of currently-selected queue transmits before switching to other queue: • PBL amount of data (indicated by ari_qN_pbl_i[]), or • Complete data of a packet The status and the timestamp are not a part of the PBL data. Therefore, the module drives the complete status (including timestamp status) during the packet's first PBL request (in store-and-forward mode) or the last PBL request (in Threshold mode). 0b - Disabled 1b - Enabled
2-0 RXQ_WEGT	Receive Queue Weight Indicates the weight assigned to the receive queue 0. Program this field with one value less than the required weight, that is reset value of 0 indicates weight of 1, value of 1 indicates weight of 2, and so on. The weight is used as the number of continuous PBL or packets requests (depending on the RXQ_FRM_ARBIT) allocated to the queue in one arbitration cycle. Note: The change in value of RXQ_WEGT takes effect only after the completion of current service round or when there is change from RAA=SP to RAA=WSP algorithm. This approach is taken so that there is smooth transition. For the RXQ_WEGT value to take effect at the start, the MTL_RxQ(#i)_Control registers must be programmed before the MTL_Operation_Mode register.

75.17.239 DMA Mode (DMA_Mode)

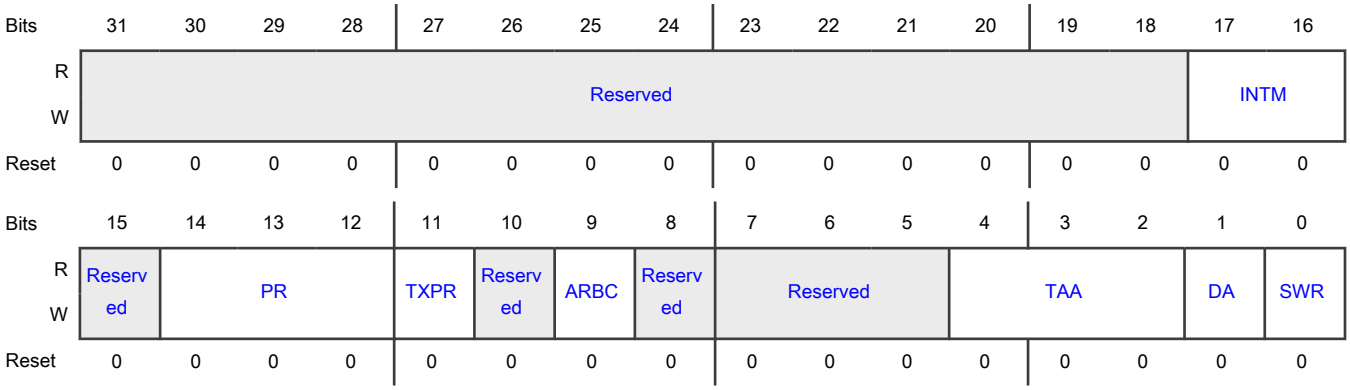
Offset

Register	Offset
DMA_Mode	1000h

Function

Establishes the bus operating modes for the DMA.

Diagram



Fields

Field	Function
31-18 —	Reserved.
17-16 INTM	Interrupt Mode Defines the interrupt mode of module. The behavior of the following outputs changes depending on these settings: sbd_perch_tx_intr_o[] (Transmit per channel interrupt) sbd_perch_rx_intr_o[] (Receive per channel interrupt) sbd_intr_o (Common interrupt) It also changes the behavior of the RI/TI bits in DMA_CH0_Status. 00 - sbd_perch_* are pulse signals for each TX/RX packet transfer completion events (irrespective of whether corresponding interrupts are enabled) for which IOC bits are enabled in descriptor. sbd_intr_o is also asserted when corresponding interrupts are enabled and cleared only when software clears the corresponding RI/TI status bits. 01 - sbd_perch_* are level signals asserted on TX/RX packet transfer completion event when corresponding interrupts are enabled and de-asserted when the software clears the corresponding RI/TI status bits. The sbd_intr_o is not asserted for these TX/RX packet transfer completion events.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10 - sbd_perch_* are level signals asserted on TX/RX packet transfer completion event when corresponding interrupts are enabled and de-asserted when the software clears the corresponding RI/TI status bits. However, the signal is asserted again if the same event occurred again before it was cleared. The sbd_intr_o is not asserted for these TX/RX packet transfer completion events. 11 - Reserved See table "Transfer Complete Interrupt Behavior" for more information. 00b - See above description 01b - See above description 10b - See above description 11b - Reserved
15 —	Reserved.
14-12 PR	Priority Ratio Controls the priority ratio in weighted round-robin arbitration between the Rx DMA and Tx DMA. These fields are valid only when DMA_Mode[DA] resets. The priority ratio is Rx:Tx or Tx:Rx depending on whether DMA_Mode[TXPR] = 1 or it resets. 000b - The priority ratio is 1:1 001b - The priority ratio is 2:1 010b - The priority ratio is 3:1 011b - The priority ratio is 4:1 100b - The priority ratio is 5:1 101b - The priority ratio is 6:1 110b - The priority ratio is 7:1 111b - The priority ratio is 8:1
11 TXPR	Transmit Priority Indicates whether the transmit priority is enabled. When this field is 1, it indicates that the Tx DMA has higher priority than the Rx DMA during arbitration for the system-side bus. 0b - Disabled 1b - Enabled
10 —	Reserved.
9 ARBC	Is reserved for NXP internal use. This field must always be 0 unless instructed by NXP. Writing 1 to this field may cause unexpected behavior.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - NXP reserved field disabled 1b - NXP reserved field enabled up on NXP request
8 —	Reserved.
7-5 —	Reserved.
4-2 TAA	Transmit Arbitration Algorithm Selects the arbitration algorithm for the transmit side when you select multiple transit DMAs. 000b - Fixed priority (Channel 0 has the lowest priority and the last channel has the highest priority) 001b - Weighted Strict Priority (WSP) 010b - Weighted Round-Robin (WRR) 011b - Reserved (for 3'b011 to 3'b111)
1 DA	DMA Tx or Rx Arbitration Scheme Specifies the arbitration scheme between the transmit and receive paths of all channels. 0 - Weighted round-robin with Rx:Tx or Tx:Rx The priority between the paths is according to the priority specified in bits[14:12] and the priority weight is specified in DMA_Mode[TXPR]. 1 - Fixed Priority The transit path has priority over the receive path when DMA_Mode[TXPR] = 1. Otherwise, the receive path has priority over the transit path. 0b - Weighted Round-Robin with Rx:Tx or Tx:Rx 1b - Fixed Priority
0 SWR	Software Reset Indicates whether the software reset is enabled. When this field is 1, it indicates that the MAC and the DMA controller resets the logic and all internal registers of the DMA, MTL, and MAC. This field automatically clears after the reset operation completes in all module clock domains. A value of zero should be read in this field before reprogramming any module register. When this field is 1, it indicates that this field must read at least 4 CSR clock cycles. **NOTE** The reset operation completes only when all resets in all active clock domains de-assert. Therefore, to complete software reset, it is essential that all PHY input clocks (applicable for the selected PHY interface) are present. The time to complete the software reset operation depends on the frequency of the slowest active clock.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Access restriction apply to this field. It clears automatically. Writing 1 sets this field and writing 0 has no effect. 0b - Disabled 1b - Enabled

75.17.240 DMA System Bus Mode (DMA_SysBus_Mode)

Offset

Register	Offset
DMA_SysBus_Mode	1004h

Function

Controls the behavior of the AHB or AXI master. It mainly controls burst splitting and the number of outstanding requests.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved	Reserved	Reserved				Reserved		Reserved						Reserved	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	RB	MB	Reserved	AAL	Reserved	Reserved	Reserved		Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	FB
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved.
30 —	Reserved.
29-26 —	Reserved.

Table continues on the next page...

Table continued from the previous page...

Field	Function
25-24 —	Reserved.
23-18 —	Reserved.
17-16 —	Reserved.
15 RB	Rebuild INCRx Burst Indicates whether the rebuild INCRx burst is enabled. When this field is 1 and the AHB master gets SPLIT, RETRY, or Early Burst Termination (EBT) response, it indicates that the AHB master interface rebuilds the pending beats of any initiated burst transfer with INCRx and SINGLE transfers. By default, the AHB master interface rebuilds pending beats of an EBT with an unspecified (INCR) burst. 0b - Disabled 1b - Enabled
14 MB	Mixed Burst Indicates whether the mixed burst is enabled. When this field is 1 and DMA_SysBus_Mode[FB] is 0, it indicates that the AHB master performs an undefined bursts transfers (INCR) for burst length of 16 or more. The AHB master performs fixed burst transfers (INCRx and SINGLE), for burst length of 16 or less. 0b - Disabled 1b - Enabled
13 —	Reserved.
12 AAL	Address-Aligned Beats Indicates whether the address-aligned beats are enabled. When this field is 1, it indicates that the EQOS-AXI or EQOS-AHB master performs address-aligned burst transfers on read and write channels. 0b - Disabled 1b - Enabled
11 —	Reserved.
10 —	Reserved.

Table continues on the next page...

Table continued from the previous page...

Field	Function
9-8 —	Reserved.
7 —	Reserved.
6 —	Reserved.
5 —	Reserved.
4 —	Reserved.
3 —	Reserved.
2 —	Reserved.
1 —	Reserved.
0 FB	Fixed Burst Length Indicates whether the fixed burst length is enabled. When this field is 1, it indicates that the AHB master initiates burst transfers of specified length (INCRx or SINGLE). When this field is 0, it indicates that the AHB master initiates transfers of unspecified length (INCR) or SINGLE transfers. 0b - Disabled 1b - Enabled

75.17.241 DMA Interrupt Status (DMA_Interrupt_Status)

Offset

Register	Offset
DMA_Interrupt_Status	1008h

Function

The application reads this Interrupt Status register during interrupt service routine or polling to determine the interrupt status of DMA channels, MTL queues, and the MAC.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved														MACIS	MTLIS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	DC1IS	DC0IS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-18 —	Reserved.
17 MACIS	MAC Interrupt Status Indicates whether the MAC interrupt status is detected. You must read the corresponding register in the MAC to get the exact cause of the interrupt and clear its source, to reset this field to 1'b0. 0b - Not detected 1b - Detected
16 MTLIS	MTL Interrupt Status Indicates whether the MTL interrupt status is detected. You must read the corresponding register in the MTL to get the exact cause of the interrupt and clear its source, to reset this field to 1'b0. 0b - Not detected 1b - Detected
15-8 —	Reserved.
7 —	Reserved.
6	Reserved.

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
5 —	Reserved.
4 —	Reserved.
3 —	Reserved.
2 —	Reserved.
1 DC1IS	DMA Channel 1 Interrupt Status Indicates whether the DMA channel 1 interrupt status is detected. You must read the corresponding register in DMA channel 1 to get the exact cause of the interrupt and clear its source, to reset this field to 1'b0. 0b - Not detected 1b - Detected
0 DC0IS	DMA Channel 0 Interrupt Status Indicates whether the DMA channel 0 interrupt status is detected. You must read the corresponding register in DMA channel 0 to get the exact cause of the interrupt and clear its source, to reset this field to 1'b0. 0b - Not detected 1b - Detected

75.17.242 DMA Debug Status 0 (DMA_Debug_Status0)

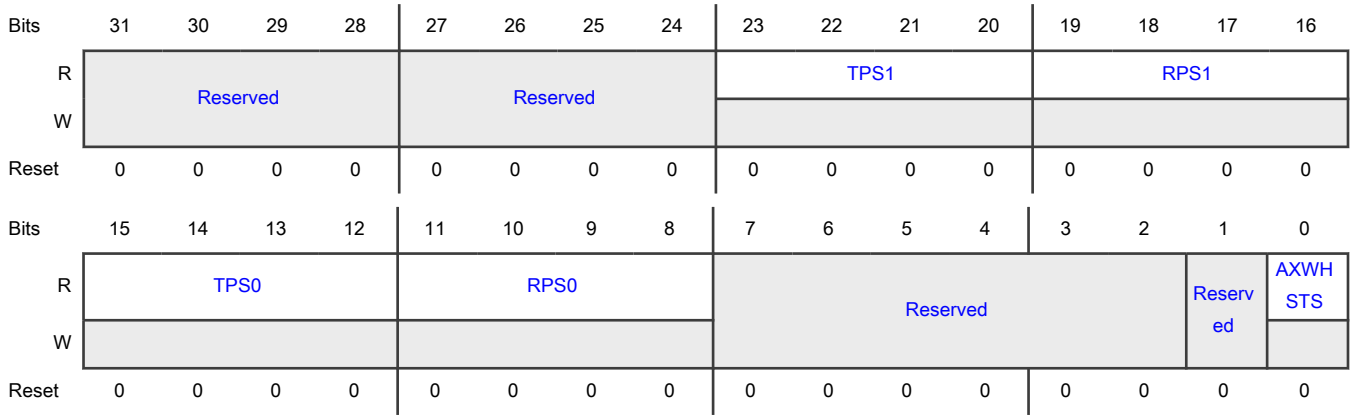
Offset

Register	Offset
DMA_Debug_Status0	100Ch

Function

Provides the receive and transmit process status for DMA Channel 0-Channel 2 for debugging purpose.

Diagram



Fields

Field	Function
31-28 —	Reserved.
27-24 —	Reserved.
23-20 TPS1	DMA Channel 1 Transmit Process State Indicates the transmit DMA FSM state for channel 1. The MSB of this field always returns 0. This field does not generate an interrupt. 0000b - Stopped (Reset or stop transmit command issued) 0001b - Running (Fetching transmit transfer descriptor) 0010b - Running (Waiting for status) 0011b - Running (Reading data from system memory buffer and queuing it to the transmit buffer (Tx FIFO)) 0100b - Timestamp write state 0101b - Reserved for future use 0110b - Suspended (Transmit descriptor unavailable or transmit buffer underflow) 0111b - Running (Closing transmit descriptor)
19-16 RPS1	DMA Channel 1 Receive Process State Indicates the receive DMA FSM state for channel 1. The MSB of this field always returns 0. This field does not generate an interrupt. 0000b - Stopped (Reset or Stop receive command issued) 0001b - Running (Fetching receive transfer descriptor) 0010b - Reserved for future use

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0011b - Running (Waiting for receive packet) 0100b - Suspended (Receive descriptor unavailable) 0101b - Running (Closing the receive descriptor) 0110b - Timestamp write state 0111b - Running (Transferring the received packet data from the receive buffer to the system memory)
15-12 TPS0	DMA Channel 0 Transmit Process State Indicates the transmit DMA FSM state for channel 0. The MSB of this field always returns 0. This field does not generate an interrupt. 0000b - Stopped (Reset or stop transmit command issued) 0001b - Running (Fetching transmit transfer descriptor) 0010b - Running (Waiting for status) 0011b - Running (Reading data from system memory buffer and queuing it to the transmit buffer (Tx FIFO)) 0100b - Timestamp write state 0101b - Reserved for future use 0110b - Suspended (Transmit descriptor unavailable or transmit buffer underflow) 0111b - Running (Closing transmit descriptor)
11-8 RPS0	DMA Channel 0 Receive Process State Indicates the receive DMA FSM state for channel 0. The MSB of this field always returns 0. This field does not generate an interrupt. 0000b - Stopped (Reset or stop receive command issued) 0001b - Running (Fetching receive transfer descriptor) 0010b - Reserved for future use 0011b - Running (Waiting for receive packet) 0100b - Suspended (Receive descriptor unavailable) 0101b - Running (Closing the receive descriptor) 0110b - Timestamp write state 0111b - Running (Transferring the received packet data from the receive buffer to the system memory)
7-2 —	Reserved.
1	Reserved.

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
0 AXWHSTS	AHB Master Status Indicates whether the AXI master write channel or AHB master status is detected. When this field is 1, it indicates that the AHB master FSMs are in the non-idle state. 0b - Not detected 1b - detected

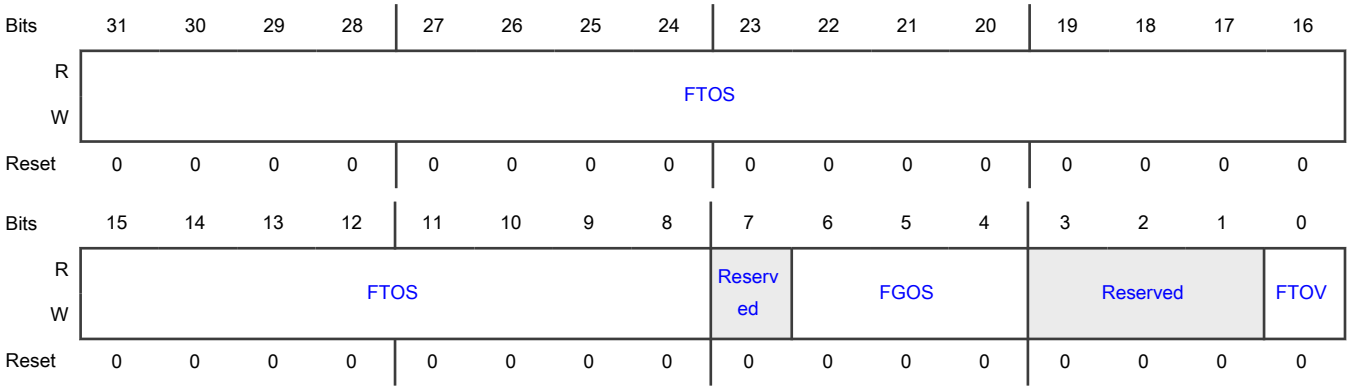
75.17.243 DMA TBS Control (DMA_TBS_CTRL)

Offset

Register	Offset
DMA_TBS_CTRL	1050h

Function
Controls the TBS attributes.

Diagram



Fields

Field	Function
31-8 FTOS	Fetch Time Offset Deduct the value in units of 256 nanoseconds, from the launch time to compute the fetch Time. Max value: 999,999,999 nanosecond, must be smaller than CTR-1 value when ESTM mode = 1, because this value is a modulo CTR value.
7	Reserved.

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
6-4 FGOS	Fetch GSN Offset Deduct the number GSN slots from the launch GSN to compute the fetch GSN. The value is valid only when DMA_TBS_CTRL[FTOV] = 1.
3-1 —	Reserved.
0 FTOV	Fetch Time Offset Valid Indicates whether the fetch time offset is valid. When this field is 1, it indicates that DMA_TBS_CTRL[FTOS] is valid. When this field is not 1, it indicates that the fetch offset is not valid and the DMA engine can fetch the frames from host memory without any time restrictions. 0b - Invalid 1b - Valid

75.17.244 DMA Safety Interrupt Status (DMA_Safety_Interrupt_Status)

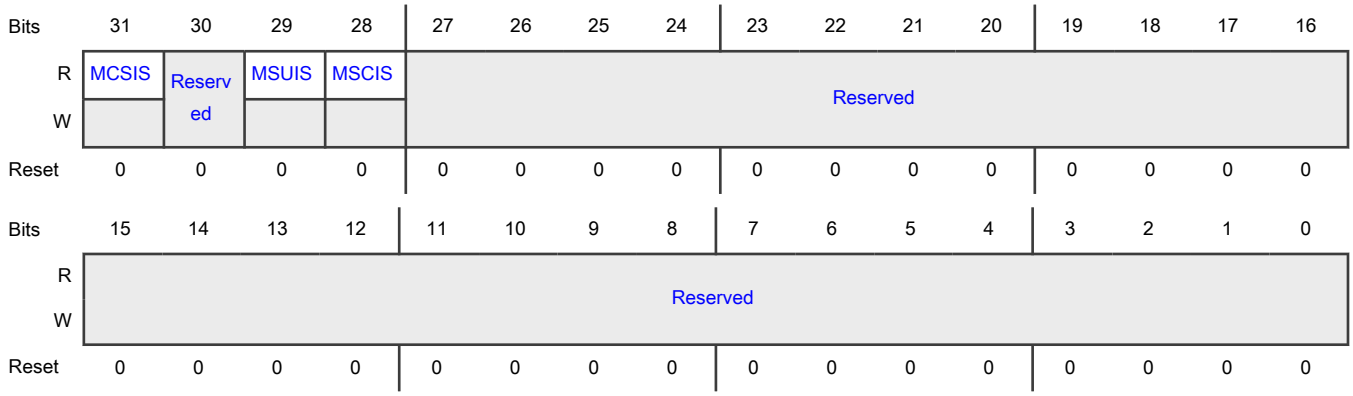
Offset

Register	Offset
DMA_Safety_Interrupt_Status	1080h

Function

Indicates summary (whether error occurred in DMA/MTL/MAC and correctable/uncorrectable) of the automotive safety related error interrupts.

Diagram



Fields

Field	Function
31 MCSIS	MAC Safety Uncorrectable Interrupt Status Indicates whether the MAC safety uncorrectable interrupt status is detected. Indicates that an uncorrectable safety related interrupt is 1 in the MAC module. When this field is 1, it indicates that you must read the MAC_DPP_FSM_Interrupt_Status, to get the cause of the safety interrupt in MAC. 0b - Not detected 1b - Detected
30 —	Reserved.
29 MSUIS	MTL Safety Uncorrectable Error Interrupt Status Indicates whether the MTL safety uncorrectable error interrupt status is detected. Indicates an uncorrectable error interrupt event in MTL. To get exact cause of the interrupt you must read the MTL_Safety_Interrupt_Status. 0b - Not detected 1b - Detected
28 MSCIS	MTL Safety Correctable Error Interrupt Status Indicates whether the MTL safety correctable error interrupt status is detected. Indicates a correctable error interrupt event in MTL. To get the exact cause of the interrupt you must read the MTL_Safety_Interrupt_Status. 0b - Not detected 1b - Detected
27-0 —	Reserved.

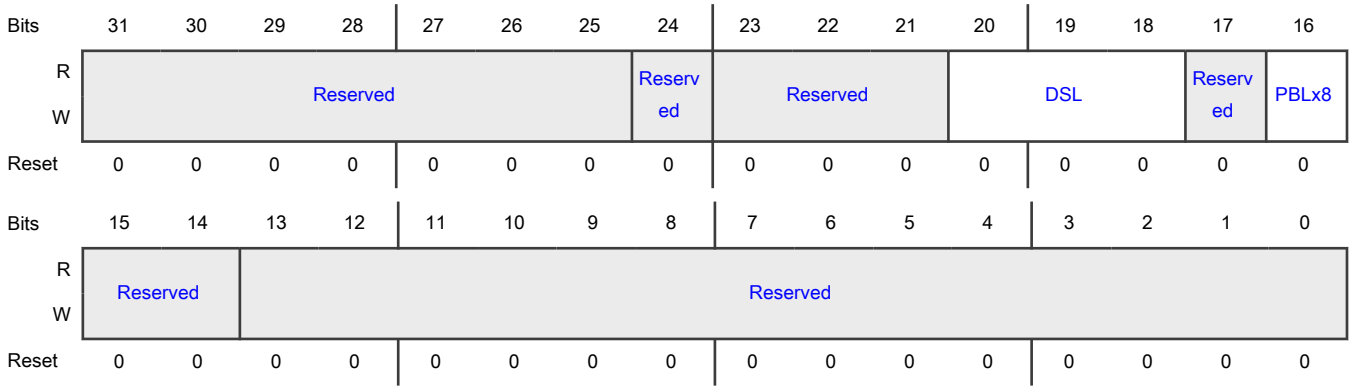
75.17.245 DMA Channel 0 Control (DMA_CH0_Control)

Offset

Register	Offset
DMA_CH0_Control	1100h

Function
Specifies the MSS value for segmentation, length to skip between two descriptors, and also the features such as header splitting and 8xPBL mode.

Diagram



Fields

Field	Function
31-25 —	Reserved.
24 —	Reserved.
23-21 —	Reserved.
20-18 DSL	Descriptor Skip Length Specifies the Word, Dword, or Lword number (depending on the 32-bit, 64-bit, or 128-bit bus) to skip between two unchained descriptors. The address skipping starts from the end of the current descriptor to the start of the next descriptor. The DMA assumes the descriptor table as contiguous, when the DSL value is equal to zero.
17 —	Reserved.
16 PBLx8	8xPBL mode Indicates whether 8xPBL mode is enabled. When this field is 1, it indicates that the PBL value programmed in Bits[21:16] in DMA_CH(#i)_Tx_Control and Bits[21:16] in DMA_CH(#i)_Rx_Control is multiplied by eight times. Therefore, the DMA transfers the data in 8, 16, 32, 64, 128, and 256 beats depending on the PBL value. 0b - Disabled 1b - Enabled
15-14 —	Reserved.

Table continues on the next page...

Table continued from the previous page...

Field	Function
13-0 —	Reserved.

75.17.246 DMA Channel Tx Control (DMA_CH0_Tx_Control)

Offset

Register	Offset
DMA_CH0_Tx_Control	1104h

Function
Controls the transmit features such as PBL, TCP segmentation, and transmit channel weights.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved			EDSE	Reserved				Reserved	ETIC	TxPBL					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved	Reserved		Reserved	Reserved							OSF	TCW			ST
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	Reserved.
28 EDSE	Enhanced Descriptor Enable Indicates whether an enhanced descriptor is enabled. When this field is 1, it indicates that the corresponding channel uses 32 bytes enhanced descriptors for both normal and context descriptors. When this field becomes 0, it indicates that the corresponding channel uses the 16 bytes descriptors. 0b - Disabled 1b - Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
27-24 —	Reserved.
23 —	Reserved.
22 ETIC	Early Transmit Interrupt Control Indicates whether an early transmit interrupt is enabled. When this field is 1, it indicates that an early transmit interrupt (ETI) status = 1 after the data transfer from buffers of a transmit descriptor completes in which IOC bit (TDES2[31]) = 1. When this field becomes 1, it indicates that ETI = 1 only after a complete packet transfers to the MTL TX FIFO memory. 0b - Disabled 1b - Enabled
21-16 TxPBL	Transmit Programmable Burst Length Indicates the maximum number of beats to transfer in one DMA block data transfer. DMA always attempts max burst as specified in PBL each time it starts a burst transfer on the application bus. You can program PBL with any of these values: 1, 2, 4, 8, 16, or 32. Any other value results in undefined behavior. Perform these steps to transfer more than 32 beats: 1. Set DMA_CH0_Control[PBLx8]. 2. Set the TxPBL. **NOTE** The maximum value of TxPBL must be less than or equal to half the transit queue size (TQS field of MTL_TxQ(#i)_Operation_Mode register) in terms of beats. This is required so that the transit queue has space to store at least another TxPBL worth of data while the MTL Tx queue controller transfers data to MAC. For example, in 64-bit data width configurations the total locations in transit queue of size 512 bytes is 64. You must program TxPBL and 8xPBL to less than or equal to 32.
15 —	Reserved.
14-13 —	Reserved.
12 —	Reserved.
11-5 —	Reserved.

Table continues on the next page...

Table continued from the previous page...

Field	Function
4 OSF	Operate on Second Packet Indicates whether the operation on second packet is enabled. When this field is 1, it instructs DMA to process the second packet of the transmit data even before you receive the status for the first packet. 0b - Disabled 1b - Enabled
3-1 TCW	Transmit Channel Weight Indicates the weight assigned to the corresponding transmit channel. When reset completes, this field is 0 for all the channels by default, which results in equal weights to all channels.
0 ST	Start or Stop Transmission Command Indicates whether to start or stop the transmission command. When this field is 1, it indicates that the transmission is in the running state. DMA checks the transmit list at the current position for a packet to be transmitted. DMA tries to acquire descriptor from either of these positions: • The current position in the list. This is the base address of the transmit list to which DMA_CH0_TxDesc_List_Address sets. • The position at which the transmission was previously stopped. The transmission enters the suspended state and DMA_CH0_Status[TBU] = 1, if DMA does not own the current descriptor. The start transmission command is effective only when the transmission stops. If the command is issued before setting DMA_CH0_TxDesc_List_Address, you cannot predict the DMA behavior. When this field becomes 0, it indicates that the transmission process is placed in the stopped state when the transmission of the current packet completes . The next descriptor position in the transmit list is saved, and it becomes the current position when you restart the transmission. To change the list address, you must program DMA_CH0_TxDesc_List_Address with a new value when this field becomes 0.You can consider the new value when this field is 1 again. The stop transmission command is effective only when the transmission of the current packet completes or the transmission is in the suspended state. 0b - Stop 1b - Start

75.17.247 DMA Channel Rx Control (DMA_CH0_Rx_Control)

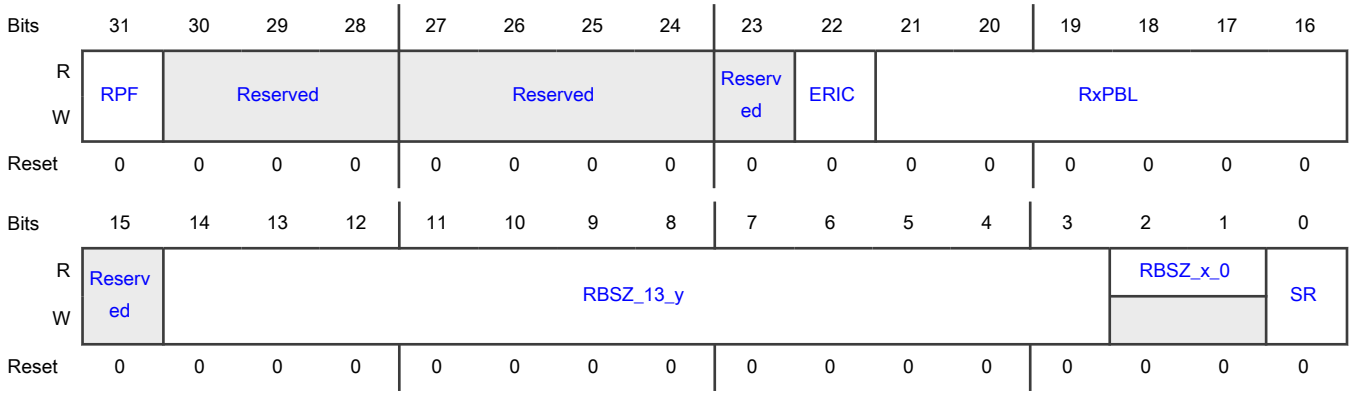
Offset

Register	Offset
DMA_CH0_Rx_Control	1108h

Function

Controls the receive features such as PBL, buffer size, and extended status.

Diagram



Fields

Field	Function
31 RPF	Rx Packet Flush Indicates whether the receive packet flush is enabled. When this field is 1, it indicates that the module automatically flushes the packet from the receive queues destined to this DMA receive channel, when it stops. When this field remains 1 and the software driver re-starts DMA, the packets residing in the receive queues that were received when this RxDMA stops and flushes out. The packets that MAC receives after the RxDMA re-starts, route to the RxDMA. The flushing takes place on the read side of the receive queue. When this field is 0, it indicates that the module do not flush the packet in the receive queue destined to this RxDMA channel when it is in STOP state. This may cause head-of-line blocking in the corresponding receive queue. **NOTE** The packet flow from a receive DMA channel to the application by writing 1 to this field stops, only when there is one-to-one mapping of receive queue to receive DMA channels. In Dynamic mapping mode, writing 1 to this field in any DMA_CH(#i)_Rx_Control register might flush packets from an unintended receive queues which are destined to the stopped receive DMA channel. 0b - Disabled 1b - Enabled
30-28 —	Reserved.
27-24 —	Reserved.
23 —	Reserved.
22	Early Receive Interrupt Control

Table continues on the next page...

Table continued from the previous page...

Field	Function
ERIC	Indicates whether an early receive interrupt is enabled. When this field is 1, it indicates that the early receive interrupt (ERI) status sets after every burst transfer of data from the Rx DMA to the buffer completes. When this field becomes 0, it indicates that ERI sets only after a complete buffer is filled by the RxDMA. 0b - Disabled 1b - Enabled
21-16 RxPBL	Receive Programmable Burst Length Indicates the maximum number of beats to be transferred in one DMA block data transfer. DMA always attempts max burst as specified in PBL every time it starts a burst transfer on the application bus. You can program PBL with any of these values: 1, 2, 4, 8, 16, or 32. Any other value results in undefined behavior. To transfer more than 32 beats, perform the following steps: 1. Set DMA_CH0_Control[PBLx8]. 2. Set the RxPBL. **NOTE** The maximum value of RxPBL must be less than or equal to half the receive queue size (RQS field of MTL_RxQ(#i)_Operation_Mode register) in terms of beats. This is required so that the receive queue has space to store at least another Rx PBL worth of data while the receive DMA transfers a block of data. For example, in 64-bit data width configurations the total locations in receive queue of size 512 bytes is 64. You must program RxPBL and 8xPBL to less than or equal to 32.
15 —	Reserved.
14-3 RBSZ_13_y	Receive Buffer size High RBSZ[13:0] splits into two fields, higher field is RBSZ_13_y and lower field is RBSZ_x_0. RBSZ[13:0] field indicates the receive buffers size specified in bytes. The maximum buffer size is limited to 16K bytes. The buffer size is applicable to payload buffers when split headers are enabled. **NOTE** The buffer size must be a multiple of 4, 8, or 16 depending on the data bus widths (32-bit, 64-bit, or 128-bit respectively). This is required even if the buffer address pointer value is not aligned with data bus width. Hence the lower RBSZ_x_0 fields are read-only and the value is considered as all-zero. Thus RBSZ_13_y field indicates the buffer size in terms of locations (with the width same as bus-width).
2-1 RBSZ_x_0	Receive Buffer size Low RBSZ[13:0] splits into two fields RBSZ_13_y and RBSZ_x_0. The RBSZ_x_0 is the lower field whose width is based on data bus width of the configuration. The field width is of 2, 3, or 4 bits for 32-bit, 64-bit, or 128-bit data bus width respectively. This field is read-only (RO).

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 SR	Start or Stop Receive Indicates whether to start or stop receive. When this field is 1, it indicates that the DMA tries to acquire the descriptor from the receive list and processes the incoming packets. DMA tries to acquire descriptor from either of these positions: • The current position in the list. This is the address which DMA_CH0_RxDesc_List_Address sets. • The position at which the receive process was previously stopped. If DMA does not own the current descriptor, the reception is suspended and DMA_CH0_Status[RBU] = 1. The start receive command is effective only when the reception stops. If the command is issued before setting DMA_CH0_RxDesc_List_Address, the DMA behavior is unpredictable. When this field becomes 0, it indicates that the receive DMA operation stops after the transfer of the current packet. The next descriptor position in the receive list is saved, and it becomes the current position after the receive process restarts. The stop receive command is effective only when the receive process is in the running (waiting for receive packet) or suspended state. 0b - Stop 1b - Start

75.17.248 DMA Channel 0 Tx Descriptor List Address (DMA_CH0_TxDesc_List_Address)

Offset

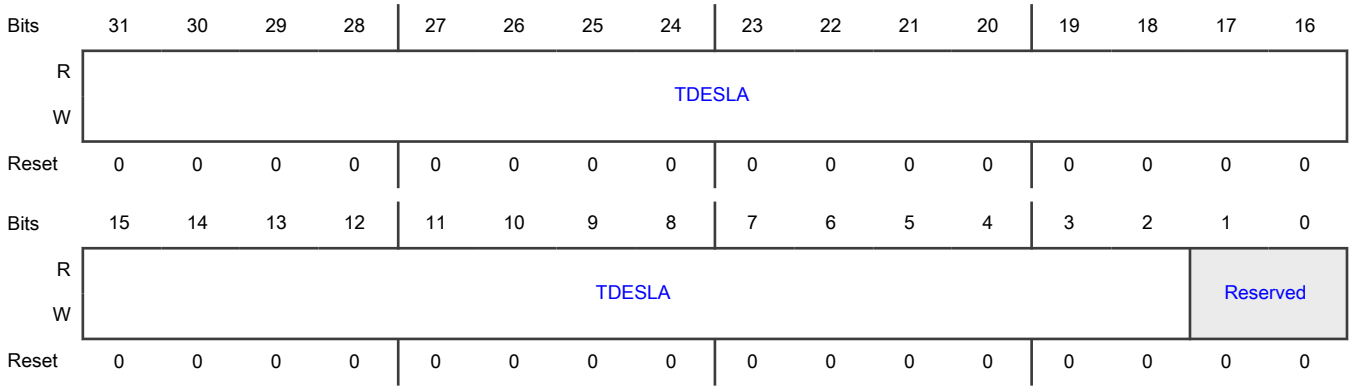
Register	Offset
DMA_CH0_TxDesc_List_Address	1114h

Function

Specifies DMA to the start of transmit descriptor list. The descriptor lists reside in the physical memory space of the application and must be Word, Dword, or Lword-aligned (for 32-bit, 64-bit, or 128-bit data bus). DMA internally converts it to bus width aligned address by making the corresponding LSB low.

You can write to this register only when the transit DMA stops, that is, DMA_CH0_Tx_Control[ST] = 0. When stopped, write this register with a new descriptor list address. When DMA_CH0_Tx_Control[ST] = 1, DMA uses the newly-programmed descriptor base address. If this register does not change when DMA_CH0_Tx_Control[ST] = 0, DMA uses the descriptor address where it was stopped earlier.

Diagram



Fields

Field	Function
31-2 TDESLA	Start of Transmit List Contains the base address of the first descriptor in the transmit descriptor list. DMA ignores the LSB bits (1:0, 2:0, or 3:0) for 32-bit, 64-bit, or 128-bit bus width and internally takes these bits as all-zero. Therefore, these LSB bits are read-only (RO). The field width depends on the configuration: 31:2 for 32-bit configuration 31:3 for 64-bit configuration 31:4 for 128-bit configuration
1-0 —	Reserved.

75.17.249 DMA Channel 0 Rx Descriptor List Address (DMA_CH0_RxDesc_List_Address)

Offset

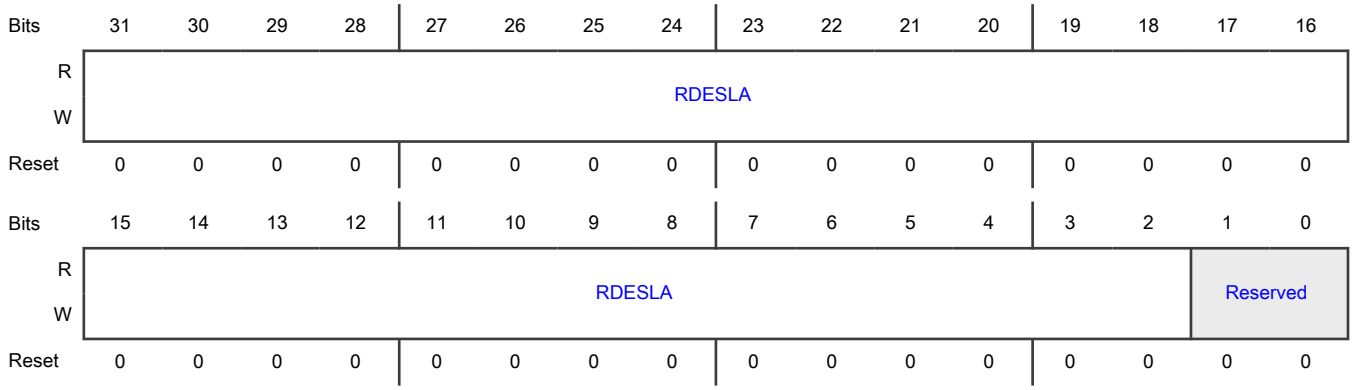
Register	Offset
DMA_CH0_RxDesc_List_Address	111Ch

Function

Specifies DMA to the start of receive descriptor list. The descriptor lists reside in the physical memory space of the application and must be Word, Dword, or Lword-aligned (for 32-bit, 64-bit, or 128-bit data bus). DMA internally converts it to bus width aligned address by making the corresponding LS bits low. You can write to this register only when reception stops. When stopped, write this register before the receive start command is given. You can write to this register only when Rx DMA stops, that is, DMA_CH0_Rx_Control[SR] = 0. When stopped, write this register with a new descriptor list address.

When DMA_CH0_Rx_Control[SR] = 1, it indicates that DMA takes the newly programmed descriptor base address.

Diagram



Fields

Field	Function
31-2 RDESLA	Start of Receive List Contains the base address of the first descriptor in the receive descriptor list. DMA ignores the LSB bits (1:0, 2:0, or 3:0) for 32-bit, 64-bit, or 128-bit bus width and internally takes these bits as all-zero. Therefore, these LSB bits are read-only (RO). The field width depends on the configuration: 31:2 for 32-bit configuration 31:3 for 64-bit configuration 31:4 for 128-bit configuration
1-0 —	Reserved.

75.17.250 DMA Channel 0 Tx Descriptor Tail Pointer (DMA_CH0_TxDesc_Tail_Pointer)

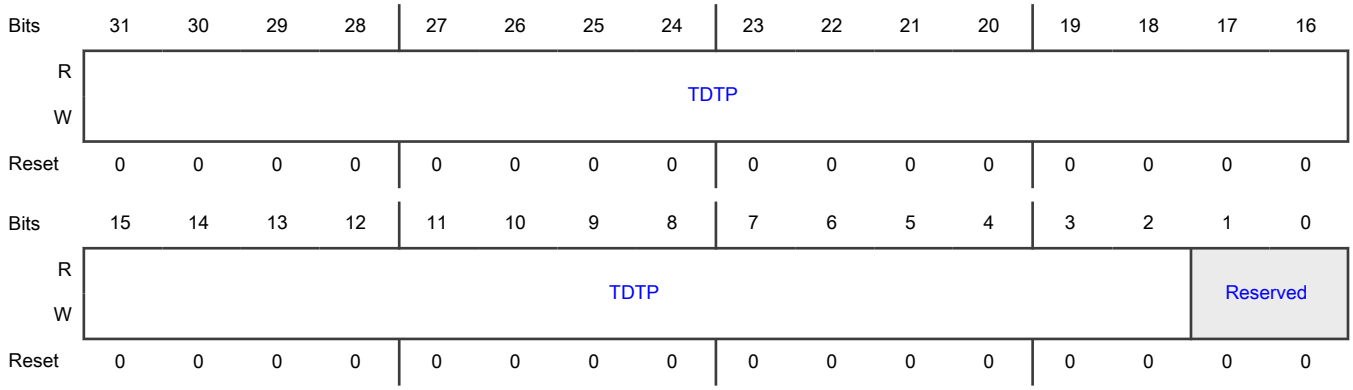
Offset

Register	Offset
DMA_CH0_TxDesc_Tail_Pointer	1120h

Function

Points to an offset from the base and indicates the location of the last valid descriptor.

Diagram



Fields

Field	Function
31-2 TDTP	Transmit Descriptor Tail Pointer Contains the tail pointer for the transit descriptor ring. You must write the tail pointer to add more descriptors to the transit channel. The hardware tries to transmit all packets to the descriptors referenced between the head and the tail pointer registers. The field width depends on the configuration: 31:2 for 32-bit configuration 31:3 for 64-bit configuration 31:4 for 128-bit configuration
1-0 —	Reserved.

75.17.251 DMA Channel 0 Rx Descriptor List Pointer (DMA_CH0_RxDesc_Tail_Pointer)

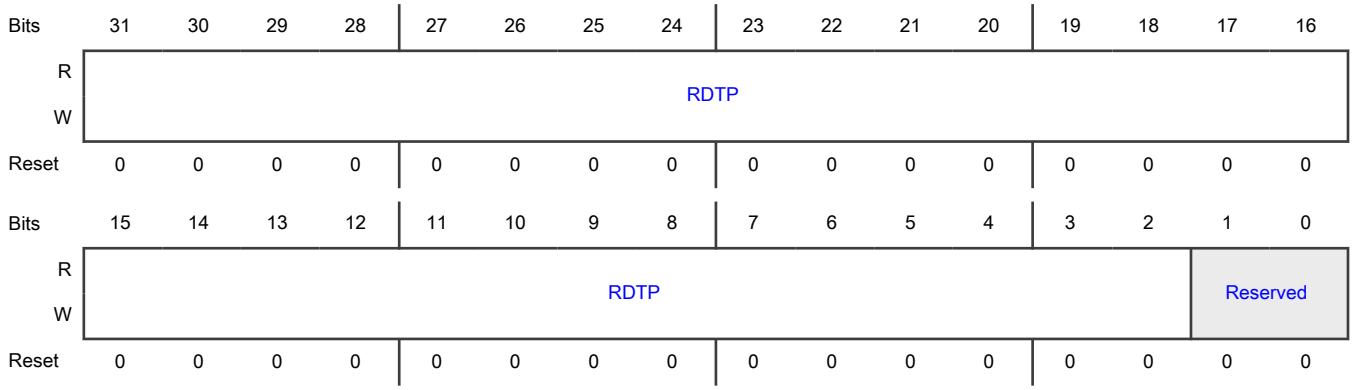
Offset

Register	Offset
DMA_CH0_RxDesc_Tail_Pointer	1128h

Function

Specifies an offset from the base and indicates the location of the last valid descriptor.

Diagram



Fields

Field	Function
31-2 RDTP	Receive Descriptor Tail Pointer Contains the tail pointer for the receive descriptor ring. You must write the tail pointer to add more descriptors to the receive channel. The hardware tries to write all received packets to the descriptors referenced between the head and the tail pointer registers. The field width depends on the configuration: 31:2 for 32-bit configuration 31:3 for 64-bit configuration 31:4 for 128-bit configuration
1-0 —	Reserved.

75.17.252 DMA Channel 0 Tx Descriptor Ring Length (DMA_CH0_TxDesc_Ring_Length)

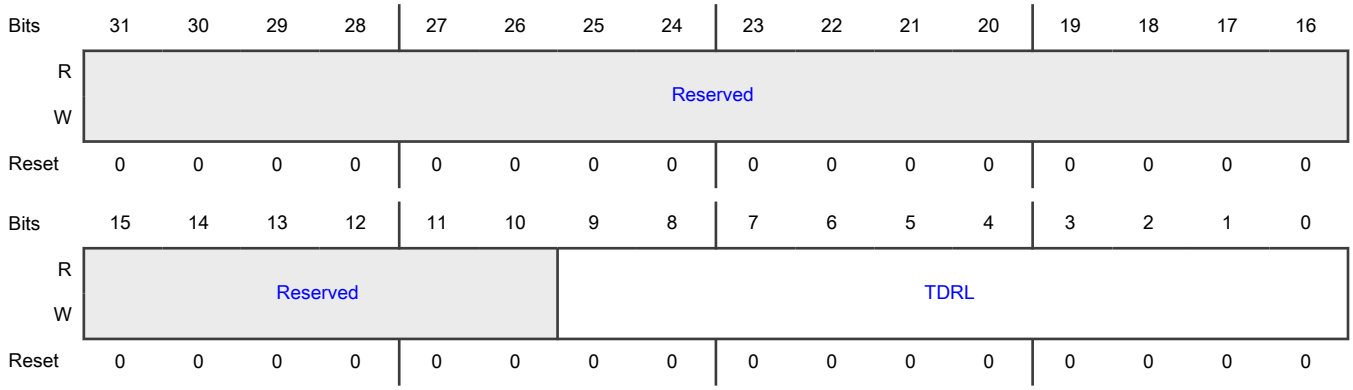
Offset

Register	Offset
DMA_CH0_TxDesc_Ring_Length	112Ch

Function

Contains the length of the transmit descriptor ring.

Diagram



Fields

Field	Function
31-10 —	Reserved.
9-0 TDRL	Transmit Descriptor Ring Length Sets the maximum number of transmit descriptors in the circular descriptor ring. The maximum number of descriptors are limited to 1K descriptors. NXP recommends a minimum ring descriptor length of 4. For example, you can program any value up to 0x3FF in this field. This field is 10 bits wide, if you program 0x3FF, you can have 1024 descriptors. Program this field to a value of 0x9, if you want to have 10 descriptors.

75.17.253 DMA Channel 0 Rx Descriptor Ring Length (DMA_CH0_RxDesc_Ring_Length)

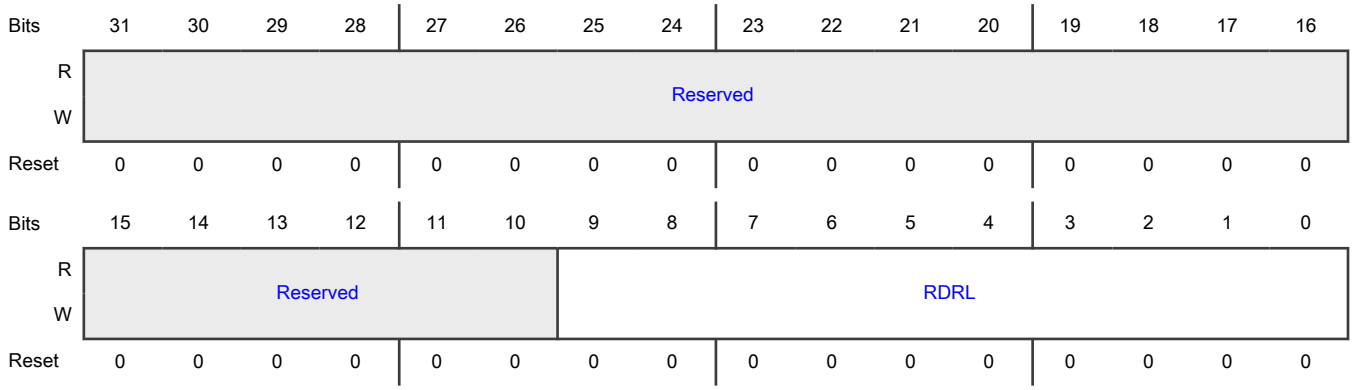
Offset

Register	Offset
DMA_CH0_RxDesc_Ring_Length	1130h

Function

Contains the length of the receive descriptor circular ring.

Diagram



Fields

Field	Function
31-10 —	Reserved.
9-0 RDRL	Receive Descriptor Ring Length Sets the maximum number of receive descriptors in the circular descriptor ring. The maximum number of descriptors is limited to 1K descriptors. For example, you can program any value up to 0x3FF in this field. This field is 10 bits wide, if you program 0x3FF, you can have 1024 descriptors. Program this field to a value of 0x9, if you want to have 10 descriptors.

75.17.254 DMA Channel 0 Interrupt Enable (DMA_CH0_Interrupt_Enable)

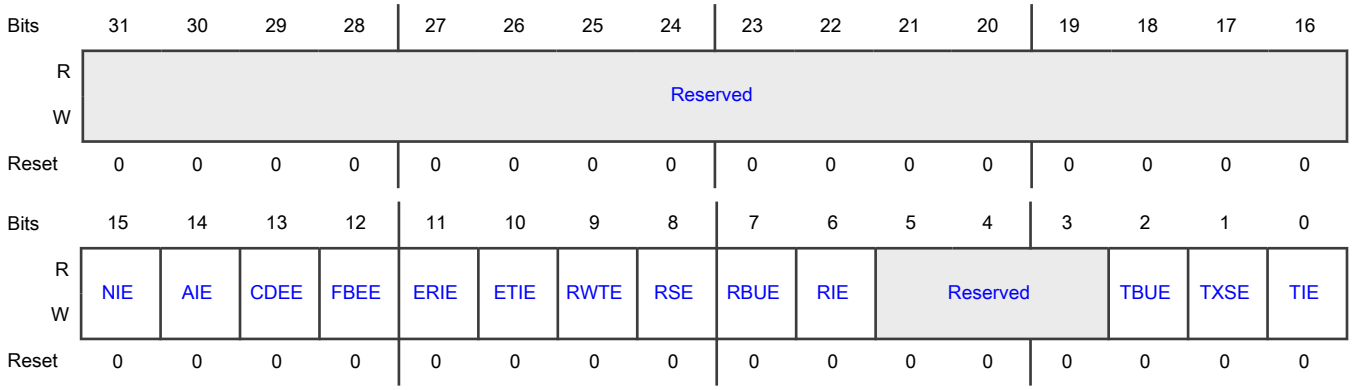
Offset

Register	Offset
DMA_CH0_Interrupt_Enable	1134h

Function

Enables the interrupts which the status register report.

Diagram



Fields

Field	Function
31-16 —	Reserved.
15 NIE	Normal Interrupt Summary Enable Enables or disables the normal interrupt summary. When this field is 1, it enables the normal interrupt summary. This field also enables the following interrupts in DMA_CH0_Status: -Bit 0 - Transmit interrupt Bit 2 - Transmit buffer unavailable Bit 6 - Receive interrupt Bit 11 - Early receive interrupt When this field becomes 0, it disables the normal interrupt summary. 0b - Disable 1b - Enable
14 AIE	Abnormal Interrupt Summary Enable Enables or disables the abnormal interrupt summary. When this field is 1, it enables the abnormal interrupt summary. This field also enables the following interrupts in DMA_CH0_Status: Bit 1 - Transmit process stopped Bit 7 - Rx buffer unavailable Bit 8 - Receive process stopped Bit 9 - Receive watchdog timeout Bit 10 - Early transmit interrupt Bit 12 - Fatal bus error Bit 13 - Context descriptor error

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When this field becomes 0, it disables the abnormal interrupt summary. 0b - Disable 1b - Enable
13 CDEE	Context Descriptor Error Enable Enables or disables the context descriptor error. When this field is 1 along with DMA_CH0_Interrupt_Enable[AIE], it enables the descriptor error interrupt. When this field becomes 0, it disables the descriptor error interrupt. 0b - Disable 1b - Enable
12 FBEE	Fatal Bus Error Enable Enables or disables the fatal bus error. When this field is 1 along with DMA_CH0_Interrupt_Enable[AIE], it enables the fatal bus error interrupt. When this field becomes 0, it disables the fatal bus error interrupt. 0b - Disable 1b - Enable
11 ERIE	Early Receive Interrupt Enable Enables or disables the early receive interrupt. When this field is 1 along with DMA_CH0_Interrupt_Enable[NIE], it enables the early receive interrupt. When this field becomes 0, it disables the early receive interrupt. 0b - Disable 1b - Enable
10 ETIE	Early Transmit Interrupt Enable Enables or disables the early transmit interrupt. When this field is 1 along with DMA_CH0_Interrupt_Enable[AIE], it enables the early transmit interrupt. When this field becomes 0, it disables the early transmit interrupt. 0b - Disable 1b - Enable
9 RWTE	Receive Watchdog Timeout Enable Enables or disables the receive watchdog timeout. When this field is 1 along with DMA_CH0_Interrupt_Enable[AIE], it enables the receive watchdog timeout interrupt. When this field becomes 0, it disables the receive watchdog timeout interrupt. 0b - Disable 1b - Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
8 RSE	Receive Stopped Enable Enables or disables the receive stopped interrupt. When this field is 1 along with DMA_CH0_Interrupt_Enable[AIE], it enables the receive stopped interrupt. When this field becomes 0, it disables the receive stopped interrupt. 0b - Disable 1b - Enable
7 RBUE	Receive Buffer Unavailable Enable Enables or disables the receive buffer unavailable interrupt. When this field is 1 along with DMA_CH0_Interrupt_Enable[AIE], it enables the receive buffer unavailable interrupt. When this field becomes 0, it disables the receive buffer unavailable interrupt. 0b - Disable 1b - Enable
6 RIE	Receive Interrupt Enable Enables or disables the receive interrupt. When this field is 1 along with DMA_CH0_Interrupt_Enable[NIE], it enables the receive interrupt. When this field becomes 0, it disables the receive interrupt. 0b - Disable 1b - Enable
5-3 —	Reserved.
2 TBUE	Transmit Buffer Unavailable Enable Enables or disables the transmit buffer unavailable interrupt. When this field is 1 along with DMA_CH0_Interrupt_Enable[NIE], it enables the transmit buffer unavailable interrupt. When this field becomes 0, it disables the transmit buffer unavailable interrupt. 0b - Disable 1b - Enable
1 TXSE	Transmit Stopped Enable Enables or disables the transmit stopped interrupt. When this field is 1 along with DMA_CH0_Interrupt_Enable[AIE], it enables the transmission stopped interrupt. When this field becomes 0, it disables the transmission stopped interrupt. 0b - Disable 1b - Enable
0	Transmit Interrupt Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
TIE	Enables or disables the transmit interrupt. When this field is 1 along with DMA_CH0_Interrupt_Enable[NIE], it enables the transmit interrupt. When this field becomes 0, it disables the transmit interrupt. 0b - Disable 1b - Enable

75.17.255 DMA Channel 0 Rx Interrupt Watchdog Timer (DMA_CH0_Rx_Interrupt_Watchdog_Timer)

Offset

Register	Offset
DMA_CH0_Rx_Interrupt_Watchdog_Timer	1138h

Function

Indicates the watchdog timeout for receive interrupt (RI) from DMA. When you write this register with a non-zero value, it enables the watchdog timer for the RI bit of the DMA_CHi_Status register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved														RWTU	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								RWT							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-18 —	Reserved.
17-16 RWTU	Receive Interrupt Watchdog Timer Count Units Indicates the system clock cycles number corresponding to one unit in DMA_CH0_Rx_Interrupt_Watchdog_Timer[RWT].

Table continues on the next page...

Table continued from the previous page...

Field	Function
	2'b00 - 256 2'b01 - 512 2'b10 - 1024 2'b11 - 2048 For example, when DMA_CH0_Rx_Interrupt_Watchdog_Timer[RWT] = 2 and DMA_CH0_Rx_Interrupt_Watchdog_Timer[RWTU] = 1, the watchdog timer sets for 2*512=1024 system clock cycles.
15-8 —	Reserved.
7-0 RWT	Receive Interrupt Watchdog Timer Count Indicates the number of system clock cycles, multiplied by factor which DMA_CH0_Rx_Interrupt_Watchdog_Timer[RWTU] indicates, for which you set the watchdog timer. The watchdog timer triggers with the programmed value after the receive DMA completes the packet transfer for which the RI bit is not set in the DMA_CH(#i)_Status register, because of the setting of interrupt enable bit in the corresponding descriptor RDES3[30]. When the watchdog timer runs out, it indicates that the RI bit is 1 and the timer stops. The watchdog timer resets when the RI bit is 1 because of automatic setting of RI per the interrupt enable bit RDES3[30] of any received packet.

75.17.256 DMA Channel 0 Slot Function Control Status (DMA_CH0_Slot_Function_Control_Status)

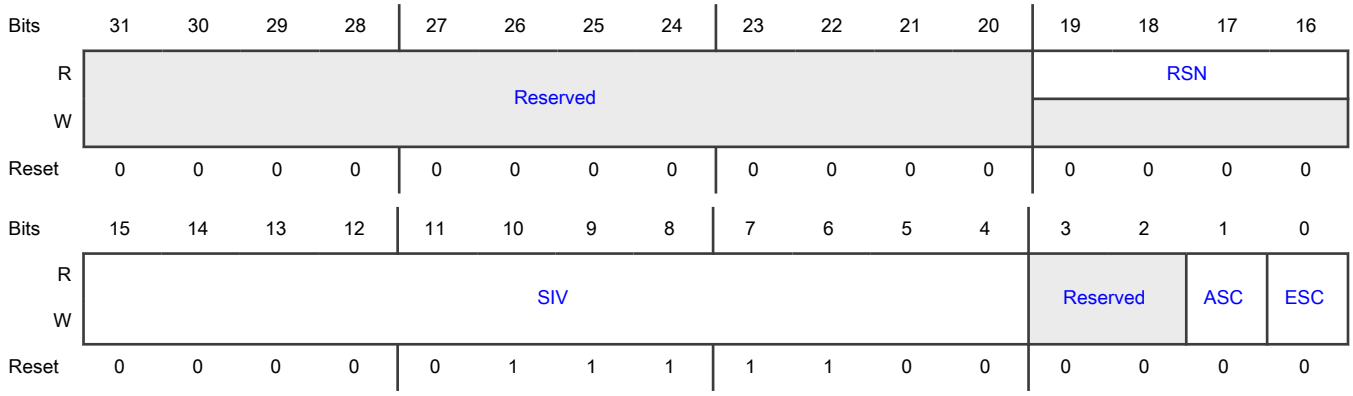
Offset

Register	Offset
DMA_CH0_Slot_Function_Control_Status	113Ch

Function

Contains the control bits for slot function and the status for transmit path.

Diagram



Fields

Field	Function
31-20 —	Reserved.
19-16 RSN	Reference Slot Number Provides the current value of the reference slot number in DMA. It is used for slot comparison.
15-4 SIV	Slot Interval Value Controls the period of the slot interval in which the TxDMA fetches the scheduled packets. A value of 0 specifies the slot interval of 1 us while the maximum value 4095 specifies the slot interval of 4096 us. The default or reset value is 0x07C which corresponds to slot interval of 125 us
3-2 —	Reserved.
1 ASC	Advance Slot Check Indicates whether the advance slot check is enabled. When this field is 1, it enables DMA to fetch the data from the buffer when the slot number (SLOTNUM) programmed in the transit descriptor is either: Equal to the reference slot number given in DMA_CH0_Slot_Function_Control_Status[RSN], or Ahead of the reference slot number by up to two slots. This field is applicable only when DMA_CH0_Slot_Function_Control_Status[ESC] = 1. 0b - Disabled 1b - Enabled
0 ESC	Enable Slot Comparison Indicates whether the slot comparison is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When this field is 1, it enables the checking of the slot numbers programmed in the transit descriptor with the current reference given in DMA_CH0_Slot_Function_Control_Status[RSN]. DMA fetches the data from the corresponding buffer only when the slot number is either: • Equal to the reference slot number, or • Ahead of the reference slot number by one slot. When this field becomes 0, it disables the checking of the slot numbers. DMA fetches the data immediately after you process the descriptor. **NOTE** You must not enable the UFO (UDP Fragmentation over IPv4)/TSO/USO along with TBS/AVB slot number check. The UFO/TSO/USO involves multiple packets/segments/ fragments transmission for single packet received from application and the slot number check are applicable for fetching only first segment/fragment. As a result it might be difficult for you to specify slot number for subsequent packets. 0b - Disabled 1b - Enabled

75.17.257 DMA Channel 0 Current Application Transmit Descriptor (DMA_CH0_Current_App_TxDesc)

Offset

Register	Offset
DMA_CH0_Current_App_TxDesc	1144h

Function
Specifies the current transmit descriptor which DMA reads.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CURTDESAPTR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CURTDESAPTR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 CURTDESAPTR	Application Transmit Descriptor Address Pointer Indicates that DMA updates this pointer during the transit operation. This pointer clears when reset.

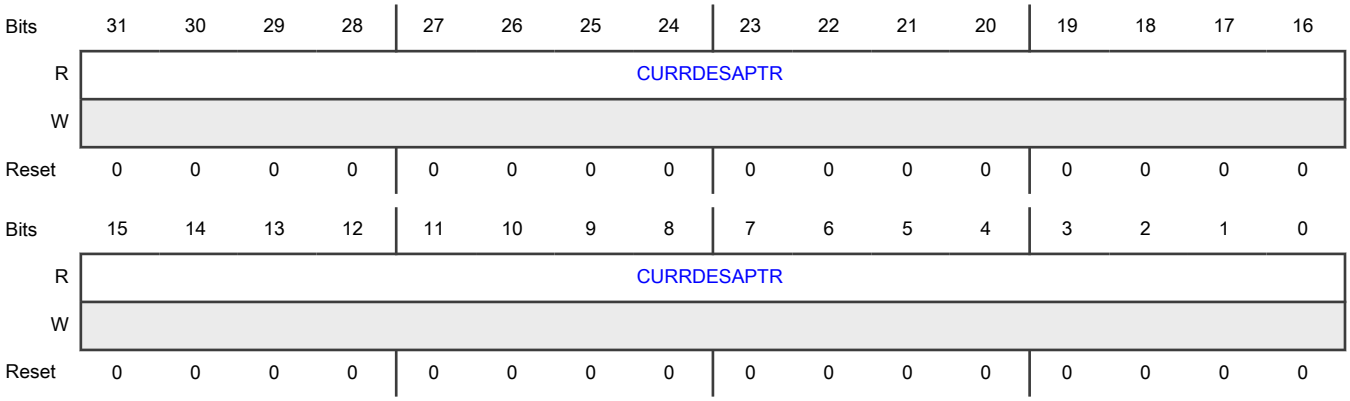
75.17.258 DMA Channel 0 Current Application Receive Descriptor (DMA_CH0_Current_App_RxDesc)

Offset

Register	Offset
DMA_CH0_Current_App_RxDesc	114Ch

Function
Specifies the current receive descriptor which DMA read.

Diagram



Fields

Field	Function
31-0 CURRDESAPTR	Application Receive Descriptor Address Pointer Indicates that the DMA updates this pointer during the receive operation. This pointer clears when reset.

75.17.259 DMA Channel 0 Current Application Transmit Descriptor (DMA_CH0_Current_App_TxBuffer)

Offset

Register	Offset
DMA_CH0_Current_App_TxBuffer	1154h

Function
Specifies the current transit buffer address which DMA reads.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CURTBUFAPTR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CURTBUFAPTR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 CURTBUFAPTR	Application Transmit Buffer Address Pointer Indicates that DMA updates this pointer during the transit operation. This pointer clears when reset.

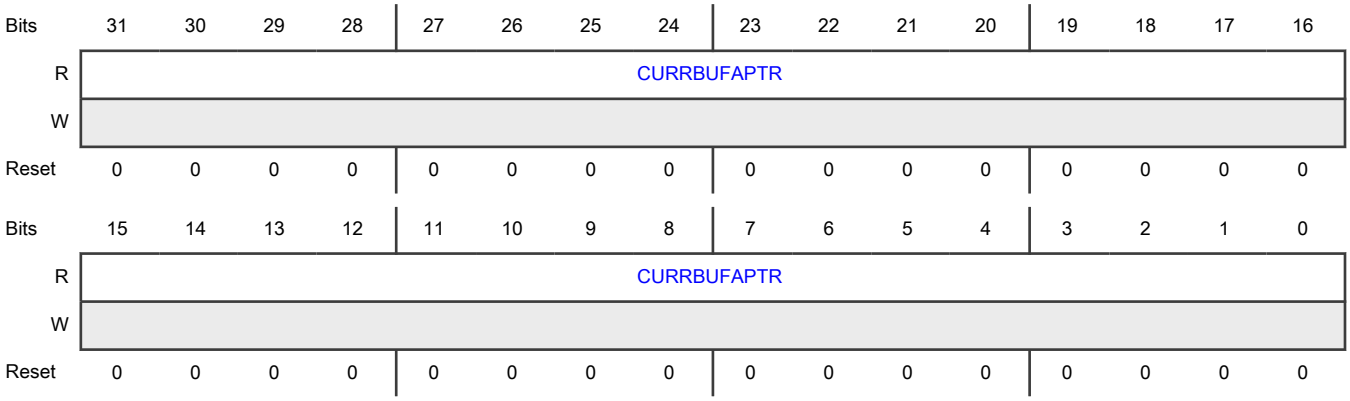
75.17.260 DMA Channel 0 Current Application Receive Buffer (DMA_CH0_Current_App_RxBuffer)

Offset

Register	Offset
DMA_CH0_Current_App_RxBuffer	115Ch

Function
Specifies the current receive buffer address which DMA read.

Diagram



Fields

Field	Function
31-0 CURRBUFAPTR	Application Receive Buffer Address Pointer Indicates that DMA updates this pointer during the receive operation. This pointer clears when reset.

75.17.261 DMA Channel 0 Status (DMA_CH0_Status)

Offset

Register	Offset
DMA_CH0_Status	1160h

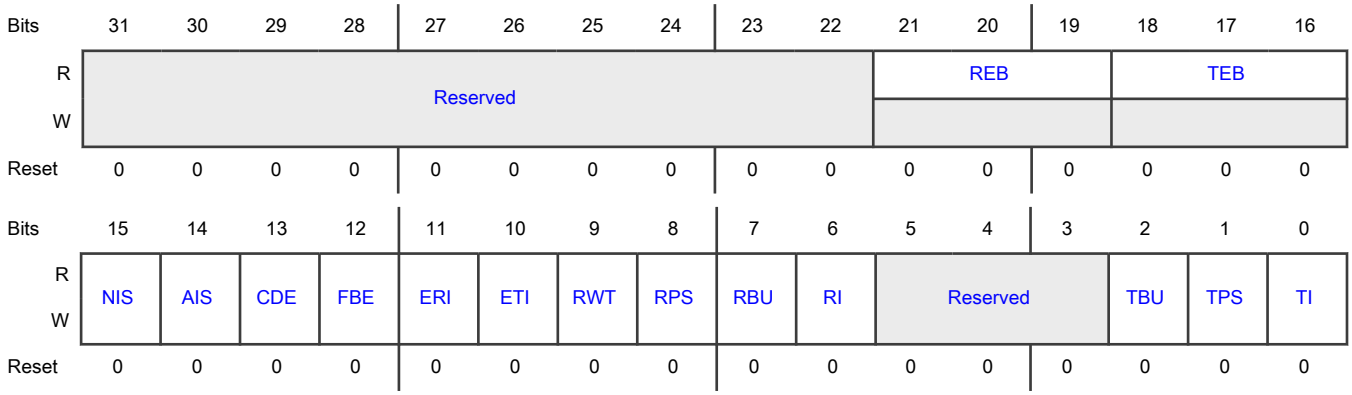
Function

Indicates that the software driver (application) reads the status register during an interrupt service routine or polling to determine the DMA status.

NOTE

The number of DMA_CH(#i)_Status register in the configuration is the higher of number of receive DMA channels and transit DMA channels.

Diagram



Fields

Field	Function
31-22 —	Reserved.
21-19 REB	Rx DMA Error Bits Indicates the type of error that causes a bus error. For example, error response on the AHB or AXI interface. Bit 21 1'b1 - Error during data transfer by Rx DMA 1'b0 - No Error during data transfer by Rx DMA Bit 20 1'b1 - Error during descriptor access 1'b0 - Error during data buffer access Bit 19 1'b1 - Error during read transfer 1'b0 - Error during write transfer This field is valid only when DMA_CH0_Status[FBE] = 1. This field does not generate an interrupt.
18-16 TEB	Tx DMA Error Bits Indicates the type of error that causes a bus error. For example, error response on the AHB or AXI interface. Bit 18 1'b1 - Error during data transfer by Tx DMA 1'b0 - No Error during data transfer by Tx DMA Bit 17 1'b1 - Error during descriptor access 1'b0 - Error during data buffer access

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Bit 16 1'b1 - Error during read transfer 1'b0 - Error during write transfer This field is valid only when DMA_CH0_Status[FBE] = 1. This field does not generate an interrupt.
15 NIS	Normal Interrupt Summary Indicates whether the normal interrupt summary status is detected. This field is the logical OR of the following bits when you enables the corresponding interrupt bits in DMA_CH0_Interrupt_Enable: Bit 0 - Transmit interrupt Bit 2 - Transmit buffer unavailable Bit 6 - Receive interrupt Bit 11 - Early receive interrupt Only unmasked bits (interrupts for which interrupt enable sets in DMA_CH0_Interrupt_Enable) affect this field. This is a sticky bit. You must clear this field by writing 1 to it each time a corresponding field which sets this field clears. Access restriction apply to this field. It automatically becomes 1 on an internal event occurrence. Writing 1 clears this field and writing 0 has no effect. 0b - Not detected 1b - Detected
14 AIS	Abnormal Interrupt Summary Indicates whether an abnormal interrupt summary status is detected. This field is the logical OR of the following bits when you enables the corresponding interrupt bits in DMA_CH0_Interrupt_Enable: Bit 1 - Transmit process stopped Bit 7 - Receive buffer unavailable Bit 8 - Receive process stopped Bit 10 - Early transmit interrupt Bit 12 - Fatal bus error Bit 13 - Context descriptor error Only unmasked bits affect this field. This is a sticky bit. You must clear this field by writing 1 to it each time a corresponding field, which sets this field, clears. Access restriction apply to this field. It automatically becomes 1 on an internal event occurrence. Writing 1 clears this field and writing 0 has no effect.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Not detected 1b - Detected
13 CDE	Context Descriptor Error Indicates whether the context descriptor error status is detected. This field indicates that the DMA Tx/Rx engine receives a descriptor error, which indicates an invalid context in the middle of packet flow (intermediate descriptor) or all one's descriptor in transit case and on receive side it indicates DMA has read a descriptor with either of the buffer address as ones which is considered to be invalid. Access restriction apply to this field. It automatically becomes 1 on an internal event occurrence. Writing 1 clears this field and writing 0 has no effect. 0b - Not detected 1b - Detected
12 FBE	Fatal Bus Error Indicates whether the fatal bus error status is detected. This field indicates that a bus error occurred (as described in the EB field). When this field is 1, it indicates that the corresponding DMA channel engine disables all bus accesses. Access restriction apply to this field. It automatically becomes 1 on an internal event occurrence. Writing 1 clears this field and writing 0 has no effect. 0b - Not detected 1b - Detected
11 ERI	Early Receive Interrupt Indicates whether an early receive interrupt status is detected. When this field is 1, it indicates that the RxDMA completes the packet data transfer to the memory. In configs supporting ERIC, this field is 1 only after the receive DMA fills a complete receive buffer with packet data, when DMA_CH0_Rx_Control[ERIC] = 0. This field is 1 after every burst transfer of data from the receive DMA to the buffer, when DMA_CH0_Rx_Control[ERIC] = 1. Writing 1 to RI field automatically clears this field. Access restriction apply to this field. It automatically becomes 1 on an internal event occurrence. Writing 1 clears this field and writing 0 has no effect. 0b - Not detected 1b - Detected
10 ETI	Early Transmit Interrupt Indicates whether an early transmit interrupt status is detected. When this field is 1, it indicates that the TxDMA completes the packet data transfer to the MTL TXFIFO memory.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	In configs supporting ERIC: this field is 1 only after the transit DMA transfers a complete packet to MTL, when DMA_CH0_Tx_Control[ETIC] = 0. This field is 1 after packet data transfers from buffer completes (partial) in the transmit descriptor in which IOC = 1, when DMA_CH0_Tx_Control[ETIC] = 1. Access restriction apply to this field. It automatically becomes 1 on an internal event occurrence. Writing 1 clears this field and writing 0 has no effect. 0b - Not detected 1b - Detected
9 RWT	Receive Watchdog Timeout Indicates whether the receive watchdog timeout status is detected. Asserts when you receives a packet with length greater than 2,048 bytes (10,240 bytes when Jumbo Packet mode is enabled). 0b - Not detected 1b - Detected
8 RPS	Receive Process Stopped Indicates whether the receive process stopped status is detected. Asserts when the receive process enters the stopped state. Access restriction apply to this field. It automatically becomes 1 on an internal event occurrence. Writing 1 clears this field and writing 0 has no effect. 0b - Not detected 1b - Detected
7 RBU	Receive Buffer Unavailable Indicates whether the receive buffer unavailable status is detected. This field indicates that the application owns the next descriptor in the receive list, and DMA cannot acquire it. The receive process is suspended. To resume processing receive descriptors, the application must change the ownership of the descriptor and issue a receive poll demand command. If this command is not issued, the receive process resumes when you receive the next recognized incoming packet. In ring mode, the application must advance the receive descriptor tail pointer register of a channel. This field is 1 only when DMA owns the previous receive descriptor. Access restriction apply to this field. It automatically becomes 1 on an internal event occurrence. Writing 1 clears this field and writing 0 has no effect. 0b - Not detected 1b - Detected
6 RI	Receive Interrupt Indicates that the receive interrupt status is detected.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field indicates that the packet reception is complete. When packet reception completes, bit 31 of RDES3 resets in the last descriptor, and you can update the specific packet status information in the descriptor. The reception remains in the running state. Access restriction apply to this field. It automatically becomes 1 on an internal event occurrence. Writing 1 clears this field and writing 0 has no effect. 0b - Not detected 1b - Detected
5-3 —	Reserved.
2 TBU	Transmit Buffer Unavailable Indicates whether the transmit buffer unavailable status is detected. This field indicates that the application owns the next descriptor in the transmit list, and the DMA cannot acquire it. Transmission is suspended. DMA_Debug_Status0[TPS0] explains the transmit process state transitions. To resume processing the transmit descriptors, the application must perform these steps: 1. Change the ownership of the descriptor by writing 1 to bit 31 of TDES3. 2. Issue a transmit poll demand command. For Ring mode, the application must advance the transmit descriptor tail pointer register of a channel. Access restriction apply to this field. It automatically becomes 1 on an internal event occurrence. Writing 1 clears this field and writing 0 has no effect. 0b - Not detected 1b - Detected
1 TPS	Transmit Process Stopped Indicates whether the transmit process stopped status is detected. This field is 1 when the transmission stops. Access restriction apply to this field. It automatically becomes 1 on an internal event occurrence. Writing 1 clears this field and writing 0 has no effect. 0b - Not detected 1b - Detected
0 TI	Transmit Interrupt Indicates whether the transmit interrupt status is detected. This field indicates that the packet transmission is complete. When transmission completes, Bit 31 of TDES3 resets in the last descriptor, and you can update the specific packet status information in the descriptor.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Access restriction apply to this field. It automatically becomes 1 on an internal event occurrence. Writing 1 clears this field and writing 0 has no effect. 0b - Not detected 1b - Detected

75.17.262 DMA Channel 0 Miss Frame Counter (DMA_CH0_Miss_Frame_Cnt)

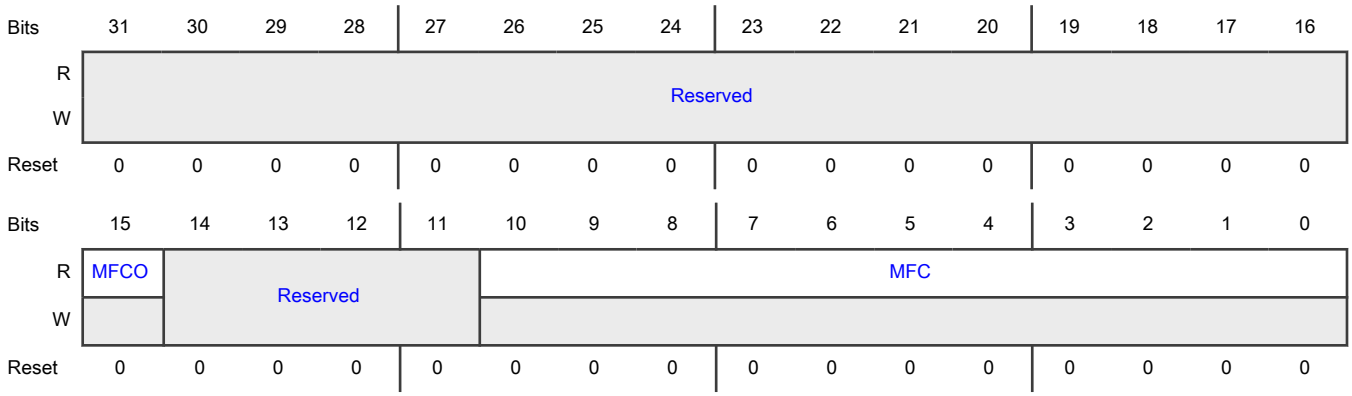
Offset

Register	Offset
DMA_CH0_Miss_Frame_Cnt	1164h

Function

Indicates the number of packet counter that DMA drops either due to bus error or due to programming RPF field in DMA_CH${i}_Rx_Control register.

Diagram



Fields

Field	Function
31-16 —	Reserved.
15 MFCO	Overflow status of the MFC Counter Indicates whether the miss frame counter overflow has occurred. When this field is 1, it indicates that the MFC counter does not increment further. This field is 0 when this register is read.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Access restriction apply to this field. It clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not occurred 1b - Occurred
14-11 —	Reserved.
10-0 MFC	Dropped Packet Counters Indicates the number of packet counters that DMA drops either because of bus error or because of programing RPF field in DMA_CH${i}_Rx_Control register. This field is 0 when this register is read. Access restriction apply to this field. It clears on read and automatically becomes 1 on an internal event occurrence.

75.17.263 DMA Channel 0 Rx Parser Accept Count (DMA_CH0_RXP_Accept_Cnt)

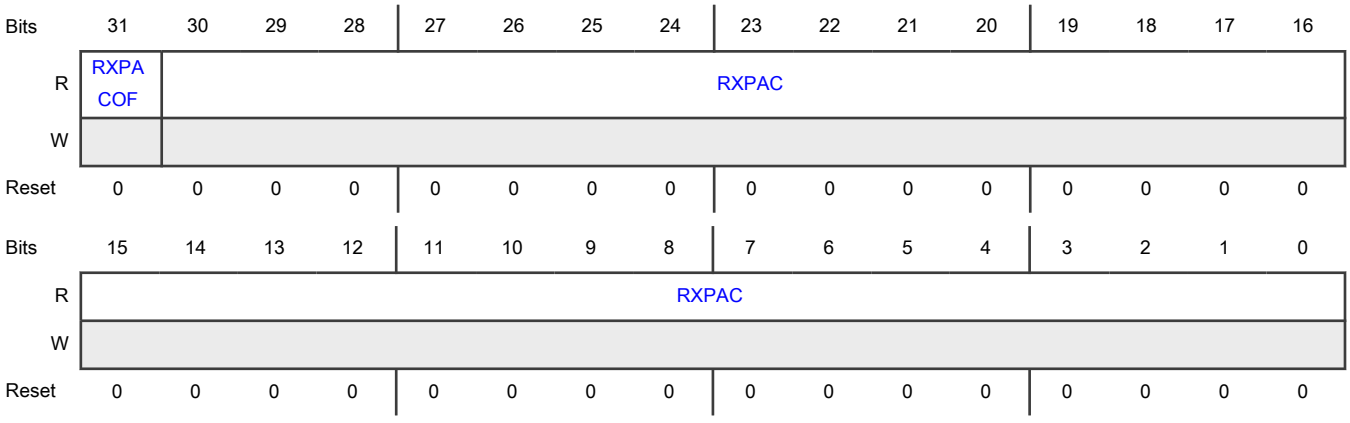
Offset

Register	Offset
DMA_CH0_RXP_Accept_Cnt	1168h

Function

Provides the count of the number of frames which the receive parser accept.

Diagram



Fields

Field	Function
31 RXPACOF	Rx Parser Accept Counter Overflow Bit Indicates whether the receive parser accept counter overflow has occurred. When this field is 1, it indicates that the RXPAC counter field has crossed the maximum limit. Access restriction apply to this field. It clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not occurred 1b - Occurred
30-0 RXPAC	Rx Parser Accept Counter Implements whenever a receive parser accept a packet because AF = 1. The counter clears when the register is read.

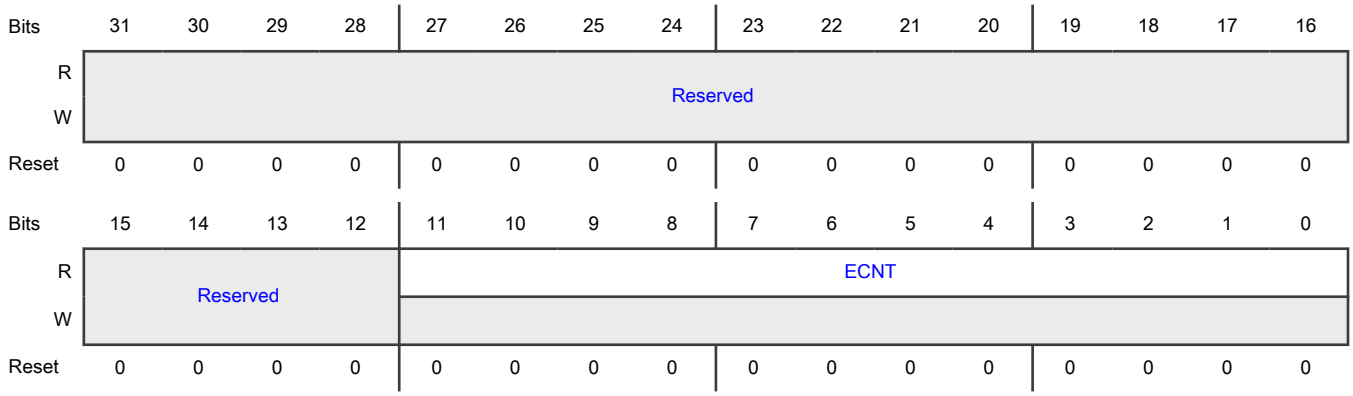
75.17.264 DMA Channel 0 Rx ERI Count (DMA_CH0_RX_ERI_Cnt)

Offset

Register	Offset
DMA_CH0_RX_ERI_Cnt	116Ch

Function
Provides the count of the number of times ERI is asserted.

Diagram



Fields

Field	Function
31-12	Reserved.

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
11-0 ECNT	ERI Counter Indicates that when ERIC bit of DMA_CH(#i)_RX_Control register is 1, this counter increments for burst transfer which the receive DMA completes from the start of packet transfer. This field becomes 0 at the start of new packet.

75.17.265 DMA Channel 1 Control (DMA_CH1_Control)

Offset

Register	Offset
DMA_CH1_Control	1180h

Function

Specifies the MSS value for segmentation, length to skip between two descriptors, and also the features such as header splitting and 8xPBL mode.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved							Reserved	Reserved				DSL			Reserved	PBLx8
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved				Reserved											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-25 —	Reserved.
24 —	Reserved.
23-21	Reserved.

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
20-18 DSL	Descriptor Skip Length Specifies the Word, Dword, or Lword number (depending on the 32-bit, 64-bit, or 128-bit bus) to skip between two unchained descriptors. The address skipping starts from the end of the current descriptor to the start of the next descriptor. When this field is 0, it indicates that DMA takes the descriptor table as contiguous.
17 —	Reserved.
16 PBLx8	8xPBL mode Indicates whether the 8xPBL mode is enabled. When this field is 1, it indicates that you must program the PBL value in Bits[21:16] in DMA_CH(#i)_Tx_Control and Bits[21:16] in DMA_CH(#i)_Rx_Control and multiply it eight times. Therefore, DMA transfers the data in 8, 16, 32, 64, 128, and 256 beats depending on the PBL value. 0b - Disabled 1b - Enabled
15-14 —	Reserved.
13-0 —	Reserved.

75.17.266 DMA Channel 1 Tx Control (DMA_CH1_Tx_Control)

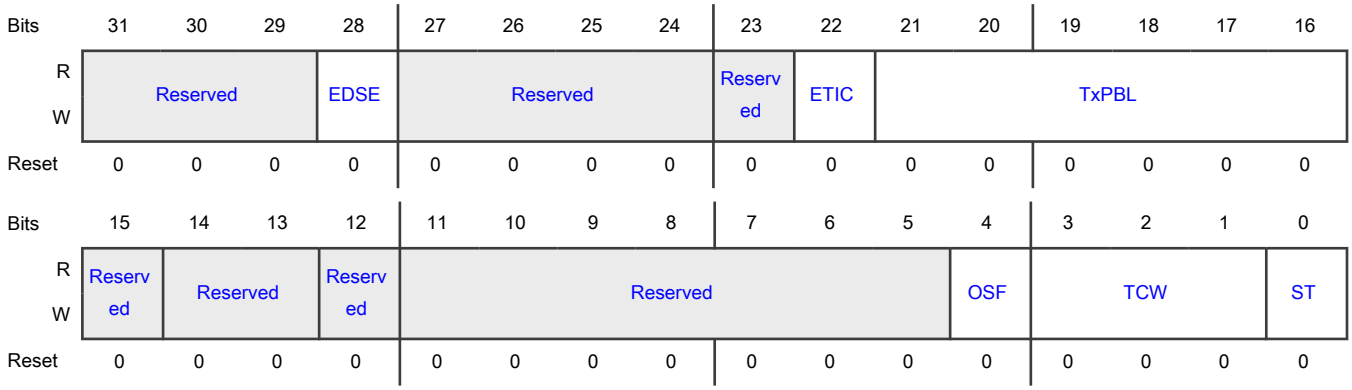
Offset

Register	Offset
DMA_CH1_Tx_Control	1184h

Function

Controls the transmit features such as PBL, TCP segmentation, and transmit channel weights.

Diagram



Fields

Field	Function
31-29 —	Reserved.
28 EDSE	Enhanced Descriptor Enable Indicates whether an enhanced descriptor is enabled. When this field is 1, it indicates that the corresponding channel uses enhanced descriptors that are 32 Bytes for both normal and context descriptors. When this field becomes 0, it indicates that the corresponding channel uses the descriptors that are 16 Bytes. 0b - Disabled 1b - Enabled
27-24 —	Reserved.
23 —	Reserved.
22 ETIC	Early Transmit Interrupt Control Indicates whether an early transmit interrupt is enabled. When this field is 1, it indicates that an early transmit interrupt (ETI) status sets after the data transfer completes from buffers of a transmit descriptor in which IOC bit (TDES2[31]) sets. When this field becomes 0, it indicates that the ETI sets only after a complete packet transfers to the MTL TX FIFO memory. 0b - Disabled 1b - Enabled
21-16	Transmit Programmable Burst Length

Table continues on the next page...

Table continued from the previous page...

Field	Function
TxPBL	Indicates the maximum number of beats to transfer in one DMA block data. DMA always attempts max burst as specified in PBL each time it starts a burst transfer on the application bus. You can program PBL with any of these values: 1, 2, 4, 8, 16, or 32. Any other value results in an undefined behavior. To transfer more than 32 beats, perform the following steps: 1. Write 1 to DMA_CH0_Control[PBLx8]. 2. Write 1 to this field. **NOTE** The maximum value of this field must be less than or equal to half the transit queue size (TQS field of MTL_TxQ(#i)_Operation_Mode register) in terms of beats. This is required so that the transit queue has space to store at least another Tx PBL worth of data while the MTL Tx Queue controller transfers data to MAC. For example, in 64-bit data width configurations the total locations in transit queue of size 512 bytes is 64. You must program TxPBL and 8xPBL to less than or equal to 32.
15 —	Reserved.
14-13 —	Reserved.
12 —	Reserved.
11-5 —	Reserved.
4 OSF	Operate on Second Packet Indicates whether an operation on second packet is enabled. When this field is 1, it instructs DMA to process the second packet of the transmit data even before the first packet status is obtained. 0b - Disabled 1b - Enabled
3-1 TCW	Transmit Channel Weight Indicates the weight assigned to the corresponding transmit channel. When reset completes, this field becomes 0 for all channels by default, resulting in equal weights to all channels.
0 ST	Start or Stop Transmission Command Indicates whether to start or stop transmission command. When this field is 1, it indicates that the transmission is placed in the running state. DMA checks the transmit list at the current position to transmit a packet.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DMA tries to acquire descriptor from either of the following positions: • The current position in the list. This is the base address of the transmit list which DMA_CH0_TxDesc_List_Address sets. • The position at which the transmission was previously stopped If DMA does not own the current descriptor, the transmission enters the suspended state and DMA_CH0_Status[TBU] = 1. The start transmission command is effective only when the transmission stops. If you issue the command before writing 1 to DMA_CH0_TxDesc_List_Address, then DMA behavior is unpredictable. When this field becomes 0, it indicates that the transmission process is placed in the stopped state after the transmission of the current packet completes. The next descriptor position in the transmit list is saved, and it becomes the current position when the transmission restarts. To change the list address, you must program DMA_CH0_TxDesc_List_Address with a new value when this field becomes 0. The new value is considered when this field is 1 again. The stop transmission command is effective only when the transmission of the current packet completes or the transmission is in the suspended state. 0b - Stop 1b - Start

75.17.267 DMA Channel 1 Rx Control (DMA_CH1_Rx_Control)

Offset

Register	Offset
DMA_CH1_Rx_Control	1188h

Function

Controls the receive features such as PBL, buffer size, and extended status.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RPF	Reserved			Reserved				Reserved	ERIC	RxPBL					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	RBSZ_13_y												RBSZ_x_0		SR
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 RPF	Rx Packet Flush Indicates whether the receive packet flush is enabled. When this field is 1, it indicates that the module automatically flushes the packet from the receive queues destined to this DMA receive channel, when it stops. When this field remains 1 and the software driver re-starts DMA, the packets residing in the receive queues that were received when this RxDMA was stopped, flushes out. Route the packets to the RxDMA that the MAC receives after the RxDMA re-starts. The flushing takes place on the read side of the receive queue. When this field is 0, it indicates that the module do not flush the packet in the receive queue destined to this RxDMA channel when it is in stop state. This might in turn cause head-of-line blocking in the corresponding RxQueue. **NOTE** The stopping of packet flow from a Rx DMA Channel to the application by writing 1 to this field works only when there is one-to-one mapping of receive queue to the receive DMA channels. In Dynamic mapping mode, writing 1 to this field in any DMA_CH(#i)_Rx_Control register might flush packets from unintended receive queues which are destined to the stopped Rx DMA Channel. 0b - Disabled 1b - Enabled
30-28 —	Reserved.
27-24 —	Reserved.
23 —	Reserved.
22 ERIC	Early Receive Interrupt Control Indicates whether an early receive interrupt control status is enabled. When this field is 1, it indicates that an early receive interrupt (ERI) status sets after every burst transfer of data from the receive DMA to the buffer completes. When this field becomes 0, it indicates that ERI sets only after the RxDMA fills the complete buffer is filled. 0b - Disabled 1b - Enabled
21-16 RxPBL	Receive Programmable Burst Length Indicates the maximum number of beats to transfer in one DMA block data. DMA always attempts max burst as specified in PBL, each time it starts a burst transfer on the application bus. You can program PBL with any of the following values: 1, 2, 4, 8, 16, or 32. Any other value results in an undefined behavior. To transfer more than 32 beats, perform the following steps:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1. Write 1 to DMA_CH0_Control[PBLx8]. 2. Write 1 to RxPBL. **NOTE** The maximum value of RxPBL must be less than or equal to half the receive queue size (RQS field of MTL_RxQ(#i)_Operation_Mode register) in terms of beats. This is required so that the receive queue has space to store at least another Rx PBL worth of data when the receive DMA transfers a block of data. For example, in 64-bit data width configurations the total locations in receive queue of size 512 bytes is 64, so you must program RxPBL and 8xPBL to less than or equal to 32.
15 —	Reserved.
14-3 RBSZ_13_y	Receive Buffer size High Indicates that RBSZ[13:0] splits into two fields higher RBSZ_13_y and lower RBSZ_x_0. RBSZ[13:0] field indicates the size of the receive buffers specified in bytes. The maximum buffer size is limited to 16K bytes. The buffer size is applicable to payload buffers when you enable split headers. **NOTE** The buffer size must be a multiple of 4, 8, or 16 depending on the data bus widths (32-bit, 64-bit, or 128-bit respectively). This is required even if the value of buffer address pointer is not aligned to data bus width. Hence the lower RBSZ_x_0 bits are read-only and the value is considered as all-zero. Thus RBSZ_13_y indicates the buffer size in terms of locations (with the width same as bus-width).
2-1 RBSZ_x_0	Receive Buffer size Low Indicates that RBSZ[13:0] splits into two fields RBSZ_13_y and RBSZ_x_0. RBSZ_x_0 is the lower field whose width is based on data bus width of the configuration. This field width is of 2, 3, or 4 bits for 32-bit, 64-bit, or 128-bit data bus width respectively. This field is read-only (RO).
0 SR	Start or Stop Receive Indicates whether to start or stop receive. When this field is 1, it indicates that DMA tries to acquire the descriptor from the receive list and processes the incoming packets. DMA tries to acquire descriptor from either of the following positions: • The current position in the list. This is the address that DMA_CH0_RxDesc_List_Address sets. • The position at which the receive process was previously stopped If DMA does not own the current descriptor, the reception suspends and DMA_CH0_Status[RBU] = 1. The start receive command is effective only when the reception stops. If the command is issued before writing 1 to DMA_CH0_RxDesc_List_Address, DMA behavior is unpredictable.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When this field becomes 0, it indicates that the receive DMA operation stops after the transfer of the current packet. The next descriptor position in the receive list is saved, and it becomes the current position after the receive process restarts. The stop receive command is effective only when the receive process is in the running (waiting for Rx packet) or suspended state. 0b - Stop 1b - Start

75.17.268 DMA Channel 1 Tx Descriptor List Address (DMA_CH1_TxDesc_List_Address)

Offset

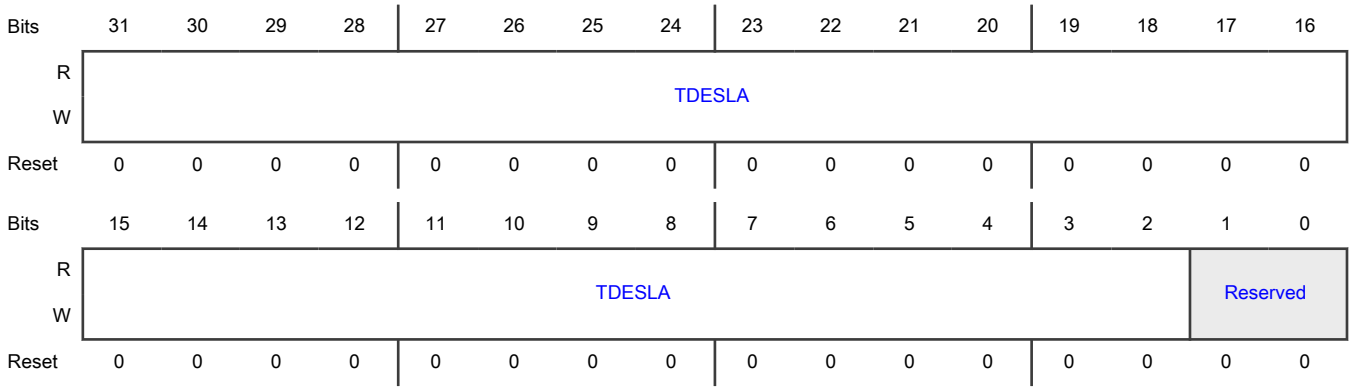
Register	Offset
DMA_CH1_TxDesc_List_Address	1194h

Function

Specifies DMA to the start of transmit descriptor list. The descriptor lists reside in the application's physical memory space and must be Word, Dword, or Lword-aligned (for 32-bit, 64-bit, or 128-bit data bus). The DMA converts the corresponding LSB bits to low to internally convert the descriptor lists to bus width aligned address.

You can write to this register only when the transmit DMA stops, that is, DMA_CH0_Tx_Control[ST] = 0. When stopped, you can write this register with a new descriptor list address. When DMA_CH0_Tx_Control[ST] = 1, the DMA takes the newly-programmed descriptor base address. If this register do not change when DMA_CH0_Tx_Control[ST] = 0, DMA takes the descriptor address where it was stopped earlier.

Diagram



Fields

Field	Function
31-2	Start of Transmit List

Table continues on the next page...

Table continued from the previous page...

Field	Function
TDESLA	Contains the first descriptor base address in the transmit descriptor list. DMA ignores the LSB bits (1:0, 2:0, or 3:0) for 32-bit, 64-bit, or 128-bit bus width and internally assumes these bits as all-zero. Therefore, these LSB bits are read-only (RO). The field width depends on the configuration: 31:2 for 32-bit configuration 31:3 for 64-bit configuration 31:4 for 128-bit configuration
1-0 —	Reserved.

75.17.269 DMA Channel 1 Rx Descriptor List Address (DMA_CH1_RxDesc_List_Address)

Offset

Register	Offset
DMA_CH1_RxDesc_List_Address	119Ch

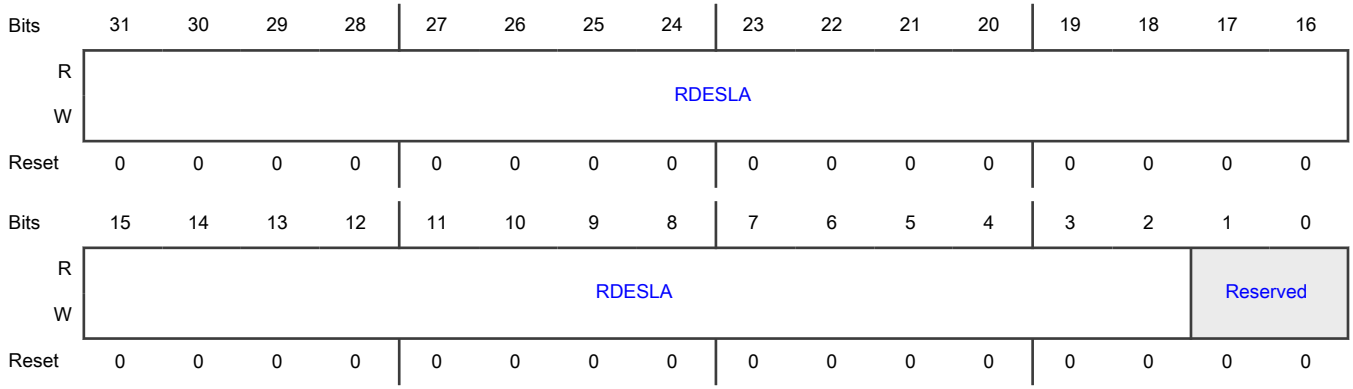
Function

Specifies DMA to the start of receive descriptor list. The descriptor lists resides in the application's physical memory space and must be Word, Dword, or Lword-aligned (for 32-bit, 64-bit, or 128-bit data bus). DMA converts the corresponding LS bits low to internally convert the descriptor lists to bus width aligned address. You can write to this register only when reception stops. When stopped, you must write this register before the receive start command is given.

You can write to this register when receive DMA has stopped, that is, DMA_CH0_Rx_Control[SR] = 0. When stopped, you can this register with a new descriptor list address.

When DMA_CH0_Rx_Control[SR] = 1, DMA takes the newly programmed descriptor base address.

Diagram



Fields

Field	Function
31-2 RDESLA	Start of Receive List Contains the first descriptor base address in the receive descriptor list. DMA ignores the LSB bits (1:0, 2:0, or 3:0) for 32-bit, 64-bit, or 128-bit bus width and internally assume these bits as all-zero. Therefore, these LSB bits are read-only (RO). The field width depends on the configuration: 31:2 for 32-bit configuration 31:3 for 64-bit configuration 31:4 for 128-bit configuration
1-0 —	Reserved.

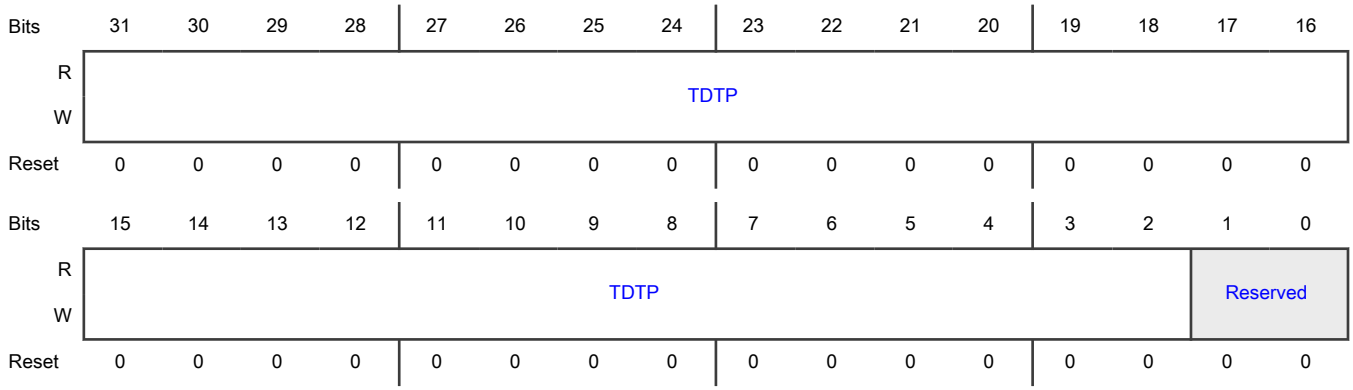
75.17.270 DMA Channel 1 Tx Descriptor Tail Pointer (DMA_CH1_TxDesc_Tail_Pointer)

Offset

Register	Offset
DMA_CH1_TxDesc_Tail_Pointer	11A0h

Function
Specifies to an offset from the base and indicates the location of the last valid descriptor.

Diagram



Fields

Field	Function
31-2	Transmit Descriptor Tail Pointer

Table continues on the next page...

Table continued from the previous page...

Field	Function
TDTP	Contains the tail pointer for the transit descriptor ring. The software writes the tail pointer to add more descriptors to the transit channel. The hardware tries to transmit all the packets which the descriptors referenced between the head and the tail pointer registers. The field width depends on the configuration: 31:2 for 32-bit configuration 31:3 for 64-bit configuration 31:4 for 128-bit configuration
1-0 —	Reserved.

75.17.271 DMA Channel 1 Rx Descriptor Tail Pointer (DMA_CH1_RxDesc_Tail_Pointer)

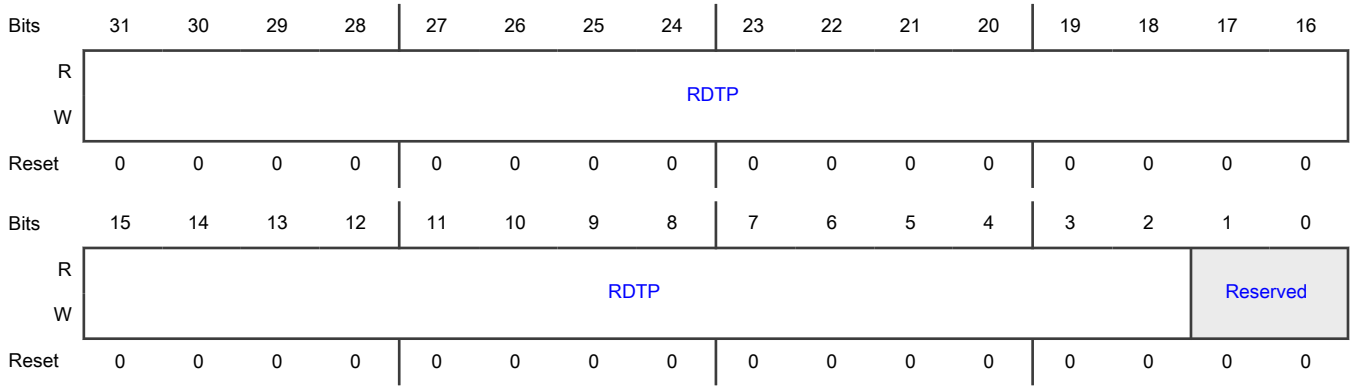
Offset

Register	Offset
DMA_CH1_RxDesc_Tail_Pointer	11A8h

Function

Specifies to an offset from the base and indicates the location of the last valid descriptor.

Diagram



Fields

Field	Function
31-2 RDTP	Receive Descriptor Tail Pointer

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Contains the tail pointer for the receive descriptor ring. The software writes the tail pointer to add more descriptors to the receive channel. The hardware tries to write all received packets to the descriptors referenced between the head and the tail pointer registers. The field width depends on the configuration: 31:2 for 32-bit configuration 31:3 for 64-bit configuration 31:4 for 128-bit configuration
1-0 —	Reserved.

75.17.272 DMA Channel 1 Tx Descriptor Ring Length (DMA_CH1_TxDesc_Ring_Length)

Offset

Register	Offset
DMA_CH1_TxDesc_Ring_Length	11ACh

Function
Contains the length of the transmit descriptor ring.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved						TDRL									
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-10 —	Reserved.

Table continues on the next page...

Table continued from the previous page...

Field	Function
9-0 TDRL	Transmit Descriptor Ring Length Sets the maximum number of transmit descriptors in the circular descriptor ring. The maximum number of descriptors is limited to 1 K descriptors. NXP recommends a minimum ring descriptor length of 4. For example, you can program any value up to 0x3FF in this field. This field is 10 bits wide, if you program 0x3FF, you can have 1024 descriptors. Program it to a value of 0x9 if you want to have 10 descriptors.

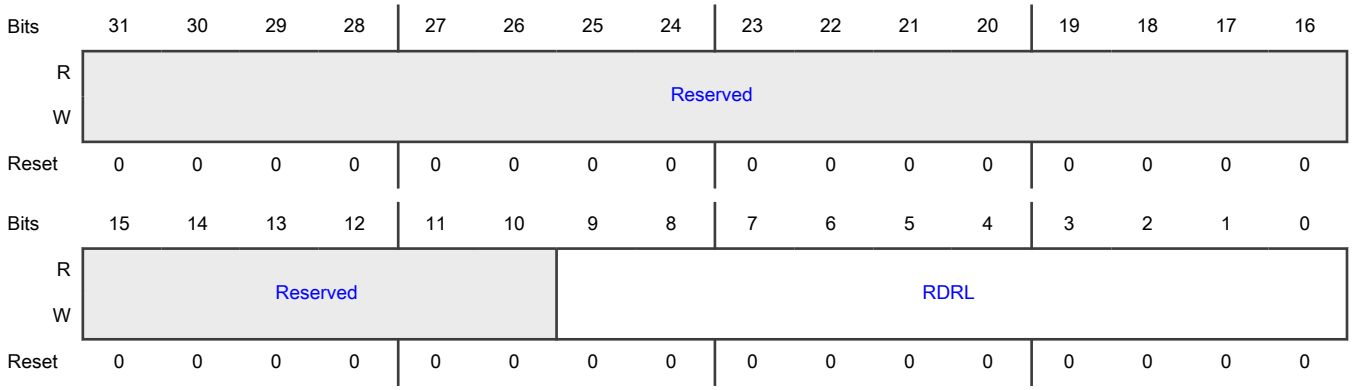
75.17.273 DMA Channel 1 Rx Descriptor Ring Length (DMA_CH1_RxDesc_Ring_Length)

Offset

Register	Offset
DMA_CH1_RxDesc_Ring_Length	11B0h

Function
Contains the length of the receive descriptor circular ring.

Diagram



Fields

Field	Function
31-10 —	Reserved.
9-0 RDRL	Receive Descriptor Ring Length Sets the maximum number of receive descriptors in the circular descriptor ring. The maximum number of descriptors are limited to 1 K descriptors. For example, you can program any value up to 0x3FF in this field. This field is 10 bits wide, if you program 0x3FF, you can have 1024 descriptors. Program it to a value of 0x9 if you want to have 10 descriptors.

75.17.274 DMA Channel 1 Interrupt Enable (DMA_CH1_Interrupt_Enable)

Offset

Register	Offset
DMA_CH1_Interrupt_Enable	11B4h

Function
Enables the interrupts which the status register reports.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	NIE	AIE	CDEE	FBEE	ERIE	ETIE	RWTE	RSE	RBUE	RIE	Reserved			TBUE	TXSE	TIE
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved.
15 NIE	Normal Interrupt Summary Enable Enables or disables the normal interrupt summary. When this field is 1, it enables the normal interrupt summary. This field also enables the following interrupts in DMA_CH0_Status: Bit 0 - Transmit interrupt Bit 2 - Transmit buffer unavailable Bit 6 - Receive interrupt Bit 11 - Early receive interrupt When this field becomes 0, it disables the normal interrupt summary. 0b - Disable 1b - Enable
14	Abnormal Interrupt Summary Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
AIE	Enables or disables the abnormal interrupt summary. When this field is 1, it enables the abnormal interrupt summary. This field also enables the following interrupts in DMA_CH0_Status: Bit 1 - Transmit process stopped Bit 7 - Rx buffer unavailable Bit 8 - Receive process stopped Bit 9 - Receive watchdog timeout Bit 10 - Early transmit interrupt Bit 12 - Fatal bus error Bit 13 - Context descriptor error When this field becomes 0, it disables the abnormal interrupt summary. 0b - Disable 1b - Enable
13 CDEE	Context Descriptor Error Enable Enables or disables the context descriptor error. When this field is 1 along with DMA_CH0_Interrupt_Enable[AIE], it enables the descriptor error interrupt. When this field becomes 0, it disables the descriptor error interrupt. 0b - Disable 1b - Enable
12 FBEE	Fatal Bus Error Enable Enables or disables fatal bus error. When this field is 1 along with DMA_CH0_Interrupt_Enable[AIE], it enables the fatal bus error interrupt. When this field becomes 0, it disables the fatal bus error interrupt. 0b - Disable 1b - Enable
11 ERIE	Early Receive Interrupt Enable Enables or disables the early receive interrupt. When this field is 1 along with DMA_CH0_Interrupt_Enable[NIE], it enables the early receive interrupt. When this field becomes 0, it disables the early receive interrupt. 0b - Disable 1b - Enable
10 ETIE	Early Transmit Interrupt Enable Enables or disables the early transmit interrupt.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When this field is 1 along with DMA_CH0_Interrupt_Enable[AIE], it enables the early transmit interrupt. When this field becomes 0, it disables the early transmit interrupt. 0b - Disable 1b - Enable
9 RWTE	Receive Watchdog Timeout Enable Enables or disables the receive watchdog timeout interrupt. When this field is 1 along with DMA_CH0_Interrupt_Enable[AIE], it enables the receive watchdog timeout interrupt. When this field becomes 0, it disables the receive watchdog timeout interrupt. 0b - Disable 1b - Enable
8 RSE	Receive Stopped Enable Enables or disables the receive stopped interrupt. When this field is 1 along with DMA_CH0_Interrupt_Enable[AIE], It enables the receive stopped interrupt. When this field becomes 0, it disables the receive stopped interrupt. 0b - Disable 1b - Enable
7 RBUE	Receive Buffer Unavailable Enable Enables or disables the receive buffer unavailable interrupt. When this field is 1 along with DMA_CH0_Interrupt_Enable[AIE], it enables the receive buffer unavailable interrupt. When this field becomes 0, it disables the receive buffer unavailable interrupt. 0b - Disable 1b - Enable
6 RIE	Receive Interrupt Enable Enables or disables the receive interrupt. When this field is 1 along with DMA_CH0_Interrupt_Enable[NIE], it enables the receive interrupt. When this field becomes 0, it disables the receive interrupt. 0b - Disable 1b - Enable
5-3 —	Reserved.
2 TBUE	Transmit Buffer Unavailable Enable Enables or disables the transmit buffer unavailable interrupt.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When this field is 1 along with DMA_CH0_Interrupt_Enable[NIE], it enables the transmit buffer unavailable interrupt. When this field becomes 0, it disables the transmit buffer unavailable interrupt. 0b - Disable 1b - Enable
1 TXSE	Transmit Stopped Enable Enables or disables the transmit stopped interrupt. When this field is 1 along with DMA_CH0_Interrupt_Enable[AIE], it enables the transmission stopped interrupt. When this field becomes 0, it disables the transmission Stopped interrupt. 0b - Disable 1b - Enable
0 TIE	Transmit Interrupt Enable Enables or disables the transmit interrupt. When this field is 1 along with DMA_CH0_Interrupt_Enable[NIE], it enables the transmit interrupt. When this field becomes 0, it disables the transmit interrupt. 0b - Disable 1b - Enable

75.17.275 DMA Channel 1 Rx Interrupt Watchdog Timer (DMA_CH1_Rx_Interrupt_Watchdog_Timer)

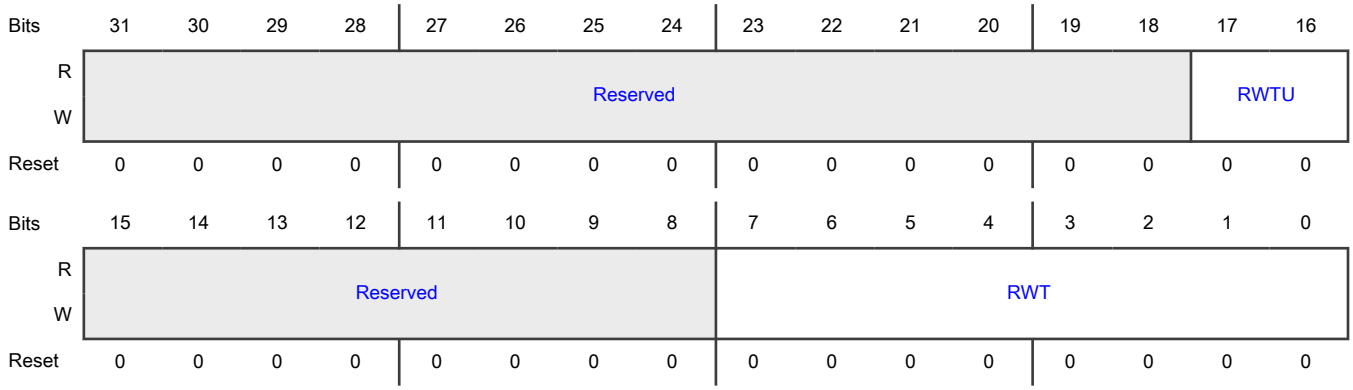
Offset

Register	Offset
DMA_CH1_Rx_Interrupt_Watchdog_Timer	11B8h

Function

Indicates the watchdog timeout for receive interrupt (RI) from DMA. When you write this register with a non-zero value, it enables the watchdog timer for the RI bit of the DMA_CHi_Status register.

Diagram



Fields

Field	Function
31-18 —	Reserved.
17-16 RWTU	Receive Interrupt Watchdog Timer Count Units Indicates the number of system clock cycles corresponding to one unit in DMA_CH1_Rx_Interrupt_Watchdog_Timer[RWT]. 2'b00 - 256 2'b01 - 512 2'b10 - 1024 2'b11 - 2048 For example, the watchdog timer sets for 2*512=1024 system clock cycles, when DMA_CH1_Rx_Interrupt_Watchdog_Timer[RWT] = 2 and DMA_CH1_Rx_Interrupt_Watchdog_Timer[RWTU] = 1.
15-8 —	Reserved.
7-0 RWT	Receive Interrupt Watchdog Timer Count Indicates the number of system clock cycles, multiplied by factor indicated in DMA_CH1_Rx_Interrupt_Watchdog_Timer[RWTU], for which the watchdog timer sets. The watchdog timer triggers with the programmed value after the receive DMA completes the packet transfer for which the RI bit is not 1 in DMA_CH(#i)_Status register, because of the setting of interrupt enable bit in the corresponding descriptor RDES3[30]. RI bit is 1 and the timer stops when the watchdog timer runs out. The watchdog timer becomes 0 when RI bit is 1 because RI sets automatically per the interrupt enable bit RDES3[30] of any received packet.

75.17.276 DMA Channel 1 Slot Function Control Status (DMA_CH1_Slot_Function_Control_Status)

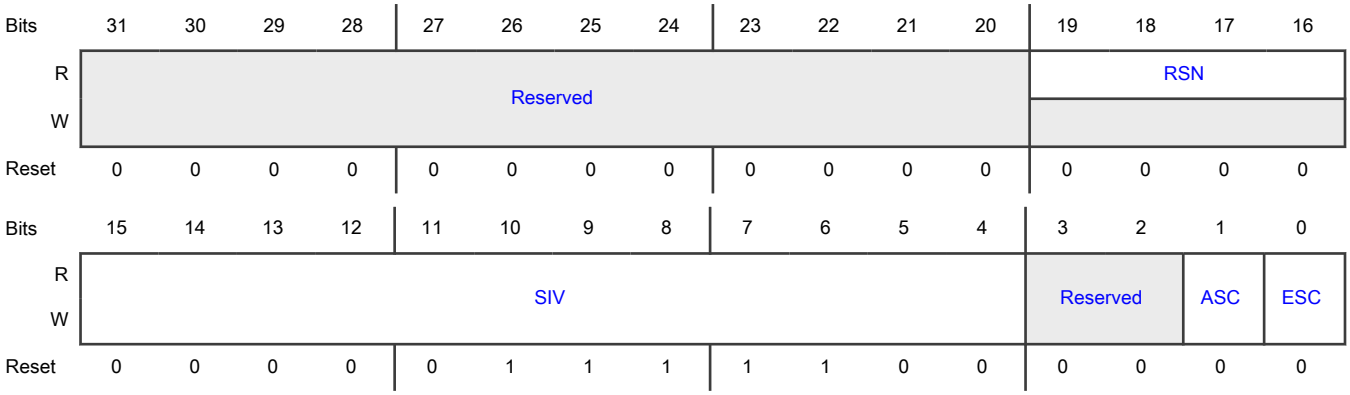
Offset

Register	Offset
DMA_CH1_Slot_Function_Control_Status	11BCh

Function

Contains the control field for slot function and the status for transmit path.

Diagram



Fields

Field	Function
31-20 —	Reserved.
19-16 RSN	Reference Slot Number Provides the current value of the reference slot number in DMA. It is used for slot comparison.
15-4 SIV	Slot Interval Value Controls the period of the slot interval in which the TxDMA fetches the scheduled packets. A value of 0 specifies the slot interval of 1 us while the maximum value 4095 specifies the slot interval of 4096 us. The default or reset value is 0x07C which corresponds to slot interval of 125 us.
3-2 —	Reserved.
1 ASC	Advance Slot Check Indicates whether an advance slot check is enabled. When this field is 1, it enables DMA to fetch the data from the buffer when the slot number (SLOTNUM) programmed in the transmit descriptor is:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• Equal to the reference slot number given in DMA_CH1_Slot_Function_Control_Status[RSN], or • Ahead of the reference slot number by up to two slots. This field is applicable only when DMA_CH1_Slot_Function_Control_Status[ESC] = 1. 0b - Disabled 1b - Enabled
0 ESC	Enable Slot Comparison Enables slot comparison. When this field is 1, it enables the checking of the slot numbers programmed in the transmit descriptor with the current reference given in DMA_CH1_Slot_Function_Control_Status[RSN]. DMA fetches the data from the corresponding buffer only when the slot number is: • Equal to the reference slot number, or • Ahead of the reference slot number by one slot. When this field becomes 0, it disables the checking of the slot numbers. DMA fetches the data immediately after you process the descriptor. **NOTE** You must not enable the UFO (UDP Fragmentation over IPv4)/TSO/USO along with TBS/AVB slot number check. The UFO/TSO/USO involves multiple packets or segments or fragments transmission for single packet received from application and the slot number check are applicable for fetching only first segment/fragment. As a result it might be difficult for you to specify slot number for subsequent packets. 0b - Disable 1b - Enable

75.17.277 DMA Channel 1 Current Application Transmit Descriptor (DMA_CH1_Current_App_TxDesc)

Offset

Register	Offset
DMA_CH1_Current_App_TxDesc	11C4h

Function

Specifies the current transmit descriptor which DMA reads.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CURTDESAPTR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CURTDESAPTR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 CURTDESAPTR	Application Transmit Descriptor Address Pointer Indicates that DMA updates this pointer during the transmit operation. This pointer is 0 on reset.

75.17.278 DMA Channel 1 Current Application Receive Descriptor (DMA_CH1_Current_App_RxDesc)

Offset

Register	Offset
DMA_CH1_Current_App_RxDesc	11CCh

Function

Specifies the current receive descriptor which DMA reads.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CURRDESAPTR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CURRDESAPTR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 CURRDESAPTR	Application Receive Descriptor Address Pointer Indicates that DMA updates this pointer during the receive operation. This pointer is 0 on reset.

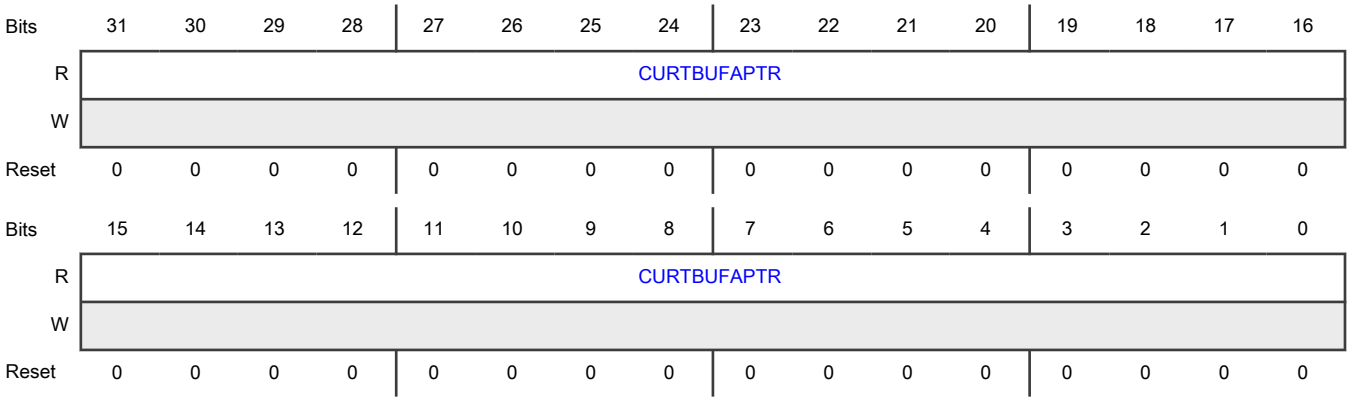
75.17.279 DMA Channel 1 Current Application Transmit Buffer (DMA_CH1_Current_App_TxBuffer)

Offset

Register	Offset
DMA_CH1_Current_App_TxBuffer	11D4h

Function
Specifies the current transmit buffer address which DMA reads.

Diagram



Fields

Field	Function
31-0 CURTBUFAPTR	Application Transmit Buffer Address Pointer Indicates that DMA updates this pointer during the transmit operation. This pointer is 0 on reset.

75.17.280 DMA Channel 1 Current Application Receive Buffer (DMA_CH1_Current_App_RxBuffer)

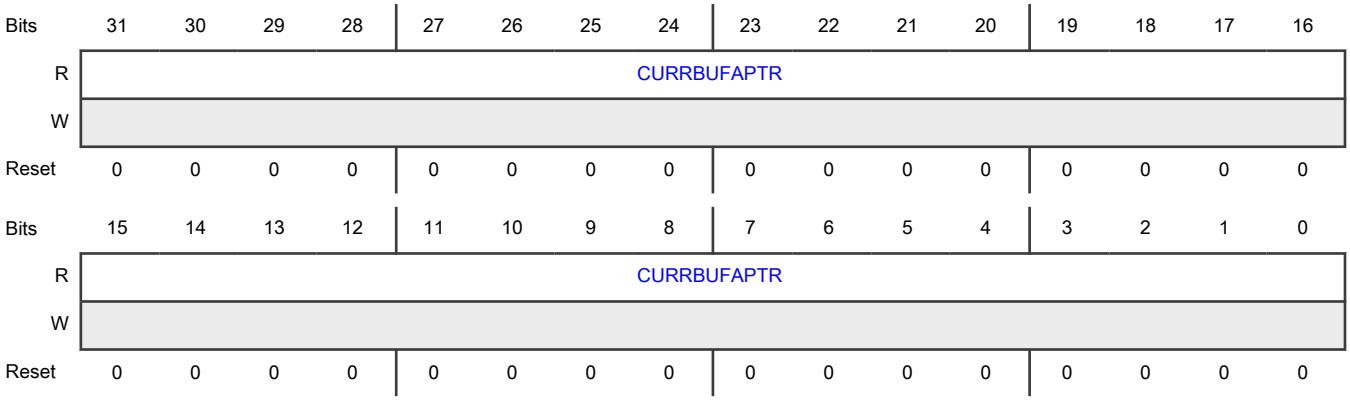
Offset

Register	Offset
DMA_CH1_Current_App_RxBuffer	11DCh

Function

Indicates that DMA_CH0_Current_App_RxBuffer points to the current receive buffer address which the DMA reads.

Diagram



Fields

Field	Function
31-0 CURRBUFAPTR	Application Receive Buffer Address Pointer Indicates that DMA update this pointer during the receive operation. This pointer is 0 on reset.

75.17.281 DMA Channel 1 Status (DMA_CH1_Status)

Offset

Register	Offset
DMA_CH1_Status	11E0h

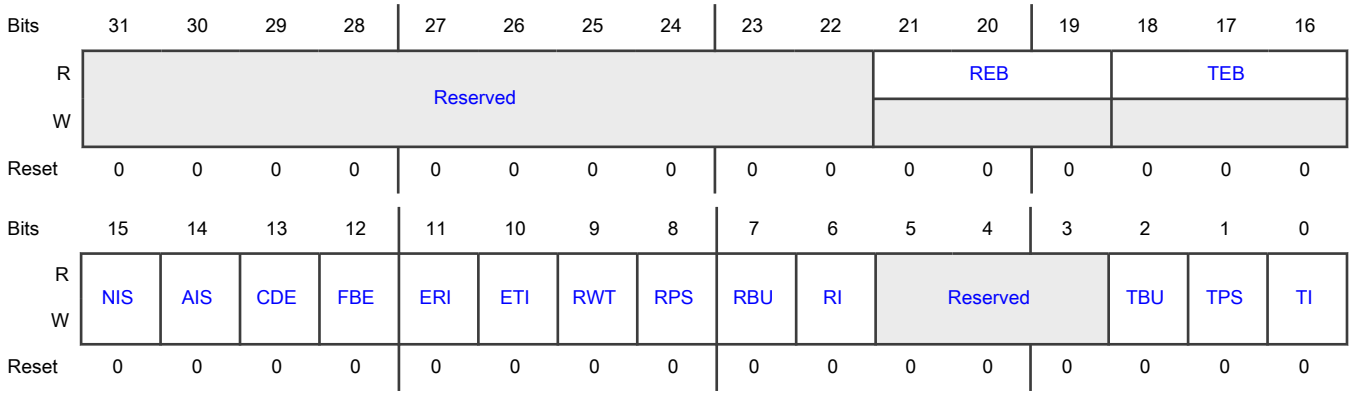
Function

Indicates that to determine the status of DMA, the application reads the status register during an interrupt service routine or polling.

NOTE

The number of DMA_CH(#i)_Status register in the configuration is the higher of number of receive DMA Channels and transmit DMA Channels.

Diagram



Fields

Field	Function
31-22 —	Reserved.
21-19 REB	Rx DMA Error Bits Indicates the type of error that causes a bus error. For example, an error response on the AHB or AXI interface. Bit 21 1'b1 - Error during data transfer by Rx DMA 1'b0 - No Error during data transfer by Rx DMA Bit 20 1'b1 - Error during descriptor access 1'b0 - Error during data buffer access Bit 19 1'b1 - Error during read transfer 1'b0 - Error during write transfer This field is valid only when DMA_CH1_Status[FBE] = 1. It does not generate an interrupt.
18-16 TEB	Tx DMA Error Bits Indicates the type of error that causes a bus error. For example, error response on the AHB or AXI interface. Bit 18 1'b1 - Error during data transfer by Tx DMA 1'b0 - No Error during data transfer by Tx DMA Bit 17 1'b1 - Error during descriptor access

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1'b0 - Error during data buffer access Bit 16 1'b1 - Error during read transfer 1'b0 - Error during write transfer This field is valid only when DMA_CH1_Status[FBE] = 1. It does not generate an interrupt.
15 NIS	Normal Interrupt Summary Indicates whether the normal interrupt summary status is detected. The field value is the logical OR of the following bits when the corresponding interrupt bits enable in DMA_CH0_Interrupt_Enable: Bit 0 - Transmit interrupt Bit 2 - Transmit buffer unavailable Bit 6 - Receive interrupt Bit 11 - Early receive interrupt Only unmasked bits (interrupts for which interrupt enable sets in DMA_CH0_Interrupt_Enable) affect this field. This is a sticky field. You must clear this field (by writing 1 to this field) each time a corresponding bit which sets it, clears. Access restriction apply to this field. Automatically becomes 1 on an internal event occurrence. Writing 1 clears this field and writing 0 has no effect. 0b - Not detected 1b - Detected
14 AIS	Abnormal Interrupt Summary Indicates whether an abnormal interrupt summary status is detected. The field value is the logical OR of the following bits when the corresponding interrupt bits enable in DMA_CH0_Interrupt_Enable: Bit 1 - Transmit process stopped Bit 7 - Receive buffer unavailable Bit 8 - Receive process stopped Bit 10 - Early transmit interrupt Bit 12 - Fatal bus error Bit 13 - Context descriptor error Only unmasked bits affect this field. This is a sticky bit. You must clear this field (by writing 1 to this bit) each time a corresponding bit, which sets it, clears.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Access restriction apply to this field. Automatically becomes 1 on an internal event occurrence. Writing 1 clears this field and writing 0 has no effect. 0b - Not detected 1b - Detected
13 CDE	Context Descriptor Error Indicates whether the context descriptor error status is detected. This field indicates that the DMA Tx/Rx engine receive a descriptor error, which indicates an invalid context in the middle of packet flow (intermediate descriptor) or all one's descriptor in transmit case and on receive side. It indicates that DMA has read a descriptor with either of the buffer address as ones which you can consider invalid. Access restriction apply to this field. Automatically becomes 1 on an internal event occurrence. Writing 1 clears this field and writing 0 has no effect. 0b - Not detected 1b - Detected
12 FBE	Fatal Bus Error Indicates whether the fatal bus error status is detected. This field indicates that a bus error has occurred (as described in the EB field). When this field is 1, it indicates that the corresponding DMA channel engine disables all bus accesses. Access restriction apply to this field. Automatically becomes 1 on an internal event occurrence. Writing 1 clears this field and writing 0 has no effect. 0b - Not detected 1b - Detected
11 ERI	Early Receive Interrupt Indicates whether an early receive interrupt status is detected. When this field is 1, it indicates that the RxDMA has completed the packet data transfer to the memory. In configs supporting ERIC, When DMA_CH1_Rx_Control[ERIC] = 0, this field is 1 only after the Rx DMA fills a complete receive buffer with packet data. When DMA_CH1_Rx_Control[ERIC] = 1, this field is 1 after every burst data transfer from the receive DMA to the buffer. If DMA_CH1_Status[RI] = 1, this field clears automatically. Access restriction apply to this field. Automatically becomes 1 on an internal event occurrence. Writing 1 clears this field and writing 0 has no effect. 0b - Not detected 1b - Detected
10 ETI	Early Transmit Interrupt Indicates whether an early transmit interrupt status is detected.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When this field is 1, it indicates that the TxDMA has completed the packet data transfer to the MTL TXFIFO memory. In configs supporting ERIC: When DMA_CH1_Tx_Control[ETIC] = 0, this field is 1 only after the Tx DMA transfers a complete packet to MTL. When DMA_CH1_Tx_Control[ETIC] = 1, this field is 1 after the packet data transfer completes (partial) from buffers in the Transmit descriptor in which IOC=1. Access restriction apply to this field. Automatically becomes 1 on an internal event occurrence. Writing 1 clears this field and writing 0 has no effect. 0b - Not detected 1b - Detected
9 RWT	Receive Watchdog Timeout Indicates whether a receive watchdog timeout status is detected. This field asserts when it receives a packet with length greater than 2,048 bytes (10,240 bytes when Jumbo Packet mode is enabled). 0b - Not detected 1b - Detected
8 RPS	Receive Process Stopped Indicates whether a receive process stopped status is detected. This field asserts when the receive process enters the stopped state. Access restriction apply to this field. Automatically becomes 1 on an internal event occurrence. Writing 1 clears this field and writing 0 has no effect. 0b - Not detected 1b - Detected
7 RBU	Receive Buffer Unavailable Indicates whether a receive buffer unavailable status is detected. This field indicates that the application owns the next descriptor in the receive list, and the DMA cannot acquire it. The receive process is suspended. To resume processing receive descriptors, the application must change the ownership of the descriptor and issue a receive poll demand command. If this command is not issued, the receive process resumes when you receives the next recognized incoming packet. In Ring mode, the application must advance the receive descriptor tail pointer register of a channel. This field is 1 only when the DMA owns the previous receive descriptor. Access restriction apply to this field. Automatically becomes 1 on an internal event occurrence. Writing 1 clears this field and writing 0 has no effect. 0b - Not detected 1b - Detected
6 RI	Receive Interrupt Indicates whether the receive interrupt status is detected.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field indicates that the packet reception is complete. Bit 31 of RDES3 resets in the last descriptor, and updates the specific packet status information in the descriptor, when the packet reception is complete. The reception remains in the running state. Access restriction apply to this field. Automatically becomes 1 on an internal event occurrence. Writing 1 clears this field and writing 0 has no effect. 0b - Not detected 1b - Detected
5-3 —	Reserved.
2 TBU	Transmit Buffer Unavailable Indicates whether the transmit buffer unavailable status is detected. This field indicates that the application owns the next descriptor in the transmit list, and the DMA cannot acquire it. Transmission is suspended. DMA_Debug_Status0[TPS0] explains the transmit process state transitions. The application must perform these actions, to resume the processing of transmit descriptors: 1. Write 1 to bit 31 of TDES3 to change the ownership of the descriptor. 2. Issue a transmit poll demand command. For Ring mode, the application must advance the transmit descriptor tail pointer register of a channel. Access restriction apply to this field. Automatically becomes 1 on an internal event occurrence. Writing 1 clears this field and writing 0 has no effect. 0b - Not detected 1b - Detected
1 TPS	Transmit Process Stopped Indicates that the transmit process stopped status is detected. This field is 1 when the transmission stops. Access restriction apply to this field. Automatically becomes 1 on an internal event occurrence. Writing 1 clears this field and writing 0 has no effect. 0b - Not detected 1b - Detected
0 TI	Transmit Interrupt Indicates whether the transmit interrupt status is detected. This field indicates that the packet transmission is complete. When transmission completes, bit 31 of TDES3 resets in the last descriptor, and the specific packet status information is updated in the descriptor. Access restriction apply to this field. Automatically becomes 1 on an internal event occurrence. Writing 1 clears this field and writing 0 has no effect.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Not detected 1b - Detected

75.17.282 DMA Channel 1 Miss Frame Counter (DMA_CH1_Miss_Frame_Cnt)

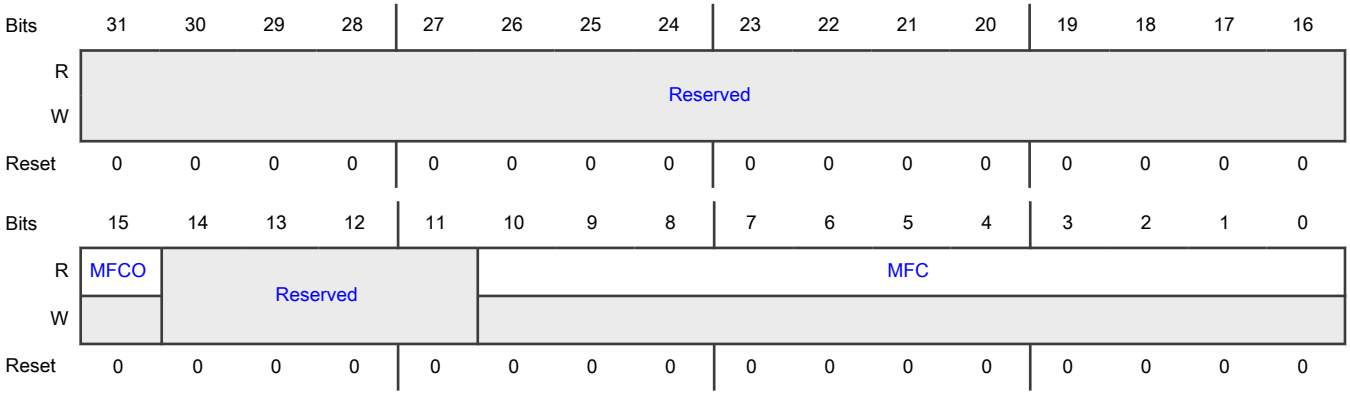
Offset

Register	Offset
DMA_CH1_Miss_Frame_Cnt	11E4h

Function

Provides the number of packet counter that the DMA drops either because of bus error or programming RPF field in DMA_CH${i}_Rx_Control register.

Diagram



Fields

Field	Function
31-16 —	Reserved.
15 MFCO	Overflow status of the MFC Counter Indicates whether the miss frame counter overflow has occurred. When this field is 1, it indicates that the MFC counter does not increments further. This field becomes 0 when this register is read. Access restriction apply to this field. It clears on read and automatically becomes 1 on an internal event occurrence.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Not occurred 1b - Occurred
14-11 —	Reserved.
10-0 MFC	Dropped Packet Counters Indicates the number of packet counters that DMA drops either because of bus error or programing RPF field in DMA_CH${i}_Rx_Control register. This counter is 0 when this register is read. Access restriction apply to this field. It clears on read and automatically becomes 1 on an internal event occurrence.

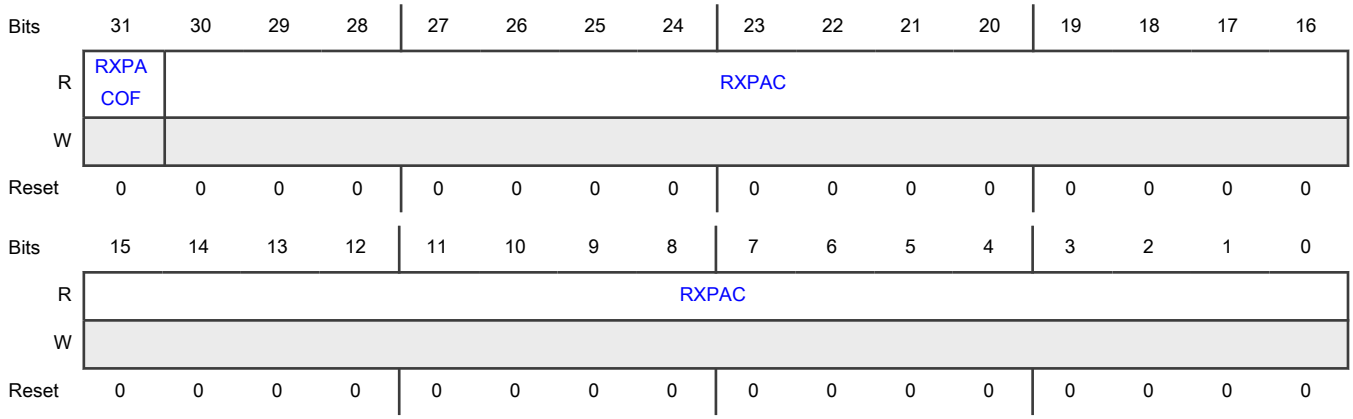
75.17.283 DMA Channel 1 Rx Parser Accept Count (DMA_CH1_RXP_Accept_Cnt)

Offset

Register	Offset
DMA_CH1_RXP_Accept_Cnt	11E8h

Function
Provides the count of the number of frames which the receive parser accepts.

Diagram



Fields

Field	Function
31	Rx Parser Accept Counter Overflow Bit

Table continues on the next page...

Table continued from the previous page...

Field	Function
RXPACOF	Indicates whether the receive parser accept counter overflow have occurred. When this field is 1, it indicates that the DMA_CH1_RXP_Accept_Cnt[RXPAC] has crossed the maximum limit. Access restriction apply to this field. It clears on read and automatically becomes 1 on an internal event occurrence. 0b - Not occurred 1b - Occurred
30-0 RXPAC	Rx Parser Accept Counter Implements whenever a receive parser accept a packet because AF = 1. The counter clears when the register is read.

75.17.284 DMA Channel 1 Rx ERI Count (DMA_CH1_RX_ERI_Cnt)

Offset

Register	Offset
DMA_CH1_RX_ERI_Cnt	11ECh

Function
Provides the count of the number of times ERI asserts.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved				ECNT											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-12	Reserved.

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
11-0 ECNT	ERI Counter ERI Counter When ERIC bit of DMA_CH(#i)_RX_Control register is 1, it indicates that this counter increments for burst transfer which the Rx DMA completes from the start of packet transfer. This counter resets at the start of new packet.

75.18 Glossary

AFM Address filtering module

ARI Application receive interface

ATI Application transmit interface

AVB Audio video bridging

AXI Advanced extensible interface

BTR Base time register

CPT Current presentation time

CRC Cyclic redundancy check

DA Destination address

DCB Data center bridging

DUT Device under test

EOP End of packet

FCS A 32-bit cyclic redundancy check used to detect an in-transit corruption of data. Also marks the end of an Ethernet frame.

FIFO First in first out

GCL Gate control list

GMII Gigabit media independent interface

MAC Media access controller

MCI MAC control interface

MDC Management data clock

MDIO Management data input/output

MII Media independent interface

MRI MAC receive interface

MTI MAC transmit interface

MTL MAC transmit layer

PBL Programmable burst length

PHY	The physical interface transceiver that implements the OSI physical layer for an ethernet network. The IEEE 802.3 standard defines it.
RGMII	Reduced gigabit media independent interface
RMII	Reduced media independent interface
RMON	Remote network monitoring
SA	Source address
SFD	Start frame delimiter
SGMII	Serial gigabit media independent interface
SMA	Station management agent
TCP	Transmission control protocol
UDP	User datagram protocol
VLAN	Virtual local area network

Chapter 76
Gigabit Ethernet Media Access Controller (GMAC)

76.1 Chip-specific GMAC information

76.1.1 GMAC instances and configuration

This chip supports up to two instances of GMAC.

Table 617. GMAC instances

Instance	S32K310/S32K311/ S32K312/S32K322/ S32K342/S32K341/ S32K314/S32K324/S32K344	S32K358/S32K356/S32K348/ S32K338/S32K328	S32K388/S32K389
GMAC	No	Yes (one instance: GMAC)	Yes (two instances: GMAC_0 and GMAC_1)

NOTE

- DWC_EQOS refer to the Synopsys GMAC IP.
- GMAC receives only octet aligned preambles.
- The description for some GMAC registers include references to configurations of the Synopsis Ethernet MAC IP. Refer to MAC_HW_Feature registers for the specific features available in this chip.
- For multi-master applications, it is recommended to configure the DMA_SysBus_Mode[FB] bit to 1 in order to fix the length and avoid any master taking over the bus which would represent a performance drop in the throughput.
- It is essential that all PHY input clocks be present and stable to complete the software reset operation and correctly initialize the GMAC IP. The MAC and DMA controller must reset the logic and all internal registers of the DMA, MTL, and MAC layers to initialize the module (see the "Programming" section within this chapter). Therefore, the reset operation is only complete when all resets in all active clock domains are de-asserted (DMA_MODE[SWR] = 0b0). Once the GMAC IP is successfully initialized, the PHY can be reset without affecting the MAC.

NOTE

- **S32K358/S32K356**: GMAC operates only in Clock options A+ and B, since the module clock becomes lower than the protocol clock (RGMII/RMII/MII clocks) in other modes. For maximum performance option A+ needs to be selected.
- **S32K388/S32K389**: GMAC operates only in Clock options A++ and A+, since the module clock becomes lower than the protocol clock (RGMII/RMII/MII clocks) in other modes. For maximum performance option A++ needs to be selected.

Following key features are supported:

- This chip supports MII/RMII/RGMII interfaces.
- 4 bit MII interface operating at 2.5/25/50 Mhz, MII can run at 10/100/200 Mbps.
- 2 bit RMII interface operating at 50 MHz, RMII can run at 10/100 Mbps
- 4 bit MII-Lite interface operating at 2.5/25 MHz
- 4 bit RGMII interface operation at 125 MHz in DDR mode
- 4 PPS are supported using DMA trigger

- Supports 4-bit Reduced Gigabit MII (RGMII) operating at 125 MHz
- The Ethernet module supports a data rate of 1Gbps (1000 Mbps)
- MTL Receive FIFO size 16384 bytes
- MTL Transmit FIFO size 16384 bytes

NOTE

The DMA referred within GMAC chapter refers to the internal GMAC DMA engine and not the device AXBS master DMA.

NOTE

Register access beyond 8k address space is not supported and will generate a transfer error.

NOTE

Always use store and forward mode if possible. In cut through mode, if multiple masters are active then there is chance (even in round robin arbitration mode) that GMAC DMA may starve for data resulting in insufficient data in TX FIFO leading to Tx underflow. To avoid this, increase MTL_TxQ0_Operation_Mode[TTC] value as needed.

The table below provides the information on the specific configuration utilized for the S32K358, S32K356, S32K348, S32K338, S32K328, S32K388, and S32K389 devices.

Table 618. Specific configuration information

Application Interface		
	Application interface configuration	EQOS-AHB
	Data width	64
	Endian mode	Little Endian
	Address width	32
	CSR interface	APB3 interface
General feature		
	Mode of operation	10/100/1000
	Enable double VLAN processing	Yes
	Enable Que/channel based VLAN tag insertion on Tx	Yes
	Enable SA and VLAN insertion on Tx	Enabled by default
Buffer management		
	MTL Receive FIFO size in bytes	16384
	MTL Receive FIFO size in bytes	16384
	Enable debug memory access	Yes
	Use single port memory	Yes
PHY interface		
	Enable RGMII, RMII only	Enable RGMII, Enable RMII, MII default
	Enabled signal phy interface	No

Table continues on the next page...

Table 618. Specific configuration information (continued)

	Enable MDIO interface	Yes
Filtering		
	Enable Additional 1-31 MAC address registers	2
	Enable Address Filter Hash Table	Yes
	Hash Table Size	256
	Enable VLAN Hash table based filtering	Yes
	Extended Rx VLAN Filter Enable	Yes
	Number of VLAN tag Filters	32
	Enable Layer3 and Layer4 Packet Filter	Yes
	Number of layer3 and Layer 4 packet filters	8
	Enable Flexible Programmable Receive Parser	Yes
	Maximum Packet Header Size for Parsing	256
	Maximum Entries of Parser Look up table	64
	Packet duplication support	Yes
IEEE 1588 timestamp		
	Enable IEEE 1588 Time stamp support	Enabled by default
	IEEE 1588 System Time Support	Both option
	Add IEEE 1588 Higher word register	Yes
	Enable IEEE 1588 sub nanoseconds time stamp support	Yes
	Add IEEE 1588 Auxillary snapshot	No
	Enable Flexible Pulse Per Seconds Output	Yes
	Number of pulse per Second outputs	4
	Enable PTP Timestamp offload feature	No
	Enable One Step timestamp for PTP over UDP/IP feature	No
	Enable One Step timestamp feature	Yes
Multiple queue support		
	Number of transmit Queue	3
	Number of receive Queue	3
	Enable data center Bridging	Yes

Table continues on the next page...

Table 618. Specific configuration information (continued)

	Enable Audio Video Bridging	Yes
	Enable support for AV in Tx queue 1	Yes
	Enable support for AV in Tx queue 2	Yes
	Enable support for AV in Tx queue 3	No
	Enable support for AV in Tx queue 4	No
	Number of DMA Channel on the receive side	3
Time sensitive networking		
	Enable Enhancement to Schedule traffic(EST)	Yes
	Width of Time Interval Field in the Gate Control List	24
	Depth of Gate Control List	256
	Enable Frame Pre-emption support	Yes
Time based scheduling		
	Enable time based Scheduling	Yes
TCP/IP offloading features		
	Enable receive TCP/IP Checksum Check	Enabled by default
	Enable transmit TCP/IP Checksum offload	Enabled by default
	Enable support for Tx COE in Tx queue 0	Yes
	Enable support for Tx COE in Tx queue 1	Yes
	Enable support for Tx COE in Tx queue 2	Yes
	Enable support for Tx COE in Tx queue 3	No
	Enable support for Tx COE in Tx queue 4	No
	Enable TCP Segmentation offloading for TCP/IP	No
	Enable Split Header Feature	Yes
	Enable IPv4 ARP offload	Yes
RMON counters		
	Enable MAC Management Counters(MMC)	Yes
	Enable the MAC Management Counters for Receiver TCP/IP Checksum offload	No
Automotive safety feature		

Table continues on the next page...

Table 618. Specific configuration information (continued)

	Enable all automotive safety feature	Yes

NOTE

Configure the ENET mode in DCM configuration and delay the software reset on IP for proper configuration of PHY mode for RGMII for GMAC_1 instance.

76.2 Introduction

This section and all its sub-sections are Synopsys Proprietary. Used with permission.

GMAC:

- Supports the 10/100/1000 Mbps application in compliance with IEEE802.3-2015 specifications
- Can support advanced applications such as time-sensitive networking and AVB
- Supports a MAC core that has prominent features such as a flexible receive parser, media clock recovery and generation, and safety

76.2.1 Features

GMAC supports these features in addition to the default feature defined in the IEEE802.3 specification:

- MII (10/100 Mbps), RMII (10/100 Mbps), and RGMII (1000 Mbps)
- Time-aware shaper (IEEE802.1bv), time synchronization (IEEE802.1AS-rev), and frame preemption (IEEE802.1Qbu) for time-sensitive networking
- Media clock recovery and generation for AVB
- AMBA 2.0 for AHB master and APB3 interface
- RMII specification version 1.2 from RMII consortium
- RGMII specification version 2.6 from HP/Marvell
- Separate interface for transmit, receive, and control paths
- Two-buffer ring
- Broadcast and multicast packet duplication
- Full-duplex flow control operations (IEEE 802.3x pause packet and priority flow control)
- Network statistics with MAC management (RMON) counters
- Flexibility to control the pulse per second (PPS) output signal
- MDIO clause 22 and clause 45 interface for configuration and management of the PHY device
- Preamble and SFD insertion and deletion

NOTE

Only byte-aligned preamble is supported.

- Automatic CRC and pad generation/stripping option
- Option to disable CRC checking
- Programmable insert packet gap
- Source address insertion or replacement

- VLAN insertion, replacement, and deletion in transmitted packets with per-packet or static-global control, insertion, replacement, and deletion of up to two VLAN tags, insertion, replacement, and deletion of queue or channel-based VLAN tags
- IEEE802.1Q VLAN tag detection and an option to delete the tags in the receive packets
- Programmable safety watchdog timeout limits
- Flexible address filtering modes that allow:
  - Up to two additional 48-bit perfect DA filters with masks for each byte
  - Up to two 48-bit SA comparison checks with masks for each byte
  - 64-bit hash filter for multicast and unicast DA addresses
  - Option to pass all multi-cast addressed packets
  - Promiscuous mode to pass all packets without any filtering for network monitoring
  - Passing of all incoming packets (according to filter) with a status report
- Additional packet filtering that is based on:
  - Virtual local area network (VLAN) tag—perfect match and hash-based filtering, which is based on either outer or inner VLAN tag, is possible
  - Layer 3 and layer 4—TCP or UDP over IPv4 or IPv6
  - Extended VLAN tag—filtering based on four filter selections

76.2.2 Functional block diagram

This figure shows the GMAC block diagram. GMAC has four main blocks to perform all the functions.

- The AHB interface is connected to all DMA channels.
- The DMA arbiter helps in the arbitration of all paths (transmit and receive) in DMA channels.
- Each channel has a separate set of control and status registers (CSRs) for managing the transmit and receive functions, descriptor handling, and interrupt handling.

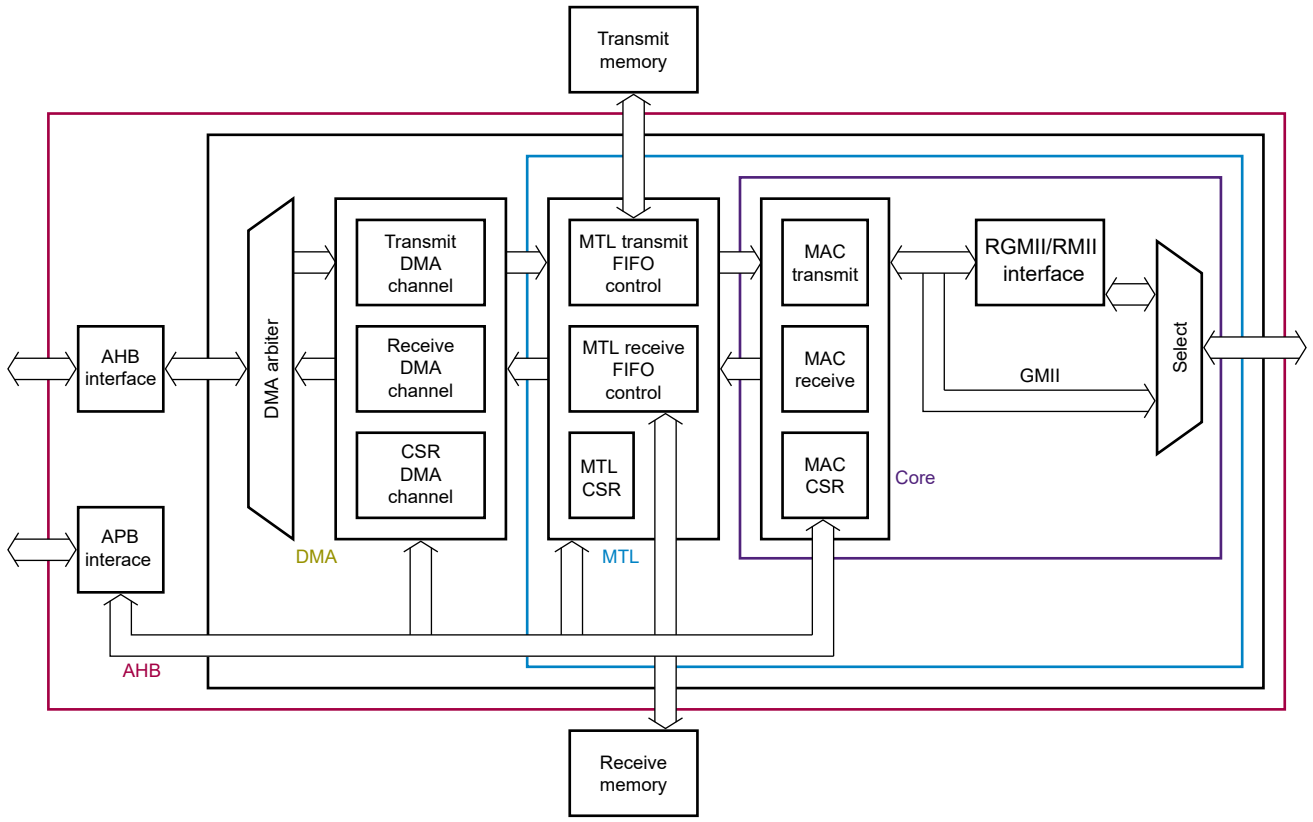


Figure 452. System-level block diagram

76.3 Architecture

This section and all its sub-sections are Synopsys Proprietary. Used with permission.

The section describes GMAC interfaces, protocols, and functionality. These are its main blocks:

- AHB master and APB3 slave interface
- DMA controller
- MTL
- MAC
- Interrupts

76.3.1 AHB master and CSR slave (APB3) interface

The 32-bit APB3 slave interface:

- Provides access to DMA, MTL, and MAC control and status registers (CSRs)
- Supports the 32-bit write and read transfers to these CSRs
- Supports single- and all-burst transfers, and an OKAY response

The DMA block uses the AHB master interface to interact with the external world. The AHB master interface:

- Controls data transfer
- Is AMBA 2.0-compliant with no restrictions
- Supports 64-bit data transfer

- Supports single and INCR transfers
- Supports burst-length modes such as Fixed, Unspecified, and Mixed

76.3.2 DMA inclusions

DMA controller:

- Includes independent transmit (Tx) and receive (Rx) engines, and a CSR space. The transmit engine transfers data from the system memory to the MTL interface, whereas the receive engine transfers data from the MTL interface to the system memory.
- Is designed for packet-oriented data transfers such as packets in Ethernet.
- Communicates with the host through CSR, descriptor lists, and data buffers.

DMA descriptors

DMA supports up to three transmit and three receive descriptor lists (or DMA channels). The base address of each list is written to the respective transmit and receive descriptor list address registers. A descriptor list is forward linked and the next descriptor is always considered at a fixed offset to the current one, and DMA_CH0_Control[DSL] controls the offset. The number of descriptors in the list is programmed using the respective DMA_CH0_TxDesc_Ring_Length and DMA_CH0_RxDesc_Ring_Length registers. After the DMA processes the last descriptor in the list, it automatically jumps to the descriptor in the list address register to create a descriptor ring.

GMAC supports the ring structure for DMA descriptors. For more information, see Descriptors .

The DMA engine uses descriptors to efficiently move data from source to destination with minimal host intervention, and the DMA controller can be programmed to interrupt the host in certain situations. These situations may include packet transmit and receive transfer completion and other normal or error conditions.

The descriptor lists reside in the physical memory address space of the application. Each descriptor can point to a maximum of two buffers in the system memory, enabling two buffers to be used and physically addressed rather than contiguous buffers in the memory.

Data buffers

A data buffer resides in the application physical memory space and consists of an entire packet or part of a packet but cannot exceed a single packet. Buffers contain only data. The buffer status is maintained in the descriptor. Data chaining refers to packets that span multiple data buffers. However, a single descriptor cannot span multiple packets. The DMA skips to the data buffer of the next packet when an EOP is detected.

The next few sections discuss DMA controller.

76.3.2.1 DMA controller bus burst access

DMA engines attempt to transfer data in a burst of the maximum size programmed using DMA_CH0_Tx_Control[TxPBL] and DMA_CH0_Rx_Control[RxPBL] of the respective DMAs, which always access the receive and transmit descriptors in the maximum possible (limited by PBL or 16 * 8/bus width) burst length for 16 bytes to be read. The burst transfers that the DMA initiates can be split into multiple burst transfers according to the AHB requirements and settings specified in DMA System Bus Mode (DMA_SysBus_Mode).

The transmit DMA initiates a data transfer only when sufficient space is available in the MTL transmit queue to accommodate either of these:

- Bytes corresponding to the configured burst (PBL * bus_width/8)
- Remaining bytes in the transmit buffer without EOP
- Number of bytes till EOP

The receive DMA initiates a data transfer in these conditions:

- Sufficient data is available in the MTL receive queue to accommodate the configured burst
- EOP (when it is less than the configured burst length) is detected in the receive queue

DMA indicates the start address and the number of transfers required to the AHB master interface. When the interface is configured for fixed-length burst, it transfers data by using the best combination of the INCR4, INCR8, or INCR16 and SINGLE transactions. If EOP is reached before the fixed burst ends on the AHB interface, DMA performs dummy transfers to complete the fixed burst. Otherwise, DMA_SysBus_Mode[FB] becomes 0. The DMA transfers the data using undefined length (INCR) and SINGLE transactions.

When the AHB interface is configured for address-aligned beats, both DMA engines ensure that the first-burst transfer that AHB initiates is less than or equal to the size of the configured PBL value, and the subsequent beats start at an address aligned to this value. The DMA can only align the address for beats up to size 16 (for PBL > 16) for the AHB interface because it does not support more than INCR16.

76.3.2.2 DMA application data buffer alignment

The transmit and receive data buffers do not have any restrictions on start address alignment. For example, in systems with 32-bit memory, the start address for buffers can be aligned to any of the four bytes. However, the DMA always initiates write transfers with an address aligned to the bus width and dummy data (old data) in the invalid byte lanes. The process of write transfer typically happens during the beginning or end of an Ethernet packet transfer. The software driver must discard the dummy bytes based on the start address of the buffer and size of the packet.

Table 619. Data buffer alignment examples

Examples	
Buffer read	If the transmit buffer address is FF2h (for a 32-bit data bus) and 15 bytes need to be transferred, the DMA reads five full words from address FF0h. However, when DMA transfers data to the MTL transmit queue, the extra bytes (the first two bytes) are dropped or ignored. Similarly, the last 3 bytes of the last transfer are also ignored. The DMA always ensures that it transfers a full 32-bit data to the MTL transmit queue, unless it is the end of the packet.
Buffer write	If the receive buffer address is FF2h (for a 64-bit data bus) and 16 bytes of a received packet need to be transferred, the DMA writes 3 full words from address FF0h. However, the first 2 bytes of the first transfer and the last 6 bytes of the third transfer have dummy data. The DMA considers the offset address only if it is the first receive buffer of the packet. It ignores the offset address and performs full-word writes for the middle and last receive buffers of the packet.

76.3.2.3 DMA buffer size calculations

DMA does not update the size fields in the transmit and receive descriptors. It updates only the status fields (RDES and TDES) of the descriptors. The driver performs the size calculations.

The transmit DMA transfers the exact number of bytes (as the buffer size field of TDES2 indicates) to MAC. If a descriptor is marked as first (TDES3[FD] = 1), the DMA marks the first transfer from the buffer as SOP, and if the descriptor is marked as last (TDES3[LD]), the DMA marks the last transfer from that data buffer as EOP to the MTL.

The receive DMA transfers data to a buffer until the buffer is full or MTL sends the end of packet. When TDES3[FD] = 1, the amount of valid data in a buffer is accurately indicated by the buffer size fields in DMA Channel 0 Rx Control (DMA_CH0_Rx_Control) minus the data buffer pointer offset. The offset is 0 when the data buffer pointer is aligned to the data bus width. If a descriptor is marked as last, the buffer may not be full (as indicated by DMA_CH0_Rx_Control[RBSZ_13_y] and DMA_CH0_Rx_Control[RBSZ_x_0]). To compute the amount of valid data in this final buffer, the driver must read the packet length (PL fields of RDES3[14:0]) and subtract the sum of the buffer sizes of the preceding buffers in this packet. The receive DMA always transfers the start of the next packet with a new descriptor.

76.3.2.4 DMA arbiter

The arbiter inside the DMA module performs the arbitration between the transmit and receive channel accesses to the AHB master interface. These two types of arbitrations are supported:

- Round-robin: If DMA_Mode[DA] = 0 and both transmit and receive DMAs simultaneously request access, the arbiter allocates the data bus in ratio sets defined in DMA_Mode[PR].
- Fixed-priority: If DMA_Mode[DA] = 1, receive DMA is always prioritized over transmit DMA for data access by default. If DMA_Mode[TXPR] = 1, transmit DMA is prioritized over receive DMA as explained in Table 636.

76.3.3 MTL

MTL provides the FIFO memory interface to buffer and regulate the packets between system memory and MAC. It also enables data to be transferred between the system clock and MAC clock domains. The MTL layer has two data paths: transmit path and receive path. MTL communicates with the host through ATI on the transmit path and through ARI on the receive path.

76.3.3.1 Transmit path

The internal DMA:

- Handles all transactions for the transmit path through ATI.
- Pushes the Ethernet packet reads from system memory to the corresponding queue.

The Ethernet packet, then, pops out and is transferred to MAC when the queue reaches its threshold (Threshold mode) or if the complete packet is in the queue (Store-and-Forward mode). When EOP is transferred, the status of the transmission is taken from MAC and transferred back to the internal DMA.

76.3.3.1.1 Transmit control word

This control information related to packet transmission is provided as part of the control word through the ATI interface:

- Packet length (if DCB is enabled with WFQ scheduling algorithm)
- CRC pad control
- Source address insertion control
- VLAN insertion and replacement control and VLAN tag for outer and inner VLAN tags
- TCP/IP checksum insertion control
- One-step timestamping control correction
- Transmit timestamp enable

76.3.3.1.2 Transmit operation

These two modes of operation trigger data read towards MAC:

- Threshold (or cut-through) mode: This is the default mode. In this mode, as soon as the number of bytes in the queue crosses the configured threshold level (or when the end of packet is written before the threshold is crossed), the data is ready to be popped out and forwarded to MAC. The threshold level is configured by using MTL_TxQ0_Operation_Mode[TTC].
- Store-and-Forward mode: In this mode, MTL pops the packet out to MAC only when one or more of the following conditions are true:
  - A complete packet is stored in the queue.
  - The transmit FIFO becomes almost full.
  - The ATI watermark becomes low.

  The watermark becomes low when the requested queue is not spacious enough to accommodate the requested burst length on ATI. Therefore, MTL, when operating in Store-and-Forward mode, allows packet transmission even if the packet length is bigger than the transmit queue size.

You can flush the complete content of the transmit queue by writing 1 to MTL_TxQ0_Operation_Mode[FTQ] or MTL_TxQ1_Operation_Mode[FTQ], depending on the queue. Doing so initializes the queue pointers to the default state. The FTQ

field returns to 0 by itself. If you write 1 to the FTQ field during a packet transfer from MTL to MAC, MTL stops further transfers because MTL considers the queue to be empty. Therefore, an underflow event occurs at the MAC transmitter.

76.3.3.1.3 Initialization flow

After reset, MTL is ready to manage the data flow between DMA and MAC.

- With single-transmit queue configurations, there are no initialization requirements for enabling MTL.
- With multiple-transmit queue configurations, initialize the queue size for each of the queues by programming MTL_TxQ0_Operation_Mode[TQS] corresponding to a transmit queue. Also, initialize the MAC block. Internal DMA controllers must be individually enabled through their respective CSRs.

76.3.3.2 Receive path

MAC sends packets to the MTL receive module and pushes them into the receive queue. MTL indicates the status (fill level) of the queue to the application or DMA when it crosses the configured receive threshold (MTL_RxQ0_Operation_Mode[RTC]), or when the MTL receive module receives the complete packet in Store-and-Forward mode. MTL also indicates the fill level of the queue so that DMA can initiate preconfigured burst transfers to the AHB interface.

76.3.3.2.1 Receive operation

During a receive operation, MTL is MAC's slave. This is the general sequence of events:

1. When MAC receives a packet, it indicates the availability of receive data.
2. MAC indicates the SOP and EOP delimiters.
3. MTL accepts the data and pushes it into the corresponding receive queue.
4. When MAC transfers EOP to the MTL receive module, MAC also drives the status word that MTL pushes into the corresponding receive queue.
5. MTL takes the data out of the queue and sends it to DMA depending on these modes:
   - Threshold mode
   - Store-and-Forward mode

   The sections that follow discuss these modes.

76.3.3.2.1.1 Threshold mode

In Threshold (default) mode, MTL reads the data and indicates its availability to the application or DMA when one of these occurs:

- Data bytes equal to the threshold amount are written to the receive queue (to MTL_RxQ0_Operation_Mode[RTC] and MTL_RxQ1_Operation_Mode[RTC]).
- A full packet of data is received into the queue.

76.3.3.2.1.2 Store-and-Forward mode

In this mode (when MTL_RxQ0_Operation_Mode[RSF] = 1), the initial receive queue locations are reserved for the status words before writing the SOP. A packet is read out only after it is completely written into the receive queue. In this mode, all error packets are dropped (if configured through MTL_RxQ0_Operation_Mode[FEP]) in such a way that only valid packets are read and forwarded to the application.

76.3.3.2.2 Multi-packet receive operation

In Threshold mode, the packet status is available immediately after the packet data. In Store-and-Forward mode, the packet data is available after the packet status. MTL is capable of storing any number of packets in the queue as long as it is not full.

If MAC receives a packet when the corresponding receive queue is full, MTL ignores that packet and overflow pulse generates on the corresponding receive queue. In addition, MTL increments the overflow counter in MTL Rx Queue 0 Missed Packet Overflow Count (MTL_RxQ0_Missed_Packet_Overflow_Cnt) for the corresponding queue.

76.3.3.2.3 Error handling in receive operation

MTL performs these actions if the MTL receive queue is full before it receives the EOP data from MAC:

1. Declares an overflow
2. Drops the whole packet (including the status word)
3. Increments the overflow counter in DMA (MTL Rx Queue 0 Missed Packet Overflow Count (MTL_RxQ0_Missed_Packet_Overflow_Cnt))

This is true even if MTL_RxQ0_Operation_Mode[FEP] is 1.

If the start address of such a packet is already transferred to the read controller, the rest of the packet is dropped and a dummy EOP is written to the queue along with the status word with overflow status. The status indicates a partial packet because of overflow. In such packets, the Packet Length field is invalid. If the MTL receive queue is configured to operate in the Store-and-Forward mode and the length of the received packet is more than the queue size, overflow occurs and all such packets are dropped.

The MTL receive control logic can filter errors and undersized packets, if enabled by using MTL_RxQ0_Operation_Mode[FEP] and MTL_RxQ0_Operation_Mode[FUP]. If the start address of one such packet is already transferred to the receive queue read controller, the packet is not filtered. The start address of the packet is transferred to the read controller after the packet crosses the receive threshold defined using MTL_RxQ0_Operation_Mode[RTC].

If the MTL receive queue is configured to operate in Store-and-Forward mode, all error packets can be filtered and dropped. For the application or DMA to flush the error packet being read from the queue, it must assert the flush signal. MTL then stops transferring data to the application (DMA). It internally reads the rest of the packet and drops it. MTL then starts transferring the next packet, if it's available.

76.3.4 MAC

MAC can support the MII and RMII PHY interface and it consists of:

- MTI
- MRI
- MCI

76.3.4.1 MAC transmission

The MAC transmission process is as follows:

1. The transmission initiates when MTL pushes in data with the SOP signal asserted.
2. After the SOP signal is detected, MAC accepts the data and begins transmitting to RMII or MII.
3. After the EOP signal is transferred to MAC, it performs one of these steps:
   - Completes normal packet transmission and sends the transmission status to MTL
   - Sends retry requests if a normal collision (in Half-Duplex mode) occurs during transmission, and until one of the following is true:
     - Packet is successfully transmitted.
     - Maximum retry requests have expired. When this happens, MAC aborts the packet transmission with "excessive collision transmit" status. MAC accepts and drops all further data until it receives the next SOP. The MTL block must retransmit the same packet from SOP on observing a retry request (in the status) from MAC.
   - If any one of the following happens, MAC aborts the packet transmission:

— No carrier (Half-Duplex mode)
— Loss of carrier (Half-Duplex mode)
— Excessive deferral (Half-Duplex mode)
— Late collisions (Half-Duplex mode)
— Jabber
— MAC accepts and drops all further data until it receives the next SOP

4. MAC issues an underflow status if MTL is not able to provide data continuously during transmission. Until the next SOP is received, MAC accepts and drops all further data.
5. During the normal transfer of a packet from MTL, if MAC receives an SOP without getting an EOP for the previous packet, it ignores the SOP and considers the new packet as a continuation of the previous packet.

76.3.4.2 MAC reception

The receive operation initiates as follows:

1. MAC detects a state-of-frame data on RMII or MII.
2. MAC strips the preamble and SFD before processing an Ethernet packet.
3. The MAC AFM:
   - Checks the header fields (SA and DA) of an incoming packet for filtering
   - Verifies the CRC contained in the packet's FCS field
4. MAC's AFM checks the header fields (SA and DA) of an incoming packet for filtering.
5. FCS verifies the CRC of the received packet.
6. MAC stores the received packet in a shallow buffer until address filtering is performed.
7. AFM drops the packet if it fails the address filter.

76.3.5 Interrupts

GMAC supports interrupt coalescing, which reduces the number of interrupts generated by the module and reduces the CPU load. MAC Interrupt Status (MAC_Interrupt_Status) captures various interrupt events. If the interrupt enable field of an interrupt is 1, the corresponding interrupt generates based on the event status in the Status registers. The Interrupt mode (INTM) field decides whether the interrupt signal is a level signal or pulse signal. See the following table and interrupt map file attached to this document for details.

Table 620. Interrupt request

Interrupt request	Interrupt Enable / Mask Register	Interrupt Flag Register
Common interrupt	MAC Interrupt Enable (MAC_Interrupt_Enable)	MAC Interrupt Status (MAC_Interrupt_Status)
	MMC Receive Interrupt Mask (MMC_Rx_Interrupt_Mask)	MMC Receive Interrupt (MMC_Rx_Interrupt)
	MMC Transmit Interrupt Mask (MMC_Tx_Interrupt_Mask)	MMC Transmit Interrupt (MMC_Tx_Interrupt)
	MMC FPE Transmit Interrupt Mask (MMC_FPE_Tx_Interrupt_Mask)	MMC Transmit FPE Fragment Counter Interrupt Status (MMC_FPE_Tx_Interrupt)

Table continues on the next page...

Table 620. Interrupt request (continued)

Interrupt request	Interrupt Enable / Mask Register	Interrupt Flag Register
	MMC FPE Receive Interrupt Mask (MMC_FPE_Rx_Interrupt_Mask)	MMC Receive Packet Assembly Error Counter Interrupt Status (MMC_FPE_Rx_Interrupt)
	MTL Debus Control (MTL_DBG_CTL)	MTL Debus Status (MTL_DBG_STS)
	MTL EST Interrupt Enable (MTL_EST_Intr_Enable)	MTL EST Status (MTL_EST_Status)
	MTL Rx Parser Interrupt Control Status (MTL_RXP_Interrupt_Control_Status)	MTL Rx Parser Interrupt Control Status (MTL_RXP_Interrupt_Control_Status)
	MTL Queue 0 Interrupt Control Status (MTL_Q0_Interrupt_Control_Status)	MTL Queue 0 Interrupt Control Status (MTL_Q0_Interrupt_Control_Status)
	MTL Queue 1 Interrupt Control Status (MTL_Q1_Interrupt_Control_Status)	MTL Queue 1 Interrupt Control Status (MTL_Q1_Interrupt_Control_Status)
	MTL Queue 2 Interrupt Control Status (MTL_Q2_Interrupt_Control_Status)	MTL Queue 2 Interrupt Control Status (MTL_Q2_Interrupt_Control_Status)
	DMA Channel 0 Interrupt Enable (DMA_CH0_Interrupt_Enable) (transfer complete interrupts only)	DMA_CH0_Interrupt_Status
	DMA Channel 1 Interrupt Enable (DMA_CH1_Interrupt_Enable) (transfer complete interrupts only)	DMA_CH1_Interrupt_Status
	DMA Channel 2 Interrupt Enable (DMA_CH2_Interrupt_Enable) (transfer complete interrupts only)	DMA_CH2_Interrupt_Status
		DMA Interrupt Status (DMA_Interrupt_Status) (First level status. Indicates major block event source: DMA channel, MTL or MAC)
		MTL Interrupt Status (MTL_Interrupt_Status) (Status summary for MTL block interrupts: Rx Parser, EST, FIFO Debug, Queue 1/Queue 0)
Tx interrupt 0/1/2	DMA Channel 0 Tx Control (DMA_CH0_Tx_Control)	DMA Channel 0 Status (DMA_CH0_Status)
	DMA Channel 0 Interrupt Enable (DMA_CH0_Interrupt_Enable)	
	DMA Channel 1 Tx Control (DMA_CH1_Tx_Control)	DMA Channel 1 Status (DMA_CH1_Status)
	DMA Channel 1 Interrupt Enable (DMA_CH1_Interrupt_Enable)	
	DMA Channel 2 Tx Control (DMA_CH2_Tx_Control)	DMA Channel 2 Status (DMA_CH2_Status)

Table continues on the next page...

Table 620. Interrupt request (continued)

Interrupt request	Interrupt Enable / Mask Register	Interrupt Flag Register
	DMA Channel 2 Interrupt Enable (DMA_CH2_Interrupt_Enable)	
Rx interrupt 0/1/2	DMA Channel 0 Rx Control (DMA_CH0_Rx_Control)	DMA Channel 0 Status (DMA_CH0_Status)
	DMA Channel 0 Interrupt Enable (DMA_CH0_Interrupt_Enable)	
	DMA Channel 1 Rx Control (DMA_CH1_Rx_Control)	DMA Channel 1 Status (DMA_CH1_Status)
	DMA Channel 1 Interrupt Enable (DMA_CH1_Interrupt_Enable)	
	DMA Channel 2 Rx Control (DMA_CH2_Rx_Control)	DMA Channel 2 Status (DMA_CH2_Status)
	DMA Channel 2 Interrupt Enable (DMA_CH2_Interrupt_Enable)	
Safety interrupt	MTL ECC Interrupt Enable (MTL_ECC_Interrupt_Enable)	MTL ECC Interrupt Status (MTL_ECC_Interrupt_Status)
		MTL Safety Interrupt Status (MTL_Safety_Interrupt_Status)
	MTL DPP Control (MTL_DPP_Control)	MAC DPP FSM Interrupt Status (MAC_DPP_FSM_Interrupt_Status)
	MAC FSM Control (MAC_FSM_Control)	

These are the interrupt output signals that are synchronous to the CSR clock.

- Sbd_intr_o – common interrupt
- Sbd_perch_tx_intr_o[max_dma_ch] – interrupt per transmit channel
- Sbd_perch_rx_intr_o [max_dma_ch] – interrupt per receive channel

NOTE

max_dma_ch = number of transmit/receive queues, which is 3

The sbd_intr_o common interrupt is a level signal. When it is asserted, a corresponding interrupt event source can be found in DMA Interrupt Status (DMA_Interrupt_Status), which is a read-only register that contains the event source fields corresponding to each DMA channel (transmit and receive queue pair), MAC transaction layer, and MAC blocks. You must then read the following registers and look for the fields that are 1:

- MAC Interrupt Status (MAC_Interrupt_Status)
- MTL Interrupt Status (MTL_Interrupt_Status)
- DMA Channel 0 Status (DMA_CH0_Status)

The sbd_intr_o interrupt deasserts only when all the enabled interrupt events are clear in their respective status registers, and correspondingly, all the fields in DMA Interrupt Status (DMA_Interrupt_Status) are 0. DMA Channel 0 Status (DMA_CH0_Status) captures all the interrupt events of that transmit DMA and receive DMA channel pair.

DMA Channel 0 Interrupt Enable (DMA_CH0_Interrupt_Enable) contains the corresponding enable fields for each of the interrupt events. These are the two groups of interrupts in the DMA channel:

- Normal—DMA_CH0_Status[NIS] indicates this interrupt, which is used for events that occur during the normal transfer of packets (TI, RI, TBU).
- Abnormal—DMA_CH0_Status[AIS] indicates this interrupt, which is used for error events.

Interrupts are not queued. If the same interrupt event occurs again before the driver responds to the previous one, no additional interrupts generate.

The common sbd_intr_o output signal asserts for the transfer complete interrupts only when the corresponding interrupts are enabled in DMA Channel 0 Interrupt Enable (DMA_CH0_Interrupt_Enable).

GMAC also supports these per-channel transfer-complete interrupt signals:

- sbd_perch_tx_intr_o[max_dma_ch] (transmit per channel interrupts)
- sbd_perch_rx_intr_o[max_dma_ch] (receive per channel interrupts)

The behavior of the RI/TI/sbd_perch_tx_intr_o[]/sbd_perch_rx_intr_o[] changes depends on the configuration specified in DMA_Mode[INTM]. This table explains the transfer complete interrupt behavior.

Table 621. Transfer complete interrupt behavior

Interrupt mode	Behavior of sbd_perch_tx_intr_o[] and sbd_perch_rx_intr_o[]	Behavior of the TI/RI interrupts and sbd_intr_o
INTM = 1	In this mode, these signals indicate the values of the corresponding DMA_CH0_Status[RI] and DMA_CH0_Status[TI] fields when DMA_CH0_Status[RI] and DMA_CH0_Status[TI] are 1, respectively. Therefore, they are level signals that you clear by writing 1 to these fields. The signals do not assert when DMA_CH0_Status[RI] or DMA_CH0_Status[TI] is 0.	DMA_CH0_Status[RI] and DMA_CH0_Status[TI] are configured as explained. For any RI/TI events: • The sbd_intr_o signal does not assert. • DMA_CH0_Status[NIS] remains 0.
INTM = 2	In this mode, RI/TI are queued and indicate the values of the corresponding DMA_CH0_Status[RI] and DMA_CH0_Status[TI] fields when any of these fields is 1. RI and TI are level signals that you clear by writing 1 to these fields. However, the fields become 1 again if another TI/RI event is detected before DMA_CH0_Status[RI] and DMA_CH0_Status[TI] become 1 for the previous event.	DMA_CH0_Status[RI] and DMA_CH0_Status[TI] become 1 whenever the transfer complete event is detected and are reset whenever the software driver changes them to 0 by writing 1. However, if another transfer complete event is detected before the software changes it to 0 (services it), GMAC automatically writes 1 to these fields again. The generation of the sbd_intr_o signal; however, is not based on DMA_CH0_Status[RI] or DMA_CH0_Status[TI].

76.4 Signal descriptions

This section and all its sub-sections are Synopsys Proprietary. Used with permission.

76.4.1 Module signals

This table provides port/signal names and their descriptions. See the IOMUX file attached to this document for details.

Table 622. Module signals

Port Name	I/O	Description
CLK_TX_I	I	MAC Transmitter Clock. The external PHY or PLL provides this transmission clock. This is 312.5 MHz in 2.5 Gbps mode, 125 MHz in 1 Gbps mode, 25 MHz in 100 Mbps mode, 2.5 MHz in 10 Mbps mode. All transmission signals generated by the MAC are synchronous to this clock. This clock is required for all PHY interfaces.
CLK_RX_I	I	MAC Receiver Clock. The external PHY provides this receive clock for RGMII, GMII, MII, and RMII interfaces. This clock is 312.5 MHz in 2.5 Gbps mode, 125 MHz in 1 Gbps mode, 25 MHz in 100 Mbps mode, 2.5 MHz in 10 Mbps mode. All RGMII, GMII, and MII receive signals received by the MAC are synchronous to this clock. This clock input is required for all PHY interfaces.
CLK_RMII_I	I	RMII Clock. This is the 50-MHz clock used by the RMII interface. When the RMII interface is selected, the CLK_RX_I clock should be 25 or 2.5 MHz and it should be derived from this clock.
CLK_PTP_REF_I	I	PTP Reference Clock. This reference clock, provided by an external oscillator or a timing source, is used for the timestamp logic.
PTP_PPS_O[3:0]	O	Pulse Per Second. This signal is high based on the PPS mode selected in the MAC_PPS_Control register. When PPS is programmed in Media Clock recovery Mode, this port indicates the recovered clock. See the device IOMUX sheet for details on the number of these signals available.
MCG_PST_TRIG_I[3:0]	I	Media Clock Generation Trigger Input Trigger Input to the DUT to capture presentation time. Based on presentation control value of MAC_PPS_Control register, this signal can be pulse or level signal. Width of this IO changes with the configured number of PPS instances. See the device IOMUX sheet for details on the number of these signals available.
PHY_INTF_SEL_I[2:0]	I	PHY Interface Select. This signal selects the PHY interface of the MAC: • 000: GMII or MII • 001: RGMII • 100: RMII This signal is sampled only during reset assertion, and it is ignored after that. The reset is generated internally in the IP, and it gets activated when either the input reset (pwr_on_rst_n) or the software reset is asserted.

Table continues on the next page...

Table 622. Module signals (continued)

Port Name	I/O	Description
PHY_TXEN_O	O	PHY Transmit Data Enable. The MAC drives this signal. This signal has multiple functions depending on the selected PHY interface as described in the following list: • GMII/MII: When high, indicates that valid data is being transmitted on the PHY_TXD_O bus. Synchronous to: CLK_TX_I • RMII: When high, indicates that valid data is being transmitted on the PHY_TXD_O bus. Synchronous to: CLK_RMII_I • RGMII: This signal is the control signal (rgmii_tctl) for the Transmit data, and it is driven on both edges of the clock. Synchronous to: CLK_TX_I, clk_tx_180_i
PHY_TXER_O	O	PHY Transmit Error. The MAC drives this signal. It has multiple functions depending on the selected PHY interface as described in the following list: • GMII: When this signal is high, it indicates a Transmit error or carrier extension on the PHY_TXD_O bus. Synchronous to:CLK_TX_I • MII, RMII, RGMII: Not used and tied to low. According to the IEEE standard, the TX_ER signal (PHY_TXER_O signal in the GMAC) is optional in the MII mode. However, if you have enabled the Energy Efficient Ethernet feature, the PHY_TXER_O signal is required.
PHY_TXD_O[7:0]	O	PHY Transmit Data. This is a group of transmit data signals driven by the MAC. It has multiple functions depending on the selected PHY interface as described in the following list: • GMII: All eight bits provide the GMII transmit data byte. The data is valid only when the PHY_TXEN_O and PHY_TXER_O signals are high. Synchronous to: CLK_TX_I • MII: Bits[3:0] provide the MII transmit data nibble. The data is valid only when the PHY_TXEN_O and PHY_TXER_O signals are high. Synchronous to: CLK_TX_I • RGMII: Bits[3:0] provide the RGMII transmit data. The data bus changes with both rising and falling edges of the transmit clock (CLK_TX_I). The data is valid only when the PHY_TXEN_O signal is high. Synchronous to: CLK_TX_I and clk_tx_180_i • RMII: Bits[1:0] provide the RMII transmit data. The data is valid only when the PHY_TXEN_O signal is high. Synchronous to: CLK_RMII_I
PHY_CRS_I	I	PHY CRS. This signal is valid only in the GMII or MII mode. The PHY drives this signal high when the Transmit or Receive medium is not idle. The PHY drives this signal low when both Transmit and Receive mediums are idle. This signal is not synchronous to any clock.

Table continues on the next page...

Table 622. Module signals (continued)

Port Name	I/O	Description
PHY_COL_I	I	PHY Collision. This signal is valid only in the GMII or MII mode. The PHY drives this signal high when a collision is detected on the medium. This signal is not synchronous to any clock.
PHY_RXDV_I	I	PHY Receive Data Valid. The PHY drives this signal. It has multiple functions depending on the selected PHY interface as described in the following list: • GMII or MII: Indicates that the data on the PHY_RXD_I bus is valid. It remains high continuously from the first recovered byte or nibble of the packet through the final recovered byte or nibble of the packet. Synchronous to: CLK_RX_I • RGMII: This is the Receive control signal that is used to qualify the data received PHY_RXD_I. This signal is sampled both edges of the clock. Synchronous to: CLK_RX_I, clk_rx_180_i • RMII: Contains the crs and data valid information of the Receive interface. Synchronous to: CLK_RMII_I
PHY_RXER_I	I	PHY Receive Error. The PHY drives this signal. It has multiple functions depending on the selected PHY interface as described in the following list: • GMII or MII: Indicates an error or carrier extension (in GMII) in the received packet the PHY_RXD_I bus. Synchronous to: CLK_RX_I • RGMII, RMII: This signal is not used.
PHY_RXD_I[7:0]	I	PHY Receive Data. This is a group of data signals received from the PHY. It has multiple functions depending on the selected PHY interface as described in the following list: • GMII: All eight bits provide the GMII receive data byte. The data is valid only when the PHY_RXDV_I and PHY_RXER_I signals are high. Synchronous to: CLK_RX_I • MII: Bits [3:0] provide the MII receive data nibble. The data is valid only when the PHY_RXDV_I and PHY_RXER_I signals are high. Synchronous to: CLK_RX_I • RGMII: Bits [3:0] provide the RGMII receive data. The data bus is sampled with both rising and falling edges of the receive clock (CLK_RX_I). The data is valid only when the PHY_RXDV_I signal is high. Synchronous to: CLK_RX_I, clk_rx_180_i • RMII: Bits [1:0] provide the RMII receive data. he data is valid only when the PHY_RXDV_I signal is high. Synchronous to: CLK_RMII_I
GMII_MDC_O	O	Management Data Clock.

Table continues on the next page...

Table 622. Module signals (continued)

Port Name	I/O	Description
		The MAC provides timing reference for the GMII_MDI_I and GMII_MDO_O signals on GMII or MII through this aperiodic clock. This clock is generated from the application clock through a clock divider controlled by CR field of MAC_MDIO_Address register.
GMII_MDI_I	I	Management Data In. The PHY generates this signal to transfer register data during a read operation. This signal is driven synchronously with the GMII_MDC_O clock.
GMII_MDO_O	O	Management Data Out. The MAC uses this signal to transfer control and data information to the PHY.

76.5 Using PHY interfaces

This section is Synopsys Proprietary. Used with permission.

This module support the following modes:

- RGMII 100/1000 Mbps inerface
- RMII 10/100 Mbps interface
- MII 10/100 Mbps interface

Phy_intf_sel input signal decides which mode is selected. Samples the Phy_intf_sel signal at reset. See Signal descriptions for more information. MAC_Configuration[PS] and MAC_Configuration[FES] selects the mode's speed.

The module supports the IEEE802.3-2015 specification for MII, RMII specification version 1.2 from RMII consortium, and RGMII specification version 2.6 from HP/Marvell.

NOTE

RMII reference clock (50 MHz) can be fed to IP from an external source or internally from PLL on SoC.

The module supports the access of the phy registers through SMA. It is a two wire Station management interface (MIM):

1. MDC
2. MDIO

According to IEEE 802.3 specification, maximum operating frequency of MDC is 2.5 Mhz which system clock derives using a divider. MAC_MDIO_Address[CR] programs to generate different MDC clock frequency.

SMA supports MDIO clause 45 and clause 22 frame structure per the IEEE802.3 specification. Writing 1 to MAC_MDIO_Address[C45E] enables the clause 45 frame structure.

The MII interface reduces the accuracy of the 1588 timestamp if it is overclocked to run at 200 Mbps. This happens because the MAC logic uses the MII interface speed,to adjust or compensate for the timestamps taken at MII, as compared to the time generated in the PTP clock domain as specified in MAC_Configuration[FES].

76.6 VLAN and double VLAN insertion, deletion, replacement and tagging

This section and all its sub-sections are Synopsys proprietary. These are used with permission.

The following sections describe the VLAN and double VLAN features.

76.6.1 Double VLAN processing

In the double VLAN tagging processing, the module supports the processing of two VLAN tags. The two VLAN tags are:

1. Inner VLAN tag (C-VLAN)
2. Outer VLAN tag (S-VLAN)

If there is only one tag in a packet then it is considered as an outer tag.

The module uses MAC_VLAN_Incl and MAC_Inner_VLAN_Incl and MAC_VLAN_Tag registers to support the following functions in a double VLAN processing feature.

- Insertion, replacement, or deletion of up to two VLAN tags in the transmit path.
- Packet filtering and stripping on the basis of any one of the two VLAN tags in the receive path. Stripping and providing up to two VLAN tags in the receive path as a part of the receive status.

76.6.1.1 Transmit path

Table 623. Double VLAN processing features in transmit path

Feature	Description
Support for C-VLAN and S-VLAN tag types	The inner or outer VLAN tag can be of C-VLAN and S-VLAN type. MAC_VLAN_Incl[CSVL] and MAC_Inner_VLAN_Incl[CSVL] specifies the VLAN type. The module supports the processing of any sequence of outer and inner VLAN tags. **NOTE** The module does not support the C-VLAN and S-VLAN sequence. MAC does not examine whether the packet, which the application provides has a valid sequence of the VLAN tag types or the insertion or replacement operation results in an invalid sequence of VLAN tag type. Therefore, the application must provide a correct sequence of VLAN tag types and program the MAC in such a way that it results in correct sequence of VLAN tag types in the transmitted packet. The application must ensure the following: • The inner tag should not be S-VLAN when outer C-VLAN Tag insertion is enabled. • The outer tag should not be C-VLAN when inner S-VLAN Tag insertion is enabled. • The inner tag should not be S-VLAN when C-VLAN replaces the outer tag. • The outer tag should not be C-VLAN when S-VLAN replaces the inner tag.
VLAN tag deletion	MAC_VLAN_Incl[VLC] or MAC_Inner_VLAN_Incl[VLC] respectively enables the VLAN tag deletion for an outer or inner tag. When you enable the VLAN deletion, MAC deletes the tag present at the corresponding position. When a packet has only one tag, it is considered as the outer tag. If you enable the inner tag deletion and if there is only one tag in the packet, then the MAC does not delete the tag.
VLAN tag insertion or replacement	MAC_VLAN_Incl[VLC] or MAC_Inner_VLAN_Incl[VLC] respectively enables the VLAN tag insertion or replacement for an outer or inner tag. When you enable the VLAN tag insertion or replacement, the MAC_VLAN_Incl[VLTI] or MAC_Inner_VLAN_Incl[VLTI] determines whether the VLAN tag must be fetched from the register or from the control word.

76.6.1.2 Receive path

Table 624. Double VLAN processing in receive path

Feature	Description
Outer or inner VLAN tag-based filtering	Mac can filter packets through the ERIVLT bit on the basis of the outer or inner VLAN tag.
C-VLAN or S-VLAN tag-based filtering	Mac can filter packets through the ERSVLM bit on the basis of the C-VLAN or S-VLAN type.
Outer and inner VLAN tag stripping	Mac can strip the outer and inner VLAN tags from received frame on the basis of the EVLS and EIVLS bits.
16-bit outer and inner VLAN tag and type in Rx status	Mac can provide the 16-bit outer and inner VLAN tag and type in the Rx status on the basis of the EVLRXS and EIVLRXS bits, respectively.
Disabling or skipping checking of outer VLAN tag type	Mac can disable or skip checking of an outer VLAN tag type to match C-VLAN or S-VLAN on the basis of the DOVLTC bit.

76.6.1.3 Source address and VLAN insertion, replacement, or deletion

This module supports the SA, and VLAN insertion, replacement and deletion feature. The SA and VLAN fields are Tx packet-related control information that interfaces provide as part of the control word. IP supports the feature to insert or replace the source address on the basis of the information in the MAC address registers, and it also supports the feature to insert, replace, or delete the VLAN fields (VLAN type and VLAN tag) on the basis of the setting of the MAC_VLAN_Incl[VLTI]. SA insertion or replacement feature is enabled for all the transmit packets or selective packets. Similarly, VLAN insertion, replacement, or deletion feature is enabled for all the Tx packets or selective packets.

76.6.1.4 Programming source address insertion or replacement

You can use the SA insertion or replacement feature to instruct MAC to perform the following actions for Tx packets:

- Insert the content of the MAC address registers in the SA field
- Replace the content of the SA field with the content of the MAC address registers

When SA insertion is enabled, the application must ensure that the packets sent to MAC do not have the SA field. The MAC does not check whether the SA field is present in the transmit packet and it inserts the content of MAC address registers in the SA field. Similarly, when the SA replacement is enabled, the application must ensure that the SA field is present in the packets sent to the MAC.

MAC replaces the six bytes following the DA field in the transmit packet with the content of the MAC address registers.

- Configure MAC_Configuration[SARC] to enable the SA insertion or replacement for all Tx packets.
- Enable the SA insertion for selective packet by configuring the SA insertion control field (bits [25:23] of TDES3) in the first transmit descriptor of the packet. When you write 1 to bit 25 of TDES3, the SA insertion control field indicates insertion or replacement by MAC Address1 registers. When bit 25 of TDES3 resets, it indicates insertion or replacement by MAC Address 0 registers.

If you do not enable the MAC Address1 registers, then the MAC Address0 registers are used for insertion or replacement irrespective of the value of the most-significant bit of the SA insertion control field.

76.6.1.5 Programming VLAN insertion, replacement, or deletion

You can use the VLAN insertion, replacement, or deletion feature to instruct MAC to perform the following actions for the Tx packets:

- Delete the VLAN type and VLAN tag fields.
- Insert or replace the VLAN type and VLAN tag fields.

Insertion or replacement is done on the basis of the setting of the MAC_VLAN_Incl[VLTI] as described in theTable 625:

Table 625. VLAN insertion or replacement based on VLTI bit

Condition	Description
VLTI bit=1	MAC inserts or replaces the following: • VLAN type field (C-VLAN or S-VLAN as indicated by the MAC_VLAN_Incl[CSVL]) • VLAN tag field depending on the content of the VT field of transmit context descriptor of the packet
VLTI bit resets	MAC inserts or replaces the following: • VLAN type field (C-VLAN or S-VLAN as indicated by the MAC_VLAN_Incl[CSVL]) • VLAN tag field with the MAC_VLAN_Incl[VLT]

When the VLAN replacement or deletion is enabled, MAC checks whether the VLAN type field (0x8100 or 0x88a8) is present after the DA and SA fields in the transmit packet. The replace or delete operation does not occur if the VLAN type field is not detected in two bytes following the DA and SA fields. However, when the VLAN insertion is enabled, MAC does not check the presence of VLAN type field in the transmit packet and just inserts the VLAN type and VLAN tag fields.

Configuring MAC_VLAN_Incl[VLC] and MAC_VLAN_Incl[VLP] enables the VLAN insertion, replacement, or deletion feature for all Tx packets.

Configuring the VTIR field of TDES2 normal descriptor enables the VLAN insertion, replacement, or deletion for selective packets.

In addition, the VLP (VLAN Priority control) field must reset in MAC_VLAN_Incl (for outer VLAN) and MAC_Inner_VLAN_Incl register (in inner VLAN) for MAC to take the control inputs from the host, depending on the configuration.

76.7 Packet Filtering

This section and all its sub-sections are Synopsys proprietary. Used with permission.

MAC receiver contains various packet filtering scheme. Figure 453 shows the filtering sequence in their precedence order of received packet.

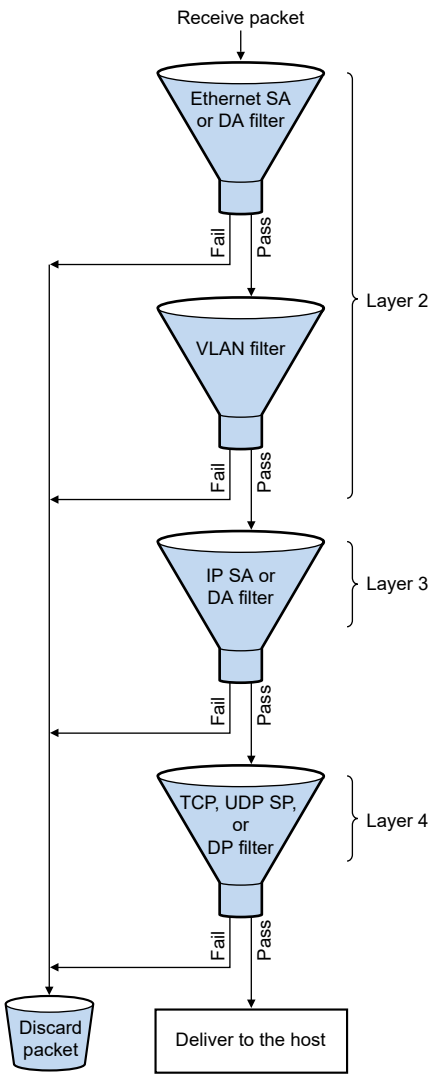


Figure 453. Packet filtering sequence

If you do not enable any of the layer filters, then that filter is bypassed and the subsequent filter is applied. Discard the packet that fails any of the filters. However, you can forward the discarded packet to the host on the basis of the register control.

76.7.1 Source address or destination address filtering

The address filtering module of MAC checks the SA and DA fields of each incoming packet. Programming of different types of address filtering is described in the sections that follows.

76.7.1.1 Unicast destination address filtering

MAC supports up to three MAC addresses for unicast perfect filtering. If perfect filtering is selected (MAC_Packet_Filter[HUC] resets), MAC compares all 48 bits of received unicast address with the programmed MAC address for any match. Default MacAddr0 is always enabled. MAC selects the MacAddr1 to MacAddr2 addresses with an individual enable field. Each byte of MacAddr1 to MacAddr2 can be masked when you compare them with corresponding received DA byte by writing 1 to the corresponding Mask byte control field in the register. This enables group address filtering for the DA.

In hash filtering mode (when you write 1 to HUC bit), MAC performs imperfect filtering by using a 64-bit hash table for unicast addresses . For hash filtering, MAC uses the upper 6 bits CRC of the received destination address to index the content of the hash table. A value of 00000 selects bit 0 of selected register, and a value of 11111 selects bit 63 of hash table register. If the value of

the corresponding bit (indicated by the 6-bit CRC) is 1, the unicast packet is considered to have passed the hash filter, otherwise, the packet is considered to have failed the hash filter.

76.7.1.2 Multicast destination address filtering

Enable MAC_Packet_Filter[PM] to pass all the multicast packets or disable to do multicast addresses filtering on the basis of MAC_Packet_Filter[HMC]. Compare the multicast address with the configured MAC destination address registers (1–2), also supports the group address filtering.

In Hash filtering mode, MAC performs imperfect filtering using a 64-bit hash table. MAC uses the upper 6-bits CRC of received multicast address to index the content of the hash table. A value of 000000 selects bit 0 of selected register and a value of 111111 selects bit 63 of the hash table register. The multicast packet is considered to have passed the hash filter if the value of the corresponding bit is equal to 1. Otherwise, the packet is considered to have failed the hash filter.

76.7.1.3 Hash or perfect address filtering

Configure MAC_Packet_Filter[HPF] to enable or disable the destination address filtering either by hash filter or perfect filter. Also configure MAC_Packet_Filter[HUC] and MAC_Packet_Filter[HMC] to select the hash filtering or perfect filtering for unicast or multicast packet.

76.7.1.4 Broadcast address filtering

MAC does not filter any broadcast packets by default. You must write 1 to MAC_Packet_Filter[DBF] to reject the broadcast packets.

76.7.1.5 Unicast source address filtering

MAC by default compares the source address field with the value configured in the source address registers. Configure bit 30 of MAC address register [1-2] to use it for source address instead of destination address comparison. MAC also supports group filtering with SA. You can filter a group of addresses by masking one or more bytes of the address. MAC drop the packets that fail the SA filter if you write 1 to MAC_Packet_Filter[SAF] field. Otherwise, the result of the SA filter is given as a status bit in the receive status word. When you write 1 to the SAF field, the SA filter and DA filter results ends to decide whether you can forward the packets. This means that the packet is dropped if either filter fails. The packet is forwarded to the application only if the packet passes both filters in-order.

76.7.1.6 Inverse filtering

For DA and SA filtering, you can invert the filter-match result at the final output by writing 1 to the DAIF and SAIF fields of MAC_Packet_Filter register. The DAIF field is applicable for both the unicast and multicast DA packets. The unicast or multicast destination address filter result is inverted in this mode. Similarly, writing 1 to the SAIF field reverses the result of unicast SA filter.

Table 626 and Table 627 summarize the DA and SA filtering on the basis of the type of packets received.

Table 626. Destination address filtering

Packet type	PR	HPF	HUC	DAIF	HMC	PM	DBF	DA filter operation
Broadcast	1	X	X	X	X	X	X	Pass
	0	X	X	X	X	X	0	Pass
	0	X	X	X	X	X	1	Fail
Unicast	1	X	X	X	X	X	X	All packets pass
	0	X	0	0	X	X	X	Pass on perfect or group filter match

Table continues on the next page...

Table 626. Destination address filtering (continued)

	0	X	0	1	X	X	X	Fail on perfect or group filter match
	0	0	1	0	X	X	X	Pass on hash filter match
	0	0	1	1	X	X	X	Fail on hash filter match
	0	1	1	0	X	X	X	Pass on hash or perfect or group filter match
	0	1	1	1	X	X	X	Fail on hash or perfect or group filter match
Multicast	1	X	X	X	X	X	X	Pass all packets
	X	X	X	X	X	1	X	Pass all packets
	0	X	X	0	0	0	X	Pass on perfect or group filter match and drop pause packets if PCF = 0x
	0	0	X	0	1	0	X	Pass on hash filter match and drop pause packets if PCF = 0x
	0	1	X	0	1	0	X	Pass on hash or perfect or group filter match and drop pause packets if PCF = 0x
	0	X	X	1	0	0	X	Fail on perfect or group filter match and drop pause packets if PCF = 0x
	0	0	X	1	1	0	X	Fail on hash filter match and drop pause packets if PCF = 0x
	0	1	X	1	1	0	X	Fail on hash or perfect/group filter match and drop pause packets if PCF = 0x

Table 627. Source address filtering

Packet type	PR	SAIF	SAF	SA filter operation
Unicast	1	X	X	Pass all packets.
	0	0	0	Pass status on perfect or group filter match but do not drop packets that fail
	0	1	0	Fail status on perfect or group filter match but do not drop packet
	0	0	1	Pass status on perfect or group filter match and drop packets that fail
	0	1	1	Fail status on perfect or group filter match and drop packets that fail

NOTE

When you write 1 to the MAC_Packet_Filter[RA] field, all packets are forwarded to the system along with the correct result of the address filtering in the Rx Status.

76.7.2 VLAN filtering

VLAN tag filter can be done either by perfect match or hash table. Depending upon the configuration, MAC can compare either lower 12 bits or all 16 bits of received VLAN tag for perfect match. MAC forward the VLAN tagged packet with match status when it drop all the packets which do not match.

MAC uses 16-bit VLAN hash table for group address filtering on the basis of the VLAN tag. If MAC enables the hash filtering then the most significant 4-bit of CRC-32 of VLAN tag are used to index the content of the MAC_VLAN_Hash_Table. A value of 1 in the VLAN hash table register, corresponding to the index indicates that the VLAN tag of the packet has matched and the packet should be forwarded. A value of 0 indicates that VLAN-tagged packet should be dropped.

MAC supports the inverse matching for VLAN packets. In the Inverse matching mode, the packet must be dropped when the VLAN tag of a packet matches the perfect or hash filter. If MAC enables the VLAN perfect and VLAN hash match, then a packet is considered as matched if either the VLAN hash or the VLAN perfect filter matches. When inverse match is set, MAC forwards a packet only when both the perfect and hash filters indicate mismatch.

Table 628 specifies the different possibilities for VLAN matching and the final VLAN match status. When you write 1 to MAC_Packet_Filter[RA] all packets are received and it indicates the VLAN match status in the VF field of RDES2 normal descriptor (write-back format). When the RA field is not set and when you write 1 to MAC_Packet_Filter[VTFE] the packet drops if the final VLAN match status fails

When VLAN VID programs to 0 in the VL field of MAC_VLAN_Tag register, consider that all the VLAN-tagged packets are perfectly matched but the status of the VLAN hash match depends on the VTHM and VTIM fields in MAC_VLAN_Tag register.

Table 628. VLAN match status

VID	VLAN perfect filter match result	VTHM bit	VLAN hash filter match result	VTIM bit	Final VLAN match status
VID = 0	Pass	0	X	X	Pass
	Pass	1	X	0	Pass
	Pass	1	Fail	1	Pass
	Pass	1	Pass	1	Fail
VID1= 0	Pass	X	X	0	Pass
	Fail	0	X	0	Fail
	Fail	1	Fail	0	Fail
	Fail	1	Pass	0	Pass
	Fail	0	X	1	Pass
	Pass	X	X	1	Fail
	Fail	1	Pass	1	Fail
	Fail	1	Fail	1	Pass

In Table 628, X represents any value.

76.7.3 VLAN filter fail packets queue

When you enable VLAN filtering, route the VLAN filter fail packets to a programmable queue (VFFQ) when the value of MAC_Packet_Filter[RA] = 1 or MAC_Packet_Filter[VTFE] = 0 and write 1 to MAC_RxQ_Ctrl4[VFFQE] for the queue.

Route the packets that passes the VLAN filtering to Rx queues on the basis of the VLAN tag priority field by configuring the PSRQ field in the corresponding MAC_RxQ_Ctrl2 and MAC_RxQ_Ctrl3 registers. Discard the packets that fail the VLAN filter if RA=0 or VTFE=1. However, when RA=1 or VTFE=0, forward the VLAN filter fail packets to the application. In such case, when you write 1 to the VLAN filter fail queue enable (VFFQE) field, the VLAN filter fail packets forward to the Rx queue number programmed in the VFFQ field. If the value of VFFQE=0, the VLAN priority mapping determines the Rx queue number per the PSRQ fields.

Table 629 shows the Rx queue routing table for unicast tagged packets, with DA or SA filter enabled.

Table 629. Rx queue routing table for unicast tagged packets

RA	VFTE	SA or DA filter result	VLAN filter result	VFFQE	Queue routing
X	X	Pass	Pass	X	PSRQ
0	0	Pass	Fail	0	PSRQ
0	0	Pass	Fail	1	VFFQ
0	X	Fail	X	X	Dropped
0	1	Pass	Fail	X	Dropped
1	X	Fail	X	0	UFFQ*/PSRQ
1	X	Fail	X	1	UFFQ*/VFFQ
1	X	Pass	Fail	0	PSRQ
1	X	Pass	Fail	1	VFFQ

X : Don't care condition * : When UFFQE is enabled else PSRQ.

76.7.4 Extended receive VLAN filtering and routing

When the extended Rx VLAN filtering and routing is enabled then both the perfect filtering and hash filtering can be enabled. The overall VLAN filter result is based on the perfect filter result and hash filter result (if enabled). Filter result is passed to application as part of the status bit. Extended routing will take place only if VLAN filter has passed. Routing is only based on perfect filter result. Each perfect filter has a DMA channel enable and DMA channel number filed which must be programmed for routing.

See the extended VLAN based DMA selection in Dynamic (per packet) mapping for more information about routing.

76.7.4.1 Comparison mode

Application has these comparison option for each VLAN tag filter:

- Programs MAC to compare the inner or outer VLAN tag either with 12 bits or 16 bits programmed VID.
- Selects whether the VID comparison is for SVLAN or CVLAN type frames, if type check is enabled for a filter.

76.7.4.2 Filtering

Perfect filtering is done on the basis of the MAC_VLAN_Tag_Filter registers. MAC compares the relevant VLAN tag ID and gives a result for each VLAN tag filter.

The results for each VLAN tag filter are:

- Pass- If any one of the VLAN tag filters gives a match.
- Fail- If the frame mismatches all the filters.

This behavior is applicable only when the inverse filtering is not enabled in MAC_VLAN_Tag_Ctrl.

If inverse filtering is enabled and the frame mis-matches all the relevant filters then it is considered to have passed the VLAN filter. If the frame matches any one of the relevant filters then it is considered to have failed. The frame is bypassed to the application if none of the enabled filters can perform a comparison or if none of the filters are enabled.

The overall filter result and the programming on MAC_Packet_Filter[VTFE] and MAC_Packet_Filter[RA] determines if the frame will drop or it is forwarded to the application. If the value of RA = 1 or VTFE = 0, then the frame is always forwarded whether the filter result is a pass or fail, . If the value of RA = 0 and VTFE = 1, only then, if the VLAN tag filter result is a pass the MAC forwards the frame. If the frame is forwarded to the application, then the relevant filter result is indicated through the status bits.

76.7.4.3 Filter status

The extended receive VLAN filtering and routing feature provides two status fields to show the comparison result of the VLAN tags.

By default, MAC indicates the VLAN filter status through one bit in the status VF field in RDES2. When you enable the extended RX VLAN filtering and routing, two status fields will show the comparison result of the VLAN tags. The outer VLAN tag filter pass and inner VLAN tag filter pass bits are defined in the following positions. The status indicated through these fields depends on the programming as described below.

In RDES2:

- Bit 15 – Outer VLAN tag filter status
- Bit 14 – Inner VLAN tag filter status

In ARI status: MAC filter status:

- Bit 15 – Outer VLAN tag filter status
- Bit 14 – Inner VLAN tag filter status

In MRI status:

- Bit 47 – Outer VLAN tag filter status
- Bit 46 – Inner VLAN tag filter status

Outer VLAN tag filter status (OTS)

- In perfect filtering, without enabling inverse filtering, if you write 1 to this field, it indicates that the frame's outer VLAN tag has matched one of the VLAN tag filters.
- If this field resets, it indicates that the frame's outer VLAN tag has either failed the relevant outer VLAN tag filters or bypassed them.
- This field resets if none of the filters are enabled for outer VLAN tag comparison.
- If inverse filtering is enabled and if you write 1 to this field, then the frame's VLAN tag has passed all the relevant VLAN tag filters. If it is resets, then it has failed at least one filter or bypassed all the filters programmed for outer VLAN tag comparison.
- This bit is valid for both single and double VLAN tagged frames.

Inner VLAN tag filter status (ITS)

- In perfect Filtering, without enabling inverse filtering, if you write 1 to this field, it indicates that the frame's inner VLAN tag has matched one of the VLAN tag filters.
- If this field resets, it indicates that the frame's inner VLAN tag has either failed the relevant inner VLAN tag filters or bypassed them.
- This field resets if none of the filters are enabled for inner VLAN tag comparison.
- If Inverse Filtering is enabled and if you write 1 to this field, then the frame's VLAN tag has passed all the relevant VLAN tag filters. If it resets, then it has failed at least one filter or bypassed all the filters programmed for inner VLAN tag comparison.

- This bit is valid for only double VLAN tagged frames, when double VLAN processing is enabled.

The application must observe the status fields and the program to determine if the frame has passed or failed the VLAN filter.

Table 630 and Table 631 describes the possible filter combinations and the corresponding filter results and they also explain the scenarios when the double VLAN processing and the hash VLAN filter are enabled in the design.

Legend for the table OTS and ITS Bit Values with at least 1 perfect filter enabled

- VTIM: VLAN tag inverse match enable – bit 17 in VLAN_Tag_Ctrl register.
- HFO: Hash filter enabled for outer VLAN tag comparison - bit 25 and bit 27 in VLAN_Tag_Ctrl register.
- HFI: Hash filter enabled for inner VLAN tag comparison – bit 25 and bit 27 in VLAN_Tag_Ctrl register.
- PFO – Perfect filter comparison enabled for outer VLAN tag - Any of the MAC_VLAN_Tag_Filter registers is enabled (write 1 to bit 16) and programmed for outer VLAN tag comparison (write 0 to bit 20).
- PFI – Perfect filter comparison enabled for Inner VLAN Tag - Any of the MAC_VLAN_Tag_Filter registers is enabled (write 1 to bit 16) and programmed for inner VLAN tag comparison (write 1 to bit 20).
- OTS – Outer VLAN tag filter status
- ITS – Inner VLAN tag filter status

Table 630 shows the possible values of status bits (OTS and ITS) when at least one perfect filter is enabled.

Table 630. OTS and ITS bit values with at least 1 perfect filter enabled

VTIM	HFO	HFI	PFO	PFI	OTS	ITS
0	0	0	0	1	0	1/0
0	0	0	1	0	1/0	0
0	0	0	1	1	1/0	1/0
0	1	0	1	1	1/0	1/0
0	1	0	1	0	1/0	0
0	1	0	0	1	1/0	1/0
0	0	1	1	1	1/0	1/0
0	0	1	1	0	1/0	1/0
0	0	1	0	1	0	1/0
1	0	0	0	1	0	1/0
1	0	0	1	0	1/0	0
1	0	0	1	1	1/0	1/0
1	1	0	1	1	1/0	1/0
1	1	0	1	0	1/0	0
1	1	0	0	1	1/0	1/0

Table continues on the next page...

Table 630. OTS and ITS bit values with at least 1 perfect filter enabled (continued)

VTIM	HFO	HFI	PFO	PFI	OTS	ITS
1	0	1	1	1	1/0	1/0
1	0	1	1	0	1/0	1/0
1	0	1	0	1	0	1/0

Table 631 shows the possible values of status bits (OTS and ITS) when no perfect filters are enabled except the VLAN hash filter is enabled.

Table 631. OTS and ITS bit values with only VLAN hash filter enabled

VTIM	HFO	HFI	OTS	ITS
0	0	0	0	0
0	1	0	1/0	0
0	0	1	1/0	1/0
1	0	0	1/0	0
1	1	0	1/0	0
1	0	1	1/0	1/0

With no perfect filters enabled, any VLAN packet is considered to have bypassed the perfect filter. If Hash Filter is enabled for one of the Tags, then the respective Status bit depends on the Filter's result. The Status bits are set to 0 if VLAN Hash Filter is also not enabled.

If the value of ITS/OTS is shown as 1/0; then it indicates that the final result is dependent on the enabled relevant filter's result.

Example 1: The second row of OTS and ITS bit values with only VLAN hash filter enabled indicates that at least one Perfect Filter is enabled for Outer VLAN tag comparison and none of the filters are enabled for Inner VLAN tag comparison. Inverse VLAN Filtering is not enabled. The bit OTS is given as 1/0. If the received frame passes atleast one of the enabled Outer VLAN Tag filters then the bit is set to 1. If the frame doesn't pass any of the enabled Outer VLAN Tag filters, then the bit is set to 0.

Example 2: The last row of table "OTS and ITS bit values with only VLAN hash filter enabled" indicates that the inverse filtering is enabled, hash filter and at least one perfect filter is enabled for Inner VLAN Tag comparison, then if the received frame's Inner VLAN tag mismatches with both the Hash Filter and all the enabled Perfect filters, then the frame will have the ITS bit set to 1 else it is set to 0. OTS will be set to 0 as no comparison is performed.

76.7.4.4 Stripping

Each VLAN tags has individual control over stripping. The programming options of always strip, never strip, strip on pass and strip on fail are available. Inner or outer VLAN tag stripping is based on the pass or fail results of the individual tag. If all the relevant filters bypasses a tag, stripping is not applicable for the tag.

- If strip on pass is enabled for the outer VLAN tag, then the stripping occurs only if the outer VLAN tag has passed the relevant filters. The outer VLAN tag filter result field will be 1.
- If strip on fail is enabled for the outer VLAN tag, then the stripping occurs only if the outer VLAN tag has failed relevant filters. The outer VLAN tag filter result field will reset.
- If the outer VLAN tag of the received frame is bypassed by the entire filter (no comparison has been made), then the tag is not stripped, though the status bit is 0.

- As multiple filters are enabled, it is possible that the received VLAN frame could have matched any one or more of the filters. The VLAN Tag's value is not always deterministic from the filter status bits.

If an application strips the VLAN tag on the basis of the filter result, it might lose the VID. You can use it, if stripping is enabled for any of the tags, place the tag in the status. For this the software or application will enable the respective VLAN tag in status bit - 24 or 31 in the MAC VLAN tag control register.

See section Programming guidelines for extended VLAN filtering and routing on receive for more information.

76.7.5 Layer 3 and layer 4 filtering

IP supports the layer 3 based packet filtering which is an IP source or destination address filtering in the IPv4 or IPv6 packets, and layer 4 based packet filtering which is a source or destination port number filtering in TCP or UDP. When you enable layer 3 and layer 4 packet filtering then packets are filtered in the following manner.

- Matched packets: MAC forward the packets that match all enabled fields to the application along with the status. It gives the matched field status only if MAC_Configuration[IPC] = 1 and if one of the following conditions is true:
  - All enabled layer 3 and layer 4 fields match
  - At least one enabled field matches and other fields are bypassed or disabled

When you enable multiple layer 3 and layer 4 filters, any filter match is considered as a match. If more than one filter matches, MAC provides the status of the lowest filter where filter 0 is the lowest filter and filter 3 is the highest filter.

- Unmatched packets: MAC drop the packets that do not match any enabled fields. You can use the inverse match feature to block or drop a packet with specific TCP or UDP over IP fields and forward all other packets.
- Non-TCP or UDP IP packets: By default, all the non-TCP or UDP IP packets bypasses the layer 3 and layer 4 filters. You can optionally program MAC to drop all the non-TCP or UDP packets over IP packets.
- IP has a control register MAC_L3_L4_Control to control the layer L3 and L4 packet filtering along with address registers to program the layer L3 and L4 fields to be matched.

76.7.6 Flexible receive parser

When you enable this function, all incoming packets are parsed per the programmable instruction stored in memory. It perform these functions:

- Packet filter (Accept or reject)
- DMA channel selection

Flexible receive parser block operates over the first 128 bytes data of receive packet on the basis of 64 bytes of instruction on each packet. Figure 454 shows the functional block diagram of flexible receive parser.

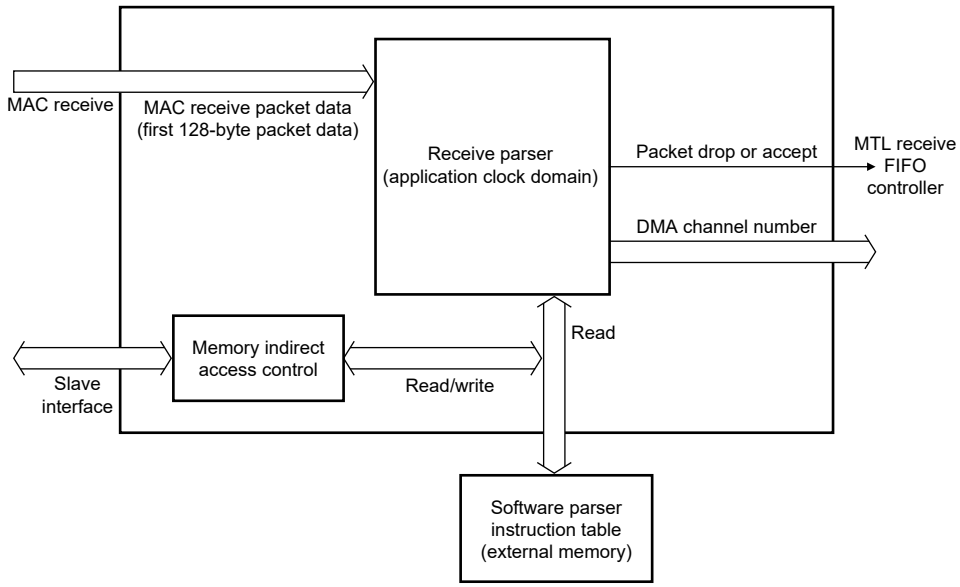


Figure 454. Functional block diagram of flexible receive parser

76.7.6.1 Instruction table format of the flexible receive parser

You must build the 96-bit wide instruction before enabling the receive parser feature and program each instruction as an entry in the instruction memory.

Table 632 shows the format of each entry in the instruction table .

Table 632. Rx parser instruction entry format

Field	Field Name	Description
31:0	MATCH_DATA	This is a 4-byte data. You can use it to compare with incoming packet data which starts at the frame offset as defined in [79:72] of the entry. The comparison is done only on those bits whose corresponding mask bits are 1 (MATCH_EN bits of [63:32]). See the notes (below this table) on the byte order relative to Ethernet arrival data and endianness which the QoS IP supports on the slave interface. The programming of this instruction table must adhere to the same.
62:32	MATCH_EN	When the MATCH_EN = 1, you can use the corresponding packet data bit for comparing. Otherwise, corresponding data bit is don’t care.
64	AF	Accept frame
65	RF	Reject frame
66	IM	Inverse match
67	NC	Next instruction control
71:68	Reserved	-

Table continues on the next page...

Table 632. Rx parser instruction entry format (continued)

Field	Field Name	Description
79:72	Frame offset	[77:72] indicates the frame offset in terms of 4 bytes. (Here [79:78] always 0) Compare the frame offset in the packet data for Match. This is in terms of 4 bytes. The value of actual frame offset in bytes are 0 01 42 8… ..63 252 The max value programmed in this instruction table is 128 B. The 'Actual frame offset' must not cross this limit.
87:80	OK index	If (NIC==0) the memory index you can use next when ENTRY_MATCH==1 and neither AF or RF = 1. If (NIC==1) the memory index you can use next when ENTRY_MATCH==0
95:88	DMA Ch No	Indicates the DMA channel number (1-bit for each). You can use this when ENTRY_MATCH ==1 and AF = 1 (Accept frame). The following bits give the encoding: bit[0]- DMA channel number o, bit[1]- DMA channel number , 1bit[2]- DMA channel number 2… bit[7]- DMA channel number 7. **NOTE** You can encode the DMA channel number using bit wise as QoS IP support multicasting or broadcast across DMA channels. See section 3.3.12 for more information and proper usage.
128:96	Reserved	0 (This field is only for software view and reserved for future enhancements. The memory width remains as 96-bits).

The parser begins parsing from the 0th entry, for each received packet. The subsequent parsing entry location (next entry or OK_INDEX[]) depends on the current entry parser result. Flexible receive parser flow chart shows the detailed flow.

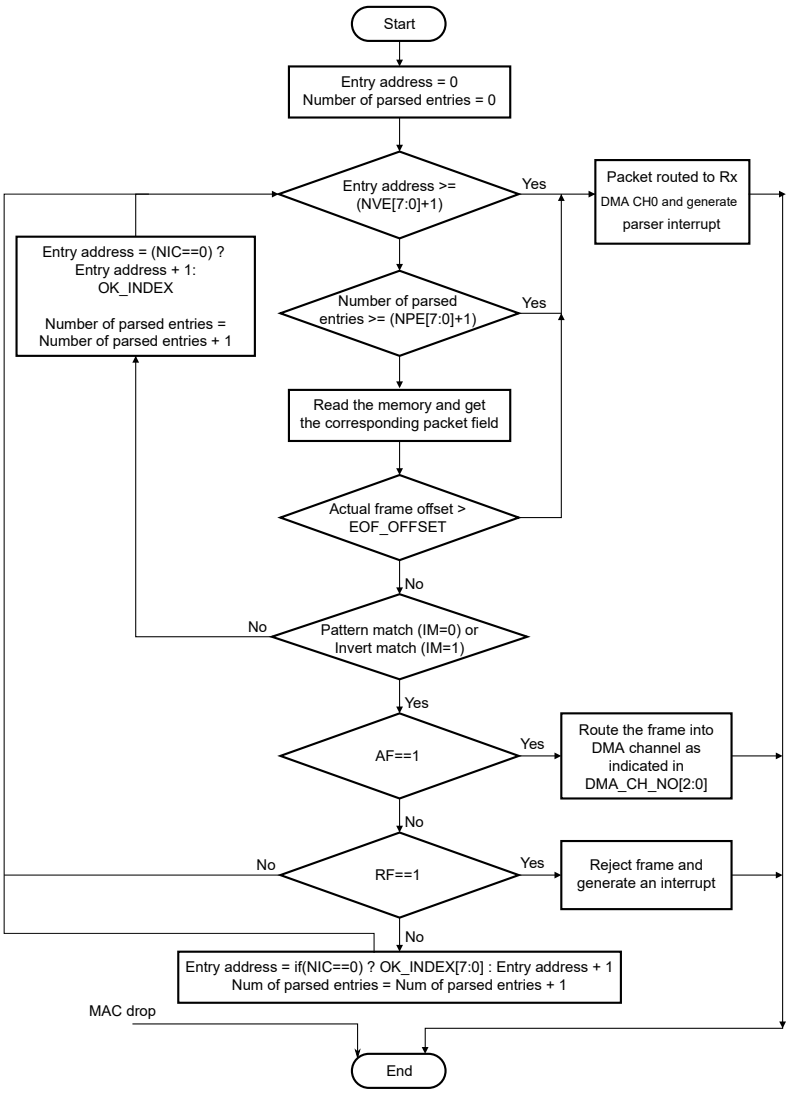


Figure 455. Flexible receive parser flow chart

The EOF_OFFSET shown in the figure is the least among the following:

- Actual packet size
- 128 Byte when frame pre-emption feature is not enabled or configured
- 128 Byte when frame pre-emption feature is enabled and packet is pre-emptible
- 64 Byte when frame pre-emption feature is enabled and packet is express packet

76.7.6.2 Number of valid entries (NVE)

NVE indicates the number of valid entries in the instruction table. The default value is 64. You can change this value by writing into MTL_RXP_Control_Status[NVE].

When parsing, an error is flagged in MTL_RXP_Interrupt_Control_Status[NVEOVIS], if the entry address is greater than MTL_RXP_Control_Status[NVE] + 1. IP generates an interrupt status (MTL_Interrupt_Status[MTLPIS]).

76.7.6.3 Number of parsable entries (NPE)

NPE indicates the number of entries that you can parse on an incoming packet. The default value is 64. Write into MTL_RXP_Control_Status[NPE] to change this value.

When parsing, an error is flagged in MTL_RXP_Interrupt_Control_Status[NPEOVIS], if the number of parsed of parsed entries is more than MTL_RXP_Control_Status[NPE] + 1. IP generates an interrupt status (MTL_Interrupt_Status[MTLPIS]).

NVE[] indicates the number of valid entries in the receive parser memory which you have built and NPE[] indicates the worstcase parsable entries considering the various possible paths that the parser can take. (NVE[] >= NPE[]).

76.7.6.4 Frame offset

It indicates the frame offset, in terms of 4 bytes which you use when comparing against the current table entry.

In an entry, an error is flagged in MTL_RXP_Interrupt_Control_Status[FOOVIS] in any of the following cases:

When frame offset is more than:

- Packet size
- 128B if pre-emption is disabled
- 128B if pre-emption is enabled and packet is pre-empitble
- 64B if pre-emption is enabled and packet is an express packet

IP generates an interrupt status (MTL_Interrupt_Status[MTLPIS]).

NOTE

If the frame ends after the (frame_offset *4) bytes but before ((frame_offset *4)+4) bytes, the receive parser declares a frame offset error if you enable comparison (via MASK bits) for the non-received bytes; from (frame_offset *4) to ((frame_offset *4)+4).

76.7.6.5 Frame reject

You can drop the frame in cases, where an entry matches (considering the inverse match, IM bit in the instruction table) and frame accept bit is not 1, and frame reject bit (RF) = 1.

When a frame drops because RF = 1, IP generates an interrupt status (MTL_Interrupt_Status[MTLPIS]).

76.7.6.6 Out of order processing

The Rx parser flow can be out of order. The decision tree is not based on the order in which packet arrives. This implies that the next entry and next frame offset (OK_INDEX[] and FRAME_OFFSET[]) can be less than the current entry address and current frame offset.

76.7.6.7 DMA channel selection

When you enables the receive parser (MTL_Operation_Mode[FRPE] = 1) it overrides the MAC DMA selection criteria.

When you disables the receive parser (MTL_Operation_Mode[FRPE] = 0) the MAC/MTL determines the DMA channel number.

76.7.6.8 Receive queue selection

The existing MAC functionality determines the receive queue, irrespective of whether the receive parser is enabled or disabled. This is because the receive parser is mainly designed to select the DMA channel and the MAC decides the receive queuing to enable the proper functioning of the pause or PFC feature.

NOTE

When there are multiple queues and multiple DMA Channels are enabled, a particular DMA channel might receive out of order packets because queue selection is based on MAC criteria (VLAN Priority and other criteria) and Rx DMA channel selection is based on receive parser.

76.7.6.9 MAC packet filtering/drop/error handling

Packet filtering is done on the basis of the received packet fields. You must disable the packet filtering features inside the MAC when the receive parser is enabled. Follow these steps to disable the packet filtering features inside the MAC :

1. Write 1 to the Promiscuous mode (MAC_Packet_Filter[PR]).
2. Write 1 to all other fields in MAC_Packet_Filter to its default values.

When MAC decides to drop the packet due to the receive MAC dependent error such as:

- GMII error
- Receive watchdog error
- CRC error
- Giant frame error

the packet drops irrespective of the receive parser decision.

76.7.6.10 Pad strip or CRC strip handling

Software controls the pad strip and CRC strip and it is applicable to all the received packets. Also,software must build the receive parser instructions accordingly, when you enable these features.

NOTE

Enable pad and CRC stripping in real use case.

76.7.6.11 VLAN strip handling

MAC supports stripping the outer VLAN as well as inner VLAN. When you program MAC to strip the VLAN, the receive parser considers those VLAN tags to be part of the incoming packet.

76.7.6.12 Multicast and broadcast support

IP supports multicast and broadcast packet and the packet route to the highest queue number.

When you enables the receive parser, the DMA channel numbers (DMA CH NO, Bits [95:88]) in the instruction table, decide the packet routing.

When you disables the receive parser, DCS (DMA Channel Select) field of the MAC_AddressX_High register determines the packet routing. See the Broadcast/multicast packet duplication.

NOTE

You can select multiple DMA channels for the packet routing because the DCS/DMA CH NO field is per the DMA channel control.

76.7.6.13 Pre-emption support

IP supports the frame pre-emption feature when the flexible receive parser feature is enabled. It has the following limitations:

- Number of bytes would be 64, so that the receive parser can operate on an express traffic
- Number of bytes would be 128, so that receive parser can operate on a pre-emptible traffic.

76.7.6.14 Software access to the flexible receive parser memory

Software can read and write into the receive parser memory. It uses the following registers via indirect addressing.

- MTL_RXP_Indirect_Acc_Control_Status
- MTL_RXP_Indirect_Acc_Data

For more details, see GMAC register descriptions.

76.7.6.15 Statistical counters

IP supports these statistics counters for flexible receive parser.

1. MTL_RXP_Drop_Cnt
2. DMA_RXP_Error_Cnt

DMA_CH(#i)_Rxp_Accept_cnt (i=0; i<2)

For more details, see GMAC register descriptions.

76.7.6.16 Changing the instruction table by software

Software can update the instruction table in the memory in the following ways:

- Disables the MAC receiver and receive parser by writing 0 to MAC_Configuration[RE] and MTL_Operation_Mode[FRPE]. It waits for receive parser to become inactive (RXPIA, MTL_RXP_Control_Status register bit[31]).
- (Optional) Programs in the entry address 0 to unconditionally (all MATCH_EN bit 0) skip to OK_INDEX[] (to certain location in the memory), where you can program it to reject or accept all the packets.

76.7.6.17 Receive cut-through functionality

In the Cut-Through mode, the receive controller transfer the received packets to DMA just after it receives the cut-through threshold amount of packet data (RTC field in the MTL_RxQx_Operation_Mode register).

However due to Rx parser's result worst case arrival time can be more than the programmed cut-through threshold. So, the receive controller delays the packet transfer to DMA accordingly, and the cut-through functionality is re-defined as follows:

The receive controller transfers packet to DMA after at least RTC number of bytes have been received and receive parser results are available.

76.7.6.18 Receive packet drop indication

In most cases, the packet drops internally in the receive queue. However, back-to-back packets can arrive for the same queue and the first packet's receive parser result is not available when the second packet arrives. In such cases, the packet cannot be flushed internally and forwarded to DMA or the application interface and indicates the status accordingly. See RDES2, bit 16(RXPD) in Receive normal descriptor (write-back format) for more information.

76.7.6.19 Runt packet handling

The receive parser can performs the operation with 64B minimum size packet. If the runt packet is received (MAC must indicate it), the receive parser assumes that it is dropped in the MAC, this is because, the back-to-back runt packets (< 64B) cannot be handled in the receive parser.

76.7.6.20 Uncorrectable ECC error handling

The packet does not drops when the parser detects an uncorrectable ECC error while parsing the parser memory. The packet is sent to the application with an error indication. The RXPI (RX parser incomplete) sets in the packet status along with error summary (ES) bit.

See the Receive normal descriptor (write-back format) for more information.

76.8 IEEE 1588 timestamp support

This section and all its sub-sections are Synopsys proprietary. Used with permission.

IP supports the IEEE 1588 precision time protocol (PTP). It is also able to do precise time stamping and captures incoming and outgoing frame because of the PTP protocol implementation in the IP. It supports all the clock types defined in IEEE 1588-2008. The typical frequency of PTP timestamp clock is 125 MHz.

IP supports these features:

- Provides an option to take snapshot of all packets or only PTP type packets
- Provides an option to take snapshot of only event messages
- Identifies the PTP message type, version, and PTP payload in packets sent directly over Ethernet and sends the status
- Provides an option to measure sub-second time in digital or binary format

76.8.1 Delay request-response mechanism

The system or network is classified into the master and slave nodes for distributing the timing and clock information. Figure 456 shows the process that PTP uses for synchronizing a slave node to a master node by exchanging PTP messages.

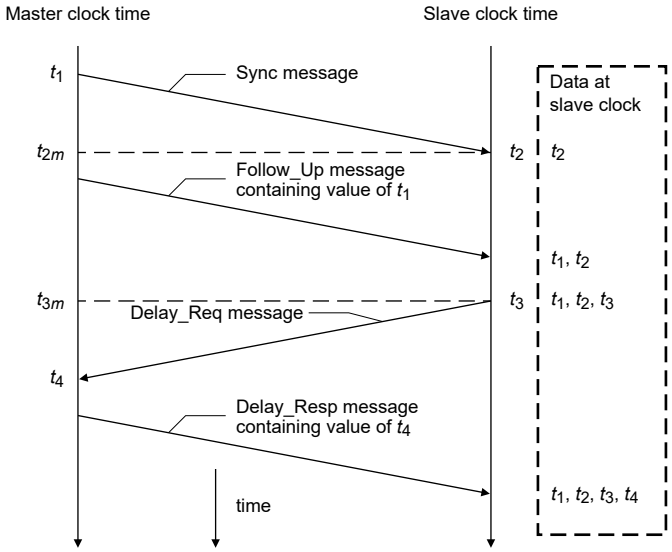


Figure 456. Networked time synchronization

As shown in the figure, PTP uses this process:

1. The master broadcast the PTP sync messages to all its nodes. The sync message contains the reference time information of the master. This message leaves the system of the master at t1. You must capture this time for Ethernet ports at GMII or MII.
2. The slave receives the sync message and it also captures the exact time, t2, using its timing reference.
3. The master sends a Follow_Up message to the slave, which contains t1 information for later use.
4. The slave sends a Delay_Req message to the master and note the exact time, t3, at which this packet leaves the GMII or MII interface.
5. The master receives the message, capturing the exact time t4, at which the message enters its system.
6. The master sends the t4 information to the slave in the Delay_Resp message.
7. The slave use the four values of t1, t2, t3, and t4 to synchronize its local timing reference to the master's timing reference.

Most of the PTP implementation is done in the software above the Ethernet layer. However, the hardware captures the exact time when specific PTP packets enter into or leave the Ethernet port at the MII interface. This timing information must be captured and returned to the software for proper implementation of PTP with high accuracy.

76.8.2 Peer-to-peer PTP transparent clock (P2P TC) message support

The IEEE 1588-2008 supports the peer-to-peer PTP (Pdelay) message in addition to the sync, delay request, follow-up, and delay response messages. Figure 457 shows the method to calculate the propagation delay in clocks supporting the peer-to-peer path correction.

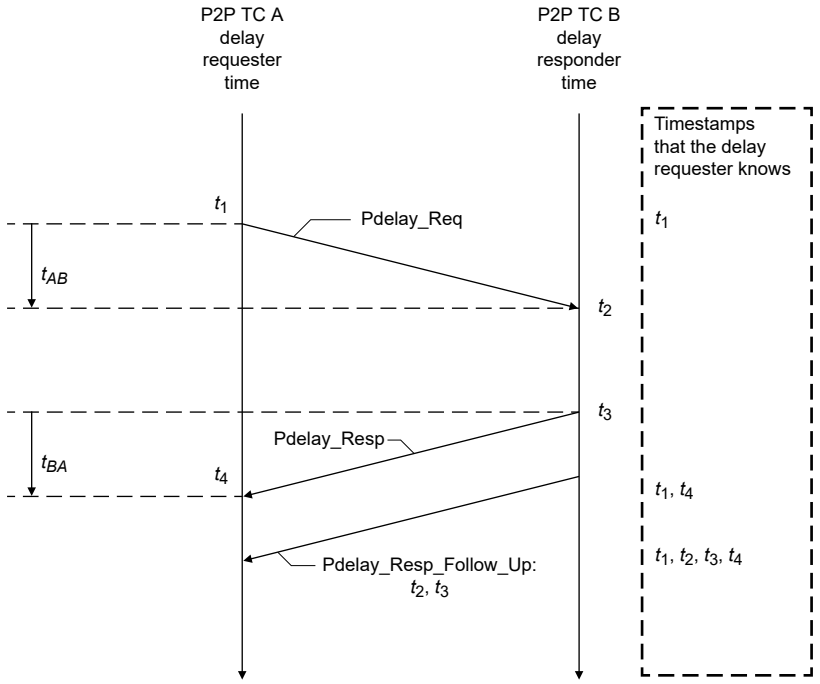


Figure 457. Propagation delay calculation in clocks supporting peer-to-peer path correction

As shown in the above figure, the propagation delay is calculated in the following way:

1. Port 1 issues a Pdelay_Req message and generates a timestamp (t1) for the Pdelay_Req message.
2. Port 2 receives a Pdelay_Req message and generates a timestamp (t2) for this message.
3. Port 2 returns a Pdelay_Resp message and generates a timestamp (t3) for this message. Port 2 returns the Pdelay_Resp message as quickly as possible after the receipt of the Pdelay_Req message, to minimize the errors because of any frequency offset between the two ports. Port 2 returns any one of the following:
   - Difference between the timestamps t2 and t3 in the Pdelay_Resp message
   - Difference between the timestamps t2 and t3 in the Pdelay_Resp_Follow_Up message
   - Timestamps t2 and t3 in the Pdelay_Resp and Pdelay_Resp_Follow_Up messages, respectively
4. Port 1 generates a timestamp (t4) after receiving the Pdelay_Resp message.
5. Port 1 uses all the four timestamps t1, t2, t3, and t4 to compute the mean link delay.

76.8.3 Timestamp correction

According to the IEEE 1588 specification, you must capture a timestamp when the PTP message timestamp point (leading edge of the first bit of the octet immediately following the start frame delimiter octet) crosses the boundary between the node and the network. As MAC takes the timestamp at an internal point far from the actual boundary of the node and network, you must correct or update the captured timestamp for the ingress or egress path latency (including the delay in the PHY layers). Further correction

is done for the inaccuracies or errors introduced because the clock (MII Tx, Rx clock) is different at the capture point as compared to the PTP clock (CLK_PTP_REF_I) that generates the time. The resultant CDC (Clock domain crossing) circuits add an error depending on the clock period of the MII and PTP clocks.

76.8.3.1 Ingress correction

The timestamp captured at the internal snapshot point In the receive side is delayed (later in time) as compared to the time at which the packet's SFD bit is received at the port's boundary. Therefore, the ingress latency and the errors in CDC sampling reduces the captured timestamp. You must determine or calculate the correction value and write into the MAC_Timestamp_Ingress_Corr_* registers.

The correction value consists of these three components:

1. External latency in the PHY layer between boundary point and the input of the core.

   If the PHY is compliant with the IEEE 802.3 Clause 45 MMD registers, it has a register which indicates the maximum and minimum ingress latency. You can read these registers and determines the average ingress latency in the PHY. Alternatively (if the PHY does not support these registers), you must determine the ingress latency from its datasheet or timing characteristics.

2. Internal latency from the core's input to the internal capture point.

   You can read the internal ingress latency from MAC_Timestamp_Ingress_Latency. This is a read-only register and provides the latency in scaled nanoseconds format as defined in IEEE 1588 Clause 5.3.2. The latency differs on the basis of the active PHY interface (MII, RMII, so on) and the operating speed. Therefore, you must read this register after any speed change in MAC, to determine the current internal latency.

3. CDC synchronization

   The CDC synchronization error is almost equal to twice the clock-period of the PTP clock (CLK_PTP_REF_I).

You must add the values that these three components determine and must write into the TSIC and TSICSNS fields of the MAC_Timestamp_Ingress_Corr_* registers.

NOTE

The value written into the register must be negative (two's complement as explained below), because it is subtracted from the captured timestamp. The MAC receiver adds the value in this register to the captured timestamp and then gives the resultant value as the timestamp of the received packet.

When MAC_Timestamp_Control[TSCTRLSSR] is 1, it indicates that the nanoseconds field of the captured timestamp is in decimal format with a granularity of 1ns. So the Bit31 of TSIC must be 1 (for negative value) and bits[30:0] must write with "10^9 - total ingress_correction_value[nanosecond part]" represented in binary. For example, if the required correction value is -5 ns, then the value is 0xBB9A_C9FB.

When MAC_Timestamp_Control[TSCTRLSSR] becomes 0, it indicates that the nanoseconds field of the captured timestamp is in binary format with a granularity of ~0.466ns. Therefore, bits[30:0] must write with "2^31 - total ingress_correction_value" represented in binary with bit[31] = 1.

76.8.3.2 Egress correction

The timestamp captured at the internal snapshot point in the transmit side is earlier (advanced in time) as compared to the time at which the packet's SFD bit is output at the port's boundary. Therefore, the egress latency and the errors in CDC sampling must compensate the captured timestamp. You must determine or calculate this correction value and write into the MAC_Timestamp_Egress_Corr_* registers.

The correction value consists of these three components:

1. External latency in the PHY layer between the core output and the port and the network boundary

   If the PHY is compliant with the IEEE 802.3 Clause 45 MMD registers, it has a register which indicates the maximum and minimum egress latency. You can read these registers and determine the average egress latency in the PHY. Alternatively (if the PHY does not support these registers), you must determine the egress latency from its datasheet or timing characteristics.

2. Internal latency from the internal capture point and the core output of the c

   You can read this internal egress latency from MAC_Timestamp_Egress_Latency. This is a read-only register and gives the latency in scaled nanoseconds format as defined in IEEE 1588 Clause 5.3.2. The latency differs on the basis of the active PHY interface (MII, RMII, so on) and the operating speed. Therefore, you must read this register after any speed change in MAC to determine the current internal latency.

3. CDC synchronization error

   The CDC synchronization error value for one-step timestamping = (1 * period of CLK_PTP_REF_I + 4 * period of CLK_TX_I). Otherwise (Two-step timestamping mode), the value = -(2 * period of CLK_PTP_REF_I).

76.8.3.3 Frequency range of reference timing clock

The timestamp information is transferred across asynchronous clock domains that is from the MAC clock domain to the application/system clock domain. Therefore, a minimum delay is required between the two consecutive timestamp captures. This delay is 4 clock cycles of MII and 3 clock cycles of PTP clocks. If the delay between the two captured timestamp is less than this delay, MAC does not take a timestamp snapshot for the second packet.

76.8.3.4 PTP processing and control

Table 633 shows the common message header for the PTP messages. This format is defined in the IEEE 1588-2008.

Table 633. Message format defined in IEEE 1588-2008

Bits								Octet	Offset
7	6	5	4	3	2	1	0		
transportSpecific				messageType				1	0
Reserved				versionPTP				1	1
messageLength								2	2
domainNumber								1	4
Reserved								1	5
flagField								2	6
correctionField								8	8
Reserved								4	16
sourcePortIdentity								10	20
sequenceId								2	30
controlField (*)								1	32
logMessageInterval								1	33

(*) – control Field is used in version 1. In version 2 message type field is used for detecting different message types.

There are some fields in the Ethernet payload that detects the PTP packet type and controls the snapshot to be taken. These fields are specified in Table 634 for PTP packets over Ethernet.

Following table provides the information about the fields that match to control the snapshots for the PTP packets sent over Ethernet for IEEE 1588 version 1 and version 2. The octet positions are offset by 4 for the tagged packets. This is based on the IEEE 1588-2008, Annex D, and the message format.

Table 634. Ethernet PTP packet fields required for control And status

Field matched	Octet position	Matched value	Description
MAC destination multicast address[1]	0–5	01-1B-19-00-00-00 01-80-C2-00-00-0E	All PTP messages can use any of the following multicast addresses[2]: • 01-1B-19-00-00-00 • 01-80-C2-00-00-0E[3]
MAC packet type	12, 13	0x88F7	PTP Ethernet packet
PTP control field (IEEE 1588 version 1)	46	0x00, 0x01, 0x02, 0x03, or 0x04	• 0x00: SYNC • 0x01: Delay_Req • 0x02: Follow_Up • 0x03: Delay_Resp • 0x04: Management
PTP message type field (IEEE 1588 version 2)	14 (nibble)	0x0, 0x1, 0x2, 0x3, 0x8, 0x9, 0xB, 0xC, or 0xD	• 0x0: SYNC • 0x1: Delay_Req • 0x2: Pdelay_Req • 0x3: Pdelay_Resp • 0x8: Follow_Up • 0x9: Delay_Resp • 0xA: Pdelay_Resp_Follow_Up • 0xB: Announce • 0xC: Signaling • 0xD: Management
PTP version	15 (nibble)	0x1 or 0x2	• 0x1: Supports PTP version 1 • 0x2: Supports PTP version 2

1. The unicast address match of destination addresses (DA), which is programmed in MAC address 0 to 31, is used if MAC_Timestamp_Control[TSENMACADDR] = 1.
2. IEEE 1588-2008, Annex F
3. MAC does not consider the PTP version 1 messages with peer delay multicast address (01-80-C2-00-00-0E) as valid PTP messages.

76.8.4 Transmit path functions

MAC captures a timestamp when the start packet delimiter (SFD) of a packet is sent on the MII interface. You can control the packets, for which the timestamps are captured on a per-packet basis and mark each transmit packet to indicate whether to capture a timestamp for it. MAC does not process the transmitted packets to identify the PTP packets. You can use the control bits in the transmit descriptor to specify the packets. MAC returns the timestamp to the software inside the corresponding transmit descriptor and therefore connects the timestamp automatically to the specific PTP packet. You must write the 64-bit timestamp information to TDES0 and TDES1 fields. The TDES0 field holds the 32 least significant bits of the timestamp.

76.8.5 Receive path functions

You can program MAC to capture the timestamp of all packets received on the MII interface or to process the packets to identify the valid PTP messages.

You can use these options of MAC_Timestamp_Control to control the snapshot of the time sent to the application:

- Enable snapshot for all packets
- Enable snapshot for IEEE 1588 version 1 or version 2 timestamp
- Enable snapshot for PTP packets transmitted directly over Ethernet
- Enable timestamp snapshot for the received packet for IPv4 or IPv6
- Enable timestamp snapshot only for EVENT messages (SYNC, DELAY_REQ, PDELAY_REQ, or PDELAY_RESP)
- Enable the node to be a master or slave and select the snapshot type

Table 635. Timestamp snapshot dependency on register bits

SNAPTYPSEL	TSMSTRENA	TSEVNTENA	PTP messages
00	x	0	SYNC, Follow_Up, Delay_Req, Delay_Resp
00	0	1	SYNC
00	1	1	Delay_Req
01	x	0	SYNC, Follow_Up, Delay_Req, Delay_Resp, Pdelay_Req, Pdelay_Resp, Pdelay_Resp_Follow_Up
01	0	1	SYNC, Pdelay_Req, Pdelay_Resp
01	1	1	Delay_Req, Pdelay_Req, Pdelay_Resp
10	x	x	SYNC, Delay_Req
11	x	x	Pdelay_Req, Pdelay_Resp

DMA returns the timestamp to the software inside the corresponding receive descriptor. The extended status, contains the timestamp message status and the IPC status. You must write the extended status in the normal descriptor RDES1 and the snapshot of the timestamp in RDES0 and RDES1 fields of context descriptor. The RDES0 field holds the 32 least significant bits of the timestamp.

See the System time correction for programming guidelines for IEEE 1588 timestamping (System time correction).

76.8.6 IEEE 1588 system time source

IP supports both the external and internal time source for reference timestamping.

- External time source uses the 64-bit external time reference and clock as an input in IP. The clock input synchronizes the external timing reference into the MAC clock domain. Upper 32-bit indicates the time in seconds and the lower 32-bit indicates the time in nanoseconds.
- Internal time source uses the only clock input to generate timing reference internally for snapshot and capture timestamps. Internal time reference has two fields.
  - UInteger48 seconds field: The seconds field is the integer portion of the timestamp in seconds units. It is 48-bit wide.

    For example, 2.000000001 seconds are represented as seconds field = 0x0000_0000_0002.

— UInteger32 nanoseconds field The nanoseconds field is the fractional portion of the timestamp in nanoseconds units.

For example, 2.000000001 seconds are represented as nanoseconds field = 0x0000_0001. The nanoseconds field supports the following two modes:

- Digital rollover mode: In this mode, the maximum value in the nanoseconds field is 0x3B9A_C9FF, that is, (10e9-1) nanoseconds.
- Binary rollover mode: In this mode, the nanoseconds field rolls over and increments the seconds field after 0x7FFF_FFFF value . Accuracy is ~0.466 ns per bit.

There is a system time register module, it is used when you use an internal time reference. The 80-bit time is maintained in this module and updated using the input reference clock (CLK_PTP_REF_I). This time is the source for taking snapshots (timestamps) of Ethernet packets which is transmitted or received at the MII interface.

Initialize or correct the system time counter using the coarse correction method. In this method, write the initial value or the offset value to the timestamp update register. For initialization, write the system time counter with the value in the timestamp update register. For system time correction, the offset value is added to or subtracted from the system time.

In the fine correction method, correct the frequency offset and/or frequency drift of a slave clock (CLK_PTP_REF_I) with respect to the master clock (as defined in IEEE 1588-2002) over a period of time instead of in one clock, as in coarse correction. This maintains linear time and does not introduce drastic changes (or a large jitter) in the reference time between the PTP sync message intervals.

See Initialization guidelines for system time generation for programming guidelines for IEEE 1588 timestamping (For internal timestamp source configuration).

76.8.7 IEEE 1588 higher word register

You can invoke the higher word register if system time source is internal. MAC timestamp is 64-bit wide. The values of the upper 16-bits of the seconds field are read from the CSR register.

76.8.8 Flexible pulse-per-second output

IP supports the flexibility of programming the start or stop time, generates pulse width and interval on pulse-per-second output. It also support four such output signals. By default, IP is in fixed pulse-per-second output mode with interval of 1 second.

Initially you must program the start time in target time registers "MAC_PPSx_Target_Time_Seconds and MAC_PPSx_Target_Time_Nanoseconds" for all desired or enabled pps output. If you re-programs start or stop time then you must do it after the synchronization of earlier programmed value. Bit 31 of MAC_PPS#_Target_Time_Nanoseconds register indicates that the synchronization is complete. If the application programs a start or stop time that has already elapsed, MAC sets an error status bit which indicates the programming error. If enabled, MAC also sets the target time reached interrupt event. The application can cancel the start or stop request only if the corresponding start or stop time has not elapsed. If the time has elapsed, the cancel command has no effect.

Program the PPS width and interval in terms of granularity of system time, that is, the number of units of sub-second increment value.

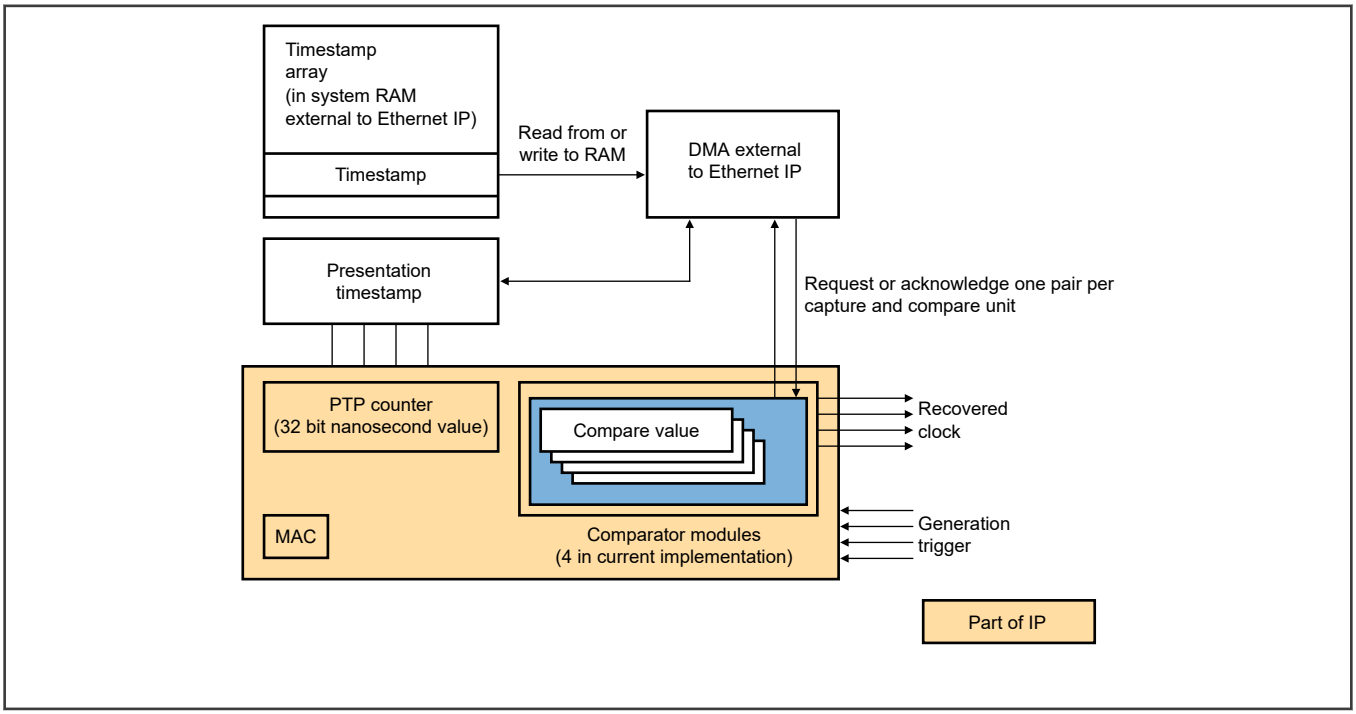
See Programming guidelines for flexible pulse-per-second output for more information.

76.8.9 Media clock generation and recovery

Media clock recovery and generation requires a dedicated hardware and software to complete the operation. IP generates a reference clock with the help of a dedicated hardware which multiplies to get the desired media clock. Software extracts the presentation time from the incoming PTP 1722 frames and program these values into CSR registers so that IP can read them for media clock recovery.

Figure 458 shows that the timestamp array and the DMA module are external to IP. IP supports a 32-bit presentation time counter in nanoseconds that completes at the full 32-bit value to match with 1722 presentation time format and an handshake mechanism with an external application (such as DMA) to program the presentation time into CSR register.

Figure 458. Media clock generation and recovery block diagram



76.8.9.1 Presentation time counter

Presentation time counter provides an another perspective of the PTP system time (1588 timer). For a given PTP system time (PTP[63:32] represents time in seconds Comparator modules and PTP[31:0] represents time in nanoseconds), there exists a corresponding presentation time (32-bit value in nanosecond). The presentation time computed is referred to as current presentation time (CPT).

CPT derives from 64-bit PTP system time. PTPNS[63:0] = (PTP[63:32] * 32'd1,000,000,000) + PTP[31:0] where, PTPNS is the PTP system time converted into a 64-bit nanosecond value.

Current presentation time[31:0] = PTPNS[31:0]

Media clock recovery is possible when the system time is internally generated in IP and compute the CPT in the same way as the internal PTP system time generation. The increment cycle of CPT and system time are same because the values of both the timers are in nanoseconds and they are synchronous. The timer updates at the same instance, but the updated value could be different. You must compute a separate 32-bit update value in nanoseconds for the CPT update. CPT sampled at different edges of a triggering input generates media clock timestamps that are inserted in 1722 based AVBTP packets, to recover the clock at the destination.

76.8.9.2 Comparator modules

Figure 458 shows that the module supports four comparator modules to handle multiple media clocks that it may require. When MAC_Timestamp_Control[PTGE] is 1, it indicates that the comparator modules handshake with the application to program the timestamps into the MAC_PPS(#i)_Target_Time_Seconds register.

Write 1 to presentation time control fields of the particular instance for recovery mode.

The timestamps received from the application are referred as target presentation time (TPT).

When MCGREN#i field of MAC_PPS_Control is 1, it indicates that the IP operates in MCGR mode. The comparator module sends upto two requests to the application for TPT write, when MAC_Timestamp_Control[PTGE] and the presentation time control bits of the particular instance are 1 for MCGR mode. Subsequent requests are generated each time a presentation time match occurs. CPT transitions from a value less than TPT, to a value greater than or equal to CPT. The first request asserts when a specific comparator instance sets to MCGR mode with a non-zero presentation time control and an additional request is made when the comparator receives the first data. This allows the application to write the next TPT value when IP processes the previous TPT value for a match.

TPT read from the MAC_PPS(#i)_Target_Time_Second is considered as a future time. A toggle or pulse is generated in the next cycle when a match is detected. Also,the mcgr_dma_req_o#i request for that comparator asserts (to obtain next TPT) until the corresponding application acknowledgment is set, when a match is detected.

The presentation control fields (PPSCMD#i field of MAC_PPS_Control) determine the shape of the generated waveform. You can program these fields to either toggle or generate a high/low pulse for one PTP clock cycle, when matched.

Media clock recovery: A match occurs, when the free-running CPT value matches the received TPT value. An output signal, EMAC_TMR signal as shown in the IO mux sheet asserts (toggle, low pulse, or high pulse) on the basis of the programmed presentation control value.

Media clock generation: On the basis of the presentation control value programmed in MAC_PPS_Control, the particular comparator captures the presentation time and programs it into MAC_PPS(#i)_Target_Time_Second register. The captured timestamp is read when a request is raised to the application. No new timestamps are captured, until the read operation is complete acknowledgment is received from the application.

76.8.9.3 Media clock generation and recovery flow

Figure 459 shows the media clock generation and recovery flow. Write 1 to MAC_Timestamp_Control[PTGE] for CPT generation and write 1 to MAC_PPS_Control MCGREN#i field to enable the corresponding instance in the MCGR mode, for both media clock generation and recovery.

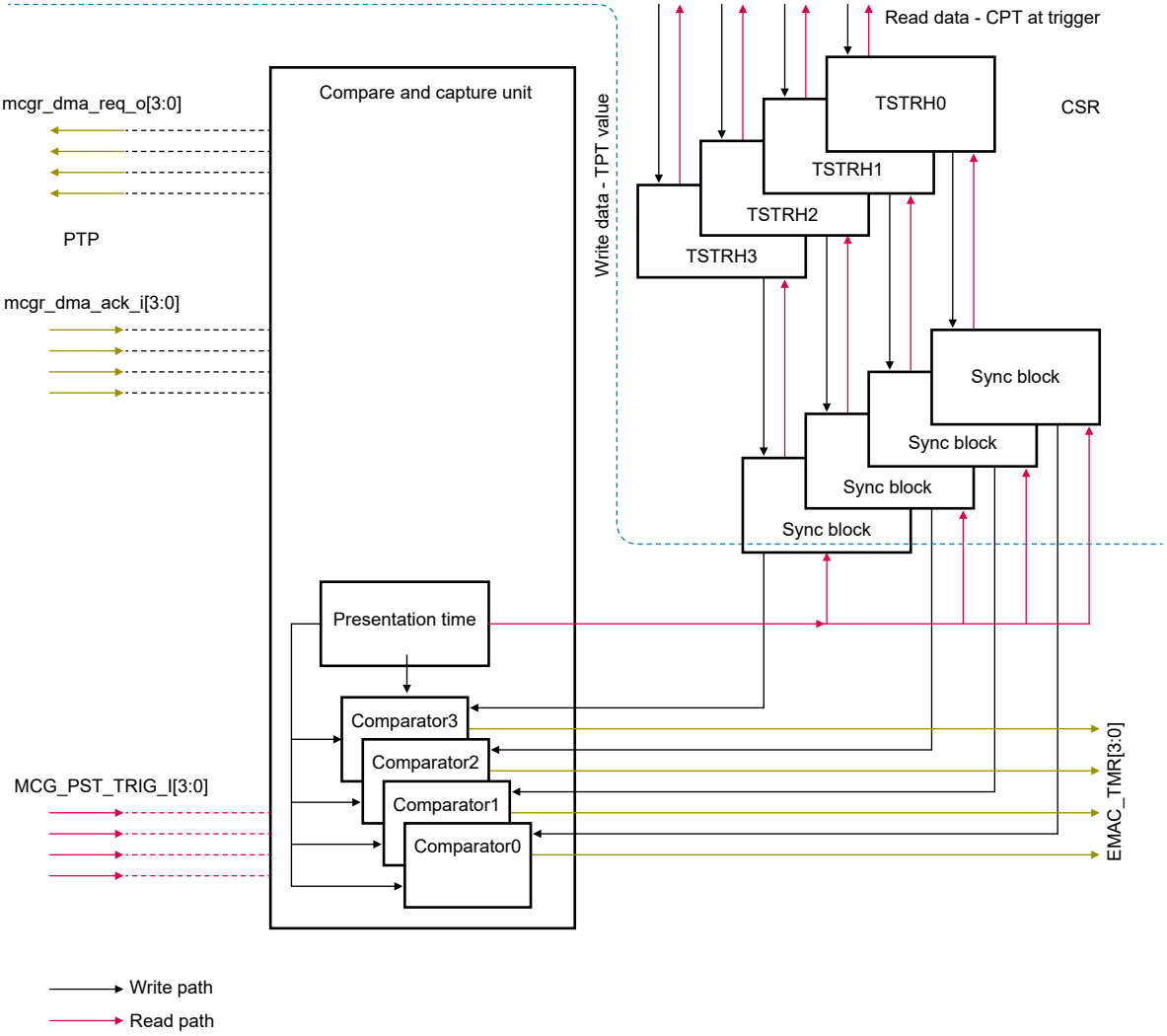


Figure 459. Media clock generation and recovery flow

NOTE

1. The consecutive triggers to sample the presentation time must assert after a few PTP clock cycles so as to allow synchronization delays. (There is no such issue when the input trigger maximum frequency is 8 KHz and the PTP clock runs at least at 1MHz)
2. In the Media Clock Recovery mode, the DMA acknowledgment is both posted as well as non-posted.

See Programming guidelines for media clock generation and recovery for more information.

76.8.10 One step timestamp

IP supports the one step timestamp feature. Writing 1 to bit 20 (OSTC) in the control word, enables the one step timestamp feature for a packet.

See the Programming guidelines for IEEE 1588 timestamping for more information.

76.8.11 IEEE 1588 sub nanoseconds timestamp

IP supports the ingress and egress correction accuracy in sub nanoseconds. It is programmed in MAC_Timestamp_Ingress_Corr_Subnanosec[TSICSNS] and MAC_Timestamp_Egress_Corr_Subnanosec[TSECSNS],

respectively. In this case, the correction has an unit of nanosecond, multiplied by 2^8. You must program the least significant 8 bits of the value in the sub-nanoseconds register.

For example, if the required egress correction is 3.6 nS and MAC_Timestamp_Control[TSCTRLSSR] is 1, then MAC_Timestamp_Egress_Corr_Nanosecond[TSEC] must be 0x3 and MAC_Timestamp_Egress_Corr_Subnanosec[TSECSNS] must be 0x99 (0.6 * 256). Similarly, if the required ingress correction is -3.6 nS and MAC_Timestamp_Control[TSCTRLSSR] is 1, then MAC_Timestamp_Ingress_Corr_Nanosecond[TSIC] must be 0xBB9A_C9FC and MAC_Timestamp_Ingress_Corr_Subnanosec[TSICSNS] must be 0x66 (fractional part of (10^9 - 3.6) * 256).

76.9 Multiple channels and queues support

This section and all its sub-sections are Synopsys Proprietary. Used with permission.

IP support three queues and channel on Tx and Rx paths. Figure 460 shows the architecture of IP with three queues and channel.

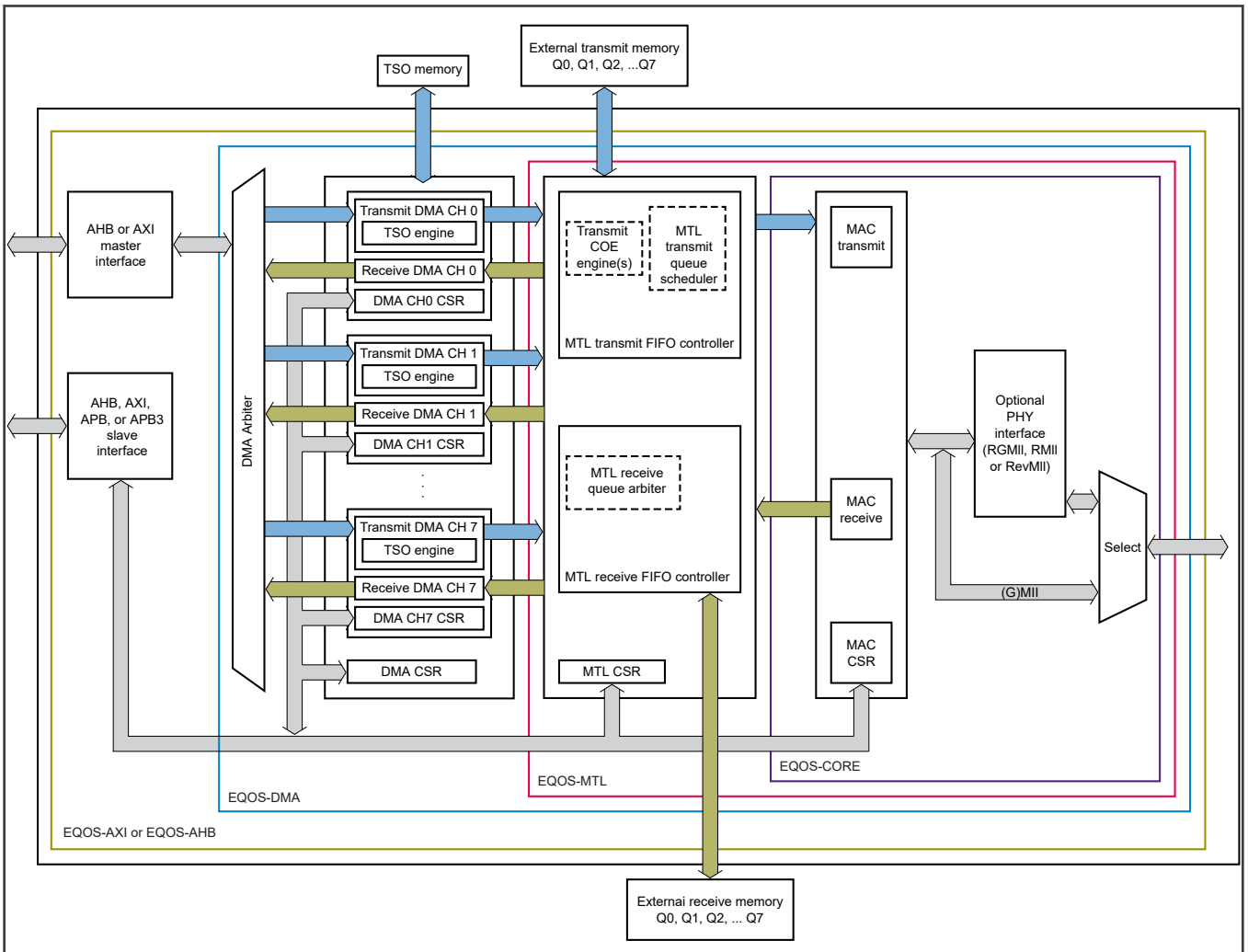


Figure 460. Module with multiple channels and queues block diagram

The above block diagram, support only three queues.

IP DMA arbiter support two types of arbitration scheme, fixed priority and weighted round-robin. The DMA arbiter performs the arbitration between the Tx and Rx paths of DMA channels to access descriptors and data buffers. DMA_Mode[DA] specifies the arbitration scheme (fixed or weighted round-robin) between the channel Tx and Rx DMA.

DMA_Mode[TXPR] sets the priority between the corresponding Tx DMA and Rx DMA. Writing 1 to DMA_Mode[PR] specifies the weighted priority between the Tx DMA and Rx DMA in round robin arbitration scheme.

Table 636 provides an information about the priority scheme between Tx DMA and Rx DMA.

Table 636. Priority scheme for Tx DMA and Rx DMA

Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Priority schemes
x	x	x	0	1	Rx always has priority over Tx
0	0	0	0	0	Tx and Rx have equal priority. Rx gets the access first on simultaneous requests
0	0	1	0	0	Rx has priority over Tx in ratio 2:1
0	1	0	0	0	Rx has priority over Tx in ratio 3:1
0	1	1	0	0	Rx has priority over Tx in ratio 4:1
1	0	0	0	0	Rx has priority over Tx in ratio 5:1
1	0	1	0	0	Rx has priority over Tx in ratio 6:1
1	1	0	0	0	Rx has priority over Tx in ratio 7:1
1	1	1	0	0	Rx has priority over Tx in ratio 8:1
x	x	x	1	1	Tx always has priority over Rx
0	0	0	1	0	Tx and Rx have equal priority. Tx gets the access first on simultaneous requests
0	0	1	1	0	Tx has priority over Rx in ratio 2:1
0	1	0	1	0	Tx has priority over Rx in ratio 3:1
0	1	1	1	0	Tx has priority over Rx in ratio 4:1
1	0	0	1	0	Tx has priority over Rx in ratio 5:1
1	0	1	1	0	Tx has priority over Rx in ratio 6:1
1	1	0	1	0	Tx has priority over Rx in ratio 7:1
1	1	1	1	0	Tx has priority over Rx in ratio 8:1

76.9.1 Multiple queues and channels support in the transmit path

IP support three transmit queues and three channels.

Fixed priority scheme is the default priority scheme for the DMA channels and channel with highest priority always wins the arbitration when it requests for the bus. Channel 0 is always of lowest priority and channel 1 is always of highest priority.

In weighted strict priority (WSP), the weight corresponds to the number of burst transfers given to a channel in one arbitration cycle. Reallocate the unused burst transfers of one or more channels on the basis of the priority. The channel with highest priority receives the unused burst transfer time before it is allocated to a channel with the next highest priority. You can process the next lower priority, when a channel uses the allocated burst transfers. After processing the allocated bandwidth of all the channels that have packets to transmit, allocate any unused burst transfer time to the channel of the highest priority (if required), and then next highest priority (if required), and so on.

In weighted round robin (WRR) priority scheme, program the weight through TCW field of DMA_CH#_Tx_Control register. IN WRR scheme, service all channels in round-robin order according to the weights settings. Table 637 shows that the TCW field of DMA_CH#_Tx_Control register provides the weight for each transmit channel.

Table 637. Weight for DMA channels

TCW Field	Transmit Channel Weight
000	1
001	2
010	3
011	4
100	5
101	6
110	7
111	8

The configured weights correspond to the number of burst transfers given to a channel in one arbitration cycle. The unused or excess burst transfers are distributed equally to all channels.

See Programming guidelines for multi-channel multi-queuing for more information.

76.9.2 Multiple queues and channels support in the receive path

IP support three receive channels and three queues.

In the receive direction, the MTL Rx controller selects the Rx DMA for which it transfers or reads the data from the receive FIFO memory. This scheduling is based on the programming done in the respective MTL_RxQ[x]_Control register.

Each Rx DMA indicates when it is ready to transfer data and the size of the burst-length (number of beats) that it will transfer. The scheduler checks whether sufficient data (of requested burst length) is available to transfer to these DMAs and then selects the receive DMA that is serviced using the programmed priorities.

See Programming guidelines for multi-channel multi-queuing for more information.

76.9.3 Rx queue to DMA mapping

You can program MTL_RxQ_DMA_Map0 (for queues 0, 1, and 2) to route the packets in the MTL receive queues to any one of the multiple DMA channels.

The following types of receive queue to DMA mapping is possible through programming.

76.9.3.1 Static mapping

In this mode, you can connect all the packets of the receive queue to a specific DMA channel. For example, all the packets from the receive queue 0 program MTL_RxQ_DMA_Map0[Q0MDMACH] (bit[3:0]) and MTL_RxQ_DMA_Map0[Q0DDMACH] (bit[7] = 0) to route to a DMA channel.

Similarly, packets from other receive queues program register fields corresponding to each queue to route to any DMA channel.

76.9.3.2 Dynamic (per packet) mapping

In this mode, the destination DMA channel of a packet read from a Rx queue is not constant but it is decided independently for each packet. For example, if you write 1 to MTL_RxQ_DMA_Map0[Q1DDMACH], it indicates that the static mapping disables for receive queue 1 and ignores the value in MTL_RxQ_DMA_Map0[Q1MDMACH]. The MAC receiver decides the destination DMA channel for each packet, depending on the following in decreasing order of priority:

1. **L3-L4 filter based DMA selection**: The TCP/UDP and IP header fields of the received packet are matched against the corresponding values programmed and enabled for comparison in the MAC_L3_L4_Address_Control register. If the match is successful, the DMA channel number programmed in the DMCHN field of the MAC_L3_L4_Address_Control register is selected as the destination DMA channel number, if DMCHEN field of the same register is 1. If none of the L3-L4 registers give a comparison match, then the module proceeds to the next step below.
2. **Extended VLAN based DMA selection**: Extended routing is applicable only if the VLAN filter has passed. Routing is done only on the basis of the perfect filter result. Each perfect filter has a DMA channel enable and a DMA channel number field which you can program. Routing is applicable for a filter, only if the DMA channel enable field is 1. The frame routes to the smallest matching filter's DMA channel, if it is enabled. If the filter's DMA channel number is not enabled, the frame route to channel 0. For example, if a frame's VLAN tag matches filters 7, 3, and 1, then the MAC checks if DMA channel number of filter 1 is enabled through programming. If yes, the frame route to the programmed value; otherwise it route to DMA. When the inverse filter is enabled; is routed to the least mismatched filter's DMA channel number provided it is enabled. If the DMA Channel enable bit is not set, then the frame routes on the basis of the DA based addressing or to channel 0.

   If hash filter is also enabled, it is determines the filter result only. Routing will still depends on the enabled perfect filters. Routing based on the VLAN filter will not occur if none of the perfect filters are enabled or if all of them are bypassed. The frame will route via DA based addressing or to channel 0. If all the perfect filters fails and the hash filter has passed, then the VLAN filter result is a pass but routing will be based on DA based addressing or to channel 0. Similar behavior is seen when you enables inverse filtering as well.
3. **Ethernet DA-based DMA selection**: The DA address of the received packet is compared against the programmed DA values in MAC address registers. If the address matches any of the programmed values, the corresponding DCS field (when enabled) determines the destination DMA channel number.

If none of the above operations make a successful match or decision, then the packet routes to the DMA channel 0 by default.

76.9.3.2.1 Broadcast/multicast packet duplication

This feature provides a mechanism to send the received broadcast or multicast packets to multiple DMA channels.

MAC_Address0_High[DCS] and other optional MAC_Address(#i)_High (i=1 to 2) registers determines the DMA channel number to which the received packet (that matches the MAC address present in that register) must route.

DMA channel routing mechanisms such as extended VLAN, or L3-L4 based routing does not support the packet duplication.

The packet duplication feature is supported only on the highest MTL queue configured. Therefore, you must program MAC_RxQ_Ctrl1[MCBCQ] to the highest RxQ present in the configuration for the packet duplication for broadcast or multicast packets.

76.9.4 Selection of tag priorities assigned to receive queues

Program the PSRQ field in the corresponding MAC_RxQ_Ctrl2 to assign the VLAN tag priorities to receive queues. The bit corresponding to the VLAN tag priority can be set in the PSRQ field for assigning that priority to the receive queue. For example, if you want to assign VLAN tag priority of 3 to receive queue 0, write 1 to bit [3] in PSRQ field of MAC_RxQ_Ctrl2. The VLAN tag priority assigned to particular receive queue must be unique, that is, you cannot assign more than one receive queue to the same VLAN tag priority. However, more than one VLAN tag priorities can be assigned to same receive queue.

The settings in the PSRQ field is used for VLAN tagged receive packet routing to receive queues as well as for PFC based transit flow control. The received VLAN tagged receive packet route to receive queue that has the VLAN tag priority match. In PFC based transit flow control, PSRQ field corresponding to a particular receive queue enables VLAN tag priorities in the PFC packet transmitted when corresponding receive queue reach the threshold level.

76.9.5 Receive side routing from MAC to queues

IP supports receive side routing from MAC to queues. MAC route the receive packets to the receive queues on the basis of the these packet types in that order

- Unicast/Multicast destination address packets that fail the destination address filter
- Multicast/Broadcast destination address packets that pass the destination address filter
- VLAN tag priority field in VLAN tagged AV data packets
- AV control packets
- VLAN tagged IEEE 1588 PTP over Ethernet packets
- Untagged IEEE1588 PTP over Ethernet packets
- VLAN tag priority field in VLAN tagged packets
- Untagged packets

You can route the outer C-VLAN tagged AV data receive packets on the basis of the priorities assigned to receive queues through PSRQ field in the corresponding MAC_RxQ_Ctrl2 and MAC_RxQ_Ctrl3 and the corresponding receive queue is enabled for AV through RXQ#EN field in MAC_RxQ_Ctrl0. These packets may be single VLAN tagged with C-VLAN type or double VLAN tagged with outer VLAN tag of C-VLAN type when double VLAN feature enables (MAC_VLAN_Tag_Ctrl[EDVLP] = 1) with an inner C-VLAN tagged or inner S-VLAN tagged when SVLAN processing is enabled (MAC_VLAN_Tag_Ctrl[ESVL] = 1). This type of Rx packet routing is available when you select the AV feature and multiple receive queues in the configuration.

You can route the AV control receive packets on the basis of the receive queue number specified in MAC_RxQ_Ctrl1[AVCPQ] and the corresponding receive queue is enabled for AV through RXQ#EN field in MAC_RxQ_Ctrl0. These packets may be single VLAN tagged with C-VLAN type or double VLAN tagged with outer VLAN tag of C-VLAN type when double VLAN feature is enabled (MAC_VLAN_Tag_Ctrl[EDVLP] = 1) with inner C-VLAN tagged or inner S-VLAN tagged when SVLAN processing is enabled (MAC_VLAN_Tag_Ctrl[ESVL] = 1). This type of receive packet routing is available when you select the AV feature and multiple receive queues in the configuration.

You can route the VLAN tagged receive packets on the basis of the priorities assigned to receive queues through PSRQ field in corresponding MAC_RxQ_Ctrl2 and MAC_RxQ_Ctrl3 and RXQ#EN field in MAC_RxQ_Ctrl0 enables the corresponding receive queue for DCB/generic. This type of receive packet routing is available when you select the multiple receive queues in the configuration.

You can route the untagged IEEE 1588 PTP over Ethernet receive packets on the basis of the receive queue number specified in MAC_RxQ_Ctrl1[PTPQ] and the corresponding receive queue is enabled through RXQ#EN field in MAC_RxQ_Ctrl0. Also, you can route the VLAN tagged IEEE 1588 PTP over Ethernet receive packets on the basis of the priorities assigned to receive queues through PSRQ field in corresponding MAC_RxQ_Ctrl2 and MAC_RxQ_Ctrl3 or the receive queue number specified in the MAC_RxQ_Ctrl1[PTPQ] and RXQ#EN field in MAC_RxQ_Ctrl0 register enables the corresponding Rx queue. Programming MAC_RxQ_Ctrl1[TPQC] determines this. This type of receive packet routing is available when IEEE 1588 timestamp feature supports and the multiple receive queues are selected in the configuration. The VLAN tagged IEEE 1588 PTP over Ethernet receive packets are detected only when you disables 802.1AS mode (MAC_Timestamp_Control[AV8021ASMEN] = 0), otherwise this type of receive packets are routed as generic VLAN tagged Rx packets.

You can route the multicast or broadcast destination address receive packets that passes the destination address filter on the basis of the receive queue number specified in MAC_RxQ_Ctrl1[MCBCQ]the MCBCQ field of MAC_RxQ_Ctrl1 register when enabled through MAC_RxQ_Ctrl1[MCBCQEN] and the corresponding receive queue is enabled through RXQ#EN field in MAC_RxQ_Ctrl0. This type of Rx packet routing is available when you select the multiple receive queues in the configuration.

You can route the untagged receive packets on the basis of the receive queue number specified in MAC_RxQ_Ctrl1[UPQ] and RXQ#EN field in MAC_RxQ_Ctrl0 enables the corresponding receive queue. This type of receive packet routing is available when you select the multiple receive queues in the configuration.

The unicast destination address receive packets that fail the destination address filter can be routed based on the receive queue number specified in MAC_RxQ_Ctrl4[UFFQ] when enabled through MAC_RxQ_Ctrl4[UFFQE]UFFQE, MAC_Packet_Filter[RA] = 1 and the corresponding receive queue is enabled through RXQ#EN field in MAC_RxQ_Ctrl0. This type of receive packet routing is available when you select the multiple receive queues in the configuration.

You can route the multicast destination address receive packets that fails the destination address filter on the basis of the receive queue number specified in MAC_RxQ_Ctrl4[MFFQ] when MAC_RxQ_Ctrl4[MFFQE] enables it. MAC_Packet_Filter[RA] = 1 and RXQ#EN field in MAC_RxQ_Ctrl0 enables the corresponding receive queue. This type of receive packet routing is available when you select the multiple receive queues in the configuration.

The receive packet will route through the receive queue 0, if it is not classified in any of the defined packet type for routing.

76.9.6 Receive side arbitration between DMA and MTL

IP controller supports receive side arbitration between DMA and MTL.

After the current packet processing completes, the DMA channel controller fetches the next descriptor and after the descriptor fetching completes, the DMA channel controller evaluates the amount of data to transfer to the Rx buffer on the basis of the programmed PBL and receive buffer length. Accordingly, it requests the MTL to transfer the data.

After servicing the current request, the MTL receive queue arbitration scheme selects receive queue on the basis of the arbitration scheme (MTL_Operation_Mode[RAA]) and the weights programmed in queue <n> receive control register. The arbitration is done among queues for which DMA is ready to service. After the receive queue is selected, PBL amount of data is read out from that queue and route to the receive DMA channel on the basis of the receive channel selection criteria.

The arbitration occur after every PBL data transfer completes and descriptors are ready for processing from at least one DMA channel.

76.9.7 Transit side arbitration between DMA and MTL

IP supports transit side arbitration between DMA and MTL. Transit DMA channels and transit queues are always mapped directly because the number of transit DMA channels are always equal to the number of transit queues in MTL. For instance, each transit DMA pushes data into its respective transit queue assigned to it.

The data inside each transit queue is stored in packets. Therefore, if two DMAs are allowed to transfer data into the same queue, when a transit DMA starts a packet transfer, the other DMA cannot transfer data unless the previous packet is completely pushed-in. This means that the second DMA remains idle until the first packet is transferred. Hence, each DMA is always connected directly to its corresponding transit queue.

76.9.8 Audio video bridging

IP supports the audio video bridging in 100 Mbps and 1000 Mbps modes, which allows the transmission of time sensitive traffic over network. IP implements these protocol for supporting the AVB feature:

- IEEE 802.1Qbv-2015 (Enhancements to scheduling traffic)
- IEEE 802.3br (Interspersing traffic)
- IEEE 802.1Qbu (Frame preemption)

The queue 0 transmit path supports the strict priority algorithm, and it is used for best-effort traffic when transmit paths of additional queues use the credit-based shaper algorithm to support traffic management. For a queue, the credit-based shaper algorithm determines that a queue is available for transmission if these conditions are true:

- The queue contains one or more packets.
- The credit for the queue is positive per the algorithm
- AV traffic is received on all queues. IP can also receive untagged PTP packets in addition to AV traffic on any queue.

76.9.9 Data Centre Bridging

The Data Center Bridging feature supports three types of algorithms:

- Weighted Round Robin: In Weighted Round Robin (WRR) algorithm, each queue is assigned a weight based on the percentage of configured bandwidth. All queues are serviced in the round-robin order according to the weight settings. This algorithm is less complex and requires less hardware as compared to other algorithms. Large packets get more bandwidth in this algorithm.

- Deficit Weighted Round Robin: In Deficit Weighted Round Robin (DWRR) algorithm, each queue is assigned a weight based on the percentage of configured bandwidth. A deficit counter holds the credit for transmission. The quantum value, proportional to the bandwidth, is added to the deficit counter every time a queue is scanned. The packet in the queue is selected for transmission only if the credit in the deficit counter is less than or equal to the size of the packet available at the head of the queue. A selected queue is serviced until the outstanding credit is lesser than the packet size at the head of the queue during a particular queue scan cycle. After this, the next lower priority queue is serviced. When the packets in a queue are over, the credit becomes zero for that queue scan cycle. This algorithm is less complex, and it provides more fairness in scheduling as compared to other algorithms.
- Weighted Fair Queuing: In Weighted Fair Queuing (WFQ) algorithm, each queue is assigned a weight based on the percentage of configured bandwidth. The algorithm computes the finish time of packets at the head of the queues based on the packet size and weights for the queue. The packets with earliest finish time is scheduled for transmission first. The algorithm is more complex and less scalable as compared to other algorithms.

76.9.10 Queue modes

Transit and receive queues both handles the AV traffic.

Queueing is based on WRR (Weighted round robin) or WSP (Weighted strict priority) algorithm for generic traffic. It is based on CBS (Credit Base Shaper) and SP (Strict Priority) algorithm for AV traffic.

The receive queue is configured for generic or AV based routing, which depends on the RXQ#EN field of corresponding queue. Queueing is done on the basis of the VLAN tag priority. The VLAN tag priority must match the PSRQ field of MAC_RxQ_Ctrl2. The receive packets can route to a particular DMA channel on the basis of the DCS field of perfectly-matched MAC address register.

By default, the untagged packets in a generic traffic route to the receive queue specified in MAC_RxQ_Ctrl1[UPQ]. Queue 0 is the default value of MAC_RxQ_Ctrl1[UPQ]. You can override the default value with any other value, for the MAC_RxQ_Ctrl1[UPQ].

The AV control packets (tagged or untagged) route on the basis of MAC_RxQ_Ctrl1[AVCPQ]. The PTP over Ethernet packets route on the basis of MAC_RxQ_Ctrl1[PTPQ].

76.9.11 Queue priorities

You can program the priority of an receive queue in the corresponding field of MAC_RxQ_Ctrl2. Also, you must program the AV queue (high priority) as an higher priority than the best-effort queue (low priority).

76.9.12 Enhancements to scheduled traffic (EST)

The IEEE 802.1Qbv-2015 defines the schedule for each queues on every egress port which makes the implementation aware of traffic arrival schedule. This information blocks the lower priority traffic from transmission in this time window or slot. This ensures that the sender forward the scheduled traffic to receiver through all the network nodes with a deterministic delay.

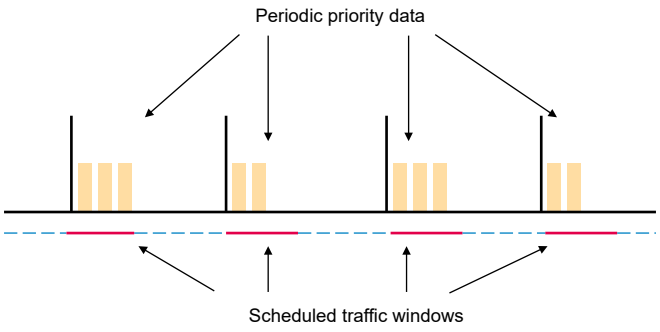


Figure 461. Time aware shaper implementation for traffic scheduling

An important requirement to achieve a low latency is to ensure that there are no interfering frames during the scheduled windows that are reserved for high priority traffic. The use of scheduled traffic imposes limitations when a transmission starts.

As shown in Figure 462 , if an interfering frame begins transmission just before the reserved time period starts, it can extend transmission into the reserved window, and potentially interfere with higher priority traffic. Therefore, a guard band whose size is equal to the largest possible interfering frame is required before the window starts.

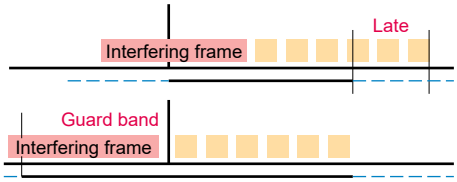


Figure 462. Implementing a guard band to avoid delays due to interfering frames

A larger guard band equates to a less efficient use of network bandwidth. However, this issue is addressed with the implementation of IEEE802.1Qbu (frame preemption). Frame preemption breaks the interfering frame into smaller fragments. Therefore, the guard band must be as large as the largest possible interfering fragment instead of the largest possible interfering frame.

During the guard band, only the frames that can complete the transmission of the entire frame before the next gate close event are permitted. This ensures that the high priority traffic can always start at the beginning of the window reserved for it.

76.9.12.1 Frequently used terms in EST

These are some of the frequently used terms in the EST support and their definitions.

- Gate control list: The list in the hardware memory that hold the gate controls and the associated time intervals.
- Gate controls: For a given schedule (row in gate control list) there is a gate open (O) and gate close (C) state associated with each TC. The set of O or C values, whose width is same as configured TCs is called the gate controls. Example: CCOCOOCO means TC7=C, TC6=C, TC5=O and so on.
- Time interval: Time interval (in nanoseconds) is a 16, 20, or 24-bit configurable field in the gate control list that indicates the time for which the associated gate controls are valid.
- Base time register: Each gate control list is associated with a 64-bit base time register that holds the start time (in PTP format) for the list.

76.9.12.2 Updates to the transmit scheduling to support EST

To support EST, these updates are required to transmit scheduling:

- Implementation of gate control list (GCL)
- Enforcing gate controls in the scheduler
- Utilizing gate open (O) duty cycle in the computation of idleSlope (CBS)

76.9.12.2.1 Implementation of gate control list (GCL)

Figure 463 shows how the gate control list governs the gate close(C) and open(O) events on the basis of the schedule provided for each event. GCL has two parts:

- Time interval: Defines the time in nanoseconds for which the gate controls are valid and must apply before reading the next gate controls from the list. The configurable width of 16, 20 or 24-bit represents a maximum of 64us, 1ms, or 16ms schedule interval respectively. It supports left shift of the time interval upto 7 bits to be able to apply a multiplication factor from 1 to 128ns (in steps of 2(power)n). The maximum value (post shifting) of this field must be 999,999,999 ns.
- Gate control: Defines the open (O) represented by logic 1 or close (C) represented by logic 0 state for the gate of each TC.

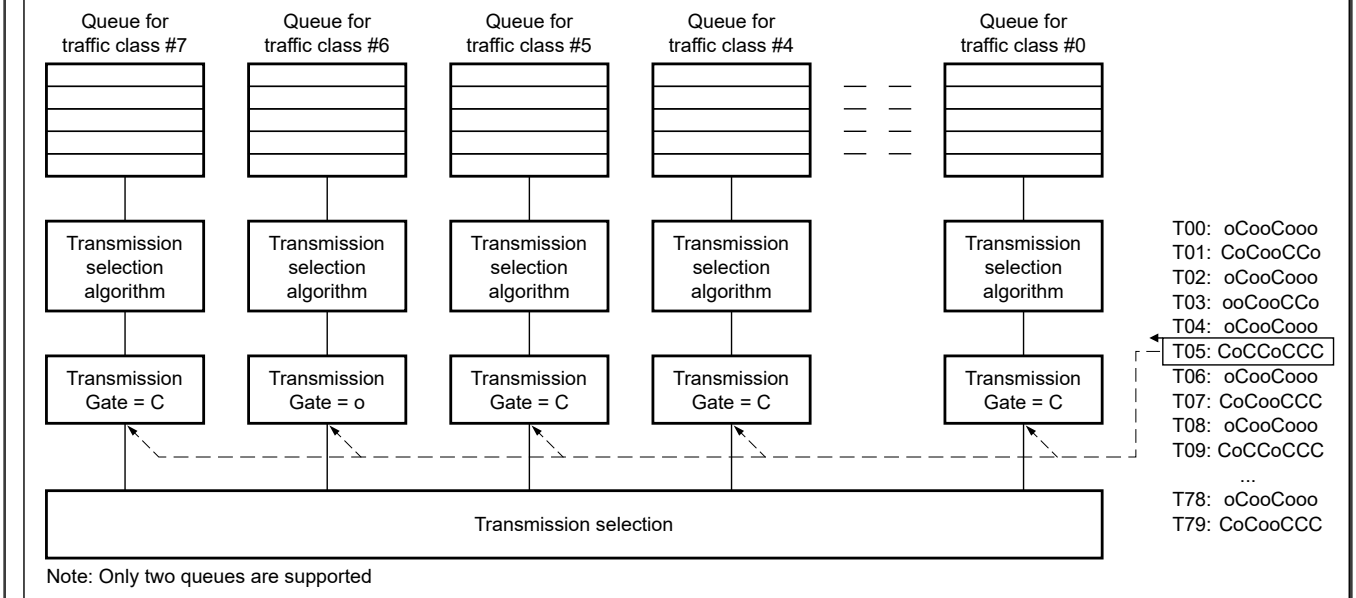


Figure 463. Block diagram from IEEE 802.1 Qbv specification - GCL governing gate close and open events

The implementation of GCL consists of these two gate control lists:

- HWOL - Hardware owned list which is a list for hardware access.
- SWOL - Software owned list which is a list for software access.

The access to these lists is mutually exclusive. Hardware sets the ownership to the list in MTL_EST_Status[SWOL]. Table 638 provides the implementation details of GCL.

Table 638. External memory used for holding the two gate-control lists

Gate control (up to 8 bits)	Time interval (ns) 16, 20, or 24 bits	
OOCCCCCC	T0	HWOL or SWOL
OOOOCCCC	T1	
CCCCOOCC	T2	
CCCCCCOO	T3	
I	I	
I	I	
OCOCOCOO		
OOOOCCCC	T last	
OOOOCCCC	T0	SWOL or HWOL
OOOOCCCC	T1	

Table continues on the next page...

Table 638. External memory used for holding the two gate-control lists (continued)

Gate control (up to 8 bits)	Time interval (ns) 16, 20, or 24 bits	
OOCCCCCC	T2	
OOOOCCCC	T3	
I	I	
I	I	
OCOCCCCC		
OOCCCCCC	T last	

76.9.12.2.2 Registers related to gate control list

The following are the set of four registers (one for each GCL) related to GCL. These registers are implemented through Indirect addressing using MTL_EST_GCL_Control and MTL_EST_GCL_Data registers.

1. 64-bit Base time register (BTR)

2. 40-bit Cycle time register (CTR)

3. m-bit Time extension register (TER)

4. n-bit [Gate control] List length register (LLR) (n depends of the configured GCL depth)

76.9.12.2.2.1 Base time register (BTR)

This is a 64-bit register that specifies the start time to execute the GCL. The format of the BTR is same as the PTP format (upper 32-bits holds time in seconds and lower 32 bits hold time in nanoseconds). After the execution of a given list starts, the implementation can update the value in BTR to indicate the next list execution start time.

76.9.12.2.2.2 Cycle time register (CTR)

This is a 40-bit register that specifies the time at which the execution of the GCL must repeat. This register consists of an 8-bit value in seconds, and a 32-bit value in nanoseconds (similar to the PTP time format with truncated seconds register). For a given GCL, the start time is "Base Time" + N * "Cycle Time" where N is an integer value that represents the iteration number starting with 0 for first iteration. If the GCL execution takes longer than the cycle time, then the list is truncated at the cycle time and the subsequent loop begins at cycle time.

76.9.12.2.2.3 Time extension register (TER)

This is a m-bit (where m = Configured time interval width + 7) register that specifies the amount of time (in nano seconds) the current GCL can extend before switching to the new GCL. This helps to avoid the execution of the small fragments of the current list before switching to a new list.

76.9.12.2.2.4 List length register (LLR)

This is an n-bit register (when n is 7, 8, 9, 10, or 11 for a GCL configured depth of 64, 128, 256, 512, or 1024 respectively) that specifies the integer value of the length of the GCL (that is the number of valid rows in each GCL). The processing of the GCL stops after the number of rows read equals to the LLR value.

76.9.12.2.3 Transmission gating implementation

A bridge or an end station can enhance to allow transmission from each TC that is yet to be scheduled relative to a known timescale. To achieve this, a transmission gate is associated with each TC; the state of the transmission gate determines if you can select the queued frames for transmission.

For a TC, the transmission gate can be in one of these two states:

1. Open: Select queued frames for transmission, in accordance with the definition of the transmission selection algorithm associated with the TC.
2. Closed: Does not select queued frames for transmission.

A frame on a traffic class queue is not available for transmission if the transmission gate is in the closed state or if there is insufficient time available to transmit the entirety of that frame before the next gate-close event associated with that queue.

The implementation has visibility to the current schedule of gate controls and the immediate next schedule of the gate controls. So the maximum Gate Open period does not exceed the sum of the two Time Intervals. This is because, a frame is selected for transmission only if the gate is currently Open and the duration of gate open interval is greater than the time taken to transfer the entire frame.

The implementation must know the frame size before the transmission, so that you can avoid the transmission overruns and only the frames that can complete are scheduled at all times.

The implementation adequately compensates for the implementation delays in the data transfer from the buffer to the line by offsetting the current time with all the relevant delays (provided by MTL_EST_Control[CTOV]). This ensures that the schedule provided is always accurately implemented at the line.

You must ensure that the GCL slot interval is always greater than the expected packet size and overhead (scheduler delay, inter frame gap (IFG), and preamble, all combined).

76.9.12.3 Idle slope computation updates

When EST is enabled, credit accumulates only when the gate is open therefore, the effective data rate of the idleSlope must increase to reflect the duty cycle for the transmission gate associated with the queue.

The idleSlope is computed on the basis of the gate open time and oper cycle time values. Program the idleSlope registers (implemented one per CBS enabled TC) based on the following equation. The existing MTL register has sufficient field width to accommodate the new values for idleSlope.

idleSlope = (operIdleSlope(N) * OperCycle/GateOpenTime)

76.9.12.3.1 Operational details of GCL

Write 1 to MTL_EST_Control[SSWL], so that the hardware can access the programmed gate control list. The first set of gate controls are applied when the current time is equal to the value in the base time register (BTR) and is held until the programmed "time interval" value.

One additional gate control event is always read ahead from the list, to avoid the transmission overruns. This enables the GCL to determine the next gate close events (if any) for the open TCs.

The scheduling is done on the basis of the gate open state and time interval of only the current and subsequent schedule. An internal accumulator adds the time intervals when gate controls are applied. BTR + Accumulator specifies the time at which the next set of gate controls are applied.

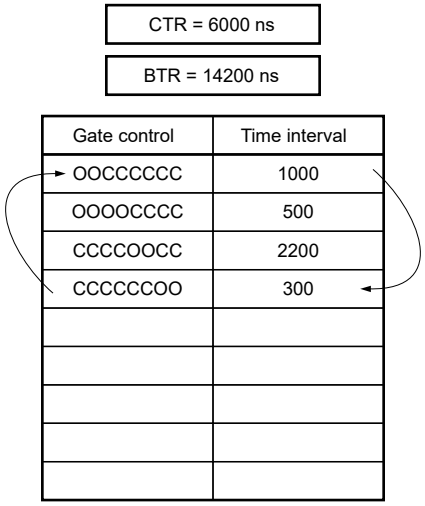


Figure 464. GCL and associated registers - basetime and cycletime

GCL is read progressively from the first row adhering to the schedule. The read operations continue until the list length (from LLR register) is reached and the execution of the list restarts at BTR + CTR time. At this point the value in CTR increments BTR to mark the beginning of a new cycle. In the absence of any gate controls, all the gates are in open state, during the execution of the list.

In cases where the execution time of the list is greater than the cycletime, the list is truncated and the next iteration starts when the current time equals BTR + CTR.

Table 639. GCL and associated registers - BTR and CTR

Current time	Gate control applied	Accumulator value	BTR (with updates)
14200	OOCCCCCC	1000	14200
15200	OOOOCCCC	1500	14200
15700	CCCCOOCC	3700	14200
17900	CCCCCCOO	4000	14200
18200	OOOOOOOO	0	20200
20200	OOCCCCCC	1000	20200
21200	OOOOCCCC	1500	20200

Table 639 describes an example in which the execution starts at 14200 and the first set of gate controls "OOCCCCCC" are applied immediately. The time interval is 1000 ns, so the next set of gate controls are applied at 14200 (BTR) + 1000 (Accumulator) = 15200 ns . The above table shows that there are no gate controls available after the execution of the last gate control and before the next iteration of the loop. The gates are deemed in open state during that time period as depicted at 18200 current time.

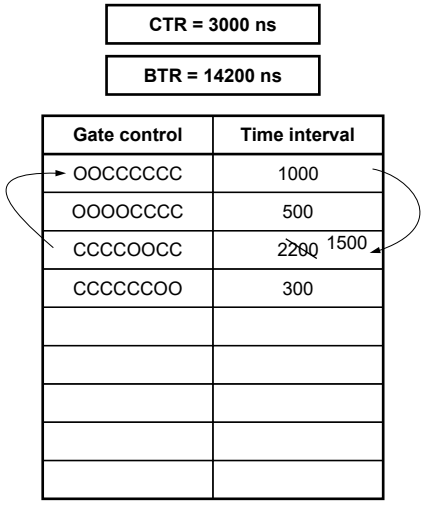


Figure 465. GCL and associated registers - BaseTime and CycleTime list execution time > CycleTime

Table 640 indicates that the list execution takes longer than the allocated CycleTime, so the list is truncated and the list starts from the BTR+CTR.

Table 640. GCL and associated registers - BTR and CTR, execution time > Cycle Time

Current Time	Gate Control Applied	Accumulator Value	BTR (with updates)
14200	OOCCCCCC	1000	14200
15200	OOOOCCCC	1500	14200
15700	CCCCOOCC	3700	14200
17200	OOCCCCCC	1000	17200
18200	OOOOCCCC	1500	17200
18700	CCCCOOCC	3700	17200

When you apply the third set of gate controls, BTR + CycleTime (17200) < BTR + Accumulator (17900), so the list is truncated and execution switches to a new iteration at 17200.

76.9.12.3.2 Installing a new GCL

The switch to the new GCL can happen in one of the following ways when a new software programmed GCL is available and executed at the new BTR value:

- New base time aligned with current schedule
- New base time unaligned with current schedule

76.9.12.3.2.1 New base time aligned with current schedule

If the choice of cycle time for the new gating cycle is unchanged from the cycle time for the current gating cycle, and if the BTR chosen for the new gating cycle (new BTR) is an integer multiple of the current cycle time (+ current BTR), then the new gating cycle exactly lines up with the old gating cycle, that is, the cycle start times for the new gating cycle is same as they would have been for the old configuration. This could be considered to be the ideal case and allows the new gating cycle to be installed and

executed with no timing issues. The implementation completes the execution of an iteration of the current list and switches to the new list at the beginning of the BaseTime listed in the new list.

If (New base time > = Current time)

ConfigChangeTime = New BaseTime

Else If (New base time < Current time)

1. Set the BTRError

2. ConfigChangeTime = (New BaseTime + N* New CycleTime)

where N is the smallest integer for which the relation ConfigChangeTime >= CurrentTime and (N =< 8) is true.

When N > 8 the hardware cannot auto recover and the loop count value in BTRError reporting is set to 1111 requires the software to reprogram the new base time.

Figure 466 illustrates the installation of the new GCL along with the timelines.

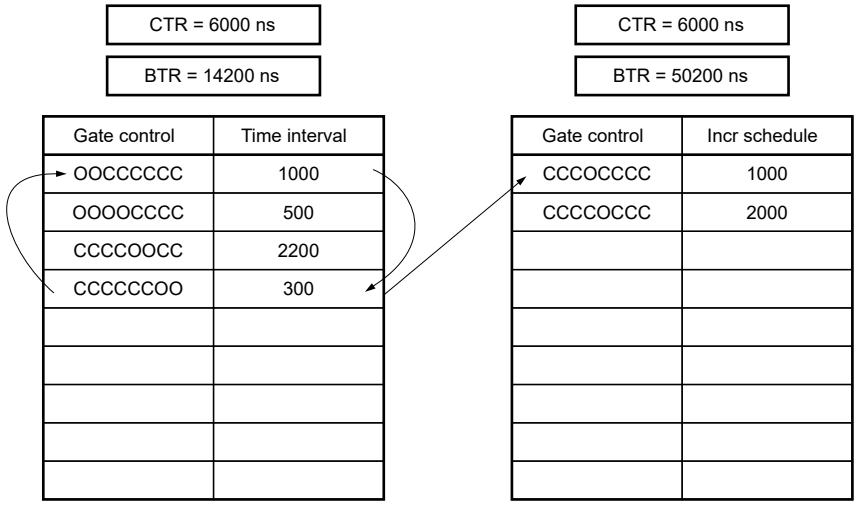


Figure 466. Switching to a new configuration that is aligned with the existing configuration

In the above example, after the sixth iteration of the first GCL, the BTR values of the old and new GCL are equal. At that point the new GCL is processed as a natural extension to the existing GCL.

Table 641. GCL and associated registers - BTR and CTR

Current time	Gate control applied	Accumulator value	BTR (with updates)
44200	OOCCCCCC	1000	44200
45200	OOOOCCCC	1500	44200
45700	CCCCOOCC	3700	44200
47900	CCCCCCOO	4000	44200
48200	OOOOOOOO	0	50200
50200	CCCOCCCC	1000	50200

Table continues on the next page...

Table 641. GCL and associated registers - BTR and CTR (continued)

Current time	Gate control applied	Accumulator value	BTR (with updates)
51200	CCCCOCCC	3000	50200
53200	OOOOOOOO	0	56200

76.9.12.3.2.2 New base time unaligned with current schedule

If the new cycletime differs from the current cycletime or new basetime in the future and is not an integer multiple of current cycletime, then the old and new cycles do not line up. When new basetime is reached (when the new configuration is installed and starts to execute), the last old cycle is normally truncated to start the first new cycle. This could be undesirable if it results in a very short last old cycle; arguably it would be better to simply extend the penultimate old cycle by that small amount, rather than starting a very short cycle. The Cycle Time Extension Register (related to the current config list) allows this extension of the last old cycle to be done in a defined way; if the last complete old cycle ends normally in less than current Cycle Time Extension (TER) ns before the new base time, then the last complete cycle before new BaseTime is reached is extended so that it ends at new BaseTime.

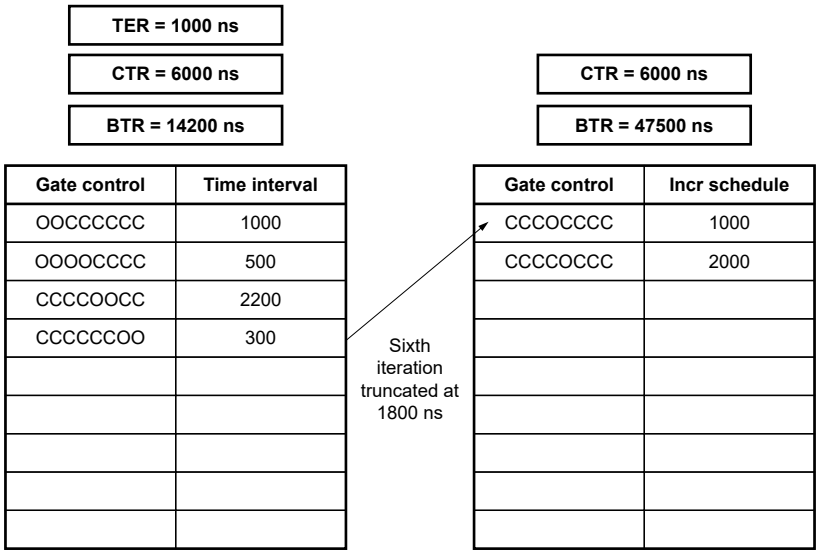


Figure 467. Switching to new list truncating the current list

At the end of the fifth iteration the Current time + Cycle time extension (TER) < New BTR so the sixth iteration of current configuration is started. During the sixth iteration of the current list when the new BTR value is smaller than the next schedule in the current list, it switches to the new list.

Table 642. Extending to new list by truncating the current list

Current time	Gate control applied	Accumulator value	BTR (with updates)
44200	OOCCCCCC	1000	44200
45200	OOOOCCCC	1500	44200
45700	CCCCOOCC	3700	44200
47500	CCCOCCCC	4000	44200

Table continues on the next page...

Table 642. Extending to new list by truncating the current list (continued)

Current time	Gate control applied	Accumulator value	BTR (with updates)
48500	CCCCOCCC	0	50200
50500	OOOOOOOO	1000	50200

Below is an example where the current config list is extended instead of starting a new iteration as the extension time of 800 ns is less than the allowed cycle extension time (TER) of 1000 ns.

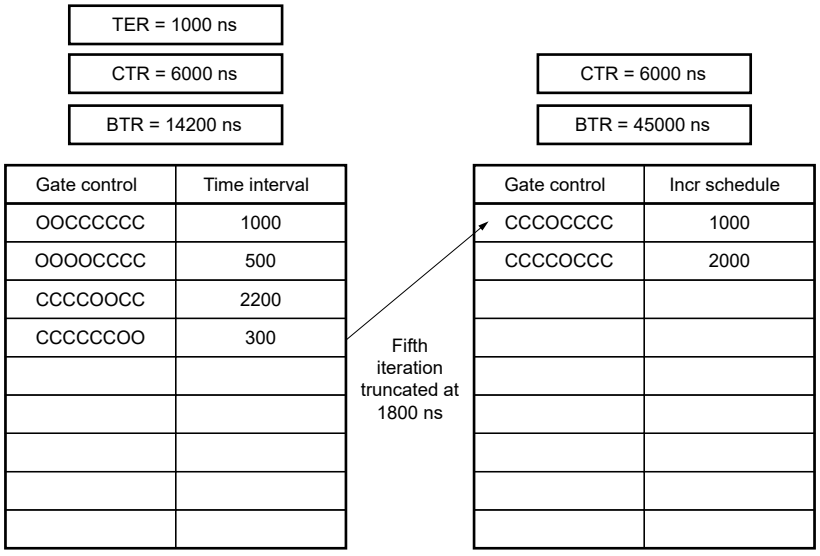


Figure 468. Switching to new list by extending the current list

Table 643. Switching to new list by extending the current list

Current time	Gate control applied	Accumulator value	BTR (with updates)
38200	OOCCCCCC	1000	38200
39200	OOOOCCCC	1500	38200
39700	CCCCOOCC	3700	38200
41900	CCCCCCOO	4000	38200
42200	OOOOOOOO	0	44200
45000	CCCOCCCC	1000	45000
46000	CCCCOCCC	3000	45000
48000	OOOOOOOO	0	51000

76.9.12.4 Impact of Transmit Scheduling Algorithms on EST

When EST is used in isolation, the Gate Control List manages the final open/close state of the Queues along with the checks carried out by the Transmission Selection Algorithm in MTL. As the Gate Controls operate on a predefined repetitive schedule, it is recommended to use Strict Priority or Credit Based Shaper (CBS) scheduling algorithms.

Other algorithms such as the Weighted Round Robin (WRR), Deficit Weighted Round Robin (DWRR) and Weighted Fair Queuing (WFQ) implement masking of the queues based on the current winning queue. The algorithm is applied only among the group of queues that open simultaneously. To ensure Queues whose gates are "Open" get priority, these algorithms are modified to treat "gate open" queues and "gate closed" queues as separate groups giving priority to gate open queues.

For example, consider 4 queues (Q3, Q2, Q1, Q0) with weights 4:3:2:1; Q3 and Q2 are in Open state in slot one slot, while Q1 and Q0 are in Open state in another slot. In this case, the scheduler works as follows:

1. In the first slot, the Q3 and Q2 are serviced in the ratio of 4:3 for the duration the slot is open.
2. In the second slot, the queues Q1 and Q0 are serviced in the ratio of 2:1.
3. Fresh arbitration is started every time a slot is opened.

In other words, the traffic does not get distributed in the intended ratio of 4:3:2:1; but as two groups with different ratios and only for the duration of the slot when the gates are open continuously

NOTE

See Programming guidelines for EST

76.9.13 Frame preemption (FPE)

Frame preemption breaks the interfering frame into smaller fragments. Therefore, the guard band needs to be only as large as the largest possible interfering fragment instead of the largest possible interfering frame. During the guard band, only the frames that can complete the transmission of the entire frame before the next gate close event are permitted. This ensures that the high priority traffic can always start at the beginning of the window reserved for it.

Preemption allows one or more higher priority (express) frames to interrupt the transmission of a lower priority (preemptable) frame; the preemptable frame transmission is resumed and completed after the express frame transmission is complete. To support frame preemption, the following two abstractions of the MAC are used:

- A preemptable MAC, called pMAC, which carries the preemptable traffic.
- An express MAC, called eMAC, which carries the express traffic.

In the implementation, only parts of the MAC that holds the state during preemption is replicated and represented as pMAC and eMAC. The MAC uses the following two ways to puts on hold, the transmission of the preemtable traffic, in the presence of express traffic:

- The MTL scheduler interrupts the preemptable traffic that is currently being transmitted. When the preemption capability is active, the MAC interrupts the transmission and reception of preemptable frames. A preempted fragment can be followed by zero or more express frames, before the continuation fragments. The preemptable frame can be fragmented any number of times, however, the minimum final and non-final fragment length criterion must be is met. However, interleaving of more than 1 preemptable packet is not permitted. This implies that if a preemptable packet is fragmented by an express packet, another preemptable packet cannot be transferred until all the remaining fragments of the first preempted packet are transferred.
- The MTL scheduler prevents starting the transmission of preemptable traffic. When the preemption capability is inactive, the pMAC entity is disabled and only express traffic is transmitted or received. Frame preemption feature can be enabled by setting EFPE field of the MAC_FPE_CTRL_STS register.

If you are unable to schedule fragmented packets after a certain number of attempts because of some reason, there is a possibility that the remaining fragmented packets are dropped. For this, software can program MTL_EST_Control[LCSE] to increase number of iterations or MTL_EST_Control[DFBS] to disable dropping of packet.

76.9.13.1 GCL Modification to Support FPE

In the EST-only configuration, the GCL entry has up to 24 bits of Time Interval and up to 8 high order bits representing the Gate Open/Close state requirements as shown in the following figure.

Gate control (up to 8 bits)	Time interval (ns) 16, 20 or 24 bits
OOCCCCCC	T0
OOOOCCCC	T1
CCCCOOCC	T2
CCCCCCOO	T3
|	|
|	|
OCOCOCOO	T126
OOOOCCCC	T127

Figure 469. Gate Control List When FPE is Disabled

EST only supports SetGate operation, which implies that the gates are set to either open or close at a given time interval. However, when both Frame Preemption (FPE) and EST are enabled, the GCL also supports Set-and-Hold-MAC and Set-and-Release-MAC operations, in addition to the SetGate operation. To enable the support of hold and release operations, the format of the GCL is slightly changed while configuring and enabling the FPE. The first Queue (Q0) is always programmed to carry preemption traffic and therefore it is always Open. The bit corresponding to Q0 is renamed as Release/Hold MAC control. The TX Queues whose packets are preemptable are indicated by setting the PEC field of the MTL_FPE_CTRL_STS register. The GCL bit of the corresponding Queue are ignored and considered as always "Open". So, even if the software writes a "0" ("C"), it is ignored for such queues.

Gate control (up to 7 bits)	Release or hold indication	Time interval (ns) 16, 20 or 24 bits
CCCOOOO	0	T0
CCCCOOO	0	T1
OCCCOOO	1	T2
COCCOOO	1	T3
CCOCOOO	1	T4
CCCOOOO	0	T5
CCCCOOO	0	T6

Figure 470. Gate Control List When FPE is Enabled

When the Release/Hold bit transitions from a '0' to '1', a Set-and-Hold-MAC operation is performed. This marks the cease of the preemptable traffic. This is achieved by sending a Hold request to MTL "ha" ns in advance (where ha is the time interval mentioned in the Hold Advance (HADV) field of the MTL_FPE_Advance register). When the Release/Hold bit transitions from a '1' to '0' a Set-and-Release-MAC operation is performed. This marks the resuming of the preemptable traffic. This is achieved by sending a Release request to MTL "ra" ns in advance (where ra is the time interval mentioned in the Release Advance (RADV) field of the MTL_FPE_Advance register). The preemptable traffic is blocked for the time duration the Release/Hold bit is set to a '1' in the Gate Control List.

76.9.13.2 Impact of Preemption on CBS

In Credit Base Shaper(CBS), the definition of "Transmit" is as follows:

- TRUE for the duration of frame transmission from the queue.
- FALSE when frame transmission from the queue is complete.
- When CBS algorithm is used along with frame preemption, the value of "Transmit" is TRUE only while the frame is being transmitted by the MAC. If the frame transmission is delayed or interrupted (for instance, a preemptable frame transmission is interrupted to allow the transmission of an express frame from a different queue, or the start of express frame is delayed because a preemptable frame is being transmitted) the value of "Transmit" is FALSE until transmission of the frame commences or is resumed.

Also, the value of "Transmit" is FALSE during the transmission of any overhead that is a consequence of frame preemption. For example, additional frame overhead (mCRC, Fragment Count) that is added to the preemptable frame.

At any given time, if there are no frames in the queue, and the value of Transmit is FALSE, and credit is positive value, the credit value is set to zero if here is no preemptable frame from the queue for which transmission is in progress but has been interrupted.

76.9.13.3 mPacket Format

When the preemption capability is active, MAC sends mPackets to the PHY. An mPacket can be one of the following:

1. A express packet
2. A preemptable packet
3. An initial fragment of a preemptable packet
4. A continuation fragment of a preemptable packet

Start mPacket Delimiter (SMD)

The value of the SMD indicates whether the mPacket contains an express packet, the start of a preemptable packet (initial fragment or complete packet), or any of continuation fragments of a preemptable packet. Following table shows the valid SMD values.

Table 644. Possible SMD Values of mPacket

mPacket Type	Notation	Frame Count	Value
verify packet	SMD-V	-	0x07
respond packet	SMD-R	-	0x19
express packet	SMD-E	-	0xD5
preemptable packet start	SMD-S0	0	0xE6
	SMD-S1	1	0x4C
	SMD-S2	2	0x7F
	SMD-S3	3	0xB3
continuation fragment	SMD-C0	0	0x61
	SMD-C1	1	0x52
	SMD-C2	2	0x9E
	SMD-C3	3	0x2A

frag_count

A frag_count is a modulo-4 counter that increments for each continuation fragment of the preemptable packet. The frag_count protects against mPacket reassembly errors by enabling detection of the loss of up to 3 packet fragments.

The frag_count field is present only in mPackets with SMD-C notation (continuation fragment). The frag_count is zero in the first continuation fragment of each preemptable packet.

Table 645. Possible frag_count Values

frag_count	Value
0	0xE6
1	0x4C
2	0x7F
3	0xB3

mData

The contents of the packet from the MAC, starting with the first byte after the SFD to the last byte before the FCS are sent in the mData fields of one or more mPackets for that frame. The minimum size of the mData field is 60 bytes.

CRC The CRC field contains a cyclic redundancy check (CRC) and has an indication of the final mPacket of a frame. In the final mPacket of a frame, the CRC field contains the last 4 octets of the MAC frame (the FCS field).

For other mPackets, the CRC field contains an mCRC value. The mCRC is calculated on the octets of the packet from the first octet of the frame (the octet following the SFD of preemption frames) to the last octet of the packet transmitted in that mPacket by performing an XOR of the calculated 32 bit CRC value of the fragment and a value of 0x0000FFFF.

Summary of Packet Formats

- Express Packet: 7bytes of PREAMBLE, SMD-E, Data, and CRC
- Complete Preemptable Packet: 7 bytes of PREAMBLE, <Current Preemptable packet SMD>, Data, CRC
- Initial Fragment (non-final) of Preemptable Packet: 7 bytes of PREAMBLE, <Current Preemptable packet SMD>, Data, mCRC
- Continuation fragments (non-final) of Preemptable packet: 6 bytes of PREAMBLE, <Current Preemptable continuation fragment SMD>, <Current Preemptable continuation fragment FC>, Data, mCRC
- Final fragment of Preemptable packet: 6 bytes of PREAMBLE, <Current Preemptable continuation fragment SMD>, <Current Preemptable continuation fragment FC>, Data, CRC

Table 646. Current and Previous SMD Values

Previous Preemptable packet SMD	Current Preemptable packet SMD
SMD-S0	SMD-S1
SMD-S1	SMD-S2
SMD-S2	SMD-S3
SMD-S3	SMD-S0

Table 647. Current and Previous SMD Values

Previous Preemptable fragment SMD	Previous Preemptable fragment FC	Current Preemptable continuation fragment SMD	Current Preemptable continuation fragment FC
SMD-S0	NA	SMD-C0	FC0
SMD-S1	NA	SMD-C1	FC0
SMD-S2	NA	SMD-C2	FC0
SMD-S3	NA	SMD-C3	FC0
SMD-C0	FC0	SMD-C0	FC1
SMD-C0	FC1	SMD-C0	FC2
SMD-C0	FC2	SMD-C0	FC3
SMD-C0	FC3	SMD-C0	FC0
SMD-C1	FC0	SMD-C1	FC1
SMD-C1	FC1	SMD-C1	FC2
SMD-C1	FC2	SMD-C1	FC3
SMD-C1	FC3	SMD-C1	FC0
SMD-C2	FC0	SMD-C2	FC1
SMD-C2	FC1	SMD-C2	FC2
SMD-C2	FC2	SMD-C2	FC3
SMD-C2	FC3	SMD-C2	FC0
SMD-C3	FC0	SMD-C3	FC1
SMD-C3	FC1	SMD-C3	FC2
SMD-C3	FC2	SMD-C3	FC3
SMD-C3	FC3	SMD-C3	FC0

76.9.13.4 Transmit Preemption

When FPE is enabled (setting EFPE=1 in MAC_FPE_CTRL_STS register), the MTL preempts a preemptable frame, when a "hold" request is asserted (EST/Qbv configured and enabled) express frames are available for transmission, that is, frame is present in MTL FIFO and is qualified for arbitration, after ensuring that the minimum mPacket mData field size is met. Therefore, preemption occurs only if at least 60 bytes of the preemptable frame have been transmitted and at least 64 bytes (including the frame CRC) remain to be transmitted.

The earliest starting position of preemption is controlled by the AFSZ field of the MTL_ FPE_CTRL_STS Register. Preemption does not occur until at least 64 * (1+ AFSZ) - 4 bytes of the preemptable frame have been sent.

When preemption occurs, all the preemptable queues are blocked and only the express queues are allowed to arbitrate (if more than one express queue has traffic) and transmit.

Continuation fragment of the preempted frame is the first frame to be transmitted after "release" request is asserted (EST/Qbv configured and enabled) and all the express traffic transmission completes.

NOTE

All the PTP packets should be transmitted as express packets

MTL communicates the following frame-type information to the MAC using a 2-bit preemption control signal on the MTI interface qualified by SoF and EoF.

- Express Frame
- Preemption Frame (Full or Fragment)
- Continuation Fragment (non-Final or Final)

Table 648. Preemption Control Values on MTI Interface for Various Frame Types

Qualifier	Preemption Control Value	Frame Type
SoF	00	Start Express
SoF	01	Start Preemption
SoF	10	Continuation Fragment
SoF	11	Reserved
EoF	00	End of Frame
EoF	01	End of Fragment

MTL should wait for the previous fragment status to be received before resuming the continuation fragments of a preempted frame. Fragment status is described in detail in the Tx Fragment Status section.

76.9.13.4.1 MAC Tx Preemption

MAC supports preemption by implementing the functionality needed to generate the mPackets as described in “mPacket Format”.

Based on the Preemption Control value received on the MTI interface (qualified with SoF and EoF), MAC determines the frame type (shown in the above table 'Preemption Control Values on MTI Interface for Various Frame Types') and generates mPackets accordingly.

When the preemption capability is active, MAC replaces the SFD of a preemption packet with an SMD-S value. A 2-bit rolling frame count is encoded in the SMD-S value.

The SMD-E value is the same as the SFD value so the SFD of an express packet does not need to be replaced.

76.9.13.4.1.1 Tx Fragment Status

MAC sends Tx fragment status to indicate successful transmission of fragmented mPackets.

In case of a transmission error (underflow, jabber, and so on) the frame status is sent (and not a fragment status) with an error indication along with all other relevant status fields. In case of receiving an error status for a transmitted fragment, the MTL drops the remaining fragments and does not send any more continuation fragments.

76.9.13.5 Receive Preemption

76.9.13.5.1 MAC Receive Preemption

When FPE is enabled, the MAC Receiver passes the incoming packets and differentiates between Express packets and preemptable packets. An SMD containing an SMD-E indicates express packet, and SMD containing an SMD-S indicates the first mPacket of a preemptable packet.

If an mPacket containing an SMD-S is received when MAC has not completed receiving the previous preempted packet, MAC sets a CRC Error status for the previously received partial packet.

When an SMD-S is detected, MAC records the frame count indicated by the SMD and then begins sending data on the MRI interface.

The MAC checks the last four bytes of the mPacket . If the last four bytes of the mPacket do not match CRC, that indicates the end of the packet with or without a CRC error as per the CRC check result. If the last four octets of the mPacket match, that indicates that the packet was preempted.

An SMD containing an SMD-C indicates an mPacket that continues the data for a preempted packet. Upon receiving an SMD value of SMD-C, MAC checks the following:

1. A preempted packet is in progress
2. The frame count indicated by the SMD matches the frame count of the packet in progress
3. The frag_count value indicates the next fragment count.

If any of the above check fails, the mPacket is discarded and MAC sets a CRC Error status for the partially received packet.

If all the checks pass, the next fragment count is incremented modulo 4.

When a packet is preempted, the MAC saves the state of the partially received packet (filter check status, timestamp, length fields etc.) and will be able to process any received Express packets before the continuation fragment is received.

The MAC Receiver sends a "dummy status" for all the mPacket fragments successfully received and sends the Rx status with the final fragment. If an error is detected during any of the fragments the Rx status is sent and the fragment is marked as final fragment. All subsequent continuation fragments received for this packet are dropped in the MAC.

The MAC communicates the following frame type information to the MTL using a 2-bit preemption control signal.

1. Express Frame
2. Preemption Frame (Full or Fragment)
3. Continuation Fragment (non-Final or Final)

Table 649. Preemption Control Values on MRI Interface for Various Frame Types

Qualifier	Preemption Control Value	Frame Type
SoF	00	Start Express
SoF	01	Start Preemption
SoF	10	Continuation Fragment
SoF	11	Reserved
EoF	00	End of Frame
EoF	01	End of Fragment

76.9.13.5.1.1 Data Alignment

When a received frame cannot be fragmented on any byte boundary, MAC retains the unaligned bytes of data in the previous fragment and resends them with the next fragment as shown in the following figure.

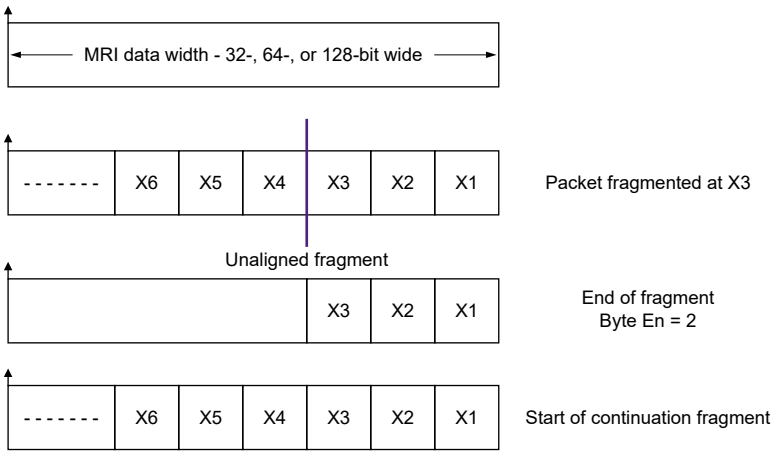


Figure 471. Data Alignment Feature of MAC

76.9.13.5.1.2 MTL Receive Preemption

The MTL Rx must have at least 2 Rx queues to support FPE function as the preemptable packets and the express packets must be routed to separate Rx queues. The destination Rx queue of a received packet is controlled by MAC_RxQ_Ctrl[n] registers. Program these registers such that Preemptable traffic and express traffic are not routed to the same Rx queue. As the queue mapping for tagged packets is based on VLAN user-priority field, this implies that priority of preemptable and express packets are mutually exclusive. In other words, packets of a certain priority (traffic class) are either express or preemptable but cannot be both.

For the preemptable frames, in addition to the PSRQ (Priority Selected in Rx queue) based queue routing, a programmable "Frame Preemption Residue Queue" (FPRQ) is supported to route all other Preemption Packets received (Untagged, SA/ DA or VLAN Filter Fails forwarded due to RA being set or VTFE being reset).

Table 650. Preemption Control Values on MRI Interface for Various Frame Types

Preemption Packet Type	Queue Routing
AV Tagged Packets passing the SA/DA and VLAN filters	PSRQ*
DCB/Generic Tagged Packets passing the SA/DA and VLAN filters	PSRQ**
Tagged Packets failing the SA/DA and VLAN filters without setting the Receive All (RA = 0) or setting VTFE = 1	Dropped
All other packets OR when RA = 1	FPRQ

76.9.13.5.1.3 MTL Receive Arbitration

On the ARI Interface, data is fetched from the MTL Rx FIFO based on the arbitration selected in the MTL_Operation_Mode register. Frame based arbitration can be used only when all the MTL Preemption Queues are operating in Store and Forward mode, else there will be loss of bandwidth on the ARI interface because all the fragments of a preemptable packet is not available. Therefore, any express packets received between the fragments are blocked until all the fragments are received and transferred; this defeats the purpose of express traffic.

When operating in either Threshold (cut through) or Store and forward modes of operation, PBL based arbitration is recommended over Frame based arbitration. In case of PBL based arbitration, the watermark check is always performed and the arbitration/ transfer of data in terms of chunks of "PBL" size of data which is almost similar to the concepts of "fragments". Therefore the express queue packets get blocked for less time as well as the ARI interface transfers the data without loss of efficiency.

76.9.13.6 Verify and Respond mPackets

When FPE function is present, the MAC can receive and detect the Verify and Respond mPackets, even when FPE is not enabled by software. When MAC detects valid Verify/Respond mPackets, it notifies the software by setting the RVER and RRSP fields of MAC_FPE_CTRL_STS register respectively. Optionally an interrupt can be generated. As such packets have empty (all-zero) data payload, they are dropped inside the MAC and not forwarded to the MRI.

Software can set the SVER and SRSP fields of MAC_FPE_CTRL_STS register to request MAC to transmit Verify and Respond mPackets respectively. Upon successful transmission of these frames, MAC clears the SVER/SRSP bits and sets the TVER & TRSP fields of MAC_FPE_CTRL_STS register. Optionally an interrupt can be generated when these events occur.

Software must ensure that frame preemption verify and response packet are not triggered at the same time. Trigger either of the packets and then wait for its completion before triggering another packet.

76.9.13.7 Frame Preemption and MMC Counter and Interrupt Registers

The following MMC counters and associated Interrupt registers are instantiated/present in the MAC when Frame Preemption feature is enabled.

Table 651. MMC Counters and Associated Interrupt Registers

Frame Assembly Error Counter	Description	Associated MMC Counter
Frame Assembly Error Counter	A 32-bit counter that provides the number of MAC frames with reassembly errors on the Receiver, due to mismatch in the Fragment Count value.	MMC_Rx_Packet_Assembly_Err_Cntr
Frame SMD Error Counter	A 32-bit counter that provides the number of received MAC frames rejected due to arriving with an SMD-C when there was no preceding preempted frame.	MMC_Rx_Packet_SMD_Err_Cntr
Frame Assembly OK Counter	A 32-bit counter that provides the number of MAC frames that were successfully reassembled.	MMC_Rx_Packet_Assembly_Ok_Cntr
MAC Rx Fragment Counter	A 32-bit counter that provides the number of additional mPackets received due to preemption.	MMC_Rx_FPE_Fragment_Cntr
MAC Tx Fragment Counter	A 32-bit counter that provides the number of additional mPackets transmitted due to preemption.	MMC_Tx_FPE_Fragment_Cntr
Hold Request Counter	A 32-bit counter that maintains the count of number of times a hold request is given to MAC.	MMC_Tx_Hold_Req_Cntr

76.9.13.7.1 Additional Registers Associated With MMC Interrupts

Following are the additional registers associated with MMC interrupts for the MMC error counters:

- MMC_FPE_Tx_Interrupt
- MMC_FPE_Tx_Interrupt_Mask
- MMC_Rx_Interrupt
- MMC_FPE_Rx_Interrupt_Mask

76.9.14 Time-Based Scheduling

Time-based scheduling feature is suitable for traffic whose periodicity and rate are predictable. To improve the quality-of-service of such traffic,

- The transmit DMA fetches the packet from the host memory for transmission at designated time. This helps the software to setup the Transmit descriptors in advance even before packet is ready/available. It reduces the overhead on the software and avoids constant monitoring of the time and preparing descriptors just in time when the packet is targeted to be transmitted.
- The MAC transmits the packet only at the designated/pre-determined time even if the packets are fetched in advance. This helps in maintaining a constant transmission rate that can be consumed by the receiver station; therefore avoiding congestion and excessive buffering in the network.

The time-based scheduling feature supports fetching and launching an Ethernet packet at (or after) a pre-determined time. The time-based scheduling is supported only in the following modes/configurations:

- Full duplex mode
- Link speed is 100Mbps or higher

The following figure shows the time-based scheduling flow.

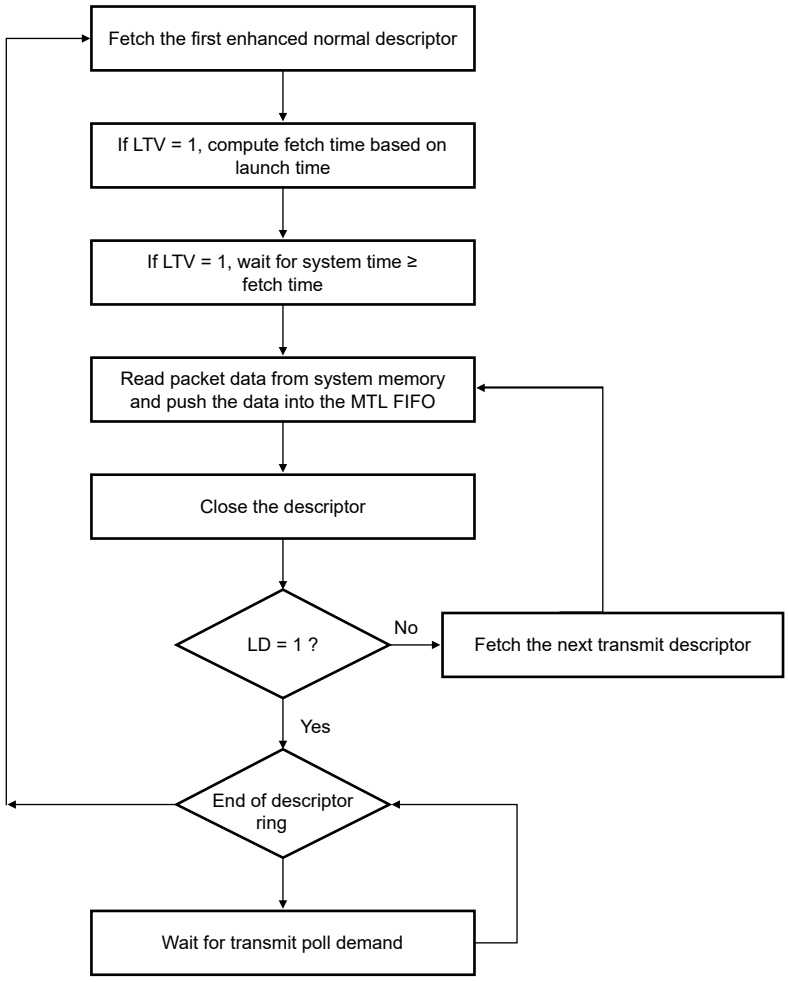


Figure 472. Time-Based Scheduling Flow

Following are the definitions specific to the time-based scheduling feature:

76.9.14.1 Launch time

The time beyond which MTL can schedule the packet for transmission. The launch time resets after the transmission of one frame. These are the two formats of the launch time.

Normal or absolute format:

In this format, the launch time is interpreted in the normal or absolute format if the ESTM bit of the MTL_TBS_CTRL register is set to 0.

The launch time is a 32-bit value, where most-significant 8-bits represent the time in seconds and the rest 24-bits represent the time in 256 ns. The launch time is compared against the IEEE 1588 based system or PTP time (bits[39:8]) and rolls over after 256 seconds.

The maximum value of the lower 32-bits of system time is 999,999,999 decimal (0x3B9AC9FF) and it wraps to 0 when reaching this value (representing a full second). Therefore, the maximum value of the lower 24-bits of the launch time (after multiplying by 256) must be 0x3B9AC9.

As the maximum value of launch time is 256 seconds,

- Launch time is greater than current system time when its value is between system time[39:8] and System time[39:8] + 128 sec.
- Launch time is less than current system time when its value is between System Time[39:8] + 128sec and System Time[39:8] + 256 sec, because this is a modulo 256 computation.

EST or offset format

In this format, the launch time is a offset value relative to the time which the BTR of the GCL list provided in the GCL slot number (GSN) indicates. The value in the BTR is always updated to the start time of the current loop of the GCL.

For each packet, the GSN value and the launch time value are specified in the enhanced transmit descriptor in the DMA configurations, or as control word in the MTL configurations.

The launch time offset is a 32-bit value; the upper 8-bits represent the time in seconds and the rest 24 bits represent the time in 256 ns, which is added to the BTR value corresponding to GCL slot number. The value of the launch time offset must be smaller than the value of the cycle time (specified in the CTR register that is implemented when EST is enabled). If the CTR is greater than or equal to 1s, the maximum value of the lower-24 bits of launch time offset must be 999,999,999 decimal (0x3B9AC9FF).

In this format, the launch time is a 64-bit value, which is interpreted as, Launch time = Launch GSN BTR[63:0] + (Launch time offset[31:0] << 8), which is compared with the system time [63:0].

GSN BTR is the Base Time value at which the Launch GSN loop started execution.

GCL slot number (GSN) A modulo 16 count of the GCL loop count is implemented and it is known as GSN. The count is incremented for every new GCL loop. Installation of new GCL list does not impact the count. Read MTL_EST_Status[CGSN] to obtain the current GCL slot number.

The maximum value of GSN is 15. Therefore the GSN values between MTL_EST_Status[CGSN] and MTL_EST_Status[CGSN]+8 represent current or future slots and all other GSN values are interpreted as elapsed slots or past slots. So, for the correct interpretation of time, GSN value must be between MTL_EST_Status[CGSN] and MTL_EST_Status[CGSN]+8.

76.9.14.2 Launch expiry time

Normal or Absolute Mode

In this mode, when MTL_TBS_CTRL[LEOV] = 1, MTL_TBS_CTRL[LEOS] determines the maximum amount of time a frame is eligible for launch, starting from the time the frame becomes eligible for launch.

The launch expiry offset is a 24-bit value defined in 256 ns units, with a maximum possible value of 999,999,999 ns (0x3B9AC9FF).

Launch expiry time = (Launch time[39:8] + LEOS[32:8]) * 256 ns

The packet with a specific launch time is considered as eligible for transmission when the launch time is less than the system time and (if MTL_TBS_CTRL[LEOV] = 1) the system time is less than the launch expiry time.

When the system time is greater than the launch expiry time, the frame is categorized as expired and it drops from the MTL FIFO.

NOTE

- For correct interpretation and meaningful operation, the fetch, launch, and launch expiry time must never set to a value larger than the Current system time + 128 sec; such a value is interpreted as time that has already elapsed.
- In Full Duplex mode, the frames dropped from the MTL FIFO have error summary (Bit 15) and excessive deferral (bit 3) of TxStatus set.

EST/Offset Mode

In EST mode, when MTL_TBS_CTRL[LEOV] = 1, MTL_TBS_CTRL[LEGOS] and MTL_TBS_CTRL[LEOS] determines the maximum amount of time a frame remains eligible for launch, starting from the time the frame becomes eligible for launch.

Launch Expiry Offset = (LEGOS: LEOS)

LEGOS holds the GSN offset (multiples of CTR time) and LEOS holds maximum value of CTR (sub CTR values) in ns.

Launch expiry offset is a 24-bit value defined in the units of 256 ns, with a maximum possible value of the smaller of 999,999,999 ns or CTR-1 ns.

The Launch expiry GSN is computed as follows:

Launch expiry GSN = (Launch GSN + LEGOS + CCMA) where, CCMA is the CTR carry due to modulo addition. This value is 1 if ((Launch time offset + LEOS) << 8) is equal to or greater than CTR.

When CCMA = 1, Launch expiry offset = (Launch time offset+ LEOS) - CTR

When CCMA = 0,

Launch expiry offset = (Launch time offset + LEOS).

Launch expiry time = Launch expiry GSN BTR [63:0] + Launch expiry time offset.

When MTL_TBS_CTRL[LEOV] is not 1, the launch expiry time is not checked.

When MTL_TBS_CTRL[LEOV] = 1, and

- the system time is greater than the launch expiry time, the frame drops from MTL FIFO. Then the frame is considered as expired.
- the launch time is smaller than the system time and the launch expiry time is greater than the system time, then the frame is considered eligible for launch.

NOTE

- Max value of MTL_TBS_CTRL[LEGOS] is 7. This indicates that when MTL_TBS_CTRL[LEOV] = 1, the frame has a maximum life time of <8 GCL loop iterations after it becomes eligible for launch.
- The slot number of the first GCL list that executes each time after EST is enabled, is zero.

76.9.14.3 Fetch time

This accounts for all possible delays in the DMA fetch operation and ensures that the frame is present in the MTL FIFO before the launch time.

If DMA_TBS_CTRL[FTOV] is 1, it indicates the fetch time for each packet. If DMA_TBS_CTRL[FTOV] is not 1, it indicates that the fetch time offset is not valid and DMA fetches packets without any time constraints.

Normal/Absolute mode

In this mode fetch time is derived or calculated by reducing the time specified in DMA_TBS_CTRL[FTOS] from the given launch time.

In Normal mode, the fetch launch time is computed as:

Fetch Time[39:8] = (Launch Time[39:8] - FTOS[31:8])

The fetch time is 32-bits and is compared against the system time[39:8] to determine whether it is eligible for fetching the frame:

- The fetch time is defined as greater than system time if the fetch time is in the range of system time[39:8]" and system time[39:8] + 128 sec. The frame is considered as not-eligible for fetch.
- The fetch time is defined as smaller than the system time if the fetch time is in the range of system time[39:8] + 128 sec and system time[39:8] + 256 sec. The frame is considered as eligible for fetch.

This is a modulo 256 computation.

EST/Offset mode

In this mode, the fetch GSN offset (DMA_TBS_CTRL[FGOS]) provides the slot number offset to deduct from the launch GSN. In this case the FTOS value must be smaller than 999,999,999 ns or CTR-1 ns.

If (Launch time offset >= FTOS):

Fetch time offset = ((Launch time offset - FTOS) * 256ns)

CBFS(CTR borrow for fetch subtraction) = 0

If (Launch time offset < FTOS) Fetch time offset = CTR + ((Launch time offset - FTOS) * 256ns)

CBFS (CTR borrow for fetch subtraction) = 1

The fetch GSN is computed as follows:

Fetch GSN = Launch GSN - FGOS - CBFS

Fetch time = Fetch GSN BTR[63:0] + Fetch time offset

The frame is eligible for DMA fetch when the fetch time is smaller than the system time.

DMA operations sequence

This is the sequence of operation when FTOV = 1:

1. Fetch the first enhanced normal descriptor (FD = 1).
2. Compute the fetch time on the basis of the launch time and wait for the system time to be greater than fetch time, if LTV = 1 and fetch is enabled.
3. Read the frame (data) from the host memory and transfer to MTL FIFO.
4. Close the normal descriptor.
5. Fetch the next normal descriptor (if the previous descriptor was not the last).
6. Repeat steps 4 to 6, until the last descriptor of the frame (LD = 1).

After the last descriptor of a frame, program the next enhanced normal descriptor with a new launch time and with LTV = 1. Otherwise, process the subsequent frames without any time restrictions.

See Programming the launch time in time-based scheduling for more information.

76.10 TCP/IP Offloading Features

This section and all other sections, along with respective sub-sections are Synopsys Proprietary. Used with permission.

76.10.1 Transmit Checksum Offload Engine

The MAC has a Checksum Offload Engine (COE) to support checksum calculation and insertion in the Transmit path, using which, the software can offload the task of checksum insertion to the hardware. In the transmit path MAC calculates the checksum and inserts it in the Tx packet. This feature helps in reducing the load on the software and can improve the overall throughput of the system. This feature is supported for only Q0 queue.

The checksum offload engine module supports two types of checksum calculation and insertion. The checksum engine can be controlled for each packet by setting the CIC bits (TDES3 Bits[17:16]).

Note: The checksum for TCP, UDP, or ICMP is calculated over a complete packet, and then inserted into its corresponding header field. Because of this requirement, when this function is enabled, the Tx FIFO automatically operates in the store-and-forward mode even if IP is configured for Threshold (cut-through) mode.

76.10.1.1 IP Header Checksum Engine

In IPv4 datagrams, the integrity of the header fields is indicated by the 16-bit Header Checksum field (the eleventh and twelfth bytes of the IPv4 datagram). The COE detects an IPv4 datagram when the Type field of Ethernet packet has the value 0x0800 and the Version field of IP datagram has the value 0x4. The checksum field of the input packet is ignored during calculation and replaced with the calculated value.

NOTE

IPv6 headers do not have a checksum field. Therefore, the COE does not modify the IPv6 header fields.

The result of this IP header checksum calculation is indicated by the IP Header Error status bit in the Transmit status (Bit 0 in Table 19-12 on page 1330). This status bit is set whenever the values of the Ethernet Type field and the Version field of IP header are not consistent, or when the Ethernet packet does not have enough data, as indicated by the IP header Length field. In other words, this bit is set when an IP header error is asserted under the following circumstances:

- For IPv4 datagrams:
  - The received Ethernet type is 0x0800, but the Version field of IP header is not equal to 0x4.
  - The IPv4 Header Length field indicates a value less than 0x5 (20 bytes).
  - The total packet length is less than the value given in the IPv4 Header Length field.
- For IPv6 datagrams:
  - The Ethernet type is 0x86dd but the IP header Version field is not equal to 0x6.
  - The packet ends before the IPv6 header (40 bytes) or extension header (as given in the corresponding Header Length field in an extension header) is completely received.

76.10.1.2 TCP/UDP/ICMP Checksum Engine

The TCP/UDP/ICMP Checksum Engine processes the IPv4 or IPv6 header (including extension headers) and determines whether the encapsulated payload is TCP, UDP, or ICMP. The checksum is calculated for the TCP, UDP, or ICMP payload and inserted into its corresponding field in the header. The Tx COE can work in the following two modes:

- The TCP, UDP, or ICMPv6 pseudo-header is not included in the checksum calculation and is assumed to be present in the Checksum field of the input packet. This engine includes the Checksum field in the checksum calculation, and then replaces the Checksum field with the final calculated checksum.
- The engine ignores the Checksum field, includes the TCP, UDP, or ICMPv6 pseudo-header data into the checksum calculation, and overwrites the checksum field with the final calculated value.

NOTE

For ICMP-over-IPv4 packets, the Checksum field in the ICMP packet must always be 16'h0000 in both modes, because pseudo-headers are not defined for such packets. If it does not equal 16’h0000, an incorrect checksum may be inserted into the packet.

The result of this operation is indicated by the Payload Checksum Error status bit in the Transmit Status vector (Bit 12 in Table 19-3 on page 1324). This engine sets the Payload Checksum Error status bit when it detects that the packet has been forwarded to the MAC Transmitter engine in the store-and-forward mode without the end of packet (EOP) being written to the FIFO, or when the packet ends before the number of bytes indicated by the Payload Length field in the IP Header is received. When the packet is longer than the indicated payload length, the COE ignores them as stuff bytes, and no error is reported. When this engine detects the first type of error, it does not modify the TCP, UDP, or ICMP header. For the second error type, it still inserts the calculated checksum into the corresponding header field. Following table describes the functions supported by Transmit Checksum Offload engine based on the packet type. When the MAC does not insert the checksum, it is indicated as "No" in the table.

NOTE

You should not enable checksum insertion for IPv4 or IPv6 packets that are greater than the frame size constraint specified in Description of Transmit Checksum Offload Engine because it may result in incorrect checksum insertion or unexpected behavior.

Table 652. Transmit Checksum Offload Engine Functions for Different Packet Types

Packet Type	Hardware IP headerchecksum insertion	Hardware TCP/ UDPchecksum insertion
Non-IPv4 or IPv6 packet	No	No
IPv4 packet is greater than 1,522 bytes (2,000 bytes when IEEE 802.3ad Support for 2K Packets is enabled in MAC) but less than or equal to the frame size constraint specified in Transmit Checksum Offload Engine.	Yes	Yes
IPv6 packet is greater than 1,522 bytes (2,000 bytes when IEEE 802.3ad Support for 2K Packets is enabled in MAC) but less than or equal to the frame size constraint specified in Transmit Checksum Offload Engine.	Not Applicable	Yes
IPv4 with TCP, UDP, or ICMP	Yes	Yes
IPv4 packet has IP options (IP header is longer than 20 bytes)	Yes	Yes
Packet is an IPv4 fragment	Yes	No
IPv6 packet with the following next header fields in main or extension headers: • Hop-by-hop options (in IPv6 main header) • Hop-by-hop options (in IPv6 extension header) • Destinations options • Routing (with segment left 0) • Routing (with segment left > 0) • TCP, UDP, or ICMP • Authentication • Any other next header field in main or extension headers	• Not Applicable • Not Applicable • Not Applicable • Not Applicable • Not Applicable • Not Applicable • Not Applicable • Not Applicable	• Yes • No • Yes • No • No • Yes • No • No
IPv4 packet has TCP header with Options fields	Yes	Yes
IPv4 Tunnels: • IPv4 packet in an IPv4 tunnel • IPv6 packet in an IPv4 tunnel	• Yes (IPv4 tunnel header) • Yes (IPv4 tunnel header)	• No • No
IPv6 Tunnels: • IPv4 packet in an IPv6 tunnel • IPv6 packet in an IPv6 tunnel	• Not applicable • Not applicable	• No • No

Table continues on the next page...

Table 652. Transmit Checksum Offload Engine Functions for Different Packet Types (continued)

Packet Type	Hardware IP headerchecksum insertion	Hardware TCP/ UDPchecksum insertion
IPv4 packet has 802.3ac tag (with C-VLAN Tag or S-VLAN Tag when enabled).	Yes	Yes
IPv6 packet has 802.3ac tag (with C-VLAN Tag or S-VLAN Tag when enabled).	Not applicable	Yes
IPv4 frames with security features (such as encapsulated security payload)	Yes	No
IPv6 frames with security features (such as encapsulated security payload)	Not applicable	No

76.10.2 Receive Checksum Offload Engine

IP provides the Checksum Offload Engine that is used to detect any error in an IPv4 or IPv6 packet in the receive path. The MAC verifies the checksum field of the received packet with the internally calculated checksum and provides the status.

The Receive Checksum Offload engine is used to detect errors in IP packets by calculating the header checksum and further matching it with the received header checksum. This engine also identifies a TCP, UDP, or ICMP payload in received IP packets and calculates the checksum of such payloads appropriately.

Here, both IPv4 and IPv6 packet in the received Ethernet packets are detected and processed for data integrity. The MAC receiver identifies IPv4 or IPv6 packets by checking for value 0x0800 or 0x86DD, respectively, in the Type field of the received Ethernet packet. This identification is applicable to single VLAN-tagged packets. It is also applicable to double VLAN-tagged packets when the Enable Double VLAN Processing option is selected and the EDVLP bit of the MAC_VLAN_Tag register is set.

The Rx COE calculates the IPv4 header checksums and checks that they match the received IPv4 header checksums. The result of this operation (pass or fail) is given to the RFC module for insertion into the receive status word. The IP Header Error bit is set for any mismatch between the indicated payload type (Ethernet Type field) and the IP header version, or when the received packet does not have enough bytes, as indicated by the Length field of the IPv4 header (or when fewer than 20 bytes are available in an IPv4 or IPv6 header).

This engine also identifies a TCP, UDP, or ICMP payload in the received IP datagrams (IPv4 or IPv6) and calculates the checksum of such payloads properly, as defined in the TCP, UDP, or ICMP specifications. This engine includes the TCP, UDP, or ICMPv6 pseudo-header bytes for checksum calculation and checks whether the received checksum field matches the calculated value. The result of this operation is given as a Payload Checksum Error bit in the receive status word. This status bit is also set if the length of the TCP, UDP, or ICMP payload does not match the expected payload length given in the IP header.

Following table describes the functions supported by the Rx COE based on the packet type. When the payload of an IP packet is not processed (indicated as "No" in the table), the information (whether the checksum engine is bypassed or not) is given in the receive status.

NOTE

The MAC does not append any payload checksum bytes to the received Ethernet packets.

Table 653. Receive Checksum Offload Engine Functions for Different Packet Types

Packet type	Hardware IP header checksum checking	Hardware TCP/UDP/ICMP checksum checking
Non-IPv4 or IPv6	No	No

Table continues on the next page...

Table 653. Receive Checksum Offload Engine Functions for Different Packet Types (continued)

Packet type	Hardware IP header checksum checking	Hardware TCP/UDP/ICMP checksum checking
IPv4 packet is greater than 1,522 bytes (2,000 bytes when IEEE 802.3ad Support for 2K Packets is enabled in the MAC)	Yes	Yes
IPv6 packet is greater than 1,522 bytes (2,000 bytes when IEEE 802.3ad Support for 2K Packets is enabled in the MAC)	Not Applicable	Yes
IPv4 with TCP, UDP, or ICMP	Yes	Yes
IPv4 header's protocol field contains a protocol other than TCP, UDP, or ICMP	Yes	No
IPv4 packet has IP options (IP header is longer than 20 bytes)	Yes	Yes
Packet is an IPv4 fragment	Yes	No
IPv6 packet with the following next header fields in main or extension headers: • Hop-by-hop options (in IPv6 main header) • Hop-by-hop options (in IPv6 extension header) • Destinations options • Routing (with segment left 0) • Routing (with segment left > 0) • TCP, UDP, or ICMP • Any other next header field in main or extension headers	• Not Applicable • Not Applicable • Not Applicable • Not Applicable • Not Applicable • Not Applicable • Not Applicable	• Yes • No • Yes • Yes • No • Yes • No
IPv4 packet has TCP header with Options fields	Yes	Yes
IPv4 Tunnels: • IPv4 packet in an IPv4 tunnel • IPv6 packet in an IPv4 tunnel	• Yes (IPv4 tunnel header) • Yes (IPv4 tunnel header)	• No • No
IPv4 Tunnels:	• Not Applicable • Not Applicable	• No • No

Table continues on the next page...

Table 653. Receive Checksum Offload Engine Functions for Different Packet Types (continued)

Packet type	Hardware IP header checksum checking	Hardware TCP/UDP/ICMP checksum checking
• IPv4 packet in an IPv6 tunnel • IPv6 packet in an IPv6 tunnel		
IPv4 packet has 802.3ac tag (with C-VLAN Tag or S-VLAN Tag when enabled)	Yes	Yes
IPv6 packet has 802.3ac tag (with C-VLAN Tag or S-VLAN Tag when enabled)	Not Applicable	Yes
IPv4 frames with security features (such as encapsulated security payload)	Yes	No
IPv6 frames with security features (such as encapsulated security payload)	Not Applicable	No

Set the IPC bit of the MAC_Configuration register for enabling Receive Checksum offload.

76.10.3 Header payload split

MAC is capable of identifying the boundary between header and payload of the received packet and store them into separate buffers. The MAC supports multiple methods and packet types for header-payload splitting. This is controlled by the setting the SPLM field in MAC_Ext1_cfg register.

76.10.4 IPv4 ARP offload engine

MAC supports the Address Recognition Protocol (ARP) Offload for IPv4 packets. This feature allows the processing of the IPv4 ARP request packet in the receive path and generating corresponding ARP response packet in the transmit path. MAC generates the ARP reply packets for appropriate ARP request packets. The ARP packet for IPv4 is L2 layer packet with Length/Type of 0x0806.

NOTE

When the received ARP request is less than 64 bytes packet length, MAC does not send a ARP response. It is treated as normal packet and forwarded to the application based on the MAC filter settings.

76.11 MAC Management Counters

This section is Synopsys Proprietary. Used with permission.

IP supports storing the statistics about the received and transmitted packets in registers that are accessible through the application/software. The MMC module maintains a set of registers for gathering statistics on the received and transmitted packets. The MMC counters are free running.

For MMC register details, see registers "MMC_Control" to "MMC_Rx_FPE_Fragment_Cntr" in GMAC register descriptions.

76.12 Flow Control

This section and its sub-sections are Synopsys Proprietary. Used with permission.

76.12.1 Transmit Flow Control

The Transmit Flow Control involves transmitting Pause packets in full-duplex mode and backpressure in half-duplex mode to control the flow of packets from the remote end.

76.12.1.1 Flow control in Full duplex mode

IP supports the IEEE802.3x Pause Packets.

Pause packet structure is shown in the following table.

Table 654. Pause Packet Fields

Field	Description
DA	Contains the special multicast address
SA	Contains the MAC address 0
Type	Contains 8808
MAC Control opcode	Contains 0001 for IEEE 802.3x Pause Control packets; 0101 for PFC packets
PT	Contains Pause time specified in the PT field of the MAC_Q#_Tx_Flow_Ctrl register

76.12.1.2 Pause Packet Control

When the FCB bit is set, the MAC generates and transmits a single Pause packet. If the FCB bit is set again after the Pause packet transmission is complete, the MAC sends another Pause packet irrespective of whether the pause time is complete or not. To extend the pause or terminate the pause prior to the time specified in the previously-transmitted Pause packet, the application should program the Pause Time register with appropriate value and then again set the FCB bit.

76.12.1.3 Flow Control in Half-Duplex Mode

In half-duplex mode, the MAC uses the deferral mechanism for the flow control (backpressure). When the application requests to stop receiving packets, the MAC sends a JAM pattern of 32 bytes when it senses a packet reception, provided the transmit flow control is enabled. This results in a collision and the remote station backs off. If the application requests a packet to be transmitted, it is scheduled and transmitted even when the backpressure is activated. If the backpressure is kept activated for a long time (and more than 16 consecutive collision events occur), the remote stations abort the transmission because of excessive collisions.

Following table describes the flow control in the Tx path for Queue 0 based on the setting of the following bits:

- EHFC bit of MTL_RxQ0_Operation_Mode register
- TFE bit of MAC_Q0_Tx_Flow_Ctrl register
- DM bit of MAC_Configuration register

Flow control is similar for all queues.

Table 655. Tx MAC Flow Control

EHFC	TFE	DM	Description
x	0	x	The MAC transmitter does not perform the flow control or backpressure operation.
0	1	0	The MAC transmitter performs back-pressure when Bit 0 of MAC_Q0_Tx_Flow_Ctrl register is set or the sideband signal sbd_flowctrl_i is 1.

Table continues on the next page...

Table 655. Tx MAC Flow Control (continued)

EHFC	TFE	DM	Description
1	1	0	The MAC transmitter performs back-pressure when Bit 0 of MAC_Q0_Tx_Flow_Ctrl register is set or the sideband signal sbd_flowctrl_i is 1. In addition, the MAC Tx performs back-pressure when Rx Queue level crosses the threshold set by Bits[10:8] of MTL_RxQ0_Operation_Mode register.
0	1	1	The MAC transmitter sends the Pause packet when Bit 0 of MAC_Q0_Tx_Flow_Ctrl register is set or the sideband signal sbd_flowctrl_i is 1.
1	1	1	The MAC transmitter sends the Pause packet when Bit 0 of MAC_Q0_Tx_Flow_Ctrl register is set or the sideband signal sbd_flowctrl_i is 1. In addition, the MAC Tx sends a Pause packet when Rx Queue level crosses the threshold set by Bits[10:8] of MTL_RxQ0_Operation_Mode register.

76.12.1.4 Enabling Transmit Flow Control

To independently enable Transmit Flow Control for each Tx queue, set the TFE bit in the MAC_Q#_Tx_Flow_Ctrl register.

76.12.2 Receive Flow Control

In the Receive path, the Flow Control is functional only in the full-duplex mode. If any Pause packet is received in the half-duplex mode, the packet is considered as a normal control packet.

NOTE

Receive pause packets should have a frame size of 64 bytes.

The Receive Flow Control is implemented by the MAC based on the bit value of the respective register, and the destination address and different fields of the received packet.

Following table describes the flow control in the Rx path based on the setting of the following bits:

- RFE bit of MAC_Rx_Flow_Ctrl register
- DM bit of MAC_Configuration register

Table 656. Rx MAC Flow Control

RFE	DM	Description
0	x	The MAC receiver does not detect the received Pause packets.
1	0	The MAC receiver does not detect the received Pause packets but recognizes such packets as Control packets.
1	1	The MAC receiver detects or processes the Pause packets and responds to such packets by stopping the MAC transmitter.

76.12.2.1 Enabling Receive Flow Control

To enable Pause Flow Control, set the RFE bit in the MAC_Rx_Flow_Ctrl register.

76.13 Loopback Mode

This section is Synopsys Proprietary. Used with permission.

The MAC supports Loopback of transmitted packets to its receiver.

The following are some guidelines for using the loopback mode:

- Enable loopback only with the full-duplex mode. In half-duplex mode, the carrier sense signal (crs) or collision (col) signal inputs get sampled which may result into issues such as packet dropping.
- If the loopback mode is enabled without connecting a PHY chip (for example, in FPGA setup), you should externally generate the Tx and Rx clocks and provide these clocks to the MAC.
- Do not loop back big packets. Big packets (>1522 bytes) may get corrupted in the loopback FIFO.

To enable this feature, program the LM bit of the MAC_Configuration register.

You can enable loopback for all PHY interfaces. The data is always looped back through internal asynchronous FIFO on to the internal Receive MII or GMII interface, irrespective of which PHY interface is selected. The loopback data is also passed through the corresponding transmit PHY interface block, on to the Ethernet line.

76.14 Automotive Safety Features

This section and all its sub-sections are Synopsys Proprietary. Used with permission.

IP supports the automotive safety feature.

76.14.1 Error Correction Code (ECC) Protection for Memories

The Error Correction Code (ECC) block can correct single-bit error and detect double-bit error.

At the write interface, ECC checkbits are generated by computing ECC on the contents of the data bus and respective address is appended with the data that is written to the memory.

At the read interface, ECC checkbits are recomputed on the content of the read data and the respective address is compared with the received checkbits in the memory.

Following figure shows the block diagram of ECC protection for memories

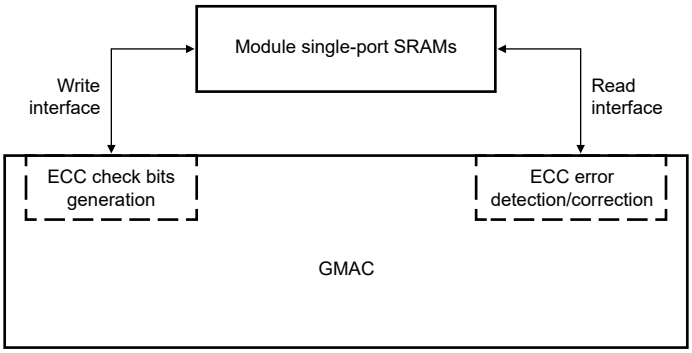


Figure 473. Block Diagram of ECC Protection for Memories

76.14.1.1 Handling the Address Mismatch

The ECC is calculated over the sum of memory data and memory location address. However only data is written in the memory along with the ECC(address is excluded).

On the read interface, the ECC is checked over the sum of memory DATA, memory ECC, and Internally generated address. When an ECC correctable error is detected over the range of the address bit position, it is treated as an uncorrectable error. This is because data might have been read from a different location.

76.14.1.2 Diagnostic Support for the Error Management

Following statistics are provided for monitoring the error behavior on each of the memory blocks.

- Error status provided to management
  - A separate status for correctable, uncorrectable, and address mismatch is specified in the MTL_EC-C_Interrupt_Status register.
  - Memory locations at which correctable and uncorrectable errors are detected is specified in the MTL_ECC_Err_Addr_Status register. In addition, a control bit (MEEAO field of MTL_ECC_Control register) decides, whether the first errored memory address is reported or the latest errored memory address is reported.
  - MTL_ECC_Err_Cntr_Status register has separate counters to count the number of correctable and uncorrectable errors
- Interrupts provided to management
  - Separate interrupts are generated for correctable and uncorrectable errors.
  - sbd_sfty_ue_intr_o interrupt is generated when uncorrectable errors are detected and these errors cannot be masked.
  - sbd_sfty_ce_intr_o interrupt is generated when correctable errors are detected. The module sets the respective interrupt enable bits (in the DMA/MTL_ECC_Interrupt_enable register).
  - To find the root cause of the error, read the DMA/MTL_Safety_Interrupt_Status and DMA/MTL_ECC_Interrupt_Status registers.
  - To clear the interrupt, write 1 in the respective interrupt status bit in DMA/MTL_ECC_Inter-rupt_Status registers.

NOTE

- The status/counters/interrupts are generated per memory.
- MTL Tx and Rx memory is logically divided into multiple queues. But, ECC diagnostics (status/counter/interrupts) are common to all queues.

76.14.1.3 ECC Error Injection Capabilities

IP supports error injection capabilities for each memory. The position where the errors are injected in the data word or address is based on EIM/BLEI field of MTL_DPP_ECC_EIC register and 3 control bits of the MTL_DBG_CTRL/MTL_EST_GCL_Control register. For each memory, 4 control bits and 8 error locations bits are provided to inject errors:

- 1 bit to enable error injection (EIEE/ESTEIEE field of MTL_DBG_CTRL/MTL_EST_GCL_Control register)
- 1 bit to enable address error injection (EIAEE/ESTEIAEE field of MTL_DBG_CTRL/MTL_EST_GCL_Control register)
- 1 bit to enable data or ECC-bit error injection (EIM field of MTL_DPP_ECC_EIC register)
- 8 bits to indicate the bit location of error injection (BLEI field of MTL_DPP_ECC_EIC register)
- 1 bit to indicate the type of error to be injected (EIEC/ESTEIEC field of the MTL_DBG_CTRL/MTL_EST_GCL_Control register)
  - 0: insert 1-bit error
  - 1: insert 2-bit error

For the 2-bit error injection, both BLEI and EIEC/ESTEIEC fields are used for the various error injection types; ECC bits, Data bits, and Address bits. The control bit descriptions for

- MTL Tx/Rx and DMA TSO memory are specified in the MTL_DBG_CTL register.
- MTL EST memory is specified in the MTL_EST_GCL_Control register.
- Rx parser memory is specified in the MTL_RXP_Indirect_Acc_Control_Status register.
- Descriptor fetch memory control is specified in the MTL_DBG_CTL register.

NOTE

While using debug mode for ECC error injection,

- There should be no traffic in the IP
- When multiple CSR writes are required for writing single data word into the memory, the application should ensure that all the CSR writes corresponding to one memory write maintains the same value for the error injection control word.

76.14.2 Finite State Machine(FSM) Parity and Timeout Protection

The FSM protection feature supports FSM parity and time out protection.

76.14.2.1 FSM State Parity Protection

Odd parity is implemented on all the FSM state register bits. Parity is monitored for every clock, after the reset is de-asserted. When a bit flips due to transient errors or permanent faults, the erroneous FSM is set to its default state and an uncorrectable error is indicated.

When the FSM state parity protection is enabled, by setting PRTYEN field of MAC_FSM_Control Register, the FSM state error is indicated by setting the FSMPES field of the MAC_DPP_FSM_Interrupt_Status register. Also, the Safety Interrupt (sbd_sfty_ue_intr_o) is asserted when an FSM Error is detected.

Error Injection mode is also supported for FSM parity error check. Program the Error Injection enable for the respective clock domain denoted by [23:16] bits of MAC_FSM_Control Register.

76.14.2.2 FSM Timeout Protection

The FSM Timeout feature provides a mechanism to ensure that all FSMs in the module can complete a transaction and return to a known completion state (IDLE or any other state that indicates completion of a transaction/transfer) within the programmed timeout value.

Program the TMR field of MAC_FSM_ACT_Timer register with a value that indicates the number of CSR cycles required to generate a 1 microsecond tic. This microsecond tic is internally used to generate the programmed timeout duration. Two timeout tics (normal mode timeout and large mode timeouts) are generated to provide flexibility to choose one per clock domain, by using the bits [31:24] of MAC_FSM_Control Register. Supported values for Timeout duration are 1us, 1.024ms, 16.384ms, 65.5536ms, 262.144ms, 1048.576ms (~1sec), 4194.304ms (~4s), 8.388s, 16.777s, and 33.554s which is based on the programmable bits NTMRMD, LTMRMD fields of MAC_FSM_ACT_Timer register. Interface timeouts are based on the normal mode tic generation; only the FSM timeouts depend on either normal or large tick selection.

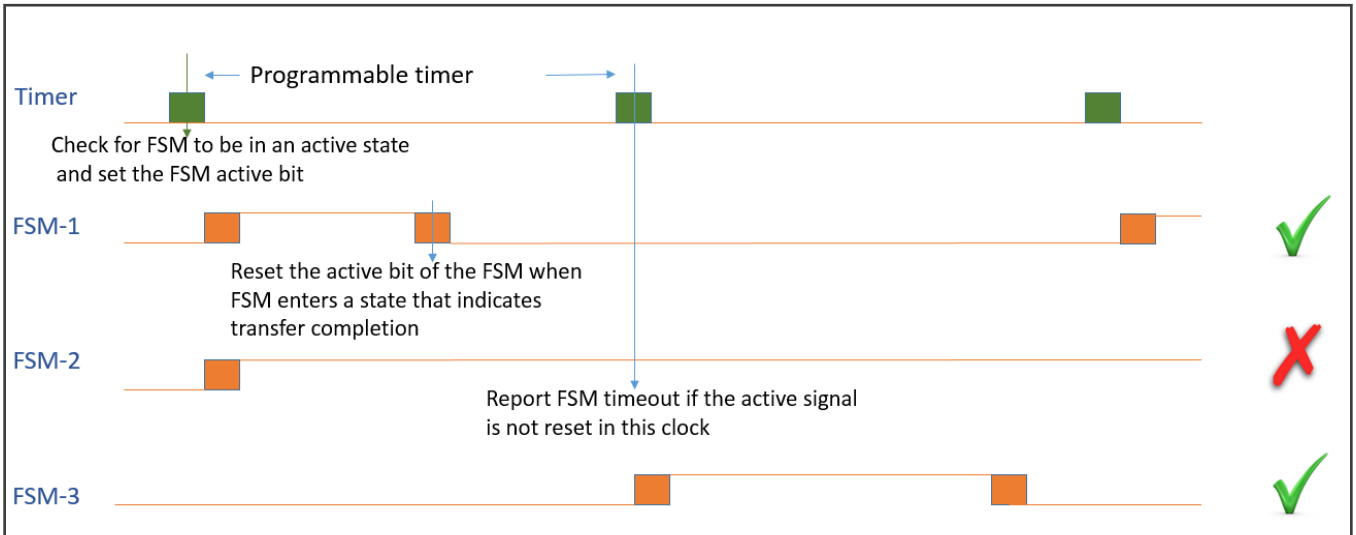


Figure 474. FSM Timeout Timing Diagram

In the above figure, the timer ticks are generated based on the programmable timeout value. The timer ticks are synchronized and made available in all the clock domains that have the FSMs.

At every timer tick, all the FSMs are monitored for being in active state (non-IDLE state, which indicates the FSM is actively processing transactions or hand-shakes) and an FSM active flag bit is set. The active flag bit is implemented for every FSM that is monitored for timeout. When the FSM reaches an IDLE or transaction completion state, the active flag bit of that FSM is reset. At the subsequent timer tick, all the FSMs' active flag bits are monitored and any flag that is set indicates a timeout for that FSM. This process of setting the flag and checking at subsequent timer tick is repeated at every tick.

Timeout Error Injection per clock domain is enabled by programming [15:8] fields of MAC_FSM_Control register. When Timeout error injection enable is set for a clock domain, the FSMs in that clock domain automatically timeout and generate an interrupt even without traffic. As error injection is a debug mode, TMR value need not indicate 1us, but can be programmed to a smaller value at which debug mode ticks are needed. FSM Timeout and Parity Error Injection modes can be used for testing at key-on/key-off.

The FSM timeout status is specified in the [15:8] field of MAC_DPP_FSM_Interrupt_Status register as per the respective clock domains. One FSM timeout status bit is made available per clock domain. The safety interrupt (sbd_sfty_ue_intr_o) is asserted when the timeout error is set for any of the clock domains. IP does not attempt to recover from a FSM timeout condition and relies on the application to take the corrective action (resetting IP).

76.14.2.3 Enabling FSM Parity and Timeout Protection

FSM timeout feature is enabled by setting the TMOUTEN field of MAC_FSM_Control register.

NOTE

See Programming guidelines for FSM parity and timeout

76.14.3 Application/CSR Interface Timeout Protection

This feature provides timeout protection to the application/CSR Interface. All the interfaces which has the handshake mechanism monitored for potential hangs due to the external agent (Master/Slave/Interconnect/Application client) not responding to the requests/transfers initiated by the QoS. After the request is initiated the response arrival interval is monitored. If the response does not arrive within a programmed time (TMR field of MAC_FSM_ACT_Timer) the timeout is triggered, using similar trigger generation as explained in the Section FSM protection.

The timeout is triggered when IP initiates SEQ, NON-SEQ and BUSY transfers on HTRANS[1:0]; and HREADY is not asserted by Slave within the programmed time.

The Timeout status for the AHB master interface is set in the MSTTES field of the MAC_DPP_FSM_Interrupt_Status register. The Safety interrupt (sbd_sfty_ue_intr_o) is asserted when application timeout status is set.

The hardware does not attempt to recover from Application/CSR Interface hangs and relies on software to take appropriate corrective action.

NOTE

- Protection is not needed for APB3 Slave interfaces because potential hangs can be only when IP does not respond to the requests. In such cases the IP internal protection logic (FSM timeout) detects such defects.
- Based on the time at which the transfer is initiated relative to the timer ticks, the timeout period could be any value between the programmed timeout and 2x times the programmed timeout.
- When an uncorrectable safety interrupt is issued and the read of the Interrupt status register returns all zeros, it implies that the CSR read is not functional. As soft reset of IP is not possible without the CSR access, and therefore, a hard reset of IP is recommended.
- See Programming guidelines for FSM parity and timeout .

76.15 Descriptors

This section and all its sub-sections are Synopsys Proprietary. Used with permission.

76.15.1 Overview

The DMA in the Ethernet subsystem transfers data based on a linked list of descriptors. The application creates the descriptors in the system memory. The module supports the following two types of descriptors:

- Normal Descriptor: Normal descriptors are used for packet data and to provide control information applicable to the packets to be transmitted.
- Context Descriptor: Context descriptors are used to provide control information applicable to the packet to be transmitted.

Each normal descriptor contains two buffers and two address pointers. These buffers enable the adapter port to be compatible with various types of memory management schemes.

NOTE

There is no limit for the number of descriptors that can be used for a single packet.

76.15.2 Descriptor Structure

The module supports the ring structure for DMA descriptor as shown in the following figure.

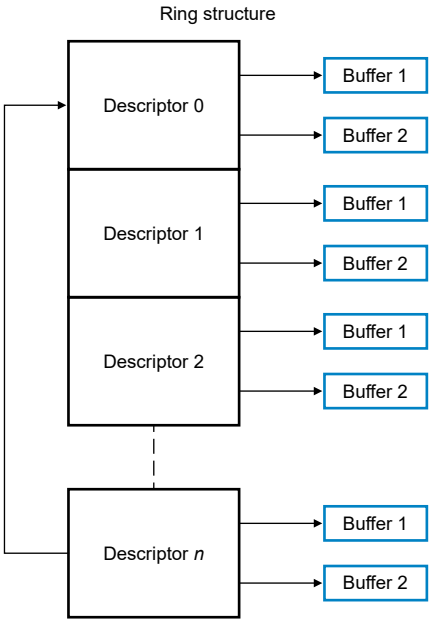


Figure 475. Descriptor Ring Structure

In Ring structure, descriptors are separated by the Word, DWord, or LWord number programmed in the DSL field of the DMA_CH#_Control register. The application needs to program the total ring length, that is, the total number of descriptors in ring span in the following registers of a DMA channel:

- Transmit Descriptor Ring Length Register (DMA_CH#_TxDesc_Ring_Length)
- Receive Descriptor Ring Length Register (DMA_CH#_RxDesc_Ring_Length)

The Descriptor Tail Pointer Register contains the pointer to the descriptor address (N). The base address and the current descriptor pointer decide the address of the current descriptor that the DMA can process. The descriptors up to one location less than the one indicated by the descriptor tail pointer (N – 1) are owned by the DMA. The DMA continues to process the descriptors until the following condition occurs:

Current Descriptor Pointer == Descriptor Tail Pointer;

The DMA goes into the Suspend mode when this condition occurs. The application must perform a write to the Descriptor Tail pointer register and update the tail pointer so that the following condition is true:

Current Descriptor Pointer < Descriptor Tail Pointer;

The DMA automatically wraps around the base address when the end of ring is reached, as shown in the following figure.

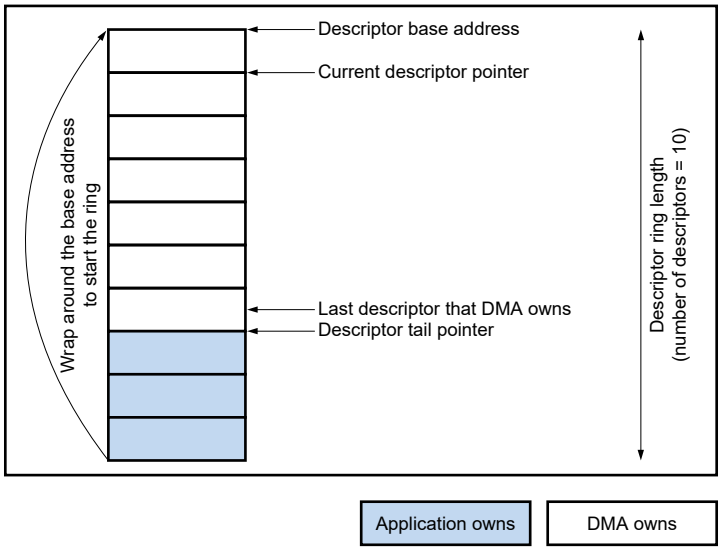


Figure 476. DMA Descriptor Ring

For descriptors owned by the application, the OWN bit of DES3 is reset to 0. For descriptors owned by the DMA, the OWN bit is set to 1. If the application has only one descriptor in the beginning, the application sets the last descriptor address (tail pointer) to Descriptor Base Address + 1. The DMA processes the first descriptor and then waits for the application to advance the tail pointer.

76.15.3 Transmit Descriptor

The DMA in the module requires at least one descriptor for a transmit packet. In addition to two buffers, two byte-count buffers, and two address pointers, the transmit descriptor has control fields which can be used to control the MAC operation on per-transmit packet basis. The Transmit Normal descriptor has two formats: Read format and Write-Back format

NOTE

TSO Split header is not supported.

Software must not write same tail pointer addresses when DMA is in Suspended state before setting new descriptor(s) for further processing.

76.15.3.1 Transmit Normal Descriptor (Read Format)

Following figure shows the Read Format for a Transmit normal descriptor. Table: 'TDES0 Normal Descriptor (Read Format)' through Table: 'TDES3 Normal Descriptor (Read Format)' describe the read format for the Transmit Normal Descriptors: TDES0, TDES1, TDES2, and TDES3.

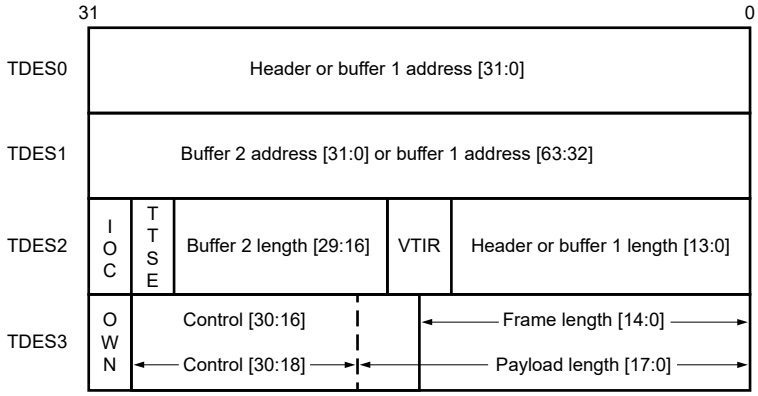


Figure 477. Transmit Descriptor Read Format

76.15.3.1.1 TDES0 Normal Descriptor (Read Format)

Table 657. TDES0 Normal Descriptor (Read Format)

Bit	Name	Description
31:0	BUF1AP	Buffer 1 Address Pointer or TSO Header Address Pointer These bits indicate the physical address of Buffer 1. These bits indicate the TSO Header Address pointer when the following bits are set: • TSE bit of TDES3 • FD bit of TDES3

76.15.3.1.2 TDES1 Normal Descriptor (Read Format)

Table 658. TDES1 Normal Descriptor (Read Format)

Bit	Name	Description
31:0	BUF2AP	Buffer 2 or Buffer 1 Address Pointer This bit indicates the physical address of Buffer 2 when a descriptor ring structure is used. There is no limitation for the buffer address alignment. In 40- or 48-bit addressing mode, these bits indicate the most-significant 8- or 16- bits of the Buffer 1 Address Pointer.

76.15.3.1.3 TDES2 Normal Descriptor (Read Format)

Table 659. TDES2 Normal Descriptor (Read Format)

31	30	29-16	15:14	13:0
IOC	TTSE/ TMWD	B2L	VTIR	HL or B1L

Table 660. TDES2 Normal Descriptor (Read Format)

Bits	Name	Description
31	IOC	Interrupt on Completion This bit controls the setting of TI and ETI status bits in the DMA_CH#_Status register. When ETIC = 1 and TDES2[LD] = 0, this bit sets the ETI bit. When TDES3[LD] = 1, this bit sets the TI status bit.
30	TTSE/ TMWD	Transmit Timestamp Enable or External TSO Memory Write Enable This bit enables the IEEE1588 time stamping for Transmit packet referenced by the descriptor, if TSE bit is not set. If TSE bit is set and external TSO memory is enabled, setting this bit disables external TSO memory writing for this packet.
29:16	B2L	Buffer 2 Length The driver sets this field. When set, this field indicates Buffer 2 length.
15:14	VTIR	VLAN Tag Insertion or Replacement These bits request the MAC to perform VLAN tagging or untagging before transmitting the packets. The application must set the CRC Pad Control bits appropriately when VLAN Tag Insertion, Replacement, or Deletion is enabled for the packet. The following list describes the values of these bits: • 2'b00: Do not add a VLAN tag. • 2'b01: Remove the VLAN tag from the packets before transmission. This option should be used only with the VLAN packets. • 2'b10: Insert a VLAN tag with the tag value programmed in the MAC_VLAN_Incl register or context descriptor. • 2'b11: Replace the VLAN tag in packets with the tag value programmed in the MAC_VLAN_Incl register or context descriptor. This option should be used only with the VLAN packets. These bits are valid when the Enable SA and VLAN Insertion on Tx option is selected while configuring the core.
13:0	HL or B1L	Header Length or Buffer 1 Length For Header length only bits [9:0] are taken. The size 13:0 is applicable only when interpreting buffer 1 length. If the TCP Segmentation Offload feature is enabled through the TSE bit of TDES3, this field is equal to the header length. When the TSE bit is set in TDES3, the header length includes the length in bytes from Ethernet Source address till the end of the TCP header. The maximum header length supported for TSO feature is 1023 bytes. The maximum header length supported for TSO feature is 1023 bytes. If the TCP Segmentation Offload feature is not enabled, this field is equal to Buffer 1 length.

76.15.3.1.4 TDES3 Normal Descriptor (Read Format)

Table 661. TDES2 Normal Descriptor (Read Format)

31	30	29	28	27:26	25:23	22:19	18	17:16	15	14:0
OWN	CTXT	FD	LD	CPC	SAIC	SLOTNUM or THL	TSE	CIC/TPL	TPL	FL/TPL

Table 662. TDES3 Normal Descriptor (Read Format)

Bits	Name	Description
31	OWN	Own Bit When this bit is set, it indicates that the DMA owns the descriptor. When this bit is reset, it indicates that the application owns the descriptor. The DMA clears this bit after it completes the transfer of data given in the associated buffer(s).
30	CTXT	Context Type This bit should be set to 1'b0 for normal descriptor.
29	FD	First Descriptor When this bit is set, it indicates that the buffer contains the first segment of a packet.
28	LD	Last Descriptor When this bit is set, it indicates that the buffer contains the last segment of the packet. When this bit is set, the B1L or B2L field should have a non-zero value.
27:26	CPC	CRC Pad Control This field controls the CRC and Pad Insertion for Tx packet. This field is valid only when the first descriptor bit (TDES3[29]) is set. The following list describes the values of Bits[27:26]: • 2'b00: CRC and Pad Insertion The MAC appends the cyclic redundancy check (CRC) at the end of the transmitted packet of length greater than or equal to 60 bytes. The MAC automatically appends padding and CRC to a packet with length less than 60 bytes. • 2'b01: CRC Insertion (Disable Pad Insertion) The MAC appends the CRC at the end of the transmitted packet but it does not append padding. The application should ensure that the padding bytes are present in the packet being transferred from the Transmit Buffer, that is, the packet being transferred from the Transmit Buffer is of length greater than or equal to 60 bytes. • 2'b10: Disable CRC Insertion The MAC does not append the CRC at the end of the transmitted packet. The application should ensure that the padding and CRC bytes are present in the packet being transferred from the Transmit Buffer. • 2'b11: CRC Replacement The MAC replaces the last four bytes of the transmitted packet with recalculated CRC bytes. The application should ensure that the padding and CRC bytes are present in the packet being transferred from the Transmit Buffer. This field is valid only for the first descriptor. Note: When the TSE bit is set, the MAC ignores this field because the CRC and pad insertion is always done for segmentation.
25:23	SAIC	SA Insertion Control These bits request the MAC to add or replace the Source Address field in the Ethernet packet with the value given in the MAC Address 0 register. The application must set the CRC Pad Control bits appropriately when SA Insertion Control is enabled for the packet.

Table continues on the next page...

Table 662. TDES3 Normal Descriptor (Read Format) (continued)

Bits	Name	Description
		Bit 25 specifies the MAC Address Register (1 or 0) value that is used for Source Address insertion or replacement. The following list describes the values of Bits[24:23]: • 2'b00: Do not include the source address • 2'b01: Include or insert the source address. For reliable transmission, the application must provide frames without source addresses. • 2'b10: Replace the source address. For reliable transmission, the application must provide frames with source addresses. • 2'b11: Reserved These bits are valid in the EQOS-DMA, EQOS-AXI, and EQOS-AHB configurations when the Enable SA and VLAN Insertion on Tx option is selected while configuring the core and when the First Segment control bit (TDES3 [29]) is set. This field is valid only for the first descriptor.
22:19	SLOTNUM or THL	SLOTNUM: Slot Number Control Bits in AV Mode These bits indicate the slot interval in which the data should be fetched from the corresponding buffers addressed by TDES0 or TDES1. When the Transmit descriptor is fetched, the DMA compares the slot number value in this field with the slot interval maintained in the RSN field DMA_CH#_Slot_Function_Control_Status. It fetches the data from the buffers only if a value matches. These bits are valid only for the AV channels. THL: TCP/UDP Header Length If the TSE bit is set, this field contains the length of the TCP/UDP header. The minimum value of this field must be 5 for TCP header. The value must be equal to 2 for UDP header. This field is valid only for the first descriptor.
18	TSE	TSO Split header is not supported. The value of this bit is always zero.
17:16	CIC/TPL	Checksum Insertion Control or TCP Payload Length These bits control the checksum calculation and insertion. The following list describes the bit encoding: • 2'b00: Checksum Insertion Disabled. • 2'b01: Only IP header checksum calculation and insertion are enabled. • 2'b10: IP header checksum and payload checksum calculation and insertion are enabled, but pseudo-header checksum is not calculated in hardware. • 2'b11: IP Header checksum and payload checksum calculation and insertion are enabled, and pseudo-header checksum is calculated in hardware. This field is valid when the Enable Transmit TCP/IP Checksum Offload option is selected and the TSE bit is reset. When the TSE bit is set, this field contains the upper bits [17:16] of the TCP Payload (or IP Payload for UDP fragmentation). This allows the TCP/UDP packet length field to be spanned across TDES3[17:0] to provide 256 KB packet length support.

Table continues on the next page...

Table 662. TDES3 Normal Descriptor (Read Format) (continued)

Bits	Name	Description
		This field is valid only for the first descriptor.
15	TPL	Reserved or TCP Payload Length When the TSE bit is reset, this bit is reserved. When the TSE bit is set, this is Bit 15 of the TCP payload length [17:0]. This field is valid only when the Enable TCP Segmentation Offloading for TCP/IP Packets option is selected while configuring the core.
14:0	FL/TPL	Frame Length or TCP Payload Length This field is equal to the length of the packet to be transmitted in bytes. When the TSE bit is not set, this field is equal to the total length of the packet to be transmitted: Ethernet Header Length + TCP /IP Header Length – Preamble Length – SFD Length + Ethernet Payload Length When the TSE bit is set, this field is equal to the lower 15 bits of the TCP payload length in case of segmentation and IP payload in case of UDP fragmentation. In case of segmentation, this length does not include Ethernet header or TCP/UDP/IP header length. In case of fragmentation, this length does not include Ethernet header and IP header. When DWRR/WFQ algorithm is NOT enabled, value written into this field is not used when TSE = 0.

76.15.3.1.5 Transmit normal descriptor (write-back format)

The transmit descriptor write-back format includes timestamp low, timestamp high, OWN, and Status bits.

The write-back format is applicable only for the last descriptor of the corresponding packet. Write 1 to the LD bit (TDES3[28]) in the descriptor where DMA writes back the status and timestamp information for the corresponding transmit packet.

Figure 478 shows the transmit descriptor write-back format.

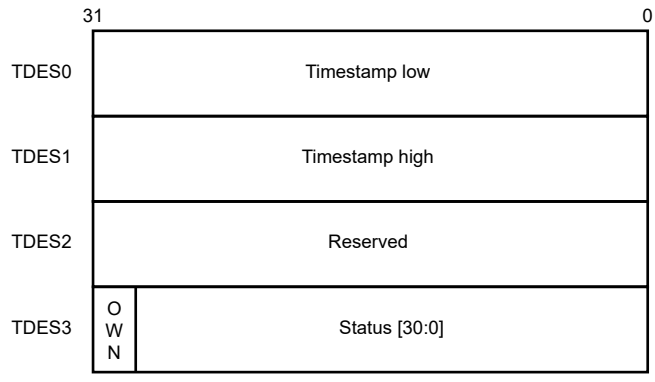


Figure 478. Transmit descriptor write-back format

TDES0 normal descriptor (write-back format)

Table 663 shows that this format is only applicable to the last descriptor of a packet.

Table 663. TDES0 normal descriptor (write-back format)TDES0 normal descriptor (write-back format)

Bit	Name	Description
31:0	TTSL	Transmit Packet Timestamp Low DMA updates this field with least significant 32 bits of the timestamp captured for the corresponding transmit packet. DMA writes the timestamp only if TTSE bit of TDES2 is 1 in the first descriptor of the packet. This field has the timestamp only if you write 1 to the Last Segment bit (LS) in the descriptor and the Timestamp status (TTSS) bit.

TDES1 normal descriptor (write-back format)

Table 664 shows that this format is only applicable to the last descriptor of a packet.

Table 664. TDES1 normal descriptor (write-back format)

Bit	Name	Description
31:0	TTSH	Transmit Packet Timestamp High DMA updates this field with the most significant 32 bits of the timestamp captured for a corresponding transmit packet. DMA writes the timestamp only if the TTSE bit of TDES2 is 1 in the first descriptor of the packet. This field has the timestamp only if you write 1 to the Last Segment bit (LS) in the descriptor and Timestamp status (TTSS) bit.

TDES2 Normal Descriptor (Write-Back Format)

This format is applicable only to the last descriptor of a packet.

Table 665. TDES2 normal descriptor (write-back format)

Bit	Description
31:0	Reserved

TDES3 Normal Descriptor (Write-Back Format)

This format is applicable only to the last descriptor of a packet.

Table 666. TDES3 normal descriptor (write-back format)

31	30	29	28	27:18	17	16	15	14	13	12	11	10	9	8	7:4	3	2	1	0
OWN	CTXT	FD	LD	Rsvd	TTSS	EUE	ES	JT	FF	PCE	LoC	NC	LC	EC	CC	ED	UF	DB	IHE

Table 667. TDES3 normal descriptor (write-back format)

Bit	Name	Description
31	OWN	Own Bit When this field is 1, it indicates that the module DMA owns the descriptor. DMA write 0 to this field when it completes the packet transmission. After the write-back is complete, this field is set to 'b0.
30	CTXT	Context Type

Table continues on the next page...

Table 667. TDES3 normal descriptor (write-back format) (continued)

Bit	Name	Description
		This field must be set to 'b0 for normal descriptor.
29	FD	First Descriptor This field indicates that the buffer contains the first segment of a packet.
28	LD	Last Descriptor This field is set to 'b1 for last descriptor of a packet. DMA writes the status fields only in the last descriptor of the packet.
27:24	Rsvd	Reserved
23	DE	Descriptor Error When this field is 1, it indicates that the descriptor content is incorrect. DMA writes 1 to this field during write-back when the descriptor is closed. Descriptor errors can be: • Incorrect sequence from the context descriptor. For example, a location after the first descriptor for a packet. • All 1s • CTXT is 1 along with LD or FD bits as 1. **NOTE** When a descriptor error occurs due to all ones or CTXT, LD, and FD bits are 1, the transmit DMA closes the transmit descriptor with DE and LD bits set to 1. When you write 1 to IOC bit in TDES2 of corresponding first descriptor, the transmit DMA write 1 to TI bit in the DMA_CH#_Status register **NOTE** Based on CTXT, LD, and FD bits of the transmit descriptor, the subsequent descriptor might consider as the first descriptor (even if FD bit is not 1) and partial packet is sent.
22:18	Rsvd	Reserved
17	TTSS	Tx Timestamp Status This field indicates that a timestamp has been captured for the corresponding transmit packet. When this field is 1, it indicates that TDES0 and TDES1 have timestamp values that were captured for the transmit packet. This field is valid only when the Last Segment control bit (TDES3 [28]) in a descriptor is 1. This field is valid only when IEEE1588 timestamping feature is enabled; otherwise, it is reserved.
16	EUE	ECC Uncorrectable Error Status Indicates the ECC uncorrectable error in the TSO memory. **NOTE** An uncorrectable error in the transmit FIFO memory is reported with (Bit 13) FF = 1. This is because the module flushes all such packets.

Table continues on the next page...

Table 667. TDES3 normal descriptor (write-back format) (continued)

Bit	Name	Description
15	ES	Error Summary This field indicates the logical OR of the following bits: • TDES3[0]: IP header error • TDES3[14]: Jabber timeout • TDES3[13]: Packet flush • TDES3[12]: Payload checksum error • TDES3[11]: Loss of carrier • TDES3[10]: No carrier • TDES3[9]: Late collision • TDES3[8]: Excessive collision • TDES3[3]: Excessive deferral • TDES3[2]: Underflow error This field is 1 when EUE (bit 16) = 1.
14	JT	Jabber Timeout This field indicates that the MAC transmitter experiences a jabber time-out. This field is 1 only when MAC_Configuration[JD] is not 1.
13	FF	Packet Flushed This field indicates that DMA or MTL flushes the packet because the CPU gives a software flush command.
12	PCE	Payload Checksum Error This field indicates that the checksum offload engine had a failure and does not insert any checksum into the encapsulated TCP, UDP, or ICMP payload. This failure can either because of insufficient bytes, as the payload length field of the IP header indicates or the MTL starting to forward the packet to the MAC transmitter in Store-and-forward mode without calculating the checksum. This second error condition only occurs when the transmit FIFO depth is less than the length of the Ethernet packet that is transmitted to avoid deadlock. The MTL starts forwarding the packet when the FIFO is full, even in the Store-and-forward mode. This error can also occur when you detects a bus error during packet transfer. When the full checksum offload engine is not enabled, this bit is reserved.
11	LoC	Loss of Carrier This field indicates that a loss of carrier occurred during packet transmission (that is, the gmii_crs_i signal was inactive for one or more transmit clock periods during packet transmission). This field is valid only for the packets transmitted without collision and when MAC operates in the Half-duplex mode.
10	NC	No Carrier This field indicates that the carrier sense signal from the PHY was not asserted during transmission.

Table continues on the next page...

Table 667. TDES3 normal descriptor (write-back format) (continued)

Bit	Name	Description
9	LC	Late Collision This bit indicates that packet transmission was aborted because a collision occurred after the collision window (64 byte times including Preamble in MII mode and 512 byte times including Preamble and Carrier Extension in GMII mode). This bit is not valid if Underflow Error is set.
8	EC	Excessive Collision This field indicates that the transmission was aborted after 16 successive collisions when you transmits the current packet. If MAC_Configuration[DR] = 1, this field is 1 after first collision and abort the packet transmission.
7:4	CC	Collision Count This 4-bit counter value indicates the number of collisions occurred before transmitting the packet. The count is not valid when the EC bit = 1.
3	ED	Excessive Deferral This field indicates that the transmission ends because of an excessive deferral of over 24,288 bit times (155,680 bits times in 1000 Mbit/s mode or Jumbo packet enabled mode) if MAC_Configuration[DC] = 1. When TBS is enabled in Full duplex mode and this field is 1, it indicates that the frame drops after the expiry time is reached.
2	UF	Underflow Error This field indicates that MAC aborts the packet because the data arrived late from the system memory. The underflow error can occur because of either of the following conditions: • DMA encounters an empty transmit buffer when it transmits the packet • The application filled the MTL Tx FIFO slower than the MAC transmit rate The transmission process enters the Suspended state and sets the underflow bit corresponding to a queue in MTL_Interrupt_Status.
1	DB	Deferred Bit This field indicates that MAC defers before transmitting because of presence of carrier. This field is valid only in the Half-Duplex mode.
0	IHE	IP Header Error When IP Header Error is 1, this field indicates that the checksum offload engine detects an IP header error. This field is valid only when transit checksum offload is enabled. Otherwise, it is reserved. If COE detects an IP header error, it inserts an IPv4 header checksum if the Ethernet type field indicates an IPv4 payload. In Full-Duplex mode, when EST/Qbv is enabled and this field is 1, it indicates the frame drop status due to the frame size error or schedule error.

76.15.3.2 Transmit context descriptor

The transmit context descriptor can provide any time before a packet descriptor. The context is valid for the current packet and subsequent packets. The context descriptor provide the timestamps for one-step timestamp correction and VLAN tag ID for VLAN insertion feature. Write back is done on a context descriptor only to write 0 to OWN bit.

NOTE

DMA internally store the VLAN tag IDs and MSS values which the application provides in a context descriptor with their corresponding valid bits set. DMA always passes the last valid VLAN tag to the MTL, when the outer or inner VLAN tag is provided with the valid bit set. The application cannot invalidate the valid VLAN tag which DMA stores. The VLAN tag is inserted or replaced based on the control inputs provided for the packet.

The inner VLAN tag control input is used only for the next packet that immediately follows the context descriptor. The application must provide a context descriptor before the normal descriptor of each packet for which DMA must use the inner VLAN tag control input.

Figure 479 shows the transmit context descriptor format.

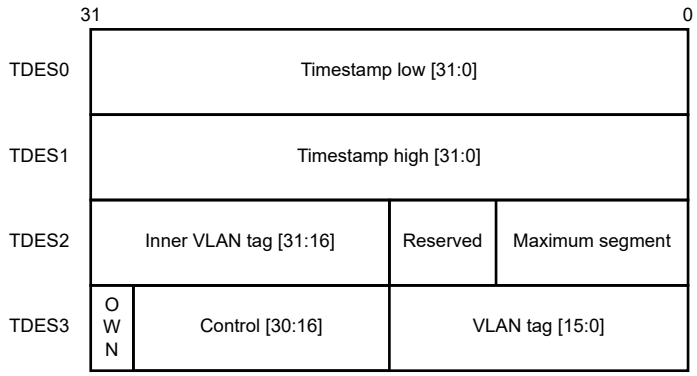


Figure 479. Transmit context descriptor format

76.15.3.2.1 TDES0 context descriptor

Table 668 describes the TDES0 context descriptor format.

Table 668. TDES0 context descriptor

Bit	Name	Description
31:0	TTSL	Transmit Packet Timestamp Low For one-step correction, the driver can provide the lower 32 bits of timestamp in this descriptor word. DMA uses this value as the low word for doing one-step timestamp correction. This field is valid only if the OSTC and TCMSSV bits of TDES3 context descriptor are 1.

76.15.3.2.2 TDES1 context descriptor

Table 669 describes the TDES1 context descriptor format.

Table 669. TDES1 context descriptor

Bit	Name	Description
31:0	TTSH	Transmit Packet Timestamp High For one-step correction, the driver can provide the upper 32 bits of timestamp in this descriptor. DMA uses this value as the high word for doing one-step timestamp correction. This field is valid only if the OSTC and TCMSSV bits of TDES3 context descriptor are 1.

76.15.3.2.3 TDES2 context descriptor

Table 670 shows the TDES2 context descriptor format.

Table 670. TDES2 context descriptor

Bit	Name	Description
31:16	IVT	Inner VLAN Tag When the IVLTV bit of TDES3 context descriptor is 1 and the TCMSSV and OSTC bits of TDES3 context descriptor becomes 0, it indicates that the TDES2[31:16] contains the inner VLAN tag to insert in the subsequent transmit packets.
15:14	Rsvd	Reserved
13:0	MSS	TSO split header is not supported. The value of this bit is always zero.

76.15.3.2.4 TDES3 context descriptor

Table 671. TDES3 context descriptor

31	30	29:28	27	26	25:24	23	22:18	17	16	15:0
OWN	CTXT	Rsvd	OSTC	TCMSSV	Rsvd	CDE	Rsvd	IVLTV	VLTV	VT

Table 672 shows the TDES3 context descriptor format.

Table 672. TDES3 context descriptor

Bit	Name	Description
31	OWN	Own Bit When this field is 1, it indicates that the module DMA owns the descriptor. When this field becomes 0, it indicates that the application owns the descriptor. DMA writes 0 to this field immediately after the read.
30	CTXT	Context Type This field must set to 1'b1 for context descriptor.
29:28	Rsvd	Reserved
27	OSTC	One-Step Timestamp Correction Enable When this field is 1, it indicates that DMA performs a one-step timestamp correction with reference to the timestamp values provided in TDES0 and TDES1.
26	TCMSSV	One-Step Timestamp Correction Input or MSS Valid When this field and the OSTC field are 1, it indicates that the timestamp correction input provided in TDES0 and TDES1 is valid. When the OSTC field becomes 0 and this field and the TSE bit of TDES3 are 1 in subsequent normal descriptor, it indicates that the MSS input in TDES2 is valid.
25:24	Rsvd	Reserved

Table continues on the next page...

Table 672. TDES3 context descriptor (continued)

Bit	Name	Description
23	DE	Descriptor Error When this field is 1, it indicates that the descriptor content is incorrect. DMA writes 1 to this field during write-back when it closes the context descriptor. Descriptor errors can be: • Incorrect sequence from the context descriptor. For example, a location before the first descriptor for a packet. • All ones. • CD, LD, and FD bits = 1. **NOTE** When a descriptor error occurs due to all ones or CTXT, LD, and FD bits are 1, the transmit DMA closes the transmit descriptor with DE and LD bits are 1. When you write 1 to IOC bit in TDES2 of corresponding first descriptor, the transmit DMA writes 1 to TI bit in the DMA_CH#_Status register **NOTE** On the basis of CTXT, LD, and FD bits of the transmit descriptor, the subsequent descriptor might considered as the first descriptor (even if FD bit is not 1) and partial packet is sent. **NOTE** This error is categorized as an abnormal event. Recovery from this event is possible by issuing a software reset only. DMA stopping, reconfiguring, and restarting recovery mechanism is not supported.
22:20	Rsvd	Reserved
19:18	IVTIR	Inner VLAN Tag Insert or Replace When this field is 1, it requests MAC to perform inner VLAN tagging or un-tagging before transmitting the packets. If the packet is modified for VLAN tags, MAC automatically recalculates and replaces the CRC bytes. The following list describes the values of these bits: • 2'b00: Do not add the inner VLAN tag. • 2'b01: Remove the inner VLAN tag from the packets before transmission. You must use this option only with the VLAN frames. • 2'b10: Insert an inner VLAN tag with the tag value programmed in MAC_Inner_VLAN_Incl or context descriptor. • 2'b11: Replace the inner VLAN tag in packets with the tag value programmed in MAC_Inner_VLAN_Incl or context descriptor. You must use this option only with the VLAN frames.. These fields are valid when you select the enable SA and VLAN insertion on transit and enable double VLAN processing options.
17	IVLTV	Inner VLAN Tag Valid When this field is 1, it indicates that IVT field of TDES2 is valid.

Table continues on the next page...

Table 672. TDES3 context descriptor (continued)

Bit	Name	Description
16	VLTV	VLAN Tag Valid When this field is 1, it indicates that VT field of TDES3 is valid.
15:0	VT	VLAN Tag Contains the VLAN tag to insert or replace in the packet. This field is used as VLAN tag only when MAC_VLAN_Incl[VLTI] becomes 0.

76.15.4 Receive descriptor

DMA in the module reads a descriptor only if the tail pointer is different from the base pointer or current pointer. It is recommended to have a descriptor ring with a length that accommodates at least two complete packets which MAC receives. Otherwise, the performance of DMA is impacted because of the unavailability of the descriptors. In such situations, the RxFIFO in MTL becomes full and starts dropping packets.

The following receive descriptors are present:

- Normal descriptors
- Context descriptors

You can prepare all the receive descriptors and give to DMA as normal descriptors with the content as shown in receive normal descriptor (read format). DMA reads this descriptor and the receive DMA closes the descriptor with the corresponding packet status after transferring a received packet (or part of) to the buffers indicated by the descriptor. The receive normal descriptor (write-back format) describes the format of this status.

For some packets, only the normal descriptor bits cannot write the complete status. For such packets, the receive DMA writes the extended status to the next descriptor (without processing or using the buffers or pointers embedded in that descriptor). Receive context descriptor describes the descriptor write back format and content.

76.15.4.1 Receive normal descriptor (read format)

The read format for a receive normal descriptor is made up of the following:

- A header or buffer 1 address
- Reserved field
- Payload or buffer 2 or next descriptor address
- A 30-bit reserved field
- OWN bit
- An interrupt field

Following figure shows the read format for a receive normal descriptor.

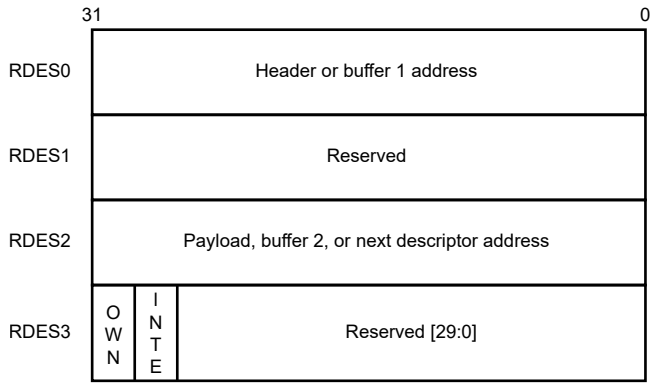


Figure 480. Receive normal descriptor read format

NOTE

In the receive descriptor (read format), if the buffer address field is all zeros, the module does not transfer data to that buffer and skips to the next buffer or next descriptor.

76.15.4.1.1 RDES0 normal descriptor (read format)

Table 673 shows the RDES0 normal descriptor read format.

Table 673. RDES0 normal descriptor (read format)

Bit	Name	Description
31:0	BUF1AP	Header or Buffer 1 Address Pointer When the SPH field of control register of a channel becomes 0, these field indicates the physical address of buffer 1. When the SPH field is 1, these field indicates the physical address of header buffer where the receive DMA writes the L2/L3/L4 header bytes of the received packet. The application can program a byte-aligned address for this buffer which means that the LS bits of this field can be non-zero. However, DMA performs a write operation with RDES0[1:0] (or RDES0[2:0]/[3:0] in case of 64-/128-bit configuration) as zero, when the start of packet is transferred. However, the packet data shifts per the actual offset which the buffer address pointer provides. If the address pointer points to a buffer where the middle or last part of the packet is stored, DMA ignores the offset address and writes to the full location which the data-width indicates .

76.15.4.1.2 RDES1 normal descriptor (read format)

Table 674 shows the RDES1 normal descriptor read format.

Table 674. RDES1 normal descriptor (read format)

Bit	Name	Description
31:0	Reserved or BUF1AP	Contains the most-significant 32 bits of the buffer 1 address pointer in 64-bit Addressing mode. Otherwise, this field is reserved.

76.15.4.1.3 RDES2 normal descriptor (read format)

Table 675 shows the RDES2 normal descriptor read format.

Table 675. RDES2 normal descriptor (read format)

Bit	Name	Description
31:0	BUF2AP	Buffer 2 Address Pointer Indicates the physical address of buffer 2. When the SPH bit of the DMA_CH#_Control register = 1, it indicates that the buffer address pointer must be bus width-aligned, that is, RDES2[3:0, 2:0, or 1:0] = 0 corresponding to 128, 64, or 32 bus width. LSBs are ignored internally. When the SPH bit of the DMA_CH#_Control register becomes 0, it indicates that there is no limitations on the RDES2 value. However, the RxDMA uses the LS fields of the pointer address only when it transfers the start bytes of a packet. If the BUF2AP provides the address of a buffer in which the middle or last part of a packet is stored, DMA ignores BUF2AP[3:0 or 2:0 or 1:0] (corresponding to 128- or 64- or 32-bit data-bus) and writes to the complete location.

76.15.4.1.4 RDES3 normal descriptor (read format)

Following table describes the RDES3 normal descriptor read format.

31	30	19-26	25	24	23:0
OWN	IOC	Rsvd	BUF2V	BUF1V	Rsvd

Table 676. RDES3 normal descriptor (read format)

Bit	Name	Description
31	OWN	Own Bit When this field is 1, it indicates that the module DMA owns the descriptor. When this field becomes 0, it indicates that the application owns the descriptor. DMA write 0 this field when either of the following conditions is true: • DMA completes the packet reception. • The buffers associated with the descriptor are full.
30	IOC	Interrupt Enabled on Completion When this field is 1, it indicates that an interrupt is issued to the application when DMA closes this descriptor.
29:26	Rsvd	Reserved
25	BUF2V	Buffer 2 Address Valid When this field is 1, it indicates to the DMA that the buffer 2 address specified in RDES2 is valid. The application must write 1 to this field so that DMA can use the address, to which the buffer 2 address in RDES2 points to write received packet data.
24	BUF1V	Buffer 1 Address Valid When this field is 1, it indicates to the DMA that the buffer 1 address specified in RDES1 is valid.

Table continues on the next page...

Table 676. RDES3 normal descriptor (read format) (continued)

Bit	Name	Description
		The application must write 1 to this field, if DMA uses the address to which the buffer 1 address in RDES1 points to write the received packet data.
23:0	Rsvd	Reserved

76.15.4.1.5 Receive normal descriptor (write-back format)

Following figure shows the write-back format for a receive normal descriptor.

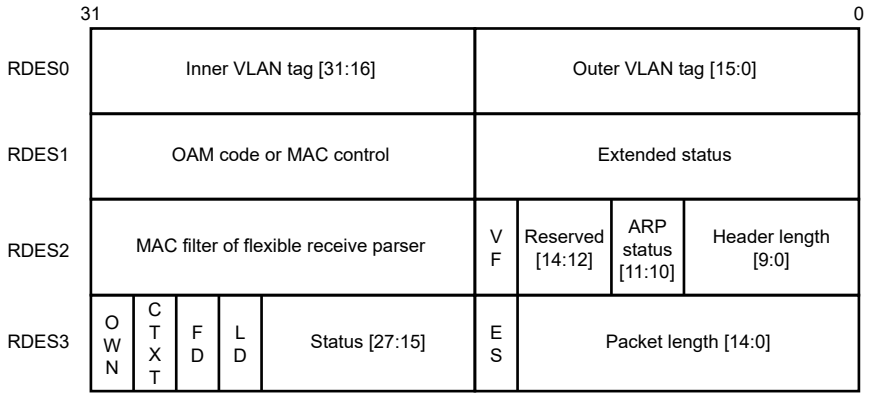


Figure 481. Receive normal descriptor (write-back format)

NOTE

When you enables the flexible receive parser, RDES2[31:16] indicates the parser status, and not the MAC filter status. The MAC filter status is not available when flexible receive parser is enabled.

76.15.4.1.5.1 RDES0 normal descriptor (write-back format)

Table 677 shows the write-back format for the RDES0 normal descriptor.

Table 677. RDES0 normal descriptor (write-back format)

Bit	Name	Description
31:16	IVT	Inner VLAN Tag Contains the inner VLAN tag of the received packet if the RS0V bit of RDES3 = 1. This field is valid only when you enables the double VLAN tag processing and VLAN tag stripping.
15:0	OVT	Outer VLAN Tag Contains the outer VLAN tag of the received packet if the RS0V bit of RDES3 = 1.

76.15.4.1.5.2 RDES1 normal descriptor (write-back format)

The status fields in the write-back format are valid only for the last descriptor (RDES3[28] = 1). Following table provide the details of the write-back format for RDES1 normal descriptor.

31:16	15	14	13	12	11:8	7	6	5	4	3	2:0
OPC	TD	TSA	PV	PFT	PMT	IPCE	IPCB	IPV6	IPV4	IPHE	PT

Table 678. RDES1 normal descriptor (write-back format)

Bit	Name	Description
31:16	OPC	Indicates any one of the following: • OAM sub-type code: If bits [18:16] of RDES3 as described in RDES3 normal descriptor (read format) are 111b, this field contains the OAM sub-type and code fields. • MAC control packet opcode: Bits 15:8 of RDES3 as described in RDES3 normal descriptor (read format) contain the subtype and bits 7:0 contain the code.
15	TD	Timestamp Dropped Indicates that the timestamp was captured for this packet but it dropped in the MTL Rx FIFO because of overflow. This field is available only when you select the timestamp feature. Otherwise, this field is reserved.
14	TSA	Timestamp Available Indicates that the timestamp value is available in a context descriptor word 2 (RDES2) and word 1(RDES1) when timestamp is present. This is valid only when the last descriptor field (RDES3 [28]) = 1 You can write the context descriptor in the next descriptor just after the last normal descriptor for a packet.
13	PV	PTP Version Indicates that the received PTP message has the IEEE 1588 version 2 format. When this field becomes 0, it indicates the IEEE 1588 version 1 format. This field is available only when you select the timestamp feature. Otherwise, this field is reserved.
12	PFT	PTP Packet Type Indicates that the PTP message is sent directly over Ethernet. This field is available only when you select the timestamp feature. Otherwise, this field is reserved.
11:8	PMT	PTP Message Type Encodes to give the type of the message received: • 0000: No PTP message received • 0001: SYNC (all clock types) • 0010: Follow_Up (all clock types) • 0011: Delay_Req (all clock types) • 0100: Delay_Resp (all clock types) • 0101: Pdelay_Req (in peer-to-peer transparent clock) • 0110: Pdelay_Resp (in peer-to-peer transparent clock) • 0111: Pdelay_Resp_Follow_Up (in peer-to-peer transparent clock)

Table continues on the next page...

Table 678. RDES1 normal descriptor (write-back format) (continued)

Bit	Name	Description
		• 1000: Announce • 1001: Management • 1010: Signaling • 1011–1110: Reserved • 1111: PTP packet with reserved message type These fields are available only when you select the timestamp feature.
7	IPCE	IP Payload Error When this field is 1, it indicates either of the following: • The 16-bit IP payload checksum (that is, the TCP, UDP, or ICMP checksum) which MAC calculates does not match the corresponding checksum field in the received segment. • The TCP, UDP, or ICMP segment length does not match the payload length value in the IP header field. • The TCP, UDP, or ICMP segment length is less than minimum allowed segment length for TCP, UDP, or ICMP. Bit 15 (ES) of RDES3 is not set when this field is 1.
6	IPCB	IP Checksum Bypassed Indicates that the checksum offload engine is bypassed. This field is available when you select the enable receive TCP/IP checksum check feature.
5	IPV6	IPv6 header Present Indicates that an IPV6 header is detected. When you select the enable split header feature option and the SPH bit of control register of a channel is 1, the IPV6 header is available in the header buffer area to which RDES0 points.
4	IPV4	IPV4 Header Present Indicates that an IPV4 header is detected. When the SPH bit of RDES3 is 1, the IPV4 header is available in the header buffer area to which RDES0 points.
3	IPHE	IP Header Error When this field is 1, it indicates either of the following: • The 16-bit IPv4 header checksum which MAC calculates does not match the received checksum bytes. • The IP datagram version is not consistent with the Ethernet type value. • Ethernet packet does not have the expected number of IP header bytes. This field is valid when either bit 5 or bit 4 is 1. This field is available when you select the enable receive TCP/IP checksum check feature.
2:0	PT	Payload Type Indicates the type of payload encapsulated in the IP datagram that the receive checksum offload engine (COE) process.

Table continues on the next page...

Table 678. RDES1 normal descriptor (write-back format) (continued)

Bit	Name	Description
		• 3'b000: Unknown type or IP/AV payload not processed • 3'b001: UDP • 3'b010: TCP • 3'b011: ICMP • 3'b110: AV tagged data packet • 3'b111: AV tagged control packet • 3'b101: AV untagged control packet • 3'b100: IGMP, if IPV4 header present field = 1, else DCB (LLDP) control packet If the COE does not process the payload of an IP datagram because there is an IP header error or fragmented IP, it sets these fields to 3'b000.

76.15.4.1.5.3 RDES2 normal descriptor (Write-back format)

31:29	28	27	26:19	18	17	16	15	14	13:11	10	9:0
L3L4FM	L4FM	L3FM	MADRM	HF	DAF	SAF	OTS	ITS	Rsvd	ARPNR	HL

Table 679. RDES2 normal descriptor (Write-back format)

Bit	Name	Description
31:29	L3L4FM	Layer 3 and Layer 4 Filter Number Matched Indicates the number of the layer 3 and layer 4 filter that matches the received packet: • 000: Filter 0 • 001: Filter 1 • 010: Filter 2 • 011: Filter 3 • 100: Filter 4 • 101: Filter 5 • 110: Filter 6 • 111: Filter 7 This field is valid only when bit 28 or bit 27 = 1. When more than one filter matches, these fields provide the number of lowest filter. **NOTE** This status is not available when Flexible RX Parser is enabled.
28	L4FM	Layer 4 Filter Match

Table continues on the next page...

Table 679. RDES2 normal descriptor (Write-back format) (continued)

Bit	Name	Description
		When this field is 1, it indicates that the received packet matches one of the enabled layer 4 port number fields. This status is given only when one of these conditions is true: • Layer 3 fields are not enabled and all enabled layer 4 fields match • All enabled layer 3 and layer 4 filter fields match When more than one filter matches, this bit gives the layer 4 filter status of filter indicated by Bits[31:29]. **NOTE** This status is not available when the flexible receive parser is enabled.
27	L3FM	Layer 3 Filter Match When this field is 1, it indicates that the received packet matches one of the enabled layer 3 IP address fields. This status is given only when one of these conditions is true: • All enabled layer 3 fields match and bypasses all enabled layer 4 fields. • All enabled filter fields match. When more than one filter matches, this field gives the layer 3 filter status of filter which bits[31:29] indicate. **NOTE** This status is not available when the flexible receive parser is enabled.
26:19	MADRM	MAC Address Match or Hash Value When HF becomes 0, it indicates that this field contains the MAC address register number that matches the received packet destination address. This field is valid only if the DAF field becomes 0. When HF = 1, it indicates that this field contains the hash value that MAC computes. A packet passes the hash filter when the field corresponding to the hash value is 1 in the hash filter register. **NOTE** This status is not available when the flexible receive parser is enabled.
18	HF	Hash Filter Status When this field is 1, it indicates that the packet passes the MAC address hash filter. Bits[26:19] indicate the hash value. **NOTE** This status is not available when the flexible receive parser is enabled.
17	DAF/RXPI	Destination Address Filter Fail When flexible receive parser is disabled, and this field is 1, it indicates that the packet fails the DA filter in MAC. When flexible receive parser is enabled, and this field is 1, it indicates that the packet parsing is incomplete (RXPI) due to ECC error.

Table continues on the next page...

Table 679. RDES2 normal descriptor (Write-back format) (continued)

Bit	Name	Description
		NOTE When this field is 1, RDES3 ES field is also 1.
16	SAF/RXPD	SA Address Filter Fail When flexible receive parser is disabled, and this field is 1, it indicates that the packet fails the SA filter in MAC. When flexible receive parser is enabled and this field is 1, it indicates that the parser drops the RXPD packet. NOTE When this field is 1, RDES3 ES field is also 1.
15	OTS	VLAN Filter Status When this field is 1, it indicates that the received packet VLAN tag passes the VLAN filter. This field redefines as an outer VLAN tag filter status (OTS). This field is valid for both single and double VLAN tagged frames.
14	ITS	Inner VLAN Tag Filter Status (ITS) This field is valid only for double VLAN tagged frames, when double VLAN processing is enabled. See Filter status for more information.
13:11	Rsvd	Reserved
10	ARPNR	ARP Reply Not Generated When this field is 1, it indicates that MAC did not generate the ARP reply for the ARP request packet it receives.. This field is 1 when MAC is busy transmitting ARP reply to earlier ARP request (only one ARP request is processed at a time). This field is reserved when you do not select the enable IPv4 ARP offload option.
9:0	HL	L3/L4 Header Length Contains the length of the packet header which MAC splits at L3 or L4 header boundary which the MAC receiver identifies. This field is valid only when the first descriptor bit = 1 (FD = 1). You can write the header data to the buffer 1 address of corresponding descriptor. If the header length is zero, this field is not valid. It implies that MAC does not identify and split the header. This field is valid when you select the enable split header feature option.

76.15.4.1.5.4 RDES3 normal descriptor (Write-back format)

Following table describes the write-back format for the RDES3 normal descriptor.

31	30	29	28	27	26	25	24	23	22	21	20	19	18:16	15	14:0
OWN	CTXT	FD	LD	RS2V	RS1V	RS0V	CE	GP	RWT	OE	RE	DE	LT	ES	PL

Table 680. RDES3 normal descriptor (Write-back format)

Bit	Name	Description
31	OWN	Own Bit When this field is 1, it indicates that the module DMA owns the descriptor. If this field becomes 0, it indicates that the application owns the descriptor. DMA writes 0 to this field when either of these conditions is true: • DMA completes the packet reception. • The buffers associated with the descriptor are full.
30	CTXT	Receive Context Descriptor When this field is 1, it indicates that the current descriptor is a context type descriptor. DMA writes 1'b0 to this field for a normal receive descriptor. When CTXT and FD bits are used together, {CTXT, FD} • 2’b00: Intermediate descriptor • 2’b01: First descriptor • 2’b10: Reserved • 2’b11: Descriptor error (due to all 1s) **NOTE** When descriptor error occurs, the receive DMA closes the receive descriptor indicating a descriptor error. This receive descriptor is skipped and the buffer addresses are not used to write the packet data. Receive DMA will write 1 to the CDE bit in DMA_CH#_Status register but not to the RI bit even when IOC = 1, as this is not marked as the last receive descriptor for the packet. The subsequent valid receive descriptor writes the packet data.
29	FD	First Descriptor When this field is 1, it indicates that this descriptor contains the first buffer of the packet. If the size of the first buffer is 0, the second buffer contains the beginning of the packet and if the size of the second buffer is also 0, the next descriptor will contain the beginning of the packet. See the CTXT bit description for more information on how to use the CTXT bit and FD bit together.
28	LD	Last Descriptor When this field is 1, it indicates that this descriptor buffers points to the last buffers of the packet.
27	RS2V	Receive Status RDES2 Valid When this field is 1, it indicates that the status in RDES2 is valid and DMA writes it. This field is valid only when the LD bit of RDES3 = 1.
26	RS1V	Receive Status RDES1 Valid When this field is 1, it indicates that the status in RDES1 is valid and DMA writes it. This field is valid only when the LD bit of RDES3 = 1.
25	RS0V	Receive Status RDES0 Valid When this field is 1, it indicates that the status in RDES0 is valid and DMA writes it. This field is valid only when the LD bit of RDES3 = 1.

Table continues on the next page...

Table 680. RDES3 normal descriptor (Write-back format) (continued)

Bit	Name	Description
24	CE	CRC Error When this field is 1, it indicates that a cyclic redundancy check (CRC) error occurs on the received packet. This field is valid only when the LD bit of RDES3 = 1.
23	GP	Giant Packet When this field is 1, it indicates that the packet length exceeds the specified maximum Ethernet size of 1518, 1522, or 2000 bytes (9018 or 9022 bytes if jumbo packet enable = 1). **NOTE** Giant packet indicates only the packet length. It does not cause any packet truncation.
22	RWT	Receive Watchdog Timeout When this field is 1, it indicates that the receive watchdog timer expires when it receives the current packet. The current packet truncates after watchdog timeout.
21	OE	Overflow Error When this field is 1, it indicates that the received packet is damaged because of buffer overflow in receive FIFO. **NOTE** This field is 1 only when DMA transfers a partial packet to the application. This happens only when the receive FIFO operates in the Threshold mode. In the Store-and-Forward mode, all the partial packets drops completely in the receive FIFO.
20	RE	Receive Error When this field is 1, it indicates that the gmii_rxer_i signal asserts when the gmii_rxdv_i signal asserts during the packet reception. This error also includes carrier extension error in the GMII and Half-duplex mode. Error can be of less or no extension, or error (rxd!= 0f) during extension.
19	DE	Dribble Bit Error When this field is 1, it indicates that the received packet has a non-integer multiple of bytes (odd nibbles). This field is valid only in the MII mode.
18:16	LT	Length/Type Field Indicates if the packet received is a length packet or a type packet. The encoding of the 3 bits is as follows: • 3'b000: The packet is a length packet. • 3'b001: The packet is a type packet. • 3'b011: The packet is a ARP request packet type • 3'b100: The packet is a type packet with VLAN tag • 3'b101: The packet is a type packet with double VLAN tag • 3'b110: The packet is a MAC control packet type • 3'b111: The packet is a OAM packet type

Table continues on the next page...

Table 680. RDES3 normal descriptor (Write-back format) (continued)

Bit	Name	Description
		• 3'b010: Reserved
15	ES	Error Summary When this field is 1, it indicates the logical OR of these bits, : • RDES3[24]: CRC error • RDES3[19]: Dribble error • RDES3[20]: Receive error • RDES3[22]: Watchdog timeout • RDES3[21]: Overflow error • RDES3[23]: Giant packet • RDES2[17]: Destination address filter fail, when the flexible receive parser is enabled • RDES2[16]: SA address filter fail, when flexible receive parser is enabled This field is valid only when the LD bit of RDES3 = 1.
14:0	PL	Packet Length Indicates the byte length of the received packet that was transferred to system memory (including CRC). This field is valid when the LD bit of RDES3 = 1 and overflow error bits becomes 0. The packet length also includes the two bytes that appends to the Ethernet packet when IP checksum calculation is enabled and the received packet is not a MAC control packet. This field is valid when the LD bit of RDES3 = 1. When the last descriptor and error summary bits are not 1, this field indicates the accumulated number of bytes that are transferred for the current packet.

76.15.4.1.6 Receive context descriptor

This descriptor is read-only for the application. Only DMA can write to this descriptor. The context descriptor provides information about the extended status related to the last received packet. Bit 30 of RDES3 indicates the context type descriptor.

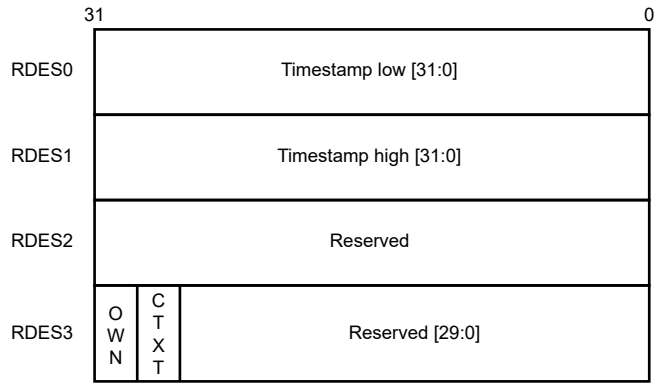


Figure 482. Receive context descriptor format

76.15.4.1.6.1 RDES0 context descriptor

Table 681. RDES0 context descriptor

Bit	Name	Description
31:0	RTSL	Receive Packet Timestamp Low DMA updates this field with least significant 32-bits of the timestamp captured for the corresponding receive packet. When this field and the RTSH field of RDES1 show all-ones value, you must consider the timestamp as corrupt.

76.15.4.1.6.2 RDES1 context descriptor

Table 682. RDES1 context descriptor

Bit	Field	Description
31:0	RTSH	Receive Packet Timestamp High DMA updates this field with the most significant 32-bits of the timestamp captured for the corresponding receive packet. When this field and the RTSL field of RDES0 show all-ones value, you must consider the timestamp as corrupt.

76.15.4.1.6.3 RDES2 context descriptor

Table 683. RDES2 context descriptor

Bit	Description
31:0	Reserved

76.15.4.1.6.4 RDES3 context descriptor

31	30	29	28:0
OWN	CTXT	DE	Rsvd

Table 684. RDES3 context descriptor

Bit	Name	Description
31	OWN	Own Bit When this field is 1, it indicates that DMA owns the descriptor. When this field becomes 0, it indicates that the application owns the descriptor. DMA clears this field when either of these conditions is true: • DMA completes the packet reception. • The buffers associated with the descriptor are full.
30	CTXT	Receive Context Descriptor

Table continues on the next page...

Table 684. RDES3 context descriptor (continued)

Bit	Name	Description
		When this field is 1, it indicates that the current descriptor is a context descriptor. DMA writes 1'b1 to this field for context descriptor. DMA writes 2'b11 to indicate a descriptor error due to all = 1. When CTXT and DE bits are used together {CTXT, DE} • 2'b00: Reserved • 2'b01: Reserved • 2'b10: Context descriptor • 2'b11: Descriptor error Note: When descriptor error occurs, the receive DMA closes the receive descriptor indicating descriptor error. This receive descriptor is skipped and the buffer addresses does not write the packet data. The receive DMA writes 1 to the CDE bit in DMA_CH#_Status register but not to the RI bit even when IOC = 1, because this is not marked as last receive descriptor for the packet. The subsequent valid receive descriptor writes the packet data.
29	DE	Descriptor Error See the CTXT bit description for more information about how to use the DE bit along with CTXT bit.
28:0	Rsvd	Reserved

76.15.5 Enhanced descriptor for time-based scheduling

This feature needs 32 bytes enhanced descriptors to enable on all the DMA channels that uses this feature (write 1 to EDSE bit of DMA_CH(#i)_TX_Control register).

The structure of 32-byte descriptor for the context and the normal descriptor in read and write formats are described in the upcoming sections.

76.15.5.1 Enhanced normal descriptor - Read (32-bit mode)

These fields are present in the first 16 bytes of the enhanced descriptor format of normal descriptor:

- LTV- Indicates that the launch time (LT) and GSN fields present in the descriptor are valid. The LTV must be = 1 only if the FD bit of the descriptor is not.
- GSN- Indicates the GCL slot number associated with the packet.
- LT- Indicates the launch time associated with the packet.

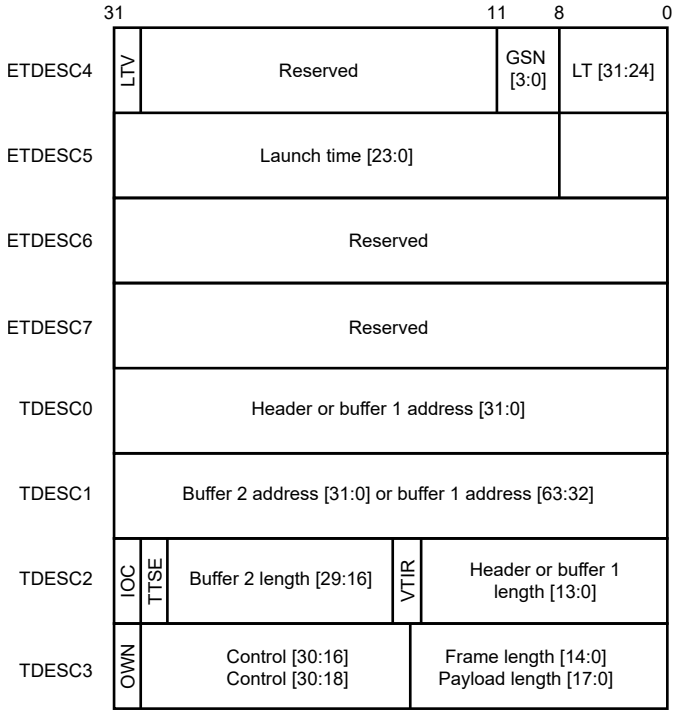


Figure 483. Enhanced normal descriptor - Read (32-bit mode)

76.15.5.2 Enhanced normal descriptor (Write, 32-bit mode)

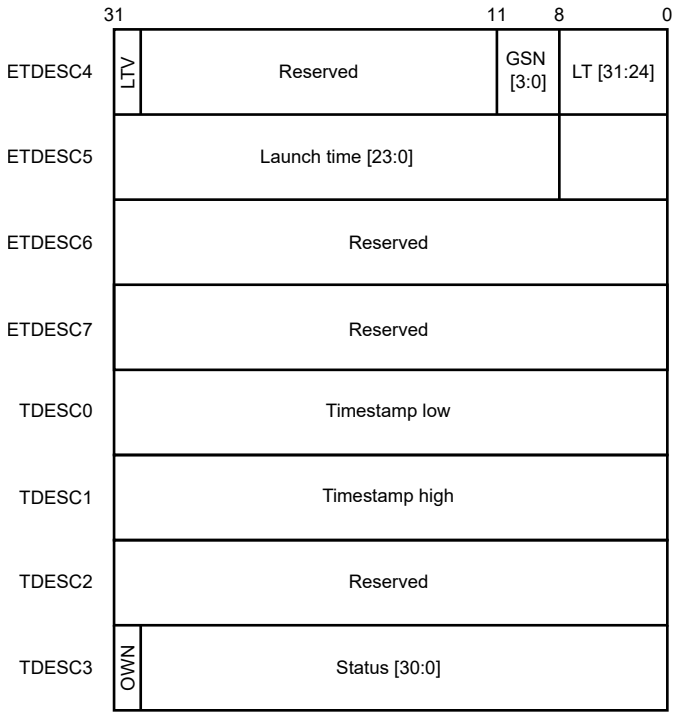


Figure 484. Enhanced normal descriptor write (32-bit mode)

NOTE

- All the enhanced descriptors, example, assumes a 32-bit data width configurations. However, the same definitions are valid in 64-bit and 128-bit configurations.
- In both the write back formats, you cannot modify the extended 16 bytes. The remaining 16 bytes (TDESC0 to TDESC3) are written back per the previous 16 bytes descriptor format.
- When you enable the fetch time it overrides the AV slot function
- When an unaligned new GCL list (with a different CTR) is installed it is recommended not to have traffic during the installation of the new list. Any traffic during the switching of the lists might have unpredictable behavior regarding fetch, launch, and launch expiry because the CTR and BTR values get updated when the frame is processed

76.15.5.3 Enhanced context descriptor (Read, 32-bit mode)

The first 16 bytes of the enhanced context descriptor (ETDESC4 to ETDESC7) are reserved and must be = 0. The fields present in the last bytes of the descriptor (TDESC0 to TDESC3) are same as the context descriptor (TDESC0 to TDESC3) in 16 byte format.

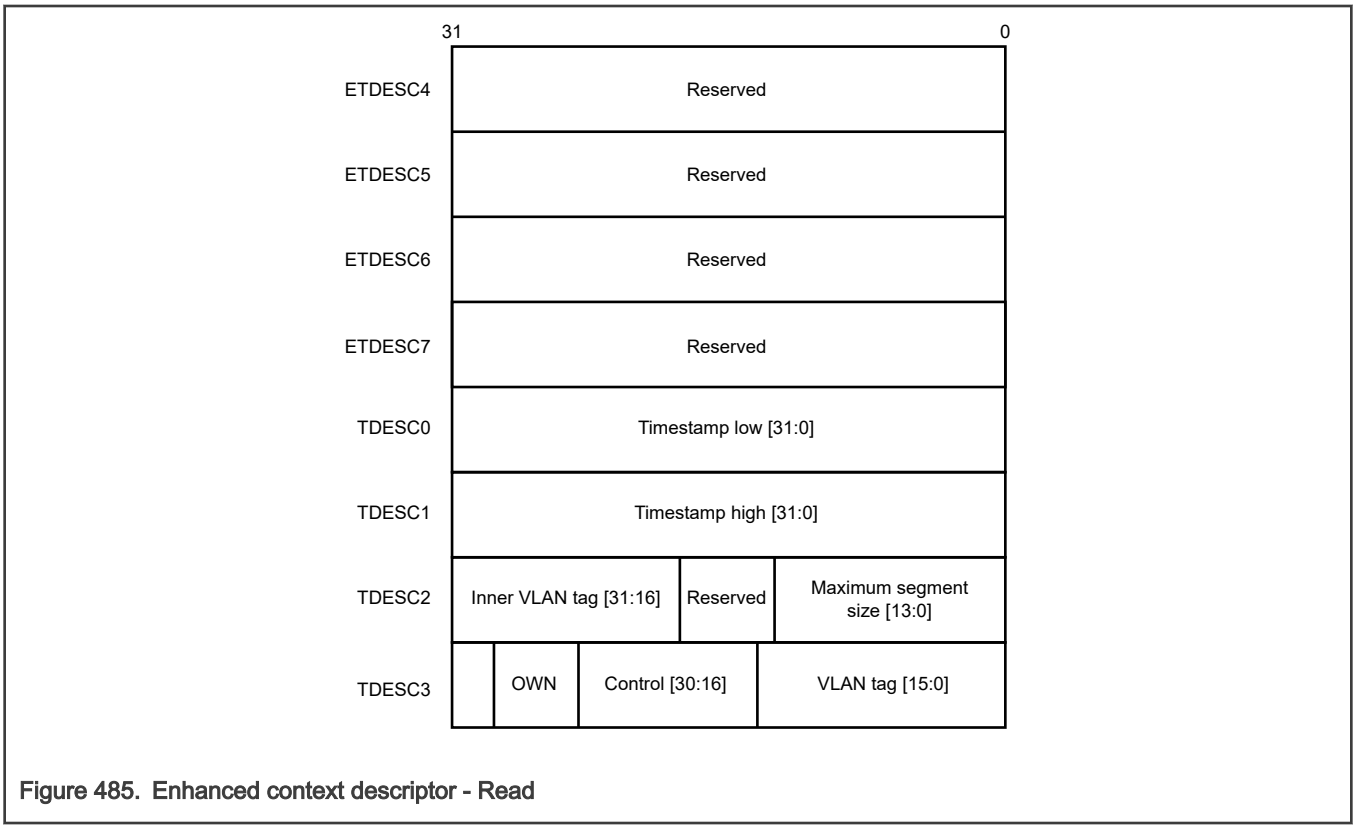


Figure 485. Enhanced context descriptor - Read

76.15.5.4 Enhanced context descriptor (Write, 32-bit mode)

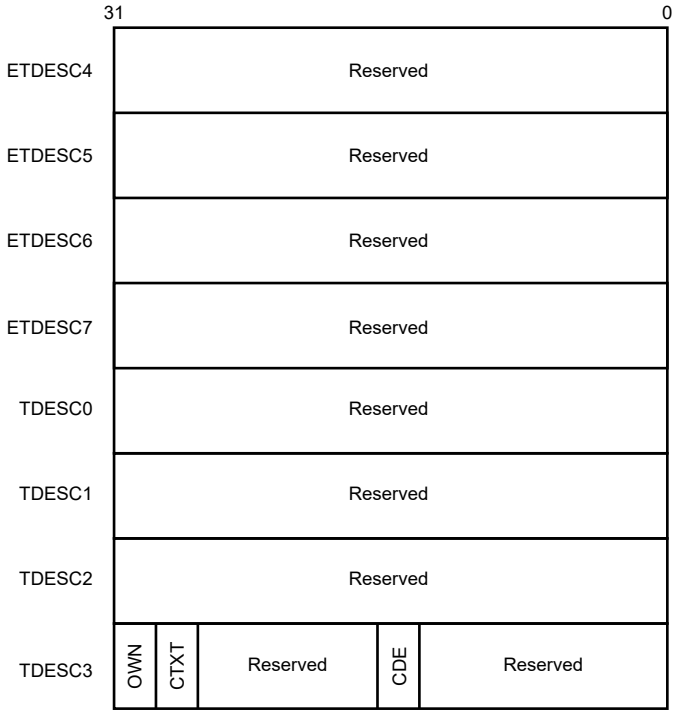


Figure 486. Enhanced context descriptor - Write

76.16 Programming

This section and all its sub-sections are Synopsys Proprietary. Used with permission.

This section provides the instructions for initializing the DMA or MAC registers in the proper sequence.

76.16.1 Initializing DMA

Perform these steps to initialize DMA:

1. Provide a software reset. This resets all the MAC internal registers and logic (bit-0 of DMA_Mode).
2. Wait for the reset process (poll bit 0 of DMA_Mode which is only clears after the reset operation completes) to complete.
3. Program these fields to initialize DMA_SysBus_Mode:
    a. DMA_SysBus_Mode[AAL]
    b. Fixed burst or undefined burst
    c. Burst mode values in case of AHB bus interface, OSR_LMT in case of AXI bus interface.
    d. Select the maximum burst length possible on the AXI bus (bits [7:1]), if fixed length value is enabled.
4. Create a descriptor list for a transmit and receive. Also, ensure that DMA (write 1 to bit 31 of descriptor TDES3/RDES3) owns the descriptors. See Descriptors for more information about descriptors.
5. Program the transmit and receive ring length registers (DMA_CH(#i)_TxDesc_Ring_Length (for i = 0; i <= 1) and DMA_CH(#i)_RxDesc_Ring_Length (for i = 0; i <= 1)). The programmed ring length must be at least 4.

NOTE

The descriptor address from the start to the end of the ring must not cross the 4 GB boundary.

6. Initialize the receive and transmit descriptor list address with the base address of the transmit and receive descriptor (DMA_CH(#i)_TxDesc_List_Address (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1), DMA_CH(#i)_RxDesc_List_Address (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1)). Also, program the transmit and receive tail pointer registers that indicates DMA about the available descriptors (DMA_CH(#i)_TxDesc_Tail_Pointer (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1) and DMA_CH(#i)_RxDesc_Tail_Pointer (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1)).

   NOTE

   For 40-bit or 48-bit addressing mode, program the higher address List registers (DMA_CH[n]_TxDesc_List_HAddress, DMA_CH[n]_RxDesc_List_HAddress).

   The tailpointer registers must advance to the location immediately after the descriptors that are set so that DMA is aware that the additional descriptors are available.

7. Program the settings of these registers for the parameters like maximum burst-length (PBL) which DMA initiates, descriptor skip lengths, OSP in case of TxDMA, RBSZ in case of RxDMA, and so on:
   - DMA_CH(#i)_Control (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1)
   - DMA_CH(#i)_TX_Control (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1)
   - DMA_CH(#i)_RX_Control (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1)
8. Program the DMA_CH(#i)_Interrupt_Enable (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1) register to enable the interrupts.
9. Write 1 to SR (bit 0) of the DMA_CH(#i)_RX_Control (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1) and ST (bit 0) of the DMA_CH(#i)_TX_Control (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1) register to start the receive and transmit DMAs.
10. Repeat steps 4 to 9 for all the transit DMA and receive DMA channels selected in the hardware.

76.16.2 Initializing MTL registers

The transaction layer (MTL) registers must initialize to establish the transmit and receive operating modes and commands.

Perform these steps to initialize the MTL registers:

1. Program MTL_Operation_Mode[SCHALG] and MTL_Operation_Mode[RAA] to initialize the MTL operation in case of multiple transit and receive queues.
2. Program the receive queue to DMA mapping in MTL_RxQ_DMA_Map0.
3. Program these fields to initialize the mode of operation in MTL_TxQ0_Operation_Mode.
   a. MTL_TxQ0_Operation_Mode[TSF] or MTL_TxQ0_Operation_Mode[TTC] in case of Threshold mode.
   b. MTL_TxQ0_Operation_Mode[TXQEN] to value 2'b10 to enable transmit queue0.
   c. MTL_TxQ0_Operation_Mode[TQS]
4. Program these fields to initialize the mode of operation in MTL_RxQ0_Operation_Mode:
   a. MTL_RxQ0_Operation_Mode[RSF] or MTL_RxQ0_Operation_Mode[RTC] in case of Threshold mode.
   b. Flow control activation and de-activation thresholds for MTL receive FIFO (MTL_RxQ0_Operation_Mode[RFA] and MTL_RxQ0_Operation_Mode[RFD]).
   c. Error Packet and undersized good Packet forwarding enable (MTL_RxQ0_Operation_Mode[FEP] and MTL_RxQ0_Operation_Mode[FUP]).
   d. MTL_RxQ0_Operation_Mode[RQS]
5. Repeat previous two steps for all MTL transit and receive queues selected in the configuration.

76.16.3 Initializing MAC

The MAC configuration registers establish the operating mode of MAC. These registers must initialize before initializing DMA.

You can perform these MAC initialization operations after DMA initialization. If the MAC initialization completes before DMA is configured, enable the MAC receiver (last step in the following sequence) only after the DMA is active. Otherwise, the received frames fill the receive FIFO and overflow.

1. Provide the MAC address registers: MAC_Address0_High and MAC_Address0_Low. If more than one MAC address is enabled, , program the MAC addresses appropriately.

2. Program these fields to set the appropriate filters for the incoming frames in MAC_Packet_Filter:

a. Receive all

b. Promiscuous mode

c. Hash or perfect filter

d. Unicast, multicast, broadcast, and control frames filter settings

3. Program these fields for proper flow control in MAC_Q0_Tx_Flow_Ctrl:

a. Pause time and other Pause frame control bits

b. Transmit flow control bits

c. Flow control busy

4. Program MAC_Interrupt_Enable as required, and if applicable, for your configuration.

5. Program the appropriate fields in MAC_Configuration. For example Inter-packet gap when transmission and jabber disable.

6. Write 1 to MAC_Configuration[RE] and MAC_Configuration[TE] to start the MAC transmitter and receiver.

76.16.4 Performing normal receive and transmit operation

During normal operation of the module, read the normal and transmit interrupts, poll the descriptors, suspend the DMA (if it does not own descriptors), and read the values of the current host transmitter or receiver descriptor pointers for debugging.

For normal operation, perform these steps:

1. Read the interrupt status for normal transmit and receive interrupts. Then poll the descriptors, read the status of the descriptor which the host owns (either transmit or receive).

2. Set appropriate values for the descriptors, ensure that the DMA owns the transmit and receive descriptors to resume the data transmission and reception.

3. If DMA does not own the descriptors (or no descriptor is available), DMA enters into SUSPEND state. The transmission or reception can resume by freeing the descriptors and writing the descriptor tail pointer to Tx/Rx tail pointer register (DMA_CH[n]_TxDesc_Tail_Pointer and DMA_CH[n]_RxDesc_Tail_Pointer).

4. Read the current host transmitter or receiver descriptor address pointer values for the debug process (DMA_CH[n]_Current_App_TxDesc and DMA_CH[n]_Current_App_RxDesc register).

5. Read the current host transmit buffer address pointer and receive buffer address pointer values for the debug process (Register DMA_CH[n]_Current_App_TxBuffer and DMA_CH[n]_Cur-rent_App_RxBuffer).

76.16.5 Stopping and starting transmission

Perform these steps to pause the transmission for some time. The steps are provided for channel 0.

1. Write 0 to bit 0 (ST) of DMA_CH(#i)_TX_Control (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1) register to disable the transmit DMA (if applicable) .

2. Wait for any previous frame transmissions to complete. Read the appropriate fields of MTL_TxQ0_Debug (MTL_TxQ0_Debug[TRCSTS] is not 01 and MTL_TxQ0_Debug[TXQSTS] = 0) to check this.

3. Write 0 to MAC_Configuration[RE] and MAC_Configuration[TE] to disable the MAC transmitter and MAC receiver.

4. Disable the receive DMA (if applicable), after you ensure that the data in the receive FIFO transfers to the system memory (read the appropriate fields MTL_RxQ0_Debug[PRXQ] = 0 and MTL_RxQ0_Debug[RXQSTS] = 00).

5. Ensure that both the transit queue and receive queue are empty (MTL_TxQ0_Debug[TXQSTS] = 0 and MTL_RxQ0_Debug[RXQSTS] = 0).

6. Restart the operation by first starting the DMAs, and then enable the MAC transmitter and receiver.

NOTE

- Do not change the configuration (such as duplex mode, speed, port, or loop back) when the MAC is actively transmitting or receiving. You can change these parameters only when the MAC transmitter and receiver are not active.
- Similarly, do not change the DMA related configuration when transmit and receive DMA are active.

76.16.6 Programming guidelines for switching to new descriptor list in RxDMA

Switching to a new descriptor list is different in the receive DMA as compared to the transit DMA. It is permitted when the RxDMA is in SUSPEND state as described below:

- RxDMA prepares the descriptors in advance.
- If the RxDMA goes to SUSPEND state because the descriptors are not available, a major failure occurs (you are not able to free the filled-up descriptors or buffers). If this issue does not rectify immediately, frames will be lost because of an RxFIFO overflow. Therefore, you can create a new descriptor list and program the RxDMA to start using it immediately, without going into STOP state.

76.16.7 Programming guidelines for multi-channel multi-queuing

76.16.7.1 Transmit

1. Program the transmit queue size in TQS field of MTL_TxQ[n]_Operation_Mode register. Based on the value programmed in TQS field, you can determine the queue size.

In the transmit operation, the number of channels is equal to the number of the queues, because of this reason, the channel-to-queue mapping is fixed.

2. Enable the queue in TXQEN in the corresponding MTL_TxQ[n]_Operation_Mode register to use the queue. In DMA configurations, enable the ST field of DMA_CH[n]_Tx_Con-trol register and corresponding TXQEN in MTL_TxQ[n]_Operation Mode register.

3. Program the scheduling method in MTL_Operation_Mode[SCHALG].

4. Program MTL_TxQ[n]_Quantum_Weight register for DCB queue per the selected algorithm. In case of CBS algorithm in AVB queues program the MTL_TxQ[n]_ETS_Control, MTL_TxQ[n]_SendSlopeCredit, MTL_TxQ[n]_HiCredit, and MTL_TxQ[n]_LoCredit registers as required.

5. Program the MAC_TxQ_Prty_Map0 register to assign a fixed priority to the queue, if DCB is enabled and PFC function is required. You can use this assigned priority to determine if the corresponding queue must stop transmitting packet on the basis of the received PFC packet.

76.16.7.2 Receive

1. Program the receive queue size in RQS field of MTL_RxQ[n]_Operation_Mode register. Determine the size of the queue, based on the value programmed in RQS field.
2. Enable the receive queues 0 to 7 in the fields RXQ0EN to RXQ7EN in MAC_RxQ_Ctrl0 for AV or DCB. In DMA configurations, SR bit of statically or dynamically map DMA_CH[n]_Rx_Control register and the corresponding RXQ[n]_EN in MAC_RxQ_Ctrl0 is enabled.

3. Based on these packet types MAC routes the receive packets to the receive queues :
    a. AV PTP packets: Based on the programming of MAC_RxQ_Ctrl1[PTPQ].
    b. AV untagged control packets: Based on the programming of MAC_RxQ_Ctrl1[AVCPQ].
    c. Data center bridging (DCB) related link layer discovery protocol (LLDP) packets. Program DCBCPQ in MAC_RxQ_Ctrl1 to indicate MAC which queue should get the DCB packets.
    d. VLAN tag priority field in VLAN tagged packets: Program PSRQ7-0 of the MAC_RxQ_Ctrl2 and MAC_RxQ_Ctrl3 register for routing the tagged packets based on the received packets USP (user priority) field to the receive queues 0 to 7.
    e. Route the AV tagged control and data packets based on PSRQ field of MAC_RxQ_Ctrl2 and MAC_RxQ_Ctrl3 registers.

NOTE
The priorities set in PSRQ7-0 must be unique.

4. If multiple receive DMA channels are enabled, you must perform the following programming for proper arbitration and mapping:
    a. Program MTL_Operation_Mode[RAA] to select the arbitration algorithm to decide which RxQ is read out from the RxFIFO memory.
    b. Program the MTL_RxQ[n]_Control register to decide the weights and the packet arbitration for each RxQ.
    c. If you program the static mapping in MTL_RxQ_DMA_Map[n] register (RXQ[n]DADMACH becomes 0), fields RXQx2DMA and others are programmed to select the channel to map each queue.
    d. Write 1 to RXQ[n]DADMACH field in MTL_RxQ_DMA_Map0 to select the dynamic mapping of packets in each receive queue.
    e. In dynamic channel mapping, the value of DCS field in the lowest MAC address register decides the routing of a packet to a specific RxDMA channel.

76.16.7.2.1 Programming guidelines for recovering from DMA channel failure

When the DMA channel issues a bus error, follow these steps to recover from the failure.

Recovering from the receive DMA channel failure.

Perform these steps if you get bus error in the receive DMA channel:

1. Write 1 to DMA_CH0_Rx_Control[RPF]. This flushes all the packets one after the other.

This step is optional. However, writing 1 to this field prevents head-of-line blocking in the receive queues when packets sent to the RXDMA are stopped due to the bus error.

2. Re-program the specific registers of the DMA channel.

3. Start the DMA channel.

Recovering from the transmit DMA channel failure.

1. Stop the specific DMA channel, even if it is in active state.

2. Flush the corresponding MTL queue.

3. Re-program the specific registers of the DMA channel.

4. Start the DMA channel.

NOTE
Due to the known limitations in the design, reprogramming the DMA channel registers might not be always successful in recovering from a bus error. If the module is not fully functional after reprogramming the DMA, you can issue a soft reset to recover from the bus error.

76.16.8 Programming guidelines for GMII link state transitions

76.16.8.1 Transmit and receive clocks running when link down

Perform these steps when the link is down but the transmit and receive clocks are running:

NOTE

The steps are provided for channel 0.

1. Write 0 to bit 0 (ST) of DMA_CH(#i)_TX_Control (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1) register to disable the transmit DMA (if applicable).

2. Write 0 to MAC_Configuration[RE] to disable the MAC receiver.

3. Wait for any previous frame transmissions to complete. You can check this by reading the appropriate fields of MTL_TxQ0_Debug (MTL_TxQ0_Debug[TRCSTS] is not 01) or, flush the Tx FIFO for faster empty operation.

4. Write 0 to MAC_Configuration[TE] to disable the MAC transmitter.

5. Make sure that both the transit queue and receive queue are empty (MTL_TxQ0_Debug[TXQSTS] and MTL_RxQ0_Debug[RXQSTS] are 0).

6. Read the PHY registers to know the latest configuration and accordingly program the MAC registers, after the link is up.

7. Start the transit DMA to restart the operation, and then enable the MAC transmitter and receiver.

You do not disable the receive DMA because the receiver is disabled and the FIFO does not get any data in the receive FIFO.

76.16.8.2 Transmit and receive clocks stopped when link down

Perform these steps when the link is down and the transmit and receive clocks are stopped:

NOTE

The steps are provided for channel 0.

1. Write 0 to MAC_Configuration[RE] and MAC_Configuration[TE] to disable the MAC transmitter and receiver. This does not take effect immediately as the clocks are absent.

2. Wait until the link is up and the clocks are restored.

3. Wait for the transfer of any partial frame, if any to complete at time of stopping of the transmit or receive clock. To check this, read MAC_Debug (should be all-zero). Some old packets may still remain in the TXFIFO because the MAC transmitter is stopped.

4. Read the PHY registers to know the latest operating mode and accordingly program the MAC registers.

5. Write 1 to MAC_Configuration[RE] and MAC_Configuration[TE] to restart the MAC transmitter and receiver.

76.16.9 Programming guidelines for IEEE 1588 timestamping

76.16.9.1 Initialization guidelines for system time generation

Write 1 MAC_Timestamp_Control[TSENA] to enable the timestamp feature. However, it is essential to initialize the timestamp counter after MAC_Timestamp_Control[TSENA] = 1. Perform these steps during the module initialization:

1. Write 0 to the bit 16 of MAC_Interrupt_Enable to mask the timestamp trigger interrupt.
2. Write 1 to MAC_Timestamp_Control[TSENA] to enable timestamping.
3. Program MAC_Sub_Second_Increment based on the PTP clock frequency.
4. Program MAC_Timestamp_Addend and write 1 to MAC_Timestamp_Control[TSADDREG], if you are using the fine correction approach.
5. Poll MAC_Timestamp_Control until MAC_Timestamp_Control[TSADDREG] = 0.
6. Program MAC_Timestamp_Control[TSCFUPDT] to select the fine update method (if required).

7. Program MAC_System_Time_Seconds_Update and MAC_System_Time_Nanoseconds_Update with the appropriate time value.
8. Write 1 to MAC_Timestamp_Control[TSINIT].
9. Initialize the timestamp counter with the value written in the timestamp update registers so that the timestamp counter can start operation. If one-step timestamping is enabled
    a. Program Bit 27 of the TDES3 context descriptor, to enable one-step timestamping.
    b. Program MAC_Timestamp_Ingress_Asym_Corr and MAC_Timestamp_Egress_Asym_Corr to update the correction field in PDelay_Req PTP messages.
10. Enable the MAC receiver and transmitter for proper timestamping.

NOTE

If the timestamp operation is disabled by writing 0 to MAC_Timestamp_Control[TSENA], repeat all these steps to restart the timestamp operation.

76.16.9.2 System time correction

76.16.9.2.1 Coarse correction method

Perform these steps to synchronize or update the system time in one process (coarse correction method):

1. Set the offset (positive or negative) in the timestamp registers that is MAC_System_Time_Seconds_Update and MAC_System_Time_Nanoseconds_Update.

2. Write 1 to MAC_Timestamp_Control[TSUPDT].

The value in the timestamp update registers is added to or subtracted from the system time when MAC_Timestamp_Control[TSUPDT] = 0.

76.16.9.2.2 Fine correction method

Perform these steps to synchronize or update the system time to reduce system-time jitter (fine correction method):

1. Calculate the rate by which you want to increment the system time slower or faster, with the help of the algorithm.

2. Update MAC_Timestamp_Addend with the new value and write 1 to MAC_Timestamp_Control[TSADDREG].

3. Wait for the time for which you want the new value of the addend register to be active. You can do this by enabling the timestamp trigger interrupt after the system time reaches the target value.

4. Program the required target time in MAC_PPS[n]_Target_Time_Seconds register and MAC_PPS[n]_Target_Time_Nanoseconds register.

5. Enable the timestamp interrupt in MAC_Interrupt_Enable[TSIE].

6. Write 1 to bit 4 in MAC_Timestamp_Control.

7. Read MAC_Interrupt_Status when this trigger causes an interrupt.

8. Reprogram MAC_Timestamp_Addend with the old value and write 1 to MAC_Timestamp_Control[TSADDREG] again.

76.16.10 Programming guidelines for AV feature

After you enable the AV feature in the module controller, follow these programming tasks:

- Initializing the DMA for QOS-AHB configurations only
- Enabling slot number checking
- Enabling average bits per slot reporting
- Disabling flow control for AV-enabled queues (transmit and receive flow control)

76.16.10.1 Initializing the DMA in audio video feature

The first step to program the AV feature in a QOS-AHB configuration is to initialize the DMA.

Use this initialization sequence for QOS-AHB/QOS-AXI/QOS-AXI4 configurations with AV feature.

1. Provide a software reset to reset all the QOS internal registers and logic (DMA_Mode[SWR]).

2. Wait for the reset process to complete. Poll DMA_Mode[SWR], which clears only after the reset operation completes.

3. Set the values in DMA_Mode to program the fields to initialize the DMA register.

4. Create a descriptor list for transmit and receive. In addition, ensure that the DMA owns the transmit and receive descriptors. When you use OSF mode, at least two transmit descriptors are required.

See Descriptors for more information about descriptors.

5. Ensure that you create three or more different transmit or receive descriptors in the list before reusing any descriptors.

6. Program the transmit and receive ring length registers (DMA_CH(#i)_TxDesc_Ring_Length (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1) and DMA_CH(#i)_RxDesc_Ring_Length (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1)). The ring length programmed must be at least 4.

7. Initialize receive and transmit descriptor list address with the base address of the transmit and receive descriptor (DMA_CH(#i)_TxDesc_List_Address (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1), DMA_CH(#i)_TxDesc_List_Address (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1)). In addition, you must program the transmit and receive tail pointer registers to indicate the DMA about the available descriptors (DMA_CH(#i)_TxDesc_Tail_Pointer (for i = 0; i <= DWC_EQOS_NUM_D-MA_TX_CH-1) and DMA_CH(#i)_RxDesc_Tail_Pointer (for i = 0; i <= DWC_EQOS_NUM_D-MA_RX_CH-1)).

8. Program these fields to initialize the mode of operation in MTL_TxQ0_Operation_Mode:

a. MTL_TxQ0_Operation_Mode[TSF]

b. MTL_TxQ0_Operation_Mode[TTC]

c. MTL_TxQ0_Operation_Mode[TXQEN] to value 2'b10 to enable transmit queue0

d. MTL_TxQ0_Operation_Mode[TQS]

9. Enable the interrupts by programming DMA_CH(#i)_Interrupt_Enable (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1) register.

10. Repeat steps 4 through 9 for all the additional channels of AV feature.

11. Program the AV queues CBS control register, idleSlope, sendSlope, hiCredit, and loCredit registers.

12. Write 1 to bit 0 of DMA_CH(#i)_TX_Control (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1) and DMA_CH(#i)_RX_Control (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1) register to start the receive and transmit DMA.

76.16.10.2 Enabling slot number checking

You can enable slot number checking feature if you specifies the intervals at which the DMA channels map to AV queues fetch the frames from the AHB/AXI system bus.

You must complete these steps after step 11 and before step 12 of Initializing the DMA in audio video feature .

You can use the slot number check feature to specify the intervals at which the DMA channels map to AV queues fetch the frames from the AHB/AXI system bus. This feature is useful for an uniform and periodic transfer of the AV traffic from the host memory and it is available only when you enable timestamping and programs MAC_Sub_Second_Increment. Perform these steps to enable the slot number checking:

1. Follow the steps described in Initialization guidelines for system time generation to enable timestamping.

2. Ensure that the SLOTNUM field (bits 22:19) of TDES3 normal descriptor (Read format) contains a valid slot number. You can read the current reference slot number from the DMA_CH(#i)_Slot_Func-tion_Control_Status (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1) register.

3. Write 1 to bit 0 (ESC) of the slot function control and status register of a channel to enable the slot number checking.

76.16.10.3 Enabling average bits per slot reporting

You can enable the reporting of average bits transmitted in a slot.

The CBS status register of the additional AV channels provide information about the average bits that are transmitted in a slot. You can asynchronously read this register to retrieve information about the average bits transmitted per slot. Perform these steps to enable average bits per slot reporting:

1. Follow the steps as described in the Initialization guidelines for system time generation to enable timestamping.

2. Program SLC bits [6:4], of the MTL_TxQ[n]_ETS_Control register of a channel with the number of slots over which the average transmitted bits per slot are computed.

3. Enable bit 9(ABPSSIE) of the MTL_Q[n]_Interrupt_Control_Status register of a channel to generate the average bits per slot interrupt.

NOTE

- The frequency of this interrupt depends on the value programmed in step 2. For example, when you program value 0 in the SLC field, the interrupt generates at every 125 microsecond.
- You can disable this interrupt to stop the interrupt flooding, when it is not required.

4. Read ABS bits [16:0], of the MTL_TxQ[n]_ETS_Status register of a channel on each interrupt.

NOTE

You can read the ABS fields in the polling mode even if ABPSIE bit is not enabled. When bit 1 (ABPSIS) of the MTL_TxQ[n]_ETS_Status register is 1, it indicates that a new value is updated in the ABS field.

76.16.10.4 Disabling flow control for AV enabled queues

76.16.10.4.1 Transmit flow

Program the EHFC (Enable Hardware Flow Control) field of the corresponding receive queue's MTL_RxQ[n]_Operation_Mode register to 0.

76.16.10.4.2 Receive flow control

Program PSTQ[n] field, corresponding to AV enabled transit queue in MAC_TxQ_Prty_Map0/1 register to 0.

76.16.10.5 Programming guidelines for flexible pulse-per-second output

After you enable the flexible pulse-per-second output feature in the module controller, you can perform these tasks:

- Generating single pulse on PPS
- Generating next pulse on PPS
- Generating a pulse train on PPS
- Generating an interrupt without affecting the PPS

76.16.10.5.1 Generating single pulse on PPS

Perform these steps to generate single pulse on PPS:

1. Program 11 or 10 (for interrupt) in TRGTMODSEL bits [6:5], of MAC_PPS_Control. This instructs MAC to use the target time registers (MAC_PPS0_Target_Time_Seconds and MAC_PPS0_Target_Time_Nanoseconds) for a start time of the PPS signal output.
2. Program the start time value in the target time registers (MAC_PPS0_Target_Time_Seconds and MAC_PPS0_Target_Time_Nanoseconds).
3. Program the PPS signal output width in MAC_PPS(#i)_Width (for i = 0; i <= 3) register.

4. Program PPSCMD bits [3:0], of MAC_PPS_Control to 0001. This instructs MAC to generate single pulse on the PPS signal output at the time programmed in the target time registers.

76.16.10.5.2 Generating next pulse on PPS

When the PPSCMD is executed (PPSCMD bits = 0), you can give the cancel start command (PPSCMD=0011) before the programmed start time elapses to cancel the pulse generation. You can also program the behavior of the next pulse in advance.

Follow these steps to program the next pulse:

1. Program the start time for the next pulse in the target time registers. This time must be more than the time at which the falling edge occurs for the previous pulse.

2. Program the next PPS signal output width in MAC_PPS(#i)_Width (for i = 0; i <= DWC_EQOS_PPS_OUT_NUM-1) register.

3. Program PPSCMD bits [3:0], of MAC_PPS_Control to generate a single pulse after the time at which the previous pulse is de-asserted. This instructs MAC to generate single pulse on the PPS signal output, at the time programmed in target time registers.

If you give this command before the previous pulse becomes low, then the new command overwrites the previous command and the QOS may generate only 1 extended pulse.

76.16.10.5.3 Generating a pulse train on PPS

Perform these steps to generate a pulse train on PPS:

1. Program 11 or 10 (for interrupt) in TRGTMODSEL bits [6:5] , of MAC_PPS_Control. This instructs MAC to use the target time registers for the start time of the PPS signal output.

2. Program the start time value in the target time registers.

3. Program the interval value between the train of pulses on the PPS signal output in MAC_PPS(#i)_Interval (for i = 0; i <= DWC_EQOS_PPS_OUT_NUM-1) register.

4. Program the PPS signal output width in MAC_PPS(#i)_Width (for i = 0; i <= DWC_EQOS_PPS_OUT_NUM-1) register.

5. Program PPSCMD bits [3:0], of MAC_PPS_Control to 0010. This instructs MAC to generate train of pulses on the PPS signal output with start time programmed in target time registers.

By default, the PPS pulse train is free-running unless the STOP pulse train at time or STOP pulse train immediately commands stop it.

6. Program the stop value in the target time registers. Ensure that bit 31 (TSTRBUSY) of MAC_PPS(#i)_Target_Time_Nanoseconds (for i = 0; i <= DWC_EQOS_PPS_OUT_NUM-1) register becomes 0 before programming the target time registers again.

7. Program PPSCMD field [bit 3:0] of MAC_PPS_Control to 0100. This stops the train of pulses on the PPS signal output after the programmed stop time specified in step 6 elapses.

You can program 0101 in PPSCMD field [bit 3:0] of MAC_PPS_Control to stop the pulse train at any time. Similarly, you can program 0110 in PPSCMD field [bit 3:0] of MAC_PPS_Control before the time (programmed in step 6) elapses to cancel the stop pulse train command (given in Step 7). Also, program 0011 in PPSCMD field [bit 3:0] of MAC_PPS_Control before the programmed start time (in Step 2) elapses to cancel the pulse train generation.

76.16.10.5.4 Generating an interrupt without affecting the PPS

MAC_PPS_Control TRGTMODSEL bits [6:5] enables you to program the target time registers to perform any one of these actions:

- Generate only interrupts.
- Generate interrupts and the PPS start and stop time.
- Generate only PPS start and stop time.

Perform these steps to program the target time registers to generate only interrupt event:

1. Program 00 (for interrupt) in MAC_PPS_Control TRGTMODSEL bits [6:5]. This instructs the MAC to use the target time registers for target time interrupt.

2. Program a target time value in the target time registers. This instructs the MAC to generate an interrupt when the target time elapses.

If TRGTMODSEL bits [6:5], are changed (for example, to control the PPS) then over-write the interrupt generation with the new mode and new programmed target time register value.

76.16.10.6 Programming guidelines for VLAN filtering on receive

Perform these steps to program VLAN filtering on receive:

1. Program MAC_VLAN_Tag register for the following fields to select the filtering method:

a. ETV: Enables 12-Bit VLAN tag comparison or 16-bit VLAN tag comparison.

b. VTHM: VLAN tag hash table match enable.

c. ERIVLT: Enables inner VLAN tag or outer VLAN Tag (to enable the inner or outer VLAN tag filtering, writing 1 to MAC_VLAN_Tag_Ctrl[EDVLP] enables the double VLAN processing).

d. ERSVLM: Enables receive S-VLAN match or C-VLAN match (write 1 to MAC_VLAN_Tag_Ctrl[ESVL] to enable S-VLAN processing).

e. DOVLTC: Ignores VLAN type for tag match

f. VTIM: Enables VLAN tag inverse match instead of the normal VLAN tag matching

2. Program VL of MAC_VLAN_Tag register for the 12-bit or 16-bit VLAN tag.

3. Program MAC_VLAN_Hash_Table, if hash filtering of VLAN tag is enabled. When MAC_VLAN_Tag_Ctrl[ETV] becomes 0, it indicates that the upper 4 bits of the calculated CRC-32 of VLAN tag are inverted and index the content of MAC_VLAN_Hash_Table. When MAC_VLAN_Tag_Ctrl[ETV] is 1, it indicates that the upper 4 bits of calculated CRC-32 of VLAN Tag index the content of MAC_VLAN_Hash_Table. For example, when MAC_VLAN_Tag_Ctrl[ETV] is 1, a hash value of 4b'1000 selects bit 8 of MAC_VLAN_Hash_Table. When MAC_VLAN_Tag_Ctrl[ETV] becomes 0 a hash value of 4'b1000 selects bit 7 of MAC_VLAN_Hash_Table.

76.16.10.7 Programming guidelines for extended VLAN filtering and routing on receive

Perform these steps, for the indirect access of the per VLAN tag registers:

- Write
  - Write the required data into MAC_VLAN_Tag_Data.
  - Program MAC_VLAN_Tag_Ctrl[OFS] with the required filter register's offset and command type to MAC_VLAN_Tag_Ctrl[CT]. For a write command, set this field to 0.
  - Write 1 to MAC_VLAN_Tag_Ctrl[OB] and wait till it becomes 0 to do the next write. This will guarantee that you have programmed the appropriate VLAN Tag Filter register.
- Read
  - Program MAC_VLAN_Tag_Ctrl[OFS] with the required register's offset and command type to MAC_VLAN_Tag_Ctrl[CT]. For a read command, set this field to 1.
  - Write 1 to MAC_VLAN_Tag_Ctrl[OB] and wait till it becomes 0. The appropriate value of the VLAN Tag Filter register is available in MAC_VLAN_Tag_Data.

76.16.10.8 Programming sequence for queue/channel based VLAN inclusion register

Perform these steps to program the queue/channel-based VLAN inclusion register:

NOTE

When MAC_VLAN_Incl[CBTI] = 1 you cannot access the indirect VLAN include registers.

1. Write 1 to MAC_VLAN_Incl[CBTI], to enable queue or channel based VLAN tag insertion on all the transmitted packets. This field must be 1 before any indirect access to the queue or channel specific MAC_VLAN_Incl(#i) register.

2. Program the VLAN tag and VLAN type to insert in packets from a particular queue or channel in MAC_VLAN_Incl[VLT] and MAC_VLAN_Incl[CSVL], corresponding offset address in ADDR field (0 for queue/channel 0, 1 for queue/channel 1, and so on) must be set. Write 0 to MAC_VLAN_Incl[RDWR] to indicate write access. The write to byte 0 (byte 3 in Big Endian mode) of MAC_VLAN_Incl initiates access to indirect access MAC_VLAN_Incl(#i) register.

3. Write 1 to MAC_VLAN_Incl[BUSY] via the module to indicate the progress of access to indirect access MAC_VLAN_Incl(#i) register. After the access completes, MAC_VLAN_Incl[BUSY] = 0. The application must not attempt subsequent access to MAC_VLAN_Incl(#i) register when the MAC_VLAN_Incl[BUSY] is 1.

4. Repeat step 2 and step 3 to program VLAN tag and VLAN type to insert in packets from the remaining queues or channels. The application must ensure that the required VLAN tag and VLAN type for all the queues or channels are programmed, otherwise an unintended VLAN tag and VLAN type might be inserted.

76.16.11 Programming guidelines for EST

Program the gate control values and time intervals in MTL_EST_Status[SWOL] along with the other EST related registers that are described in GMAC register descriptions to appropriate values. The upcoming sub-sections provide step by step details for programming the GCL and the other EST related registers.

76.16.11.1 Programming the GCL and GCL linked registers

Perform these steps to program the GCL and the four other registers implemented per GCL:

1. Access the GCL and the four other GCL-linked registers via indirect addressing using MTL_EST_GCL_Control and MTL_EST_GCL_Data. MTL_EST_Status[SWOL] indicates whether the software owns GCL0 or GCL1.
2. Write the 32-bit write data to MTL_EST_GCL_Data, to program the GCL. Then program MTL_EST_GCL_Control to write the write address and other control information.
3. In MTL_EST_GCL_Data, write data consists of up to 8 bits (configurable) of gate controls and up to 24 bits (configurable) of time interval. Programming a 0 indicates gate close and programming a 1 indicates gate open. For a 4-TC and 20-bit time interval configuration, the data width is 24-bits and the remaining 8-bits are reserved/read-only. You must write the data in the following format.

   {8'h0, TC3, TC2, TC1, TC0, 20-bit Time Interval} where TCx = 0 or 1.
4. Program MTL_EST_GCL_Control[SRWO] to 1 (to start a Write Op) and program the address and R/W fields appropriately.
5. Poll and check that the hardware clears MTL_EST_GCL_Control[SRWO] to indicate that the previous operation completes before initiating a new read-write operation via the same indirect addressing mode.
6. Repeat steps 3, 4, 5 until the programming of the GCL completes.
7. Program the BTR, CTR, TER and LLR registers, using the same indirect addressing method as described above. Write 1 to MTL_EST_GCL_Control[GCRR] appropriately. MTL_EST_GCL_Control[GCRR] interprets the address field as belonging to these registers (instead of the GCL).

After the GCL and the related registers are programmed, program MTL_EST_GCL_Control to allow hardware to own and process the GCL. When the list length (as indicated in LLR) is 1, the associated time interval must be smaller than the cycle time register value. Otherwise, an error is reported (as described in the Error handling section) as a single set of gate controls add no value in the TSN context.

NOTE

The time unit in all the GCL related registers is seconds and nanoseconds. In cases where internally generated PTP system time is used, you must program the nanoseconds field to use the Digital Rollover mode. (The value of MAC_Timestamp_Control[TSCTRLSSR]must be 1)

76.16.11.2 Programming the EST registers

After the steps mentioned in "Programming the GCL and GCL Linked Registers" completes, program MTL_EST_Control

1. Write 1 to MTL_EST_Control[CTOV] and MTL_EST_Control[TILS]. Also, write 1 to MTL_EST_Control[EEST] and MTL_EST_Control[SWOL].

   NOTE

   The CTOV recommendations for GMII for SPRAM configurations are:

   - 96 * Tx clock period, for 32 and 64 bit data width configurations.
   - 128 * Tx clock period for 128 bit data width configurations.

   The CTOV recommendations for GMII for non-SPRAM configurations are:

   - 30 * Tx clock period, when SA/VLAN insertion is enabled.
   - 22 * Tx clock period, when SA/VLAN insertion is not enabled.

2. Enables the hardware to own and process the new GCL and make a switch to the new GCL at the BTR value. The hardware provides an interrupt (if enabled) when the switch to the new list happens.
3. Performs an appropriate action to address any other interrupts (explained in the Interrupts section) received during the hardware execution of the GCL.

Write 1 to MTL_EST_Control[SSWL] to handoff to hardware. Software is not allowed to write to the GCL and GCL linked registers when MTL_EST_Control[SSWL] = 1, because the hardware might be using the new GCL.

The hardware resets or clears MTL_EST_Control[SSWL] when it successfully switches to the new list. The hardware also flips MTL_EST_Status[SWOL] to indicate the new GCL that the software owns.

Program the GCL that software owns (MTL_EST_Status[SWOL] indicates) as described in "Programming the GCL and GCL Linked Registers" and then program MTL_EST_Control as described above, to install a new GCL. Ensure that the new BTR is set to an appropriate value to avoid BTR Error that may require software intervention in some cases.

The packet length (frame size) information must be available at all times, to avoid transmission overruns. Therefore, in the DMA configurations, program the packet length in the first descriptor of every transmit frame. Similarly, in the MTL configuration, provide the packet length in the control word.

NOTE

The packet length provided in the transmit descriptor must account for the SA and VLAN insertion, to avoid transmission overruns, if applicable, that is, packet length must represent the actual packet length on the Ethernet line.

76.16.12 Programming the launch time in time-based scheduling

Configure the launch time in the enhanced normal transmit descriptors in DMA configurations and is driven as a control word in MTL configurations as follows:

The OSTC and launch time features are mutually exclusive and must not be used together. In case of a conflict (if OSTC = LTV = 1 in MTL configuration), give priority to OSTC and ignore the launch time.

In DMA configuration, if a context descriptor is received with a valid OSTC values immediately before receiving a first normal descriptor with LTV = 1, then the LTV is ignored.

76.16.13 Programming guidelines for media clock generation and recovery

76.16.13.1 Programming guidelines for media clock generation

These are the guidelines for the media clock generation:

1. Program the appropriate presentation time control (supported generation modes "1001-1011") to PPSCTRL_PPSCMD (for 0th instance)/PPSCMD#i (for 1,2,3 instances) of MAC_PPS_Control, to define the PPS instance in Media clock generation mode.

2. Based on the selected PPS instance, application must drive the appropriate trigger signal to the corresponding MCG_PST_TRIG_I[#i].

3. Based on the programmed mode, DUT captures the timestamp and programs it in the MAC_PPS(#i)_Target_Time_Seconds register. Then, mcgr_dma_req_o[#i] is set.

4. For every request which DUT generates, the module reads the corresponding MAC_PPS(#i)_Target_Time_Seconds register and asserts the corresponding mcgr_d-ma_ack_i[#i] to acknowledge the read request.

76.16.13.2 Programming guidelines for media clock recovery

These are the guidelines for the media clock recovery:

- Writing 1 to MAC_Timestamp_Control[PTGE] enables the CPT counter. In addition to configuring the initialization values for the system time, update MAC_Presn_Time_Updt with the equivalent presentation time initialization value, then MAC_Timestamp_Control[TSINIT] = 1.
- Use the increment values for the system time and also for the CPT because the increment value is in sub-seconds and sub-nanoseconds.
- Update MAC_Presn_Time_Updt with the equivalent presentation time update value (32 bit ns), when an update is required, and also configure the updated values for the system time. Finally, MAC_Timestamp_Control[TSUPDT] must be 1 for the update.
- Program the MCGREN#i field of MAC_PPS_Control to enable the PPS instance to operate in MCGR mode.
- Program the appropriate presentation time control (supported recovery modes "0001 - 0011") to PPSCTRL_PPSCMD (for 0th instance) or PPSCMD#i (for 1,2,3 instances) of MAC_PPS_Control register, to set the PPS instance in Media clock recovery mode.
- After the PPS instance is set in Recovery mode, DUT sets the corresponding mcgr_dma_req_o[#i]. For every request which DUT generates, program the target presentation time into the MAC_PPS(#i)_Target_Time_Seconds register and assert the corresponding mcgr_dma_ack_i[#i] to acknowledge the request. Acknowledgment can be given before/after programming the Target presentation time.
- Observe the recovered media clock on the corresponding PTP_PPS_O[#i].

76.16.14 Programming guidelines for ECC protection for memories

These are the guidelines for the ECC protection for memories:

- Write 1 to the appropriate field of MTL_ECC_Control to enable the ECC features for the respective memory.
- Generate a correctable interrupt (sbd_sfty_ce_intr_o) or an uncorrectable interrupt (sbd_sfty_ue_intr_o) to indicate the application, if any correctable, uncorrectable, or address errors are detected. DMA_Interrupt_Status or MTL_ECC_Interrupt_Status indicates an appropriate status.
- Provide debug mode for each memory to specify errors.

NOTE

Enable the ECC feature before the traffic is online (or after the reset), otherwise the false interrupts may trigger.

76.16.14.1 Programming guidelines for ECC hardware error injection (Debug mode)

Follow these steps for ECC hardware error injection:

- Write 1 to the appropriate field in MTL_DBG_CTL to enable the ECC error injection feature for MTL TX/RX and DMA TSO.
- Write 1 to the appropriate field in MTL_EST_GCL_Control to enable the ECC error injection feature for the EST memory.
- Write 1 to the appropriate field in MTL_RXP_Indirect_Acc_Control_Status to enable the ECC error injection for the receive parser memory. To access the receive parser memory in Debug mode, write 0 to MTL_RXP_Indirect_Acc_Control_Status[FRPE] to disable the receive parser feature.
- Ensure that all the CSR writes corresponding to one memory write must have the same value for the error injection control word, where multiple CSR writes are required for writing single data word into the memory.

76.16.15 Programming guidelines for on-chip datapath parity protection

Follow these steps for on-chip datapath parity protection:

- Writing 1 to MTL_DPP_Control[EDPP] enables the parity generation and parity detection mismatch on datapath.
- Generates an uncorrectable interrupt (sbd_sfty_ue_intr_o) to indicate the application if any parity mismatch is detected. DMA_Safety_Interrupt_Status or MTL_Safety_Interrupt_Status and MAC_DPP_FSM_Interrupt_Status indicates the appropriate status.
- Supports Debug mode for each parity generator to insert an error.

NOTE

Enable the datapath parity protection before the receive or transit traffic starts, to avoid the false safety interrupts.

76.16.16 Programming guidelines for FSM parity and timeout

Follow these steps to configure the FSM parity and timeout:

- Write 1 to MAC_FSM_Control[PRTYEN] and MAC_FSM_Control[TMOUTEN] to enable the FSM parity and timeout feature respectively.
- Force the FSMs to enter into a state with even number of 1s and wait for the safety uncorrectable interrupt to define with MAC_DPP_FSM_Interrupt_Status[FSMPES] for a parity error detection.
- Write 1 to the parity error injection using the [23:16] bits of MAC_FSM_Control to select the appropriate clock domain, for Error injection mode in FSM parity. Wait for the safety uncorrectable interrupt to be defined with MAC_DPP_FSM_Interrupt_Status[FSMPES].
- Configure the large and normal mode values which MAC_FSM_ACT_Timer[LTMRMD] and MAC_FSM_ACT_Timer[NTMRMD] indicates , for FSM timeouts.
- Use the [31:24] bits of MAC_FSM_Control to select the large or normal mode tic generation per clock domain.
- Write 1 to MAC_FSM_ACT_Timer[TMR] with the appropriate number of CSR clock cycles to generate 1us tic.
- Force the FSM in a particular clock domain to remain in an active state for timeout duration (for example, T-2T). You can check the safety interrupt and the status field for the corresponding clock domain that sets in the [15:8] bits of the MAC_DPP_FSM_Interrupt_Status .
- Select the appropriate clock domain for which timeout error injection must be set using the [15:8] bits of MAC_FSM_Control, for Error injection mode in FSM timeout. Wait for the safety uncorrectable interrupt to define with the corresponding error status in the MAC_DPP_FSM_Interrupt_Status .
- Use only normal mode tic generation, for application or CSR interface timeout. On the basis of the time-outs of configured interface, appropriate fields (MSTTES, SLVTES) of MAC_DPP_FSM_Interrupt_Status is defined with the safety interrupt.

76.17 GMAC register descriptions

76.17.1 GMAC memory map

NOTE

This chip has 16 KB allocated per Ethernet controller for configuration space. GMAC uses only 8 KB of this memory and the remaining 8 KB is reserved for future use. Writing to the reserved memory may lead to unintended behavior.

GMAC_0 base address: 4048_4000h

GMAC_1 base address: 4048_8000h

Offset	Register	Width (In bits)	Access	Reset value
0h	MAC Configuration (MAC_Configuration)	32	RW	0000_0000h
4h	MAC Extended Configuration (MAC_Ext_Configuration)	32	RW	0000_0000h
8h	MAC Packet Filter (MAC_Packet_Filter)	32	RW	0000_0000h
Ch	MAC Watchdog Timeout (MAC_Watchdog_Timeout)	32	RW	0000_0000h
10h	MAC Hash Table First 32 Bits (MAC_Hash_Table_Reg0)	32	RW	0000_0000h
14h	MAC Hash Table Second 32 Bits (MAC_Hash_Table_Reg1)	32	RW	0000_0000h
18h	MAC Hash Table Third 32 Bits (MAC_Hash_Table_Reg2)	32	RW	0000_0000h
1Ch	MAC Hash Table Fourth 32 Bits (MAC_Hash_Table_Reg3)	32	RW	0000_0000h
20h	MAC Hash Table Fifth 32 Bits (MAC_Hash_Table_Reg4)	32	RW	0000_0000h
24h	MAC Hash Table Sixth 32 Bits (MAC_Hash_Table_Reg5)	32	RW	0000_0000h
28h	MAC Hash Table Seventh 32 Bits (MAC_Hash_Table_Reg6)	32	RW	0000_0000h
2Ch	MAC Hash Table Eighth 32 Bits (MAC_Hash_Table_Reg7)	32	RW	0000_0000h
50h	MAC VLAN Tag Control (MAC_VLAN_Tag_Ctrl)	32	RW	0000_0000h
54h	MAC VLAN Tag Data (MAC_VLAN_Tag_Data)	32	RW	0000_0000h
54h	MAC VLAN Tag Filter 0 (MAC_VLAN_Tag_Filter0)	32	RW	0000_0000h
54h	MAC VLAN Tag Filter 1 (MAC_VLAN_Tag_Filter1)	32	RW	0000_0000h
54h	MAC VLAN Tag Filter 10 (MAC_VLAN_Tag_Filter10)	32	RW	0000_0000h
54h	MAC VLAN Tag Filter 11 (MAC_VLAN_Tag_Filter11)	32	RW	0000_0000h
54h	MAC VLAN Tag Filter 12 (MAC_VLAN_Tag_Filter12)	32	RW	0000_0000h
54h	MAC VLAN Tag Filter 13 (MAC_VLAN_Tag_Filter13)	32	RW	0000_0000h
54h	MAC VLAN Tag Filter 14 (MAC_VLAN_Tag_Filter14)	32	RW	0000_0000h
54h	MAC VLAN Tag Filter 15 (MAC_VLAN_Tag_Filter15)	32	RW	0000_0000h
54h	MAC VLAN Tag Filter 16 (MAC_VLAN_Tag_Filter16)	32	RW	0000_0000h
54h	MAC VLAN Tag Filter 17 (MAC_VLAN_Tag_Filter17)	32	RW	0000_0000h
54h	MAC VLAN Tag Filter 18 (MAC_VLAN_Tag_Filter18)	32	RW	0000_0000h
54h	MAC VLAN Tag Filter 19 (MAC_VLAN_Tag_Filter19)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
54h	MAC VLAN Tag Filter 2 (MAC_VLAN_Tag_Filter2)	32	RW	0000_0000h
54h	MAC VLAN Tag Filter 20 (MAC_VLAN_Tag_Filter20)	32	RW	0000_0000h
54h	MAC VLAN Tag Filter 21 (MAC_VLAN_Tag_Filter21)	32	RW	0000_0000h
54h	MAC VLAN Tag Filter 22 (MAC_VLAN_Tag_Filter22)	32	RW	0000_0000h
54h	MAC VLAN Tag Filter 23 (MAC_VLAN_Tag_Filter23)	32	RW	0000_0000h
54h	MAC VLAN Tag Filter 24 (MAC_VLAN_Tag_Filter24)	32	RW	0000_0000h
54h	MAC VLAN Tag Filter 25 (MAC_VLAN_Tag_Filter25)	32	RW	0000_0000h
54h	MAC VLAN Tag Filter 26 (MAC_VLAN_Tag_Filter26)	32	RW	0000_0000h
54h	MAC VLAN Tag Filter 27 (MAC_VLAN_Tag_Filter27)	32	RW	0000_0000h
54h	MAC VLAN Tag Filter 28 (MAC_VLAN_Tag_Filter28)	32	RW	0000_0000h
54h	MAC VLAN Tag Filter 29 (MAC_VLAN_Tag_Filter29)	32	RW	0000_0000h
54h	MAC VLAN Tag Filter 3 (MAC_VLAN_Tag_Filter3)	32	RW	0000_0000h
54h	MAC VLAN Tag Filter 30 (MAC_VLAN_Tag_Filter30)	32	RW	0000_0000h
54h	MAC VLAN Tag Filter 31 (MAC_VLAN_Tag_Filter31)	32	RW	0000_0000h
54h	MAC VLAN Tag Filter 4 (MAC_VLAN_Tag_Filter4)	32	RW	0000_0000h
54h	MAC VLAN Tag Filter 5 (MAC_VLAN_Tag_Filter5)	32	RW	0000_0000h
54h	MAC VLAN Tag Filter 6 (MAC_VLAN_Tag_Filter6)	32	RW	0000_0000h
54h	MAC VLAN Tag Filter 7 (MAC_VLAN_Tag_Filter7)	32	RW	0000_0000h
54h	MAC VLAN Tag Filter 8 (MAC_VLAN_Tag_Filter8)	32	RW	0000_0000h
54h	MAC VLAN Tag Filter 9 (MAC_VLAN_Tag_Filter9)	32	RW	0000_0000h
58h	MAC VLAN Hash Table (MAC_VLAN_Hash_Table)	32	RW	0000_0000h
60h	MAC VLAN Inclusion Or Replacement (MAC_VLAN_Incl)	32	RW	0000_0000h
60h	MAC VLAN Inclusion 0 (MAC_VLAN_Incl0)	32	RW	0000_0000h
60h	MAC VLAN Inclusion 1 (MAC_VLAN_Incl1)	32	RW	0000_0000h
60h	MAC VLAN Inclusion 2 (MAC_VLAN_Incl2)	32	RW	0000_0000h
60h	MAC VLAN Inclusion 3 (MAC_VLAN_Incl3)	32	RW	0000_0000h
60h	MAC VLAN Inclusion 4 (MAC_VLAN_Incl4)	32	RW	0000_0000h
60h	MAC VLAN Inclusion 5 (MAC_VLAN_Incl5)	32	RW	0000_0000h
60h	MAC VLAN Inclusion 6 (MAC_VLAN_Incl6)	32	RW	0000_0000h
60h	MAC VLAN Inclusion 7 (MAC_VLAN_Incl7)	32	RW	0000_0000h
64h	Inner VLAN Tag Inclusion Or Replacement (MAC_Inner_VLAN_Incl)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
70h	MAC Q0 Tx Flow Control (MAC_Q0_Tx_Flow_Ctrl)	32	RW	0000_0000h
74h	MAC Q1 Tx Flow Control (MAC_Q1_Tx_Flow_Ctrl)	32	RW	0000_0000h
78h	MAC Q2 Tx Flow Control (MAC_Q2_Tx_Flow_Ctrl)	32	RW	0000_0000h
90h	MAC Receive Flow Control (MAC_Rx_Flow_Ctrl)	32	RW	0000_0000h
94h	MAC Rx Queue Control 4 (MAC_RxQ_Ctrl4)	32	RW	0000_0000h
98h	MAC Tx Queue Parity Map 0 (MAC_TxQ_Prty_Map0)	32	RW	0000_0000h
A0h	MAC Rx Queue Control 0 (MAC_RxQ_Ctrl0)	32	RW	0000_0000h
A4h	MAC Rx Queue Control 1 (MAC_RxQ_Ctrl1)	32	RW	0000_0000h
A8h	MAC Rx Queue Control 2 (MAC_RxQ_Ctrl2)	32	RW	0000_0000h
B0h	MAC Interrupt Status (MAC_Interrupt_Status)	32	R	0000_0000h
B4h	MAC Interrupt Enable (MAC_Interrupt_Enable)	32	RW	0000_0000h
B8h	MAC Rx Transmit Status (MAC_Rx_Tx_Status)	32	RW	0000_0000h
F8h	MAC Physical Interface Control Status (MAC_PHYIF_Control_Status)	32	RW	0000_0000h
110h	MAC Version (MAC_Version)	32	R	0000_1052h
114h	MAC Debug (MAC_Debug)	32	R	0000_0000h
11Ch	MAC Hardware Feature 0 (MAC_HW_Feature0)	32	R	0E09_5337h
120h	MAC Hardware Feature 1 (MAC_HW_Feature1)	32	R	431B_29E7h
124h	MAC Hardware Feature 2 (MAC_HW_Feature2)	32	R	0408_2082h
128h	MAC Hardware Feature 3 (MAC_HW_Feature3)	32	R	2C37_1635h
140h	MAC DPP FSM Interrupt Status (MAC_DPP_FSM_Interrupt_Status)	32	RW	0000_0000h
148h	MAC FSM Control (MAC_FSM_Control)	32	RW	0000_0000h
14Ch	MAC FSM ACT Timer (MAC_FSM_ACT_Timer)	32	RW	0000_0000h
150h	SCS REG 1 (SCS_REG1)	32	RW	0000_0000h
200h	MAC MDIO Address (MAC_MDIO_Address)	32	RW	0000_0000h
204h	MAC MDIO Data (MAC_MDIO_Data)	32	RW	0000_0000h
210h	MAC ARP Address (MAC_ARP_Address)	32	RW	0000_0000h
230h	MAC CSR Software Control (MAC_CSR_SW_Ctrl)	32	RW	0000_0000h
234h	MAC FPE Control STS (MAC_FPE_CTRL_STS)	32	RW	0000_0000h
238h	MAC Extended Configuration 1 (MAC_Ext_Cfg1)	32	RW	0002_0002h
240h	MAC Presentation Time (MAC_Presn_Time_ns)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
244h	MAC Presentation Time Update (MAC_Presn_Time_Updt)	32	RW	0000_0000h
300h	MAC Address 0 High (MAC_Address0_High)	32	RW	8000_FFFFh
304h	MAC Address 0 Low (MAC_Address0_Low)	32	RW	FFFF_FFFFh
308h	MAC Address 1 High (MAC_Address1_High)	32	RW	0000_FFFFh
30Ch	MAC Address 1 Low (MAC_Address1_Low)	32	RW	FFFF_FFFFh
310h	MAC Address 2 High (MAC_Address2_High)	32	RW	0000_FFFFh
314h	MAC Address 2 Low (MAC_Address2_Low)	32	RW	FFFF_FFFFh
700h	MMC Control (MMC_Control)	32	RW	0000_0000h
704h	MMC Receive Interrupt (MMC_Rx_Interrupt)	32	R	0000_0000h
708h	MMC Transmit Interrupt (MMC_Tx_Interrupt)	32	R	0000_0000h
70Ch	MMC Receive Interrupt Mask (MMC_Rx_Interrupt_Mask)	32	RW	0000_0000h
710h	MMC Transmit Interrupt Mask (MMC_Tx_Interrupt_Mask)	32	RW	0000_0000h
714h	Transmit Octet Count Good Bad (Tx_Octet_Count_Good_Bad)	32	R	0000_0000h
718h	Transmit Packet Count Good Bad (Tx_Packet_Count_Good_Bad)	32	R	0000_0000h
71Ch	Transmit Broadcast Packets Good (Tx_Broadcast_Packets_Good)	32	R	0000_0000h
720h	Transmit Multicast Packets Good (Tx_Multicast_Packets_Good)	32	R	0000_0000h
724h	Transmit 64-Octet Packets Good Bad (Tx_64Octets_Packets_Good_Bad)	32	R	0000_0000h
728h	Transmit 65 to 127 Octet Packets Good Bad (Tx_65To127Octets_Packets_Good_Bad)	32	R	0000_0000h
72Ch	Transmit 128 to 255 Octet Packets Good Bad (Tx_128To255Octets_Packets_Good_Bad)	32	R	0000_0000h
730h	Transmit 256 to 511 Octet Packets Good Bad (Tx_256To511Octets_Packets_Good_Bad)	32	R	0000_0000h
734h	Transmit 512 to 1023 Octet Packets Good Bad (Tx_512To1023Octets_Packets_Good_Bad)	32	R	0000_0000h
738h	Transmit 1024 to Max Octet Packets Good Bad (Tx_1024ToMaxOctets_Packets_Good_Bad)	32	R	0000_0000h
73Ch	Transmit Unicast Packets Good Bad (Tx_Unicast_Packets_Good_Bad)	32	R	0000_0000h
740h	Transmit Multicast Packets Good Bad (Tx_Multicast_Packets_Good_Bad)	32	R	0000_0000h
744h	Transmit Broadcast Packets Good Bad (Tx_Broadcast_Packets_Good_Bad)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
748h	Transmit Underflow Error Packets (Tx_Underflow_Error_Packets)	32	R	0000_0000h
74Ch	Transmit Single Collision Good Packets (Tx_Single_Collision_Good_Packets)	32	R	0000_0000h
750h	Transmit Multiple Collision Good Packets (Tx_Multiple_Collision_Good_Packets)	32	R	0000_0000h
754h	Transmit Deferred Packets (Tx_Deferred_Packets)	32	R	0000_0000h
758h	Transmit Late Collision Packets (Tx_Late_Collision_Packets)	32	R	0000_0000h
75Ch	Transmit Excessive Collision Packets (Tx_Excessive_Collision_Packets)	32	R	0000_0000h
760h	Transmit Carrier Error Packets (Tx_Carrier_Error_Packets)	32	R	0000_0000h
764h	Transmit Octet Count Good (Tx_Octet_Count_Good)	32	R	0000_0000h
768h	Transmit Packet Count Good (Tx_Packet_Count_Good)	32	R	0000_0000h
76Ch	Transmit Excessive Deferral Error (Tx_Excessive_Deferral_Error)	32	R	0000_0000h
770h	Transmit Pause Packets (Tx_Pause_Packets)	32	R	0000_0000h
774h	Transmit VLAN Packets Good (Tx_VLAN_Packets_Good)	32	R	0000_0000h
778h	Transmit O Size Packets Good (Tx_OSize_Packets_Good)	32	R	0000_0000h
780h	Receive Packets Count Good Bad (Rx_Packets_Count_Good_Bad)	32	R	0000_0000h
784h	Receive Octet Count Good Bad (Rx_Octet_Count_Good_Bad)	32	R	0000_0000h
788h	Receive Octet Count Good (Rx_Octet_Count_Good)	32	R	0000_0000h
78Ch	Receive Broadcast Packets Good (Rx_Broadcast_Packets_Good)	32	R	0000_0000h
790h	Receive Multicast Packets Good (Rx_Multicast_Packets_Good)	32	R	0000_0000h
794h	Receive CRC Error Packets (Rx_CRC_Error_Packets)	32	R	0000_0000h
798h	Receive Alignment Error Packets (Rx_Alignment_Error_Packets)	32	R	0000_0000h
79Ch	Receive Runt Error Packets (Rx_Runt_Error_Packets)	32	R	0000_0000h
7A0h	Receive Jabber Error Packets (Rx_Jabber_Error_Packets)	32	R	0000_0000h
7A4h	Receive Undersize Packets Good (Rx_Undersize_Packets_Good)	32	R	0000_0000h
7A8h	Receive Oversize Packets Good (Rx_Oversize_Packets_Good)	32	R	0000_0000h
7ACh	Receive 64 Octets Packets Good Bad (Rx_64Octets_Packets_Good_Bad)	32	R	0000_0000h
7B0h	Receive 65 to 127 Octets Packets Good Bad (Rx_65To127Octets_Packets_Good_Bad)	32	R	0000_0000h
7B4h	Receive 128 to 255 Octets Packets Good Bad (Rx_128To255Octets_Packets_Good_Bad)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
7B8h	Receive 256 to 511 Octets Packets Good Bad (Rx_256To511Octets_Packets_Good_Bad)	32	R	0000_0000h
7BCh	Receive 512 to 1023 Octets Packets Good Bad (Rx_512To1023Octets_Packets_Good_Bad)	32	R	0000_0000h
7C0h	Receive 1024 to Max Octets Packets Good Bad (Rx_1024ToMaxOctets_Packets_Good_Bad)	32	R	0000_0000h
7C4h	Receive Unicast Packets Good (Rx_Unicast_Packets_Good)	32	R	0000_0000h
7C8h	Receive Length Error Packets (Rx_Length_Error_Packets)	32	R	0000_0000h
7CCh	Receive Out of Range Type Packet (Rx_Out_Of_Range_Type_Packets)	32	R	0000_0000h
7D0h	Receive Pause Packets (Rx_Pause_Packets)	32	R	0000_0000h
7D4h	Receive FIFO Overflow Packets (Rx_FIFO_Overflow_Packets)	32	R	0000_0000h
7D8h	Receive VLAN Packets Good Bad (Rx_VLAN_Packets_Good_Bad)	32	R	0000_0000h
7DCh	Receive Watchdog Error Packets (Rx_Watchdog_Error_Packets)	32	R	0000_0000h
7E0h	Receive Error Packets (Rx_Receive_Error_Packets)	32	R	0000_0000h
7E4h	Receive Control Packets Good (Rx_Control_Packets_Good)	32	R	0000_0000h
8A0h	MMC Transmit FPE Fragment Counter Interrupt Status (MMC_FPE_Tx_Interrupt)	32	R	0000_0000h
8A4h	MMC FPE Transmit Interrupt Mask (MMC_FPE_Tx_Interrupt_Mask)	32	RW	0000_0000h
8A8h	Transmit FPE Fragment Counter (MMC_Tx_FPE_Fragment_Cntr)	32	R	0000_0000h
8ACh	Transmit Hold Request Counter (MMC_Tx_Hold_Req_Cntr)	32	R	0000_0000h
8C0h	MMC Receive Packet Assembly Error Counter Interrupt Status (MMC_FPE_Rx_Interrupt)	32	R	0000_0000h
8C4h	MMC FPE Receive Interrupt Mask (MMC_FPE_Rx_Interrupt_Mask)	32	RW	0000_0000h
8C8h	MMC Receive Packet Assembly Error Counter (MMC_Rx_Packet_Assembly_Err_Cntr)	32	R	0000_0000h
8CCh	MMC Receive Packet SMD Error Counter (MMC_Rx_Packet_SMD_Err_Cntr)	32	R	0000_0000h
8D0h	MMC Receive Packet Assembly OK Counter (MMC_Rx_Packet_Assembly_OK_Cntr)	32	R	0000_0000h
8D4h	MMC Receive FPE Fragment Counter (MMC_Rx_FPE_Fragment_Cntr)	32	R	0000_0000h
900h	MAC Layer 3 Layer 4 Control 0 (MAC_L3_L4_Control0)	32	RW	0000_0000h
904h	MAC Layer 4 Address 0 (MAC_Layer4_Address0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
910h	MAC Layer 3 Address 0 Reg 0 (MAC_Layer3_Addr0_Reg0)	32	RW	0000_0000h
914h	MAC Layer 3 Address 1 Reg 0 (MAC_Layer3_Addr1_Reg0)	32	RW	0000_0000h
918h	MAC Layer 3 Address 2 Reg 0 (MAC_Layer3_Addr2_Reg0)	32	RW	0000_0000h
91Ch	MAC Layer 3 Address 3 Reg 0 (MAC_Layer3_Addr3_Reg0)	32	RW	0000_0000h
930h	MAC Layer 3 Layer 4 Control 1 (MAC_L3_L4_Control1)	32	RW	0000_0000h
934h	MAC Layer 4 Address 1 (MAC_Layer4_Address1)	32	RW	0000_0000h
940h	MAC Layer 3 Address 0 Reg 1 (MAC_Layer3_Addr0_Reg1)	32	RW	0000_0000h
944h	MAC Layer 3 Address 1 Reg 1 (MAC_Layer3_Addr1_Reg1)	32	RW	0000_0000h
948h	MAC Layer 3 Address 2 Reg 1 (MAC_Layer3_Addr2_Reg1)	32	RW	0000_0000h
94Ch	MAC Layer 3 Address 3 Reg 1 (MAC_Layer3_Addr3_Reg1)	32	RW	0000_0000h
960h	MAC Layer 3 Layer 4 Control 2 (MAC_L3_L4_Control2)	32	RW	0000_0000h
964h	MAC Layer 4 Address 2 (MAC_Layer4_Address2)	32	RW	0000_0000h
970h	MAC Layer 3 Address 0 Reg 2 (MAC_Layer3_Addr0_Reg2)	32	RW	0000_0000h
974h	MAC Layer 3 Address 1 Reg 2 (MAC_Layer3_Addr1_Reg2)	32	RW	0000_0000h
978h	MAC Layer 3 Address 2 Reg 2 (MAC_Layer3_Addr2_Reg2)	32	RW	0000_0000h
97Ch	MAC Layer 3 Address 3 Reg 2 (MAC_Layer3_Addr3_Reg2)	32	RW	0000_0000h
990h	MAC Layer 3 Layer 4 Control 3 (MAC_L3_L4_Control3)	32	RW	0000_0000h
994h	MAC Layer 4 Address 3 (MAC_Layer4_Address3)	32	RW	0000_0000h
9A0h	MAC Layer 3 Address 0 Reg 3 (MAC_Layer3_Addr0_Reg3)	32	RW	0000_0000h
9A4h	MAC Layer 3 Address 1 Reg 3 (MAC_Layer3_Addr1_Reg3)	32	RW	0000_0000h
9A8h	MAC Layer 3 Address 2 Reg 3 (MAC_Layer3_Addr2_Reg3)	32	RW	0000_0000h
9ACh	MAC Layer 3 Address 3 Reg 3 (MAC_Layer3_Addr3_Reg3)	32	RW	0000_0000h
9C0h	MAC Layer 3 Layer 4 Control 4 (MAC_L3_L4_Control4)	32	RW	0000_0000h
9C4h	MAC Layer 4 Address 4 (MAC_Layer4_Address4)	32	RW	0000_0000h
9D0h	MAC Layer 3 Address 0 Reg 4 (MAC_Layer3_Addr0_Reg4)	32	RW	0000_0000h
9D4h	MAC Layer 3 Address 1 Reg 4 (MAC_Layer3_Addr1_Reg4)	32	RW	0000_0000h
9D8h	MAC Layer 3 Address 2 Reg 4 (MAC_Layer3_Addr2_Reg4)	32	RW	0000_0000h
9DCh	MAC Layer 3 Address 3 Reg 4 (MAC_Layer3_Addr3_Reg4)	32	RW	0000_0000h
9F0h	MAC Layer 3 Layer 4 Control 5 (MAC_L3_L4_Control5)	32	RW	0000_0000h
9F4h	MAC Layer 4 Address 5 (MAC_Layer4_Address5)	32	RW	0000_0000h
A00h	MAC Layer 3 Address 0 Reg 5 (MAC_Layer3_Addr0_Reg5)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
A04h	MAC Layer 3 Address 1 Reg 5 (MAC_Layer3_Addr1_Reg5)	32	RW	0000_0000h
A08h	MAC Layer 3 Address 2 Reg 5 (MAC_Layer3_Addr2_Reg5)	32	RW	0000_0000h
A0Ch	MAC Layer 3 Address 3 Reg 5 (MAC_Layer3_Addr3_Reg5)	32	RW	0000_0000h
A20h	MAC Layer 3 Layer 4 Control 6 (MAC_L3_L4_Control6)	32	RW	0000_0000h
A24h	MAC Layer 4 Address 6 (MAC_Layer4_Address6)	32	RW	0000_0000h
A30h	MAC Layer 3 Address 0 Reg 6 (MAC_Layer3_Addr0_Reg6)	32	RW	0000_0000h
A34h	MAC Layer 3 Address 1 Reg 6 (MAC_Layer3_Addr1_Reg6)	32	RW	0000_0000h
A38h	MAC Layer 3 Address 2 Reg 6 (MAC_Layer3_Addr2_Reg6)	32	RW	0000_0000h
A3Ch	MAC Layer 3 Address 3 Reg 6 (MAC_Layer3_Addr3_Reg6)	32	RW	0000_0000h
A50h	MAC Layer 3 Layer 4 Control 7 (MAC_L3_L4_Control7)	32	RW	0000_0000h
A54h	MAC Layer 4 Address 7 (MAC_Layer4_Address7)	32	RW	0000_0000h
A60h	MAC Layer 3 Address 0 Reg 7 (MAC_Layer3_Addr0_Reg7)	32	RW	0000_0000h
A64h	MAC Layer 3 Address 1 Reg 7 (MAC_Layer3_Addr1_Reg7)	32	RW	0000_0000h
A68h	MAC Layer 3 Address 2 Reg 7 (MAC_Layer3_Addr2_Reg7)	32	RW	0000_0000h
A6Ch	MAC Layer 3 Address 3 Reg 7 (MAC_Layer3_Addr3_Reg7)	32	RW	0000_0000h
A70h	MAC Indirect Access Control (MAC_Indir_Access_Ctrl)	32	RW	0000_0000h
A74h	MAC Indirect Access Data (MAC_Indir_Access_Data)	32	RW	0000_0000h
A74h	MAC TMR Queue Regs 0 (MAC_TMRQ_Regs0)	32	RW	0000_0000h
A74h	MAC TMR Queue Regs 1 (MAC_TMRQ_Regs1)	32	RW	0000_0000h
A74h	MAC TMR Queue Regs 2 (MAC_TMRQ_Regs2)	32	RW	0000_0000h
A74h	MAC TMR Queue Regs 3 (MAC_TMRQ_Regs3)	32	RW	0000_0000h
A74h	MAC TMR Queue Regs 4 (MAC_TMRQ_Regs4)	32	RW	0000_0000h
A74h	MAC TMR Queue Regs 5 (MAC_TMRQ_Regs5)	32	RW	0000_0000h
A74h	MAC TMR Queue Regs 6 (MAC_TMRQ_Regs6)	32	RW	0000_0000h
A74h	MAC TMR Queue Regs 7 (MAC_TMRQ_Regs7)	32	RW	0000_0000h
B00h	MAC Timestamp Control (MAC_Timestamp_Control)	32	RW	0000_2000h
B04h	MAC Sub Second Increment (MAC_Sub_Second_Increment)	32	RW	0000_0000h
B08h	MAC System Time In Seconds (MAC_System_Time_Seconds)	32	R	0000_0000h
B0Ch	MAC System Time In Nanoseconds (MAC_System_Time_Nanoseconds)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
B10h	MAC System Time Seconds Update (MAC_System_Time_Seconds_Update)	32	RW	0000_0000h
B14h	MAC System Time Nanoseconds Update (MAC_System_Time_Nanoseconds_Update)	32	RW	0000_0000h
B18h	MAC Timestamp Addend (MAC_Timestamp_Addend)	32	RW	0000_0000h
B1Ch	MAC System Time Higher Word In Seconds (MAC_System_Time_Higher_Word_Seconds)	32	RW	0000_0000h
B20h	MAC Timestamp Status (MAC_Timestamp_Status)	32	R	0000_0000h
B30h	MAC Transmit Timestamp Status In Nanoseconds (MAC_Tx_Timestamp_Status_Nanoseconds)	32	R	0000_0000h
B34h	MAC Transmit Timestamp Status In Seconds (MAC_Tx_Timestamp_Status_Seconds)	32	R	0000_0000h
B50h	MAC Timestamp Ingress Asymmetry Correction (MAC_Timestamp_Ingress_Asym_Corr)	32	RW	0000_0000h
B54h	MAC Timestamp Egress Asymmetry Correction (MAC_Timestamp_Egress_Asym_Corr)	32	RW	0000_0000h
B58h	MAC Timestamp Ingress Correction In Nanoseconds (MAC_Timestamp_Ingress_Corr_Nanosecond)	32	RW	0000_0000h
B5Ch	MAC Timestamp Egress Correction In Nanoseconds (MAC_Timestamp_Egress_Corr_Nanosecond)	32	RW	0000_0000h
B60h	MAC Timestamp Ingress Correction In Subnanoseconds (MAC_Timestamp_Ingress_Corr_Subnanosec)	32	RW	0000_0000h
B64h	MAC Timestamp Egress Correction In Subnanoseconds (MAC_Timestamp_Egress_Corr_Subnanosec)	32	RW	0000_0000h
B68h	MAC Timestamp Ingress Latency (MAC_Timestamp_Ingress_Latency)	32	R	0000_0000h
B6Ch	MAC Timestamp Egress Latency (MAC_Timestamp_Egress_Latency)	32	R	0000_0000h
B70h	MAC PPS Control (MAC_PPS_Control)	32	RW	0000_0000h
B80h	MAC PPS0 Target Time In Seconds (MAC_PPS0_Target_Time_Seconds)	32	RW	0000_0000h
B84h	MAC PPS0 Target Time In Nanoseconds (MAC_PPS0_Target_Time_Nanoseconds)	32	RW	0000_0000h
B88h	MAC PPS0 Interval (MAC_PPS0_Interval)	32	RW	0000_0000h
B8Ch	MAC PPS0 Width (MAC_PPS0_Width)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
B90h	MAC PPS1 Target Time In Seconds (MAC_PPS1_Target_Time_Seconds)	32	RW	0000_0000h
B94h	MAC PPS1 Target Time In Nanoseconds (MAC_PPS1_Target_Time_Nanoseconds)	32	RW	0000_0000h
B98h	MAC PPS1 Interval (MAC_PPS1_Interval)	32	RW	0000_0000h
B9Ch	MAC PPS1 Width (MAC_PPS1_Width)	32	RW	0000_0000h
BA0h	MAC PPS2 Target Time In Seconds (MAC_PPS2_Target_Time_Seconds)	32	RW	0000_0000h
BA4h	MAC PPS2 Target Time In Nanoseconds (MAC_PPS2_Target_Time_Nanoseconds)	32	RW	0000_0000h
BA8h	MAC PPS2 Interval (MAC_PPS2_Interval)	32	RW	0000_0000h
BACh	MAC PPS2 Width (MAC_PPS2_Width)	32	RW	0000_0000h
BB0h	MAC PPS3 Target Time In Seconds (MAC_PPS3_Target_Time_Seconds)	32	RW	0000_0000h
BB4h	MAC PPS3 Target Time In Nanoseconds (MAC_PPS3_Target_Time_Nanoseconds)	32	RW	0000_0000h
BB8h	MAC PPS3 Interval (MAC_PPS3_Interval)	32	RW	0000_0000h
BBCh	MAC PPS3 Width (MAC_PPS3_Width)	32	RW	0000_0000h
C00h	MTL Operation Mode (MTL_Operation_Mode)	32	RW	0000_0000h
C08h	MTL Debus Control (MTL_DBG_CTL)	32	RW	0000_0000h
C0Ch	MTL Debus Status (MTL_DBG_STS)	32	RW	0000_0018h
C10h	MTL FIFO Debug Data (MTL_FIFO_Debug_Data)	32	RW	0000_0000h
C20h	MTL Interrupt Status (MTL_Interrupt_Status)	32	R	0000_0000h
C30h	MTL Receive Queue DMA Map 0 (MTL_RxQ_DMA_Map0)	32	RW	0000_0000h
C40h	MTL TBS Control (MTL_TBS_CTRL)	32	RW	0000_0000h
C50h	MTL EST Control (MTL_EST_Control)	32	RW	0000_0000h
C54h	MTL EST Extended Control (MTL_EST_Ext_Control)	32	RW	0000_0000h
C58h	MTL EST Status (MTL_EST_Status)	32	RW	0000_0000h
C60h	MTL EST Scheduling Error (MTL_EST_Sch_Error)	32	RW	0000_0000h
C64h	MTL EST Frame Size Error (MTL_EST_Frm_Size_Error)	32	RW	0000_0000h
C68h	MTL EST Frame Size Capture (MTL_EST_Frm_Size_Capture)	32	R	0000_0000h
C70h	MTL EST Interrupt Enable (MTL_EST_Intr_Enable)	32	RW	0000_0000h
C80h	MTL EST GCL Control (MTL_EST_GCL_Control)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
C84h	MTL EST GCL Data (MTL_EST_GCL_Data)	32	RW	0000_0000h
C90h	MTL FPE Control Status (MTL_FPE_CTRL_STS)	32	RW	0000_0000h
C94h	MTL FPE Advance (MTL_FPE_Advance)	32	RW	0000_0000h
CA0h	MTL Rx Parser Control Status (MTL_RXP_Control_Status)	32	RW	803F_003Fh
CA4h	MTL Rx Parser Interrupt Control Status (MTL_RXP_Interrupt_Control_Status)	32	RW	0000_0000h
CA8h	MTL Rx Parser Drop Count (MTL_RXP_Drop_Cnt)	32	R	0000_0000h
CACh	MTL Rx Parser Error Count (MTL_RXP_Error_Cnt)	32	R	0000_0000h
CB0h	MTL Rx Parser Indirect Access Control Status (MTL_RXP_Indirect_Acc_Control_Status)	32	RW	0000_0000h
CB4h	MTL Rx Parser Indirect Access Data (MTL_RXP_Indirect_Acc_Data)	32	R	0000_0000h
CB8h	MTL Rx Parser Bypass Count (MTL_RXP_Bypass_Cnt)	32	R	0000_0000h
CC0h	MTL ECC Control (MTL_ECC_Control)	32	RW	0000_000Fh
CC4h	MTL Safety Interrupt Status (MTL_Safety_Interrupt_Status)	32	R	0000_0000h
CC8h	MTL ECC Interrupt Enable (MTL_ECC_Interrupt_Enable)	32	RW	0000_1111h
CCCh	MTL ECC Interrupt Status (MTL_ECC_Interrupt_Status)	32	RW	0000_0000h
CD0h	MTL ECC Error Status (MTL_ECC_Err_Sts_Rctl)	32	RW	0000_0000h
CD4h	MTL ECC Error Address Status (MTL_ECC_Err_Addr_Status)	32	R	0000_0000h
CD8h	MTL ECC Error Control Status (MTL_ECC_Err_Cntr_Status)	32	R	0000_0000h
CE0h	MTL DPP Control (MTL_DPP_Control)	32	RW	0000_0003h
CE4h	MTL DPP ECC Error Injection Channel (MTL_DPP_ECC_EIC)	32	RW	0000_0000h
D00h	MTL Tx Queue 0 Operation Mode (MTL_TxQ0_Operation_Mode)	32	RW	0000_0000h
D04h	MTL Tx Queue 0 Underflow (MTL_TxQ0_Underflow)	32	R	0000_0000h
D08h	MTL Tx Queue 0 Debug (MTL_TxQ0_Debug)	32	R	0000_0000h
D14h	MTL Tx Queue 0 ETS Staus (MTL_TxQ0_ETS_Status)	32	R	0000_0000h
D18h	MTL Tx Queue 0 Quantum Weight (MTL_TxQ0_Quantum_Weight)	32	RW	0000_0000h
D2Ch	MTL Queue 0 Interrupt Control Status (MTL_Q0_Interrupt_Control_Status)	32	RW	0000_0000h
D30h	MTL Rx Queue 0 Operation Mode (MTL_RxQ0_Operation_Mode)	32	RW	0000_0000h
D34h	MTL Rx Queue 0 Missed Packet Overflow Count (MTL_RxQ0_Missed_Packet_Overflow_Cnt)	32	R	0000_0000h
D38h	MTL Rx Queue 0 Debug (MTL_RxQ0_Debug)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
D3Ch	MTL Tx Queue 0 Control (MTL_RxQ0_Control)	32	RW	0000_0000h
D40h	MTL Tx Queue 1 Operation Mode (MTL_TxQ1_Operation_Mode)	32	RW	0000_0000h
D44h	MTL Tx Queue 1 Underflow (MTL_TxQ1_Underflow)	32	R	0000_0000h
D48h	MTL Tx Queue 1 Debug (MTL_TxQ1_Debug)	32	R	0000_0000h
D50h	MTL Tx Queue 1 ETS Control (MTL_TxQ1_ETS_Control)	32	RW	0000_0000h
D54h	MTL Tx Queue 1 ETS_Status (MTL_TxQ1_ETS_Status)	32	R	0000_0000h
D58h	MTL Tx Queue 1 Quantum Weight (MTL_TxQ1_Quantum_Weight)	32	RW	0000_0000h
D5Ch	MTL Tx Queue 1 Sendslope Credit (MTL_TxQ1_SendSlopeCredit)	32	RW	0000_0000h
D60h	MTL Tx Queue 1 Hi Credit (MTL_TxQ1_HiCredit)	32	RW	0000_0000h
D64h	MTL Tx Queue 1 Lo Credit (MTL_TxQ1_LoCredit)	32	RW	0000_0000h
D6Ch	MTL Queue 1 Interrupt Control Status (MTL_Q1_Interrupt_Control_Status)	32	RW	0000_0000h
D70h	MTL Rx Queue 1 Operation Mode (MTL_RxQ1_Operation_Mode)	32	RW	0000_0000h
D74h	MTL Rx Queue 1 Missed Packet Overflow Counter (MTL_RxQ1_Missed_Packet_Overflow_Cnt)	32	R	0000_0000h
D78h	MTL Rx Queue 1 Debug (MTL_RxQ1_Debug)	32	R	0000_0000h
D7Ch	MTL Rx Queue 1 Control (MTL_RxQ1_Control)	32	RW	0000_0000h
D80h	MTL Tx Queue 2 Operation Mode (MTL_TxQ2_Operation_Mode)	32	RW	0000_0000h
D84h	MTL Tx Queue 2 Underflow (MTL_TxQ2_Underflow)	32	R	0000_0000h
D88h	MTL Tx Queue 2 Debug (MTL_TxQ2_Debug)	32	R	0000_0000h
D90h	MTL Tx Queue 2 ETS Control (MTL_TxQ2_ETS_Control)	32	RW	0000_0000h
D94h	MTL Tx Queue 2 ETS Status (MTL_TxQ2_ETS_Status)	32	R	0000_0000h
D98h	MTL Tx Queue 2 Quantum Weight (MTL_TxQ2_Quantum_Weight)	32	RW	0000_0000h
D9Ch	MTL Tx Queue 2 SendSlope Credit (MTL_TxQ2_SendSlopeCredit)	32	RW	0000_0000h
DA0h	MTL Tx Queue 2 Hi Credit (MTL_TxQ2_HiCredit)	32	RW	0000_0000h
DA4h	MTL Tx Queue 2 Lo Credit (MTL_TxQ2_LoCredit)	32	RW	0000_0000h
DACh	MTL Queue 2 Interrupt Control Status (MTL_Q2_Interrupt_Control_Status)	32	RW	0000_0000h
DB0h	MTL Rx Queue 2 Operation Mode (MTL_RxQ2_Operation_Mode)	32	RW	0000_0000h
DB4h	MTL Rx Queue 2 Missed Packet Overflow Counter (MTL_RxQ2_Missed_Packet_Overflow_Cnt)	32	R	0000_0000h
DB8h	MTL Rx Queue 2 Debug (MTL_RxQ2_Debug)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
DBCh	MTL Rx Queue 2 Control (MTL_RxQ2_Control)	32	RW	0000_0000h
1000h	DMA Mode (DMA_Mode)	32	RW	0000_0000h
1004h	DMA System Bus Mode (DMA_SysBus_Mode)	32	RW	0000_0000h
1008h	DMA Interrupt Status (DMA_Interrupt_Status)	32	R	0000_0000h
100Ch	DMA Debug Status 0 (DMA_Debug_Status0)	32	R	0000_0000h
1050h	DMA TBS Control 0 (DMA_TBS_CTRL0)	32	RW	0000_0000h
1054h	DMA TBS Control 1 (DMA_TBS_CTRL1)	32	RW	0000_0000h
1058h	DMA TBS Control 2 (DMA_TBS_CTRL2)	32	RW	0000_0000h
105Ch	DMA TBS Control 3 (DMA_TBS_CTRL3)	32	RW	0000_0000h
1080h	DMA Safety Interrupt Status (DMA_Safety_Interrupt_Status)	32	R	0000_0000h
1100h	DMA Channel 0 Control (DMA_CH0_Control)	32	RW	0000_0000h
1104h	DMA Channel 0 Tx Control (DMA_CH0_Tx_Control)	32	RW	0000_0000h
1108h	DMA Channel 0 Rx Control (DMA_CH0_Rx_Control)	32	RW	0000_0000h
1114h	DMA Channel 0 Tx Descriptor List Address (DMA_CH0_TxDesc_List_Address)	32	RW	0000_0000h
111Ch	DMA Channel 0 Rx Descriptor List Address (DMA_CH0_RxDesc_List_Address)	32	RW	0000_0000h
1120h	DMA Channel 0 Tx Descriptor Tail Pointer (DMA_CH0_TxDesc_Tail_Pointer)	32	RW	0000_0000h
1128h	DMA Channel 0 Rx Descriptor Tail Pointer (DMA_CH0_RxDesc_Tail_Pointer)	32	RW	0000_0000h
112Ch	DMA Channel 0 Tx Descriptor Ring Length (DMA_CH0_TxDesc_Ring_Length)	32	RW	0000_0000h
1130h	DMA Channel 0 Rx Control 2 (DMA_CH0_Rx_Control2)	32	RW	0000_0000h
1134h	DMA Channel 0 Interrupt Enable (DMA_CH0_Interrupt_Enable)	32	RW	0000_0000h
1138h	DMA Channel 0 Rx Interrupt Watchdog Timer (DMA_CH0_Rx_Interrupt_Watchdog_Timer)	32	RW	0000_0000h
113Ch	DMA Channel 0 Slot Function Control Status (DMA_CH0_Slot_Function_Control_Status)	32	RW	0000_07C0h
1144h	DMA Channel 0 Current Application Transmit Descriptor (DMA_CH0_Current_App_TxDesc)	32	R	0000_0000h
114Ch	DMA Channel 0 Current Application Receive Descriptor (DMA_CH0_Current_App_RxDesc)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1154h	DMA Channel 0 Current Application Transmit Buffer (DMA_CH0_Current_App_TxBuffer)	32	R	0000_0000h
115Ch	DMA Channel 0 Current Application Receive Buffer (DMA_CH0_Current_App_RxBuffer)	32	R	0000_0000h
1160h	DMA Channel 0 Status (DMA_CH0_Status)	32	RW	0000_0000h
1164h	DMA Channel 0 Miss Frame Counter (DMA_CH0_Miss_Frame_Cnt)	32	R	0000_0000h
1168h	DMA Channel 0 Rx Parser Accept Count (DMA_CH0_RXP_Accept_Cnt)	32	R	0000_0000h
116Ch	DMA Channel 0 Rx ERI Count (DMA_CH0_RX_ERI_Cnt)	32	R	0000_0000h
1180h	DMA Channel 1 Control (DMA_CH1_Control)	32	RW	0000_0000h
1184h	DMA Channel 1 Tx Control (DMA_CH1_Tx_Control)	32	RW	0000_0000h
1188h	DMA Channel 1 Rx Control (DMA_CH1_Rx_Control)	32	RW	0000_0000h
1194h	DMA Channel 1 Tx Descriptor List Address (DMA_CH1_TxDesc_List_Address)	32	RW	0000_0000h
119Ch	DMA Channel 1 Rx Descriptor List Address (DMA_CH1_RxDesc_List_Address)	32	RW	0000_0000h
11A0h	DMA Channel 1 Tx Descriptor Tail Pointer (DMA_CH1_TxDesc_Tail_Pointer)	32	RW	0000_0000h
11A8h	DMA Channel 1 Rx Descriptor Tail Pointer (DMA_CH1_RxDesc_Tail_Pointer)	32	RW	0000_0000h
11ACh	DMA Channel 1 Tx Descriptor Ring Length (DMA_CH1_TxDesc_Ring_Length)	32	RW	0000_0000h
11B0h	DMA Channel 1 Rx Control 2 (DMA_CH1_Rx_Control2)	32	RW	0000_0000h
11B4h	DMA Channel 1 Interrupt Enable (DMA_CH1_Interrupt_Enable)	32	RW	0000_0000h
11B8h	DMA Channel 1 Rx Interrupt Watchdog Timer (DMA_CH1_Rx_Interrupt_Watchdog_Timer)	32	RW	0000_0000h
11BCh	DMA Channel 1 Slot Function Control Status (DMA_CH1_Slot_Function_Control_Status)	32	RW	0000_07C0h
11C4h	DMA Channel 1 Current Application Transmit Descriptor (DMA_CH1_Current_App_TxDesc)	32	R	0000_0000h
11CCh	DMA Channel 1 Current Application Receive Descriptor (DMA_CH1_Current_App_RxDesc)	32	R	0000_0000h
11D4h	DMA Channel 1 Current Application Transmit Buffer (DMA_CH1_Current_App_TxBuffer)	32	R	0000_0000h
11DCh	DMA Channel 1 Current Application Receive Buffer (DMA_CH1_Current_App_RxBuffer)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
11E0h	DMA Channel 1 Status (DMA_CH1_Status)	32	RW	0000_0000h
11E4h	DMA Channel 1 Miss Frame Counter (DMA_CH1_Miss_Frame_Cnt)	32	R	0000_0000h
11E8h	DMA Channel 1 Rx Parser Accept Count (DMA_CH1_RXP_Accept_Cnt)	32	R	0000_0000h
11ECh	DMA Channel 1 Rx ERI Count (DMA_CH1_RX_ERI_Cnt)	32	R	0000_0000h
1200h	DMA Channel 2 Control (DMA_CH2_Control)	32	RW	0000_0000h
1204h	DMA Channel 2 Tx Control (DMA_CH2_Tx_Control)	32	RW	0000_0000h
1208h	DMA Channel 2 Rx Control (DMA_CH2_Rx_Control)	32	RW	0000_0000h
1214h	DMA Channel 2 Tx Descriptor List Address (DMA_CH2_TxDesc_List_Address)	32	RW	0000_0000h
121Ch	DMA Channel 2 Rx Descriptor List Address (DMA_CH2_RxDesc_List_Address)	32	RW	0000_0000h
1220h	DMA Channel 2 Tx Descriptor Tail Pointer (DMA_CH2_TxDesc_Tail_Pointer)	32	RW	0000_0000h
1228h	DMA Channel 2 Rx Descriptor Tail Pointer (DMA_CH2_RxDesc_Tail_Pointer)	32	RW	0000_0000h
122Ch	DMA Channel 2 Tx Descriptor Ring Length (DMA_CH2_TxDesc_Ring_Length)	32	RW	0000_0000h
1230h	DMA Channel 2 Rx Control 2 (DMA_CH2_Rx_Control2)	32	RW	0000_0000h
1234h	DMA Channel 2 Interrupt Enable (DMA_CH2_Interrupt_Enable)	32	RW	0000_0000h
1238h	DMA Channel 2 Rx Interrupt Watchdog Timer (DMA_CH2_Rx_Interrupt_Watchdog_Timer)	32	RW	0000_0000h
123Ch	DMA Channel 2 Slot Function Control Status (DMA_CH2_Slot_Function_Control_Status)	32	RW	0000_07C0h
1244h	DMA Channel 2 Current Application Transmit Descriptor (DMA_CH2_Current_App_TxDesc)	32	R	0000_0000h
124Ch	DMA Channel 2 Current Application Receive Descriptor (DMA_CH2_Current_App_RxDesc)	32	R	0000_0000h
1254h	DMA Channel 2 Current Application Transmit Buffer (DMA_CH2_Current_App_TxBuffer)	32	R	0000_0000h
125Ch	DMA Channel 2 Current Application Receive Buffer (DMA_CH2_Current_App_RxBuffer)	32	R	0000_0000h
1260h	DMA Channel 2 Status (DMA_CH2_Status)	32	RW	0000_0000h
1264h	DMA Channel 2 Miss Frame Count (DMA_CH2_Miss_Frame_Cnt)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1268h	DMA Channel 2 Rx Parser Accept Count (DMA_CH2_RXP_Accept_Cnt)	32	R	0000_0000h
126Ch	DMA Channel 2 Rx ERI Count (DMA_CH2_RX_ERI_Cnt)	32	R	0000_0000h

76.17.2 MAC Configuration (MAC_Configuration)

Offset

Register	Offset
MAC_Configuration	0h

Function
The MAC Configuration Register establishes the operating mode of the MAC.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ARPEN	SARC			IPC	IPG			GPSLCE	S2KP	CST	ACS	WD	BE	JD	JE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PS	FES	DM	LM	ECRSFD	DO	DCRS	DR	Reserved	BL		DC	PRELEN		TE	RE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 ARPEN	ARP Offload Enable When this bit is set, the MAC can recognize an incoming ARP request packet and schedules the ARP packet for transmission. It forwards the ARP packet to the application and also indicate the events in the RxStatus. When this bit is reset, the MAC receiver does not recognize any ARP packet and indicates them as Type frame in the RxStatus. This bit is available only when the Enable IPv4 ARP Offload is selected. 0b - ARP Offload is disabled 1b - ARP Offload is enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
30-28 SARC	Source Address Insertion or Replacement Control This field controls the source address insertion or replacement for all transmitted packets. Bit 30 specifies which MAC Address register (0 or 1) is used for source address insertion or replacement based on the values of Bits[29:28]: 2'b0x: - The mti_sa_ctrl_i and ati_sa_ctrl_i input signals control the SA field generation. 2'b10: - If Bit 30 is set to 0, the MAC inserts the content of the MAC Address 0 registers in the SA field of all transmitted packets. - If Bit 30 is set to 1 and the Enable MAC Address Register 1 option is selected while configuring the IP, the MAC inserts the content of the MAC Address 1 registers in the SA field of all transmitted packets. 2'b11: - If Bit 30 is set to 0, the MAC replaces the content of the MAC Address 0 registers in the SA field of all transmitted packets. - If Bit 30 is set to 1 and the MAC Address Register 1 is enabled, the MAC replaces the content of the MAC Address 1 registers in the SA field of all transmitted packets. Note: Changes to this field take effect only on the start of a packet. If you write to this register field when a packet is being transmitted, only the subsequent packet can use the updated value, that is, the current packet does not use the updated value. 000b - mti_sa_ctrl_i and ati_sa_ctrl_i input signals control the SA field generation 010b - Contents of MAC Addr-0 inserted in SA field 011b - Contents of MAC Addr-0 replaces SA field 110b - Contents of MAC Addr-1 inserted in SA field 111b - Contents of MAC Addr-1 replaces SA field
27 IPC	Checksum Offload When set, this bit enables the IPv4 header checksum checking and IPv4 or IPv6 TCP, UDP, or ICMP payload checksum checking. When this bit is reset, the COE function in the receiver is disabled. The Layer 3 and Layer 4 Packet Filter and Enable Split Header features automatically selects the IPC Full Checksum Offload Engine on the Receive side. When any of these features are enabled, you must set the IPC bit. 0b - IP header/payload checksum checking is disabled 1b - IP header/payload checksum checking is enabled
26-24 IPG	Inter-Packet Gap These bits control the minimum IPG between packets during transmission. This range of minimum IPG is valid in full-duplex mode. In the half-duplex mode, the minimum IPG can be configured only for 64-bit times (IPG = 100). Lower values are not considered. When a JAM pattern is being transmitted because of backpressure activation, the MAC does not consider the minimum IPG. This function (IPG less than 96 bit times) is valid only when EIPGEN bit in MAC_Ext_Configuration register is reset. When EIPGEN is set, then the minimum IPG (greater than 96 bit times) is controlled as per the description given in EIPG field in MAC_Ext_Configuration register. 000b - 96 bit times IPG 001b - 88 bit times IPG 010b - 80 bit times IPG 011b - 72 bit times IPG 100b - 64 bit times IPG

Table continues on the next page...

Table continued from the previous page...

Field	Function
	101b - 56 bit times IPG 110b - 48 bit times IPG 111b - 40 bit times IPG
23 GPSLCE	Giant Packet Size Limit Control Enable When this bit is set, the MAC considers the value in GPSL field in MAC_Ext_Configuration register to declare a received packet as Giant packet. This field must be programmed to more than 1,518 bytes. Otherwise, the MAC considers 1,518 bytes as giant packet limit. When this bit is reset, the MAC considers a received packet as Giant packet when its size is greater than 1,518 bytes (1522 bytes for tagged packet). The watchdog timeout limit, Jumbo Packet Enable and 2K Packet Enable have higher precedence over this bit, that is the MAC considers a received packet as Giant packet when its size is greater than 9,018 bytes (9,022 bytes for tagged packet) with Jumbo Packet Enabled and greater than 2,000 bytes with 2K Packet Enabled. The watchdog timeout, if enabled, terminates the received packet when watchdog limit is reached. Therefore, the programmed giant packet limit should be less than the watchdog limit to get the giant packet status. 0b - Giant Packet Size Limit Control is disabled 1b - Giant Packet Size Limit Control is enabled
22 S2KP	IEEE 802.3as Support for 2K Packets When this bit is set, the MAC considers all packets with up to 2,000 bytes length as normal packets. When the JE bit is not set, the MAC considers all received packets of size more than 2K bytes as Giant packets. When this bit is reset and the JE bit is not set, the MAC considers all received packets of size more than 1,518 bytes (1,522 bytes for tagged) as giant packets. For more information about how the setting of this bit and the JE bit impact the Giant packet status, see the Table, Giant Packet Status based on S2KP and JE Bits. Note: When the JE bit is set, setting this bit has no effect on the giant packet status. 0b - Support upto 2K packet is disabled 1b - Support upto 2K packet is Enabled
21 CST	CRC stripping for Type packets When this bit is set, the last four bytes (FCS) of all packets of Ether type (type field greater than 1,536) are stripped and dropped before forwarding the packet to the application. Note: For information about how the settings of the ACS bit and this bit impact the packet length, see the Table, Packet Length based on the CST and ACS Bits. 0b - CRC stripping for Type packets is disabled 1b - CRC stripping for Type packets is enabled
20 ACS	Automatic Pad or CRC Stripping When this bit is set, the MAC strips the Pad or FCS field on the incoming packets only if the value of the length field is less than 1,536 bytes. All received packets with length field greater than or equal to 1,536 bytes are passed to the application without stripping the Pad or FCS field. When this bit is reset, the MAC passes all incoming packets to the application, without any modification. Note: For information

Table continues on the next page...

Table continued from the previous page...

Field	Function
	about how the settings of CST bit and this bit impact the packet length, see the Table, Packet Length based on the CST and ACS Bit . 0b - Automatic Pad or CRC Stripping is disabled 1b - Automatic Pad or CRC Stripping is enabled
19 WD	Watchdog Disable When this bit is set, the MAC disables the watchdog timer on the receiver. The MAC can receive packets of up to 16,383 bytes. When this bit is reset, the MAC does not allow more than 2,048 bytes (10,240 if JE is set high) of the packet being received. The MAC cuts off any bytes received after 2,048 bytes. 0b - Watchdog is enabled 1b - Watchdog is disabled
18 BE	Packet Burst Enable When this bit is set, the MAC allows packet bursting during transmission in the GMII half-duplex mode. 0b - Packet Burst is disabled 1b - Packet Burst is enabled
17 JD	Jabber Disable When this bit is set, the MAC disables the jabber timer on the transmitter. The MAC can transfer packets of up to 16,383 bytes. When this bit is reset, if the application sends more than 2,048 bytes of data (10,240 if JE is set high) during transmission, the MAC does not send rest of the bytes in that packet. 0b - Jabber is enabled 1b - Jabber is disabled
16 JE	Jumbo Packet Enable When this bit is set, the MAC allows jumbo packets of 9,018 bytes (9,022 bytes for VLAN tagged packets) without reporting a giant packet error in the Rx packet status. 0b - Jumbo packet is disabled 1b - Jumbo packet is enabled
15 PS	Port Select This bit selects the Ethernet line speed. This bit, along with Bit 14, selects the exact line speed. In the 10/100 Mbps-only (always 1) or 1000 Mbps-only (always 0) configurations, this bit is read-only (RO) with appropriate value. In default 10/100/1000 Mbps configurations, this bit is read-write (R/W). The mac_speed_o[1] signal reflects the value of this bit. 0b - For 1000 or 2500 Mbps operations 1b - For 10 or 100 Mbps operations
14 FES	Speed This bit selects the speed mode. The mac_speed_o[0] signal reflects the value of this bit.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - 10 Mbps when PS bit is 1 and 1 Gbps when PS bit is 0 1b - 100 Mbps when PS bit is 1 and 2.5 Gbps when PS bit is 0
13 DM	Duplex Mode When this bit is set, the MAC operates in the full-duplex mode in which it can transmit and receive simultaneously. This bit is RO with default value of 1'b1 in the full-duplex-only configurations. 0b - Half-duplex mode 1b - Full-duplex mode
12 LM	Loopback Mode When this bit is set, the MAC operates in the loopback mode at GMII or MII. The (G)MII Rx clock input (CLK_RX_I) is required for the loopback to work properly. This is because the Tx clock is not internally looped back. 0b - Loopback is disabled 1b - Loopback is enabled
11 ECRSFD	Enable Carrier Sense Before Transmission in Full-Duplex Mode When this bit is set, the MAC transmitter checks the CRS signal before packet transmission in the full-duplex mode. The MAC starts the transmission only when the CRS signal is low. When this bit is reset, the MAC transmitter ignores the status of the CRS signal. 0b - ECRSFD is disabled 1b - ECRSFD is enabled
10 DO	Disable Receive Own When this bit is set, the MAC disables the reception of packets when the gmii_txen_o is asserted in the half-duplex mode. When this bit is reset, the MAC receives all packets given by the PHY. This bit is not applicable in the full-duplex mode. 0b - Enable Receive Own 1b - Disable Receive Own
9 DCRS	Disable Carrier Sense During Transmission When this bit is set, the MAC transmitter ignores the (G)MII CRS signal during packet transmission in the half-duplex mode. As a result, no errors are generated because of Loss of Carrier or No Carrier during transmission. When this bit is reset, the MAC transmitter generates errors because of Carrier Sense. The MAC can even abort the transmission. 0b - Enable Carrier Sense During Transmission 1b - Disable Carrier Sense During Transmission
8 DR	Disable Retry When this bit is set, the MAC attempts only one transmission. When a collision occurs on the GMII or MII interface, the MAC ignores the current packet transmission and reports a Packet Abort with excessive

Table continues on the next page...

Table continued from the previous page...

Field	Function
	collision error in the Tx packet status. When this bit is reset, the MAC retries based on the settings of the BL field. This bit is applicable only in the half-duplex mode. 0b - Enable Retry 1b - Disable Retry
7 —	Reserved
6-5 BL	Back-Off Limit The back-off limit determines the random integer number (r) of slot time delays (4,096 bit times for 1000/2500 Mbps; 512 bit times for 10/100 Mbps) for which the MAC waits before rescheduling a transmission attempt during retries after a collision. n = retransmission attempt. The random integer r takes the value in the range 0 <= r < 2^k This bit is applicable only in the half-duplex mode. 00b - k = min(n,10) 01b - k = min(n,8) 10b - k = min(n,4) 11b - k = min(n,1)
4 DC	Deferral Check When this bit is set, the deferral check function is enabled in the MAC. The MAC issues a Packet Abort status, along with the excessive deferral error bit set in the Tx packet status, when the Tx state machine is deferred for more than 24,288 bit times in 10 or 100 Mbps mode. If the MAC is configured for 1000/2500 Mbps operation, the threshold for deferral is 155,680 bits times. Deferral begins when the transmitter is ready to transmit, but it is prevented because of an active carrier sense signal (CRS) on GMII or MII. The defer time is not cumulative. For example, if the transmitter defers for 10,000 bit times because the CRS signal is active and the CRS signal becomes inactive, the transmitter transmits and collision happens. Because of collision, the transmitter needs to back off and then defer again after back off completion. In such a scenario, the deferral timer is reset to 0, and it is restarted. When this bit is reset, the deferral check function is disabled and the MAC defers until the CRS signal goes inactive. This bit is applicable only in the half-duplex mode. 0b - Deferral check function is disabled 1b - Deferral check function is enabled
3-2 PRELEN	Preamble Length for Transmit packets These bits control the number of preamble bytes that are added to the beginning of every Tx packet. The preamble reduction occurs only when the MAC is operating in the full-duplex mode. 00b - 7 bytes of preamble 01b - 5 bytes of preamble 10b - 3 bytes of preamble 11b - Reserved
1	Transmitter Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
TE	When this bit is set, the Tx state machine of the MAC is enabled for transmission on the GMII or MII interface. When this bit is reset, the MAC Tx state machine is disabled after it completes the transmission of the current packet. The Tx state machine does not transmit any more packets. 0b - Transmitter is disabled 1b - Transmitter is enabled
0 RE	Receiver Enable When this bit is set, the Rx state machine of the MAC is enabled for receiving packets from the GMII or MII interface. When this bit is reset, the MAC Rx state machine is disabled after it completes the reception of the current packet. The Rx state machine does not receive any more packets from the GMII or MII interface. 0b - Receiver is disabled 1b - Receiver is enabled

76.17.3 MAC Extended Configuration (MAC_Ext_Configuration)

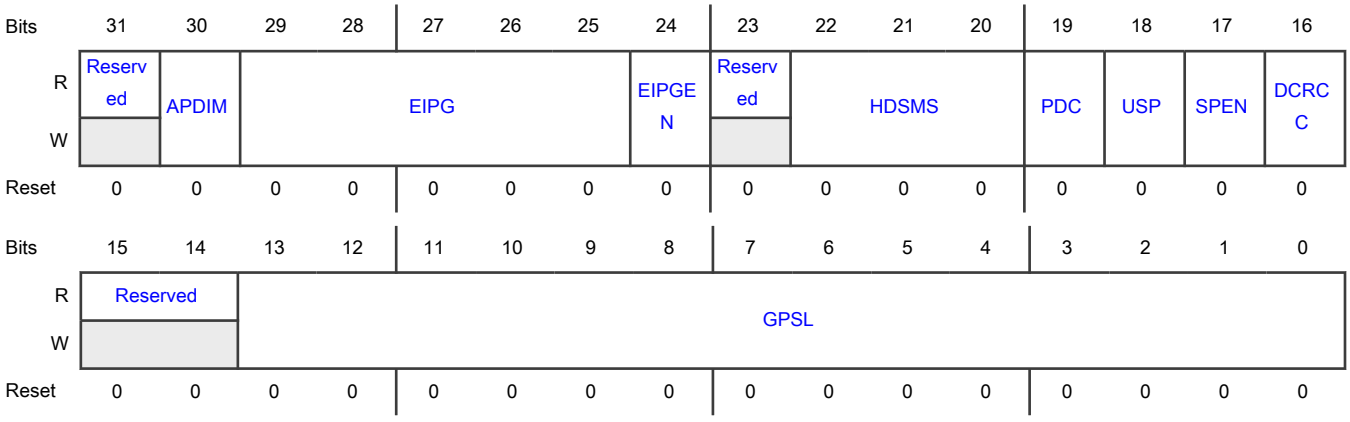
Offset

Register	Offset
MAC_Ext_Configuration	4h

Function

The MAC Extended Configuration Register establishes the operating mode of the MAC.

Diagram



Fields

Field	Function
31 —	Reserved
30 APDIM	ARP Packet Drop if IP Address Mismatch When this bit is set, Packet for which Target Protocol Address does not match IPv4 address is dropped in the MTL layer. When this bit is reset, when target Protocol Address does not match, packet is forwarded to MTL maintaining backward compatibility 0b - mux select to drop the arp packet if target protocol address mismatches IPv4 address disabled 1b - mux select to drop the arp packet if target protocol address mismatches IPv4 address enabled
29-25 EIPG	Extended Inter-Packet Gap The value in this field is applicable when the EIPGEN bit is set. This field (as Most Significant bits), along with IPG field in MAC_Configuration register, gives the minimum IPG greater than 96 bit times in steps of 8 bit times: {EIPG, IPG} 8'h00 - 104 bit times 8'h01 - 112 bit times 8'h02 - 120 bit times ---------- 8'hFF - 2144 bit times
24 EIPGEN	Extended Inter-Packet Gap Enable When this bit is set, the MAC interprets EIPG field and IPG field in MAC_Configuration register together as minimum IPG greater than 96 bit times in steps of 8 bit times. When this bit is reset, the MAC ignores EIPG field and interprets IPG field in MAC_Configuration register as minimum IPG less than or equal to 96 bit times in steps of 8 bit times. Note: Enable the extended Inter-Packet Gap feature only when operating in Full-Duplex mode. There might be undesirable effects on back-pressure function and frame transmission if it is enabled in Half-Duplex mode. 0b - Extended Inter-Packet Gap is disabled 1b - Extended Inter-Packet Gap is enabled
23 —	Reserved
22-20 HDSMS	Maximum Size for Splitting the Header Data These bits indicate the maximum header size allowed for splitting the header data in the received packet. 000b - Maximum Size for Splitting the Header Data is 64 bytes 001b - Maximum Size for Splitting the Header Data is 128 bytes 010b - Maximum Size for Splitting the Header Data is 256 bytes 011b - Maximum Size for Splitting the Header Data is 512 bytes 100b - Maximum Size for Splitting the Header Data is 1024 bytes 101b - Reserved
19	Packet Duplication Control

Table continues on the next page...

Table continued from the previous page...

Field	Function
PDC	When this bit is set, the received packet with Multicast/Broadcast Destination address is routed to multiple Receive DMA Channels. The Receive DMA Channels is identified by the DCS field of MAC_Address(#i)_High register corresponding to the MAC Address register that matches the Multicast/ Broadcast Destination address in the received packet. The DCS field is interpreted to be a one-hot value, each bit corresponding to the Receive DMA Channel. When this bit is reset, the received packet is routed to single Receive DMA Channel. The Receive DMA Channel is identified by the DCS field of MAC_Address(#i)_High register corresponding to the MAC Address register that matches the Destination address in the received packet. The DCS field is interpreted as a binary value. 0b - Packet Duplication Control is disabled 1b - Packet Duplication Control is enabled
18 USP	Unicast Slow Protocol Packet Detect When this bit is set, the MAC detects the Slow Protocol packets with unicast address of the station specified in the MAC_Address0_High and MAC_Address0_Low registers. The MAC also detects the Slow Protocol packets with the Slow Protocols multicast address (01-80-C2-00-00-02). When this bit is reset, the MAC detects only Slow Protocol packets with the Slow Protocol multicast address specified in the IEEE 802.3-2015, Section 5. 0b - Unicast Slow Protocol Packet Detection is disabled 1b - Unicast Slow Protocol Packet Detection is enabled
17 SPEN	Slow Protocol Detection Enable When this bit is set, MAC processes the Slow Protocol packets (Ether Type 0x8809) and provides the Slow Protocol Sub-Type and Code fields in Rx status. When this bit is reset, the MAC forwards all error-free Slow Protocol packets to the application. The MAC considers such packets as normal Type packets. 0b - Slow Protocol Detection is disabled 1b - Slow Protocol Detection is enabled
16 DCRCC	Disable CRC Checking for Received Packets When this bit is set, the MAC receiver does not check the CRC field in the received packets. When this bit is reset, the MAC receiver always checks the CRC field in the received packets. 0b - CRC Checking is enabled 1b - CRC Checking is disabled
15-14 —	Reserved
13-0 GPSL	Giant Packet Size Limit If the received packet size is greater than the value programmed in this field in units of bytes, the MAC declares the received packet as Giant packet. The value programmed in this field must be greater than or equal to 1,518 bytes. Any other programmed value is considered as 1,518 bytes. For VLAN tagged packets, the MAC adds 4 bytes to the programmed value. When the Enable Double VLAN Processing

Table continues on the next page...

Table continued from the previous page...

Field	Function
	option is selected, the MAC adds 8 bytes to the programmed value for double VLAN tagged packets. The value in this field is applicable when the GPSLCE bit is set in MAC_Configuration register.

76.17.4 MAC Packet Filter (MAC_Packet_Filter)

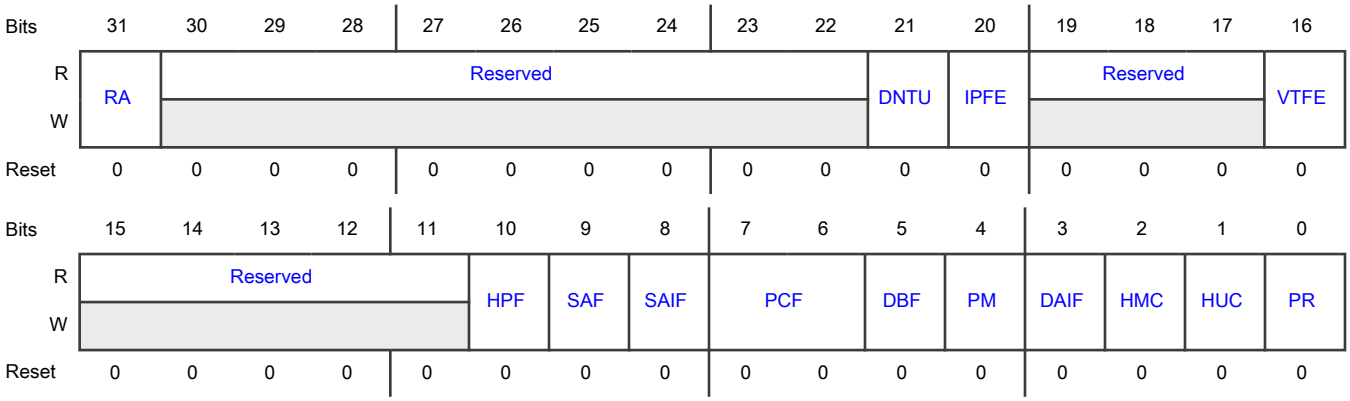
Offset

Register	Offset
MAC_Packet_Filter	8h

Function

The MAC Packet Filter register contains the filter controls for receiving packets. Some of the controls from this register go to the address check block of the MAC which performs the first level of address filtering. The second level of filtering is performed on the incoming packet based on other controls such as Pass Bad Packets and Pass Control Packets.

Diagram



Fields

Field	Function
31 RA	Receive All When this bit is set, the MAC Receiver module passes all received packets to the application, irrespective of whether they pass the address filter or not. The result of the SA or DA filtering is updated (pass or fail) in the corresponding bit in the Rx Status Word. When this bit is reset, the Receiver module passes only those packets to the application that pass the SA or DA address filter. 0b - Receive All is disabled 1b - Receive All is enabled
30-22 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
21 DNTU	Drop Non-TCP/UDP over IP Packets When this bit is set, the MAC drops the non-TCP or UDP over IP packets. The MAC forward only those packets that are processed by the Layer 4 filter. When this bit is reset, the MAC forwards all non-TCP or UDP over IP packets. 0b - Forward Non-TCP/UDP over IP Packets 1b - Drop Non-TCP/UDP over IP Packets
20 IPFE	Layer 3 and Layer 4 Filter Enable When this bit is set, the MAC drops packets that do not match the enabled Layer 3 and Layer 4 filters. If Layer 3 or Layer 4 filters are not enabled for matching, this bit does not have any effect. When this bit is reset, the MAC forwards all packets irrespective of the match status of the Layer 3 and Layer 4 fields. 0b - Layer 3 and Layer 4 Filters are disabled 1b - Layer 3 and Layer 4 Filters are enabled
19-17 —	Reserved
16 VTFE	VLAN Tag Filter Enable When this bit is set, the MAC drops the VLAN tagged packets that do not match the VLAN Tag. When this bit is reset, the MAC forwards all packets irrespective of the match status of the VLAN Tag. 0b - VLAN Tag Filter is disabled 1b - VLAN Tag Filter is enabled
15-11 —	Reserved
10 HPF	Hash or Perfect Filter When this bit is set, the address filter passes a packet if it matches either the perfect filtering or hash filtering as set by the HMC or HUC bit. When this bit is reset and the HUC or HMC bit is set, the packet is passed only if it matches the Hash filter. 0b - Hash or Perfect Filter is disabled 1b - Hash or Perfect Filter is enabled
9 SAF	Source Address Filter Enable When this bit is set, the MAC compares the SA field of the received packets with the values programmed in the enabled SA registers. If the comparison fails, the MAC drops the packet. When this bit is reset, the MAC forwards the received packet to the application with updated SAF bit of the Rx Status depending on the SA address comparison. Note: According to the IEEE specification, Bit 47 of the SA is reserved. However, in DWC_ether_qos, the MAC compares all 48 bits. The software driver should take this into consideration while programming the MAC address registers for SA. 0b - SA Filtering is disabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - SA Filtering is enabled
8 SAIF	SA Inverse Filtering When this bit is set, the Address Check block operates in the inverse filtering mode for SA address comparison. If the SA of a packet matches the values programmed in the SA registers, it is marked as failing the SA Address filter. When this bit is reset, if the SA of a packet does not match the values programmed in the SA registers, it is marked as failing the SA Address filter. 0b - SA Inverse Filtering is disabled 1b - SA Inverse Filtering is enabled
7-6 PCF	Pass Control Packets These bits control the forwarding of all control packets (including unicast and multicast Pause packets). 00b - MAC filters all control packets from reaching the application 01b - MAC forwards all control packets except Pause packets to the application even if they fail the Address filter 10b - MAC forwards all control packets to the application even if they fail the Address filter 11b - MAC forwards the control packets that pass the Address filter
5 DBF	Disable Broadcast Packets When this bit is set, the AFM module blocks all the incoming broadcast packets. In addition, it overrides all other filter settings. When this bit is reset, the AFM module passes all received broadcast packets. 0b - Enable Broadcast Packets 1b - Disable Broadcast Packets
4 PM	Pass All Multicast When this bit is set, it indicates that all the received packets with a multicast destination address (first bit in the destination address field is '1') are passed. When this bit is reset, filtering of multicast packet depends on HMC bit. 0b - Pass All Multicast is disabled 1b - Pass All Multicast is enabled
3 DAIF	DA Inverse Filtering When this bit is set, the Address Check block operates in inverse filtering mode for the DA address comparison for both unicast and multicast packets. When this bit is reset, normal filtering of packets is performed. 0b - DA Inverse Filtering is disabled 1b - DA Inverse Filtering is enabled
2 HMC	Hash Multicast When this bit is set, the MAC performs the destination address filtering of received multicast packets according to the hash table. When this bit is reset, the MAC performs the perfect destination address

Table continues on the next page...

Table continued from the previous page...

Field	Function
	filtering for multicast packets, that is, it compares the DA field with the values programmed in DA registers. 0b - Hash Multicast is disabled 1b - Hash Multicast is enabled
1 HUC	Hash Unicast When this bit is set, the MAC performs the destination address filtering of unicast packets according to the hash table. When this bit is reset, the MAC performs a perfect destination address filtering for unicast packets, that is, it compares the DA field with the values programmed in DA registers. 0b - Hash Unicast is disabled 1b - Hash Unicast is enabled
0 PR	Promiscuous Mode When this bit is set, the Address Filtering module passes all incoming packets irrespective of the destination or source address. The SA or DA Filter Fails status bits of the Rx Status Word are always cleared when PR is set. 0b - Promiscuous Mode is disabled 1b - Promiscuous Mode is enabled

76.17.5 MAC Watchdog Timeout (MAC_Watchdog_Timeout)

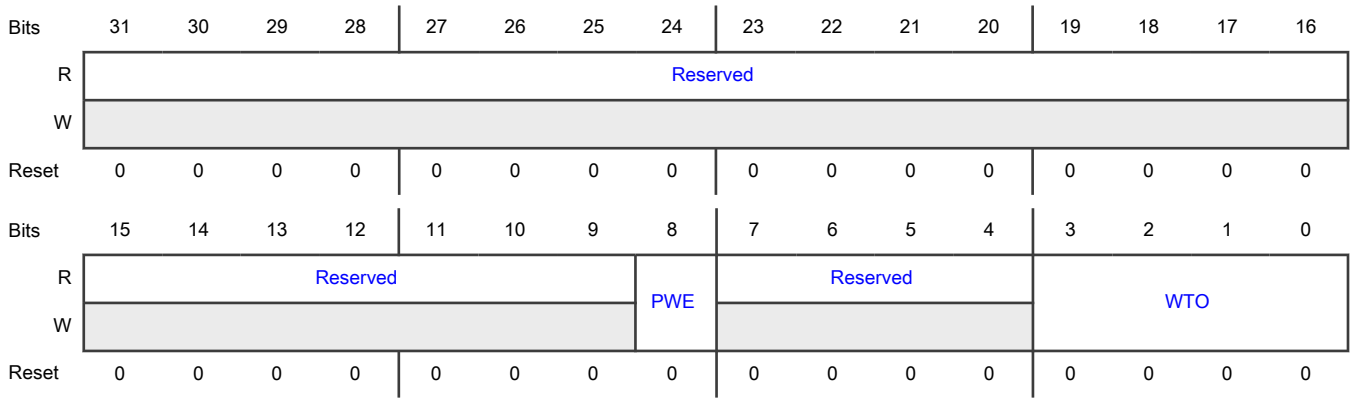
Offset

Register	Offset
MAC_Watchdog_Timeout	Ch

Function

The Watchdog Timeout register controls the watchdog timeout for received packets.

Diagram



Fields

Field	Function
31-9 —	Reserved
8 PWE	Programmable Watchdog Enable When this bit is set and the WD bit of the MAC_Configuration register is reset, the WTO field is used as watchdog timeout for a received packet. When this bit is cleared, the watchdog timeout for a received packet is controlled by setting of WD and JE bits in MAC_Configuration register. 0b - Programmable Watchdog is disabled 1b - Programmable Watchdog is enabled
7-4 —	Reserved
3-0 WTO	Watchdog Timeout When the PWE bit is set and the WD bit of the MAC_Configuration register is reset, this field is used as watchdog timeout for a received packet. If the length of a received packet exceeds the value of this field, such packet is terminated and declared as an error packet. Note: When the PWE bit is set, the value in this field should be more than 1,522 (0x05F2). Otherwise, the IEEE 802.3-specified valid tagged packets are declared as error packets and then dropped. 0000b - 2 KB 0001b - 3 KB 0010b - 4 KB 0011b - 5 KB 0100b - 6 KB 0101b - 7 KB 0110b - 8 KB 0111b - 9 KB 1000b - 10 KB 1001b - 11 KB 1010b - 12 KB 1011b - 13 KB 1100b - 14 KB 1101b - 15 KB 1110b - 16383 Bytes 1111b - Reserved

76.17.6 MAC Hash Table First 32 Bits (MAC_Hash_Table_Reg0)

Offset

Register	Offset
MAC_Hash_Table_Reg0	10h

Function

The Hash Table Register 0 contains the first 32 bits of the hash table, when the width of the hash table is 128 or 256 bits. You can specify the width of the hash table by using the Hash Table Size option in coreConsultant.

NOTE

In this chip, upper 6 bits should be used for 64-bit Hash Table.

The Hash table is used for group address filtering. For hash filtering, the content of the destination address in the incoming packet is passed through the CRC logic and the upper six (seven in 128-bit Hash or eight in 256-bit Hash) bits of the CRC register are used to index the content of the Hash table. The most significant bits determines the register to be used (Hash Table Register X), and the least significant five bits determine the bit within the register. For example, a hash value of 6'b100000 (in 64-bit Hash) selects Bit 0 of the Hash Table Register 1, a value of 7b'1110000 (in 128-bit Hash) selects Bit 16 of the Hash Table Register 3 and a value of 8b'10111111 (in 256-bit Hash) selects Bit 31 of the Hash Table Register 5.

The hash value of the destination address is calculated in the following way:

- Calculate the 32-bit CRC for the DA (See IEEE 802.3, Section 3.2.8 for the steps to calculate CRC32).

- Perform bitwise reversal for the value obtained in Step 1.

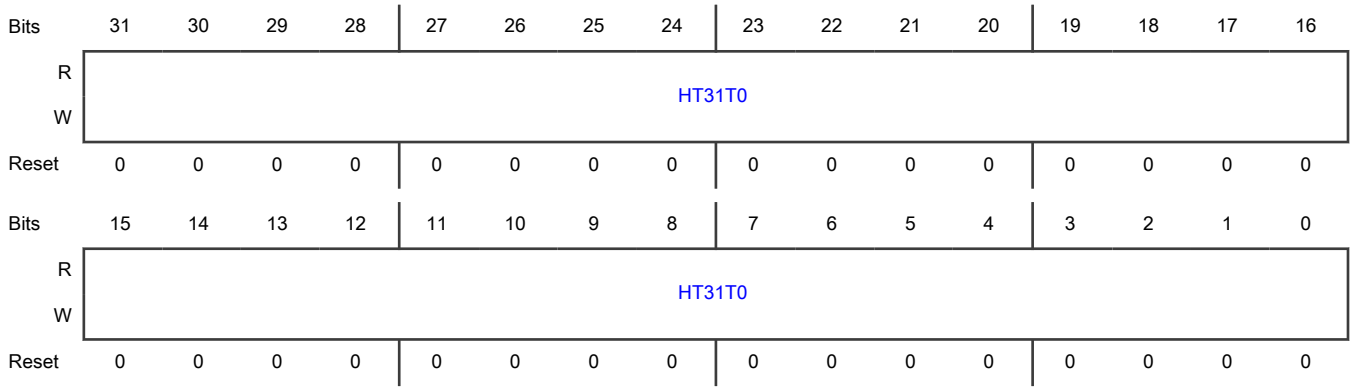
- Take the upper 6 (or 7 or 8) bits from the value obtained in Step 2.

If the corresponding bit value of the register is 1'b1, the packet is accepted. Otherwise, it is rejected. If the PM bit is set in MAC_Packet_Filter, all multicast packets are accepted regardless of the multicast hash values.

If the Hash Table register is configured to be double-synchronized to the (G)MII clock domain, the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the Hash Table Register X registers are written.

If double-synchronization is enabled, consecutive writes to this register should be performed after at least four clock cycles in the destination clock domain.

Diagram



Fields

Field	Function
31-0 HT31T0	MAC Hash Table First 32 Bits This field contains the first 32 Bits [31:0] of the Hash table.

76.17.7 MAC Hash Table Second 32 Bits (MAC_Hash_Table_Reg1)

Offset

Register	Offset
MAC_Hash_Table_Reg1	14h

Function

The Hash Table Register 0 contains the first 32 bits of the hash table, when the width of the hash table is 128 or 256 bits. You can specify the width of the hash table by using the Hash Table Size option in coreConsultant.

NOTE

In this chip, upper 6 bits should be used for 64-bit Hash Table.

The Hash table is used for group address filtering. For hash filtering, the content of the destination address in the incoming packet is passed through the CRC logic and the upper six (seven in 128-bit Hash or eight in 256-bit Hash) bits of the CRC register are used to index the content of the Hash table. The most significant bits determines the register to be used (Hash Table Register X), and the least significant five bits determine the bit within the register. For example, a hash value of 6'b100000 (in 64-bit Hash) selects Bit 0 of the Hash Table Register 1, a value of 7b'1110000 (in 128-bit Hash) selects Bit 16 of the Hash Table Register 3 and a value of 8b'10111111 (in 256-bit Hash) selects Bit 31 of the Hash Table Register 5.

The hash value of the destination address is calculated in the following way:

- Calculate the 32-bit CRC for the DA (See IEEE 802.3, Section 3.2.8 for the steps to calculate CRC32).

- Perform bitwise reversal for the value obtained in Step 1.

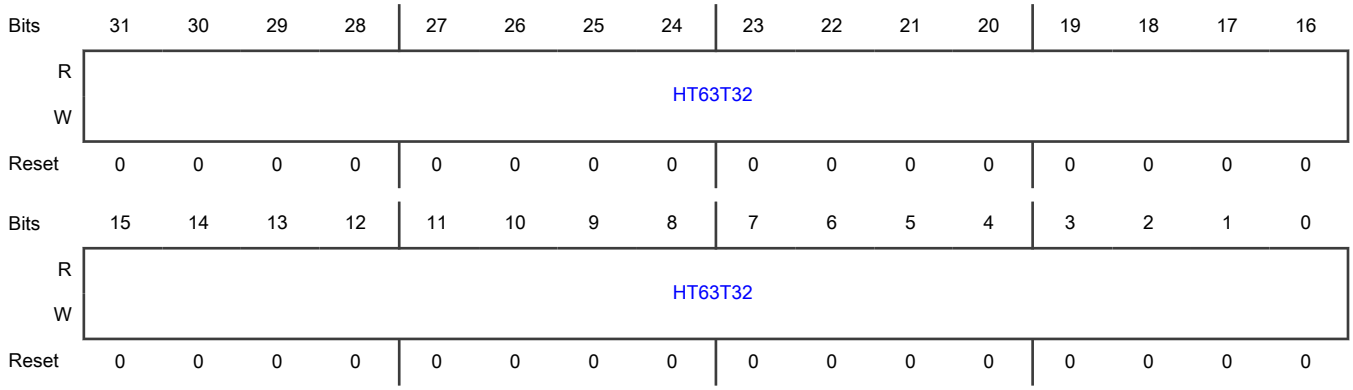
- Take the upper 6 (or 7 or 8) bits from the value obtained in Step 2.

If the corresponding bit value of the register is 1'b1, the packet is accepted. Otherwise, it is rejected. If the PM bit is set in MAC_Packet_Filter, all multicast packets are accepted regardless of the multicast hash values.

If the Hash Table register is configured to be double-synchronized to the (G)MII clock domain, the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the Hash Table Register X registers are written.

If double-synchronization is enabled, consecutive writes to this register should be performed after at least four clock cycles in the destination clock domain.

Diagram



Fields

Field	Function
31-0 HT63T32	MAC Hash Table Second 32 Bits This field contains the second 32 Bits [63:32] of the Hash table.

76.17.8 MAC Hash Table Third 32 Bits (MAC_Hash_Table_Reg2)

Offset

Register	Offset
MAC_Hash_Table_Reg2	18h

Function

The Hash Table Register 0 contains the first 32 bits of the hash table, when the width of the hash table is 128 or 256 bits. You can specify the width of the hash table by using the Hash Table Size option in coreConsultant.

NOTE

In this chip, upper 6 bits should be used for 64-bit Hash Table.

The Hash table is used for group address filtering. For hash filtering, the content of the destination address in the incoming packet is passed through the CRC logic and the upper six (seven in 128-bit Hash or eight in 256-bit Hash) bits of the CRC register are used to index the content of the Hash table. The most significant bits determines the register to be used (Hash Table Register X), and the least significant five bits determine the bit within the register. For example, a hash value of 6'b100000 (in 64-bit Hash) selects Bit 0 of the Hash Table Register 1, a value of 7b'1110000 (in 128-bit Hash) selects Bit 16 of the Hash Table Register 3 and a value of 8b'10111111 (in 256-bit Hash) selects Bit 31 of the Hash Table Register 5.

The hash value of the destination address is calculated in the following way:

- Calculate the 32-bit CRC for the DA (See IEEE 802.3, Section 3.2.8 for the steps to calculate CRC32).

- Perform bitwise reversal for the value obtained in Step 1.

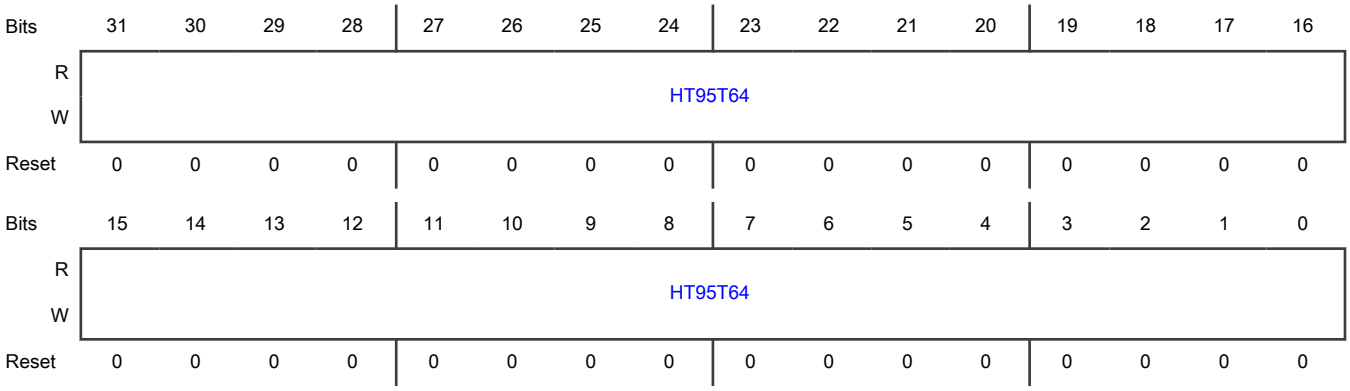
- Take the upper 6 (or 7 or 8) bits from the value obtained in Step 2.

If the corresponding bit value of the register is 1'b1, the packet is accepted. Otherwise, it is rejected. If the PM bit is set in MAC_Packet_Filter, all multicast packets are accepted regardless of the multicast hash values.

If the Hash Table register is configured to be double-synchronized to the (G)MII clock domain, the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the Hash Table Register X registers are written.

If double-synchronization is enabled, consecutive writes to this register should be performed after at least four clock cycles in the destination clock domain.

Diagram



Fields

Field	Function
31-0 HT95T64	MAC Hash Table Third 32 Bits This field contains the third 32 Bits [95:64] of the Hash table.

76.17.9 MAC Hash Table Fourth 32 Bits (MAC_Hash_Table_Reg3)

Offset

Register	Offset
MAC_Hash_Table_Reg3	1Ch

Function

The Hash Table Register 0 contains the first 32 bits of the hash table, when the width of the hash table is 128 or 256 bits. You can specify the width of the hash table by using the Hash Table Size option in coreConsultant.

NOTE

In this chip, upper 6 bits should be used for 64-bit Hash Table.

The Hash table is used for group address filtering. For hash filtering, the content of the destination address in the incoming packet is passed through the CRC logic and the upper six (seven in 128-bit Hash or eight in 256-bit Hash) bits of the CRC register are used to index the content of the Hash table. The most significant bits determines the register to be used (Hash Table Register X), and the least significant five bits determine the bit within the register. For example, a hash value of 6'b100000 (in 64-bit Hash) selects Bit 0 of the Hash Table Register 1, a value of 7b'1110000 (in 128-bit Hash) selects Bit 16 of the Hash Table Register 3 and a value of 8b'10111111 (in 256-bit Hash) selects Bit 31 of the Hash Table Register 5.

The hash value of the destination address is calculated in the following way:

- Calculate the 32-bit CRC for the DA (See IEEE 802.3, Section 3.2.8 for the steps to calculate CRC32).

- Perform bitwise reversal for the value obtained in Step 1.

- Take the upper 6 (or 7 or 8) bits from the value obtained in Step 2.

If the corresponding bit value of the register is 1'b1, the packet is accepted. Otherwise, it is rejected. If the PM bit is set in MAC_Packet_Filter, all multicast packets are accepted regardless of the multicast hash values.

If the Hash Table register is configured to be double-synchronized to the (G)MII clock domain, the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the Hash Table Register X registers are written.

If double-synchronization is enabled, consecutive writes to this register should be performed after at least four clock cycles in the destination clock domain.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	HT127T96															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	HT127T96															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 HT127T96	MAC Hash Table Fourth 32 Bits This field contains the fourth 32 Bits [127:96] of the Hash table.

76.17.10 MAC Hash Table Fifth 32 Bits (MAC_Hash_Table_Reg4)

Offset

Register	Offset
MAC_Hash_Table_Reg4	20h

Function

The Hash Table Register 0 contains the first 32 bits of the hash table, when the width of the hash table is 128 or 256 bits. You can specify the width of the hash table by using the Hash Table Size option in coreConsultant.

NOTE

In this chip, upper 6 bits should be used for 64-bit Hash Table.

The Hash table is used for group address filtering. For hash filtering, the content of the destination address in the incoming packet is passed through the CRC logic and the upper six (seven in 128-bit Hash or eight in 256-bit Hash) bits of the CRC register are used to index the content of the Hash table. The most significant bits determines the register to be used (Hash Table Register X), and the least significant five bits determine the bit within the register. For example, a hash value of 6'b100000 (in 64-bit Hash)

selects Bit 0 of the Hash Table Register 1, a value of 7b'1110000 (in 128-bit Hash) selects Bit 16 of the Hash Table Register 3 and a value of 8b'10111111 (in 256-bit Hash) selects Bit 31 of the Hash Table Register 5.

The hash value of the destination address is calculated in the following way:

- Calculate the 32-bit CRC for the DA (See IEEE 802.3, Section 3.2.8 for the steps to calculate CRC32).

- Perform bitwise reversal for the value obtained in Step 1.

- Take the upper 6 (or 7 or 8) bits from the value obtained in Step 2.

If the corresponding bit value of the register is 1'b1, the packet is accepted. Otherwise, it is rejected. If the PM bit is set in MAC_Packet_Filter, all multicast packets are accepted regardless of the multicast hash values.

If the Hash Table register is configured to be double-synchronized to the (G)MII clock domain, the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the Hash Table Register X registers are written.

If double-synchronization is enabled, consecutive writes to this register should be performed after at least four clock cycles in the destination clock domain.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	HT159T128															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	HT159T128															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 HT159T128	MAC Hash Table Fifth 32 Bits This field contains the fifth 32 Bits [159:128] of the Hash table.

76.17.11 MAC Hash Table Sixth 32 Bits (MAC_Hash_Table_Reg5)

Offset

Register	Offset
MAC_Hash_Table_Reg5	24h

Function

The Hash Table Register 0 contains the first 32 bits of the hash table, when the width of the hash table is 128 or 256 bits. You can specify the width of the hash table by using the Hash Table Size option in coreConsultant.

NOTE

In this chip, upper 6 bits should be used for 64-bit Hash Table.

The Hash table is used for group address filtering. For hash filtering, the content of the destination address in the incoming packet is passed through the CRC logic and the upper six (seven in 128-bit Hash or eight in 256-bit Hash) bits of the CRC register are used to index the content of the Hash table. The most significant bits determines the register to be used (Hash Table Register X), and the least significant five bits determine the bit within the register. For example, a hash value of 6'b100000 (in 64-bit Hash) selects Bit 0 of the Hash Table Register 1, a value of 7b'1110000 (in 128-bit Hash) selects Bit 16 of the Hash Table Register 3 and a value of 8b'10111111 (in 256-bit Hash) selects Bit 31 of the Hash Table Register 5.

The hash value of the destination address is calculated in the following way:

- Calculate the 32-bit CRC for the DA (See IEEE 802.3, Section 3.2.8 for the steps to calculate CRC32).

- Perform bitwise reversal for the value obtained in Step 1.

- Take the upper 6 (or 7 or 8) bits from the value obtained in Step 2.

If the corresponding bit value of the register is 1'b1, the packet is accepted. Otherwise, it is rejected. If the PM bit is set in MAC_Packet_Filter, all multicast packets are accepted regardless of the multicast hash values.

If the Hash Table register is configured to be double-synchronized to the (G)MII clock domain, the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the Hash Table Register X registers are written.

If double-synchronization is enabled, consecutive writes to this register should be performed after at least four clock cycles in the destination clock domain.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	HT191T160															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	HT191T160															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 HT191T160	MAC Hash Table Sixth 32 Bits This field contains the sixth 32 Bits [191:160] of the Hash table.

76.17.12 MAC Hash Table Seventh 32 Bits (MAC_Hash_Table_Reg6)

Offset

Register	Offset
MAC_Hash_Table_Reg6	28h

Function

The Hash Table Register 0 contains the first 32 bits of the hash table, when the width of the hash table is 128 or 256 bits. You can specify the width of the hash table by using the Hash Table Size option in coreConsultant.

NOTE

In this chip, upper 6 bits should be used for 64-bit Hash Table.

The Hash table is used for group address filtering. For hash filtering, the content of the destination address in the incoming packet is passed through the CRC logic and the upper six (seven in 128-bit Hash or eight in 256-bit Hash) bits of the CRC register are used to index the content of the Hash table. The most significant bits determines the register to be used (Hash Table Register X), and the least significant five bits determine the bit within the register. For example, a hash value of 6'b100000 (in 64-bit Hash) selects Bit 0 of the Hash Table Register 1, a value of 7b'1110000 (in 128-bit Hash) selects Bit 16 of the Hash Table Register 3 and a value of 8b'10111111 (in 256-bit Hash) selects Bit 31 of the Hash Table Register 5.

The hash value of the destination address is calculated in the following way:

- Calculate the 32-bit CRC for the DA (See IEEE 802.3, Section 3.2.8 for the steps to calculate CRC32).

- Perform bitwise reversal for the value obtained in Step 1.

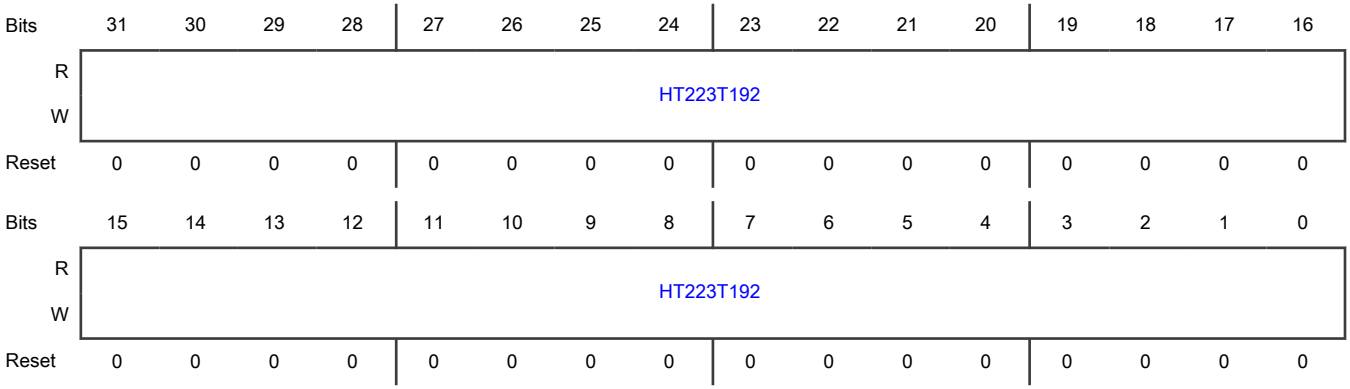
- Take the upper 6 (or 7 or 8) bits from the value obtained in Step 2.

If the corresponding bit value of the register is 1'b1, the packet is accepted. Otherwise, it is rejected. If the PM bit is set in MAC_Packet_Filter, all multicast packets are accepted regardless of the multicast hash values.

If the Hash Table register is configured to be double-synchronized to the (G)MII clock domain, the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the Hash Table Register X registers are written.

If double-synchronization is enabled, consecutive writes to this register should be performed after at least four clock cycles in the destination clock domain.

Diagram



Fields

Field	Function
31-0 HT223T192	MAC Hash Table Seventh 32 Bits This field contains the seventh 32 Bits [223:192] of the Hash table.

76.17.13 MAC Hash Table Eighth 32 Bits (MAC_Hash_Table_Reg7)

Offset

Register	Offset
MAC_Hash_Table_Reg7	2Ch

Function

The Hash Table Register 0 contains the first 32 bits of the hash table, when the width of the hash table is 128 or 256 bits. You can specify the width of the hash table by using the Hash Table Size option in coreConsultant.

NOTE

In this chip, upper 6 bits should be used for 64-bit Hash Table.

The Hash table is used for group address filtering. For hash filtering, the content of the destination address in the incoming packet is passed through the CRC logic and the upper six (seven in 128-bit Hash or eight in 256-bit Hash) bits of the CRC register are used to index the content of the Hash table. The most significant bits determines the register to be used (Hash Table Register X), and the least significant five bits determine the bit within the register. For example, a hash value of 6'b100000 (in 64-bit Hash) selects Bit 0 of the Hash Table Register 1, a value of 7b'1110000 (in 128-bit Hash) selects Bit 16 of the Hash Table Register 3 and a value of 8b'10111111 (in 256-bit Hash) selects Bit 31 of the Hash Table Register 5.

The hash value of the destination address is calculated in the following way:

- Calculate the 32-bit CRC for the DA (See IEEE 802.3, Section 3.2.8 for the steps to calculate CRC32).

- Perform bitwise reversal for the value obtained in Step 1.

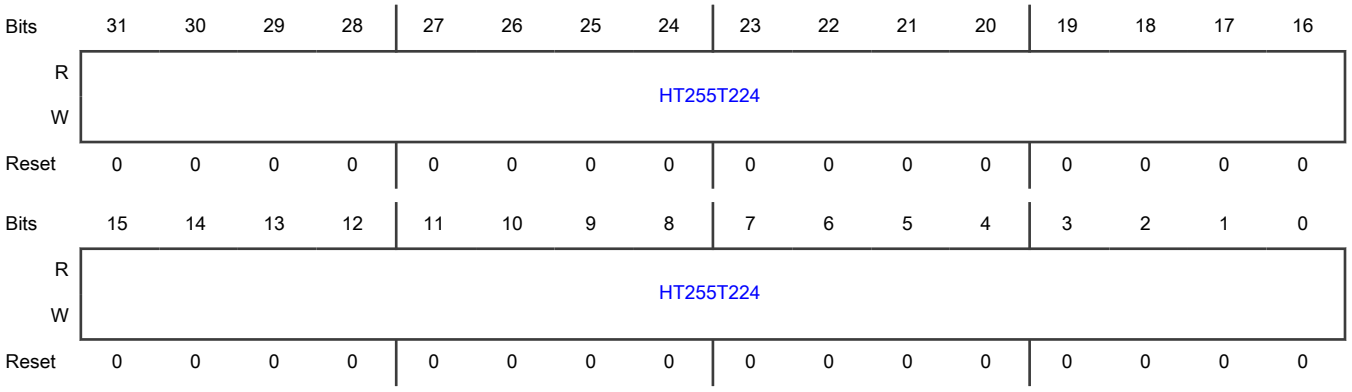
- Take the upper 6 (or 7 or 8) bits from the value obtained in Step 2.

If the corresponding bit value of the register is 1'b1, the packet is accepted. Otherwise, it is rejected. If the PM bit is set in MAC_Packet_Filter, all multicast packets are accepted regardless of the multicast hash values.

If the Hash Table register is configured to be double-synchronized to the (G)MII clock domain, the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the Hash Table Register X registers are written.

If double-synchronization is enabled, consecutive writes to this register should be performed after at least four clock cycles in the destination clock domain.

Diagram



Fields

Field	Function
31-0 HT255T224	MAC Hash Table Eighth 32 Bits This field contains the eighth 32 Bits [255:224] of the Hash table.

76.17.14 MAC VLAN Tag Control (MAC_VLAN_Tag_Ctrl)

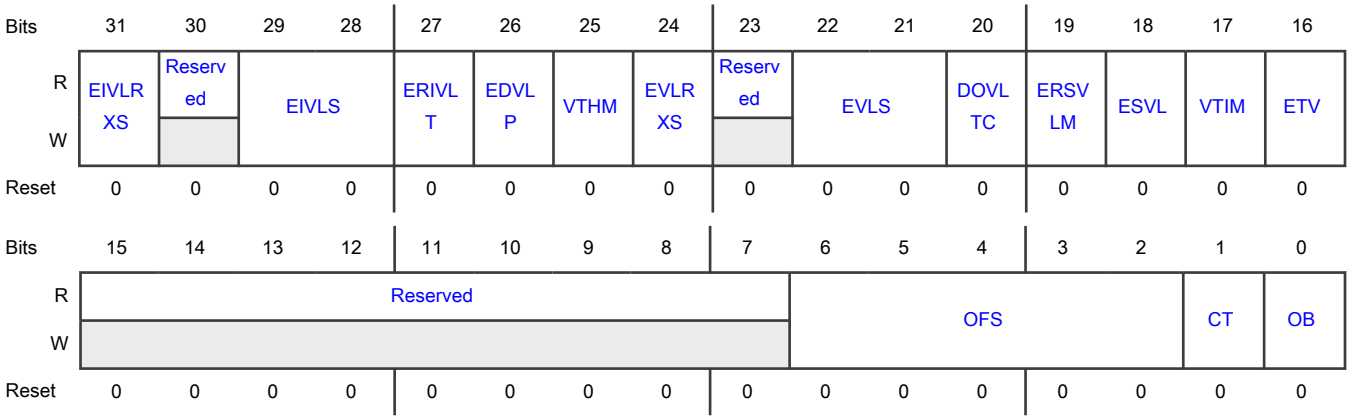
Offset

Register	Offset
MAC_VLAN_Tag_Ctrl	50h

Function

This register is the redefined format of the MAC VLAN Tag Register. It is used for indirect addressing. It contains the address offset, command type and Busy Bit for CSR access of the Per VLAN Tag registers.

Diagram



Fields

Field	Function
31 EIVLRXS	Enable Inner VLAN Tag in Rx Status When this bit is set, the MAC provides the inner VLAN Tag in the Rx status. When this bit is reset, the MAC does not provide the inner VLAN Tag in Rx status. 0b - Inner VLAN Tag in Rx status is disabled 1b - Inner VLAN Tag in Rx status is enabled
30 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
29-28 EIVLS	Enable Inner VLAN Tag Stripping on Receive This field indicates the stripping operation on inner VLAN Tag in received packet. 00b - Do not strip 01b - Strip if VLAN filter passes 10b - Strip if VLAN filter fails 11b - Always strip
27 ERIVLT	Enable Inner VLAN Tag Comparison When this bit, VTHM bit and the EDVLP field are set, the MAC receiver enables VLAN Hash filtering operation on the inner VLAN Tag (if present). When this bit is reset and VTHM bit is set, the MAC receiver enables VLAN Hash filtering operation on the outer VLAN Tag (if present). The ERSVLM bit and DOVLTC bit determines which VLAN type is enabled for filtering. 0b - Inner VLAN tag is disabled 1b - Inner VLAN tag is enabled
26 EDVLP	Enable Double VLAN Processing When this bit is set, the MAC enables processing of up to two VLAN Tags on Tx and Rx (if present). When this bit is reset, the MAC enables processing of up to one VLAN Tag on Tx and Rx (if present). 0b - Double VLAN Processing is disabled 1b - Double VLAN Processing is enabled
25 VTHM	VLAN Tag Hash Table Match Enable When this bit is set, the most significant four bits of CRC of VLAN Tag are used to index the content of the MAC_VLAN_Hash_Table register. A value of 1 in the VLAN Hash Table register, corresponding to the index, indicates that the packet matched the VLAN hash table. When the ETV bit is set, the CRC of the 12-bit VLAN Identifier (VID) is used for comparison. When the ETV bit is reset, the CRC of the 16-bit VLAN tag is used for comparison. When this bit is reset, the VLAN Hash Match operation is not performed. 0b - VLAN Tag Hash Table Match is disabled 1b - VLAN Tag Hash Table Match is enabled
24 EVLRXS	Enable VLAN Tag in Rx status When this bit is set, MAC provides the outer VLAN Tag in the Rx status. When this bit is reset, the MAC does not provide the outer VLAN Tag in Rx status. 0b - VLAN Tag in Rx status is disabled 1b - VLAN Tag in Rx status is enabled
23 —	Reserved
22-21	Enable VLAN Tag Stripping on Receive

Table continues on the next page...

Table continued from the previous page...

Field	Function
EVLS	This field indicates the stripping operation on the outer VLAN Tag in received packet. 00b - Do not strip 01b - Strip if VLAN filter passes 10b - Strip if VLAN filter fails 11b - Always strip
20 DOVLTC	Disable VLAN Type Check for VLAN Hash Filtering When this bit is set, the MAC VLAN Hash Filter does not check whether the VLAN Tag specified by the ERIVLT bit is of type S-VLAN or C-VLAN. When this bit is reset, the MAC VLAN Hash Filter filters or matches the VLAN Tag specified by the ERIVLT bit only when VLAN Tag type is similar to the one specified by the ERSVLM bit. 0b - VLAN Type Check is enabled 1b - VLAN Type Check is disabled
19 ERSVLM	Enable Receive S-VLAN Match for VLAN Hash Filtering When this bit is set, the MAC receiver enables VLAN Hash filtering or matching for S-VLAN (Type = 0x88A8) packets. When this bit is reset, the MAC receiver enables VLAN Hash filtering or matching for C-VLAN (Type = 0x8100) packets. The ERIVLT bit determines the VLAN tag position considered for VLAN Hash filtering or matching. 0b - Receive S-VLAN Match is disabled 1b - Receive S-VLAN Match is enabled
18 ESVL	Enable S-VLAN When this bit is set, the MAC transmitter and receiver consider the S-VLAN packets (Type = 0x88A8) as valid VLAN tagged packets. 0b - S-VLAN is disabled 1b - S-VLAN is enabled
17 VTIM	VLAN Tag Inverse Match Enable When this bit is set, this bit enables the VLAN Tag inverse matching. The packets without matching VLAN Tag are marked as matched. When reset, this bit enables the VLAN Tag perfect matching. The packets with matched VLAN Tag are marked as matched. 0b - VLAN Tag Inverse Match is disabled 1b - VLAN Tag Inverse Match is enabled
16 ETV	Enable 12-Bit VLAN Tag Comparison for VLAN Hash Filtering When this bit is set, a 12-bit VLAN identifier is used for VLAN Hash filtering instead of the complete 16-bit VLAN tag. Bits[11:0] of VLAN tag in the received VLAN-tagged packet are used for hash-based VLAN filtering. When this bit is reset, all 16 bits of the 15th and 16th bytes of the received VLAN packet are used for VLAN hash filtering.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - 12-Bit VLAN Tag Comparison is disabled 1b - 12-Bit VLAN Tag Comparison is enabled
15-7 —	Reserved
6-2 OFS	Offset This field holds the address offset of the MAC VLAN Tag Filter Register which the application is trying to access. The width of the field depends on the number of MAC VLAN Tag Registers enabled.
1 CT	Command Type This bit indicates if the current register access is a read or a write. When set, it indicate a read operation. When reset, it indicates a write operation. 0b - Write operation 1b - Read operation
0 OB	Operation Busy This bit is set along with a read or write command for initiating the indirect access to per VLAN Tag Filter register. This bit is reset when the read or write command to per VLAN Tag Filter indirect access register is complete. The next indirect register access can be initiated only after this bit is reset. During a write operation, the bit is reset only after the data has been written into the Per VLAN Tag register. During a read operation, the data should be read from the MAC_VLAN_Tag_Data register only after this bit is reset. **NOTE** This field is not testable by an automated register test. 0b - Operation Busy is disabled 1b - Operation Busy is enabled

76.17.15 MAC VLAN Tag Data (MAC_VLAN_Tag_Data)

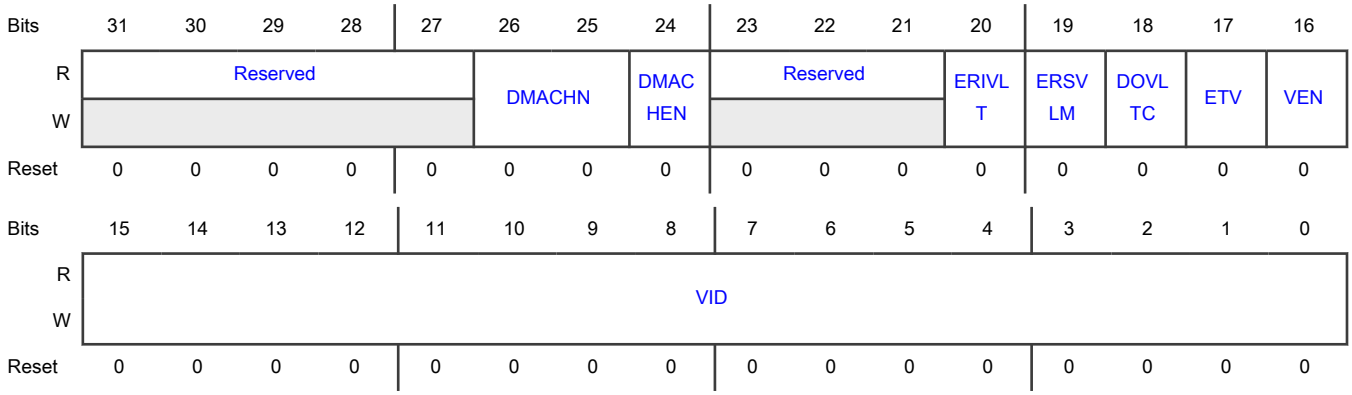
Offset

Register	Offset
MAC_VLAN_Tag_Data	54h

Function

This register holds the read/write data for Indirect Access of the Per VLAN Tag registers. During the read access, this field contains valid read data only after the OB bit is reset. During the write access, this field should be valid prior to setting the OB bit in the MAC_VLAN_Tag_Ctrl Register.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-25 DMACHN	DMA Channel Number The DMA Channel number to which the VLAN Tagged Frame is to be routed if it passes this VLAN Tag Filter is programmed in this field. If the Routing based on VLAN Tag Filter is not necessary, this field need not be programmed.
24 DMACHEN	DMA Channel Number Enable This bit is the Enable for the DMA Channel Number value programmed in the field DMACH. When this bit is reset, the Routing does not occur based on VLAN Filter result. The frame is routed based on DA Based DMA Channel Routing. 0b - DMA Channel Number is disabled 1b - DMA Channel Number is enabled
23-21 —	Reserved
20 ERIVLT	Enable Inner VLAN Tag Comparison This bit is valid only when Double VLAN Tag Enable of the Filter is set. When this bit and the EDVLP field are set, the MAC receiver enables operation on the inner VLAN Tag (if present). When this bit is reset, the MAC receiver enables operation on the outer VLAN Tag (if present). 0b - Inner VLAN tag comparison is disabled 1b - Inner VLAN tag comparison is enabled
19 ERSVLM	Enable S-VLAN Match for received Frames This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC receiver enables filtering or matching for S-VLAN (Type = 0x88A8) packets. When this bit is reset, the MAC receiver enables filtering or matching for C-VLAN (Type = 0x8100) packets.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Receive S-VLAN Match is disabled 1b - Receive S-VLAN Match is enabled
18 DOVLTC	Disable VLAN Type Comparison This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC does not check whether the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit is of type S-VLAN or C-VLAN. When this bit is reset, the MAC filters or matches the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit only when VLAN Tag type is similar to the one specified by the Enable S-VLAN Match for received Frames bit. 0b - VLAN type comparison is enabled 1b - VLAN type comparison is disabled
17 ETV	12bits or 16bits VLAN comparison This bit is valid only when VEN of the Filter is set. When this bit is set, a 12-bit VLAN identifier is used for comparing and filtering instead of the complete 16-bit VLAN tag. Bits [11:0] of VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. 0b - 16 bit VLAN comparison 1b - 12 bit VLAN comparison
16 VEN	VLAN Tag Enable This bit is used to enable or disable the VLAN Tag. When this bit is set, the MAC compares the VLAN Tag of received packet with the VLAN Tag ID. When this bit is reset, no comparison is performed irrespective of the programming of the other fields. 0b - VLAN Tag is disabled 1b - VLAN Tag is enabled
15-0 VID	VLAN Tag ID This field holds the VLAN Tag value which is used by the MAC for perfect comparison. It is valid when VLAN Tag Enable is set.

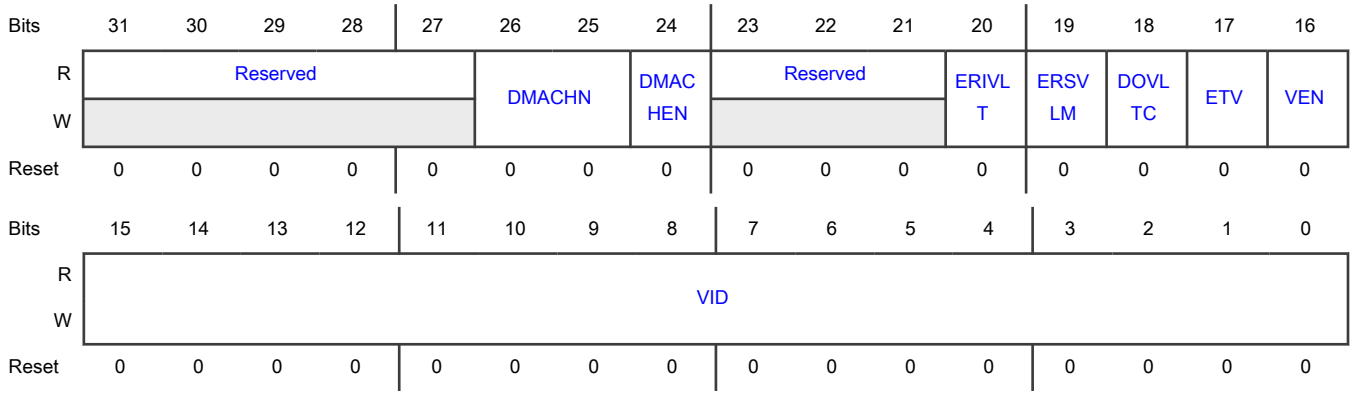
76.17.16 MAC VLAN Tag Filter 0 (MAC_VLAN_Tag_Filter0)

Offset

Register	Offset
MAC_VLAN_Tag_Filter0	54h

Function
This register contains VLAN Tag filter 0 control information.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-25 DMACHN	DMA Channel Number The DMA Channel number to which the VLAN Tagged Frame is to be routed if it passes this VLAN Tag Filter is programmed in this field. If the Routing based on VLAN Tag Filter is not necessary, this field need not be programmed.
24 DMACHEN	DMA Channel Number Enable This bit is the Enable for the DMA Channel Number value programmed in the field DMACH. When this bit is reset, the Routing does not occur based on VLAN Filter result. The frame is routed based on DA Based DMA Channel Routing. 0b - DMA Channel Number is disabled 1b - DMA Channel Number is enabled
23-21 —	Reserved
20 ERIVLT	Enable Inner VLAN Tag Comparison This bit is valid only when Double VLAN Tag Enable of the Filter is set. When this bit and the EDVLP field are set, the MAC receiver enables operation on the inner VLAN Tag (if present). When this bit is reset, the MAC receiver enables operation on the outer VLAN Tag (if present). 0b - Inner VLAN tag comparison is disabled 1b - Inner VLAN tag comparison is enabled
19 ERSVLM	Enable S-VLAN Match for received Frames This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC receiver enables filtering or matching for S-VLAN (Type = 0x88A8) packets. When this bit is reset, the MAC receiver enables filtering or matching for C-VLAN (Type = 0x8100) packets.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Receive S-VLAN Match is disabled 1b - Receive S-VLAN Match is enabled
18 DOVLTC	Disable VLAN Type Comparison This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC does not check whether the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit is of type S-VLAN or C-VLAN. When this bit is reset, the MAC filters or matches the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit only when VLAN Tag type is similar to the one specified by the Enable S-VLAN Match for received Frames bit. 0b - VLAN type comparison is enabled 1b - VLAN type comparison is disabled
17 ETV	12bits or 16bits VLAN comparison This bit is valid only when VEN of the Filter is set. When this bit is set, a 12-bit VLAN identifier is used for comparing and filtering instead of the complete 16-bit VLAN tag. Bits [11:0] of VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. 0b - 16 bit VLAN comparison 1b - 12 bit VLAN comparison
16 VEN	VLAN Tag Enable This bit is used to enable or disable the VLAN Tag. When this bit is set, the MAC compares the VLAN Tag of received packet with the VLAN Tag ID. When this bit is reset, no comparison is performed irrespective of the programming of the other fields. 0b - VLAN Tag is disabled 1b - VLAN Tag is enabled
15-0 VID	VLAN Tag ID This field holds the VLAN Tag value which is used by the MAC for perfect comparison. It is valid when VLAN Tag Enable is set.

76.17.17 MAC VLAN Tag Filter 1 (MAC_VLAN_Tag_Filter1)

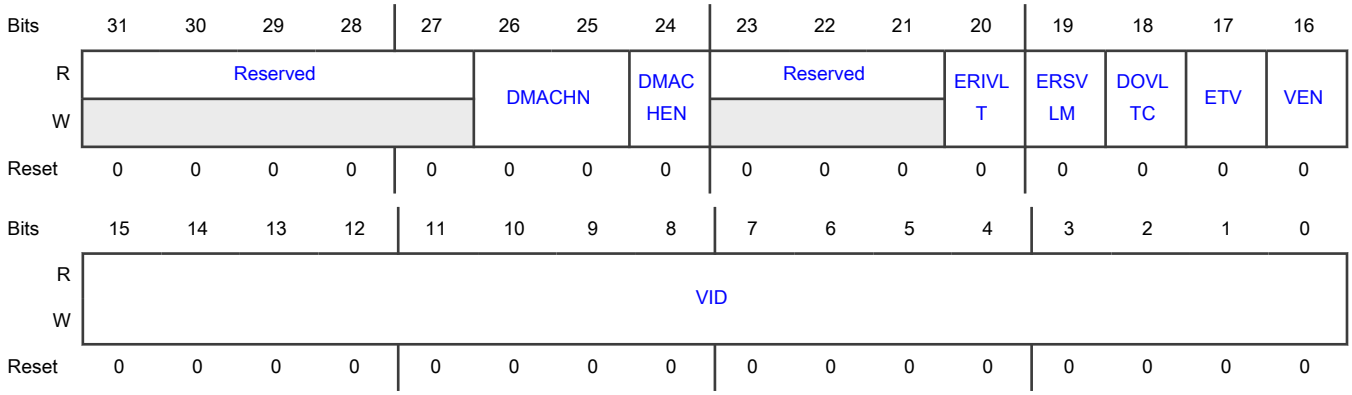
Offset

Register	Offset
MAC_VLAN_Tag_Filter1	54h

Function

This register contains VLAN Tag filter 1 control information.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-25 DMACHN	DMA Channel Number The DMA Channel number to which the VLAN Tagged Frame is to be routed if it passes this VLAN Tag Filter is programmed in this field. If the Routing based on VLAN Tag Filter is not necessary, this field need not be programmed.
24 DMACHEN	DMA Channel Number Enable This bit is the Enable for the DMA Channel Number value programmed in the field DMACH. When this bit is reset, the Routing does not occur based on VLAN Filter result. The frame is routed based on DA Based DMA Channel Routing. 0b - DMA Channel Number is disabled 1b - DMA Channel Number is enabled
23-21 —	Reserved
20 ERIVLT	Enable Inner VLAN Tag Comparison This bit is valid only when Double VLAN Tag Enable of the Filter is set. When this bit and the EDVLP field are set, the MAC receiver enables operation on the inner VLAN Tag (if present). When this bit is reset, the MAC receiver enables operation on the outer VLAN Tag (if present). 0b - Inner VLAN tag comparison is disabled 1b - Inner VLAN tag comparison is enabled
19 ERSVLM	Enable S-VLAN Match for received Frames This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC receiver enables filtering or matching for S-VLAN (Type = 0x88A8) packets. When this bit is reset, the MAC receiver enables filtering or matching for C-VLAN (Type = 0x8100) packets.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Receive S-VLAN Match is disabled 1b - Receive S-VLAN Match is enabled
18 DOVLTC	Disable VLAN Type Comparison This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC does not check whether the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit is of type S-VLAN or C-VLAN. When this bit is reset, the MAC filters or matches the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit only when VLAN Tag type is similar to the one specified by the Enable S-VLAN Match for received Frames bit. 0b - VLAN type comparison is enabled 1b - VLAN type comparison is disabled
17 ETV	12bits or 16bits VLAN comparison This bit is valid only when VEN of the Filter is set. When this bit is set, a 12-bit VLAN identifier is used for comparing and filtering instead of the complete 16-bit VLAN tag. Bits [11:0] of VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. 0b - 16 bit VLAN comparison 1b - 12 bit VLAN comparison
16 VEN	VLAN Tag Enable This bit is used to enable or disable the VLAN Tag. When this bit is set, the MAC compares the VLAN Tag of received packet with the VLAN Tag ID. When this bit is reset, no comparison is performed irrespective of the programming of the other fields. 0b - VLAN Tag is disabled 1b - VLAN Tag is enabled
15-0 VID	VLAN Tag ID This field holds the VLAN Tag value which is used by the MAC for perfect comparison. It is valid when VLAN Tag Enable is set.

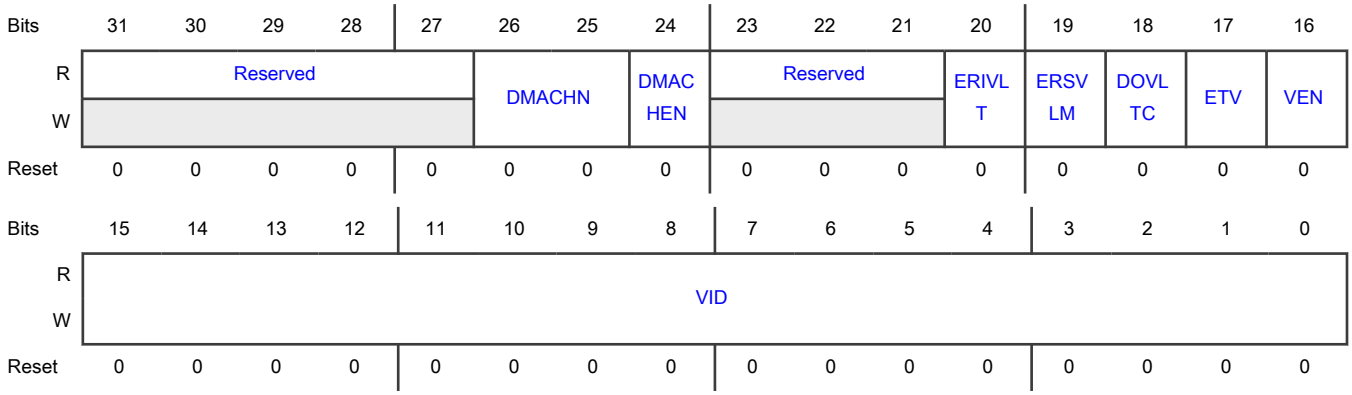
76.17.18 MAC VLAN Tag Filter 10 (MAC_VLAN_Tag_Filter10)

Offset

Register	Offset
MAC_VLAN_Tag_Filter10	54h

Function
This register contains VLAN Tag filter 10 control information.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-25 DMACHN	DMA Channel Number The DMA Channel number to which the VLAN Tagged Frame is to be routed if it passes this VLAN Tag Filter is programmed in this field. If the Routing based on VLAN Tag Filter is not necessary, this field need not be programmed.
24 DMACHEN	DMA Channel Number Enable This bit is the Enable for the DMA Channel Number value programmed in the field DMACH. When this bit is reset, the Routing does not occur based on VLAN Filter result. The frame is routed based on DA Based DMA Channel Routing. 0b - DMA Channel Number is disabled 1b - DMA Channel Number is enabled
23-21 —	Reserved
20 ERIVLT	Enable Inner VLAN Tag Comparison This bit is valid only when Double VLAN Tag Enable of the Filter is set. When this bit and the EDVLP field are set, the MAC receiver enables operation on the inner VLAN Tag (if present). When this bit is reset, the MAC receiver enables operation on the outer VLAN Tag (if present). 0b - Inner VLAN tag comparison is disabled 1b - Inner VLAN tag comparison is enabled
19 ERSVLM	Enable S-VLAN Match for received Frames This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC receiver enables filtering or matching for S-VLAN (Type = 0x88A8) packets. When this bit is reset, the MAC receiver enables filtering or matching for C-VLAN (Type = 0x8100) packets.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Receive S-VLAN Match is disabled 1b - Receive S-VLAN Match is enabled
18 DOVLTC	Disable VLAN Type Comparison This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC does not check whether the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit is of type S-VLAN or C-VLAN. When this bit is reset, the MAC filters or matches the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit only when VLAN Tag type is similar to the one specified by the Enable S-VLAN Match for received Frames bit. 0b - VLAN type comparison is enabled 1b - VLAN type comparison is disabled
17 ETV	12bits or 16bits VLAN comparison This bit is valid only when VEN of the Filter is set. When this bit is set, a 12-bit VLAN identifier is used for comparing and filtering instead of the complete 16-bit VLAN tag. Bits [11:0] of VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. 0b - 16 bit VLAN comparison 1b - 12 bit VLAN comparison
16 VEN	VLAN Tag Enable This bit is used to enable or disable the VLAN Tag. When this bit is set, the MAC compares the VLAN Tag of received packet with the VLAN Tag ID. When this bit is reset, no comparison is performed irrespective of the programming of the other fields. 0b - VLAN Tag is disabled 1b - VLAN Tag is enabled
15-0 VID	VLAN Tag ID This field holds the VLAN Tag value which is used by the MAC for perfect comparison. It is valid when VLAN Tag Enable is set.

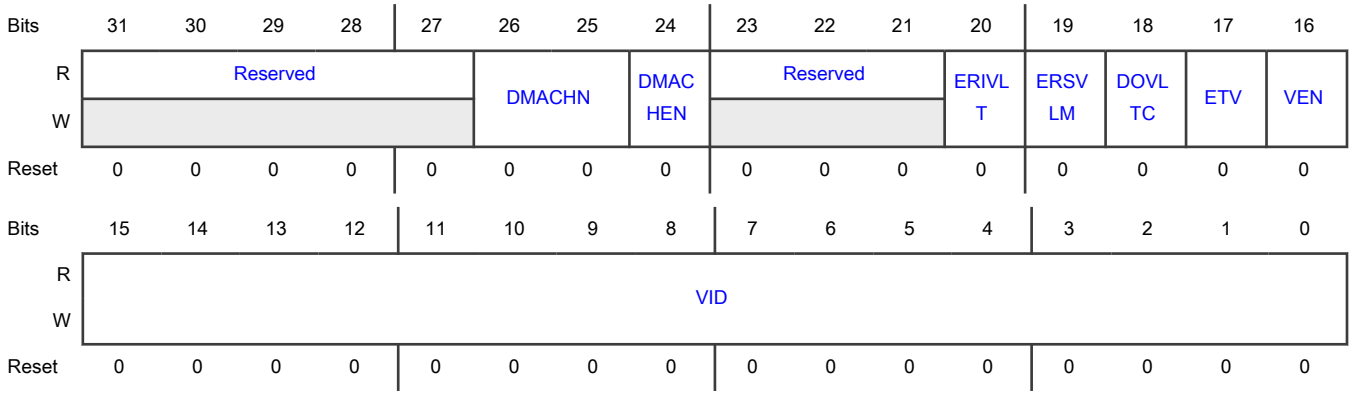
76.17.19 MAC VLAN Tag Filter 11 (MAC_VLAN_Tag_Filter11)

Offset

Register	Offset
MAC_VLAN_Tag_Filter11	54h

Function
This register contains VLAN Tag filter 11 control information.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-25 DMACHN	DMA Channel Number The DMA Channel number to which the VLAN Tagged Frame is to be routed if it passes this VLAN Tag Filter is programmed in this field. If the Routing based on VLAN Tag Filter is not necessary, this field need not be programmed.
24 DMACHEN	DMA Channel Number Enable This bit is the Enable for the DMA Channel Number value programmed in the field DMACH. When this bit is reset, the Routing does not occur based on VLAN Filter result. The frame is routed based on DA Based DMA Channel Routing. 0b - DMA Channel Number is disabled 1b - DMA Channel Number is enabled
23-21 —	Reserved
20 ERIVLT	Enable Inner VLAN Tag Comparison This bit is valid only when Double VLAN Tag Enable of the Filter is set. When this bit and the EDVLP field are set, the MAC receiver enables operation on the inner VLAN Tag (if present). When this bit is reset, the MAC receiver enables operation on the outer VLAN Tag (if present). 0b - Inner VLAN tag comparison is disabled 1b - Inner VLAN tag comparison is enabled
19 ERSVLM	Enable S-VLAN Match for received Frames This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC receiver enables filtering or matching for S-VLAN (Type = 0x88A8) packets. When this bit is reset, the MAC receiver enables filtering or matching for C-VLAN (Type = 0x8100) packets.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Receive S-VLAN Match is disabled 1b - Receive S-VLAN Match is enabled
18 DOVLTC	Disable VLAN Type Comparison This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC does not check whether the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit is of type S-VLAN or C-VLAN. When this bit is reset, the MAC filters or matches the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit only when VLAN Tag type is similar to the one specified by the Enable S-VLAN Match for received Frames bit. 0b - VLAN type comparison is enabled 1b - VLAN type comparison is disabled
17 ETV	12bits or 16bits VLAN comparison This bit is valid only when VEN of the Filter is set. When this bit is set, a 12-bit VLAN identifier is used for comparing and filtering instead of the complete 16-bit VLAN tag. Bits [11:0] of VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. 0b - 16 bit VLAN comparison 1b - 12 bit VLAN comparison
16 VEN	VLAN Tag Enable This bit is used to enable or disable the VLAN Tag. When this bit is set, the MAC compares the VLAN Tag of received packet with the VLAN Tag ID. When this bit is reset, no comparison is performed irrespective of the programming of the other fields. 0b - VLAN Tag is disabled 1b - VLAN Tag is enabled
15-0 VID	VLAN Tag ID This field holds the VLAN Tag value which is used by the MAC for perfect comparison. It is valid when VLAN Tag Enable is set.

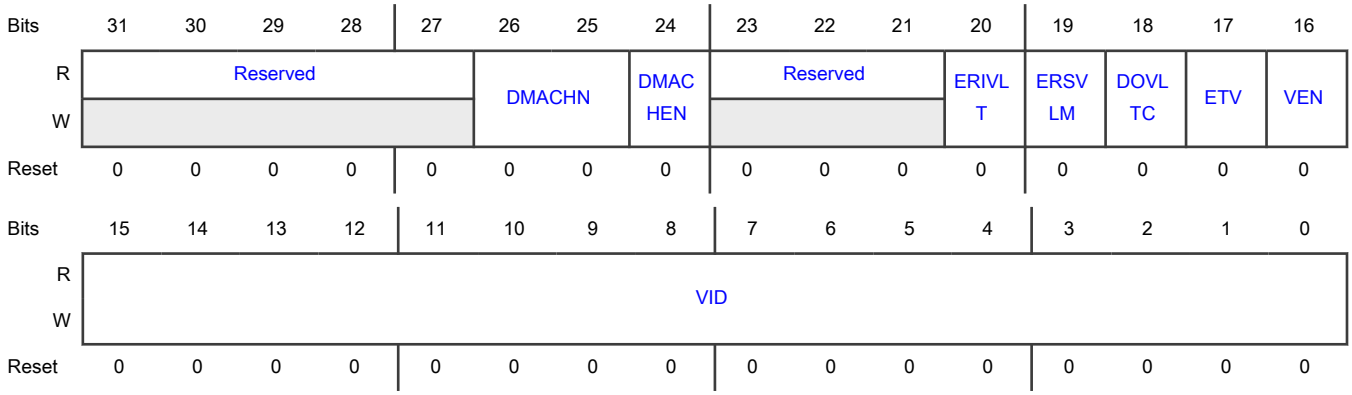
76.17.20 MAC VLAN Tag Filter 12 (MAC_VLAN_Tag_Filter12)

Offset

Register	Offset
MAC_VLAN_Tag_Filter12	54h

Function
This register contains VLAN Tag filter 12 control information.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-25 DMACHN	DMA Channel Number The DMA Channel number to which the VLAN Tagged Frame is to be routed if it passes this VLAN Tag Filter is programmed in this field. If the Routing based on VLAN Tag Filter is not necessary, this field need not be programmed.
24 DMACHEN	DMA Channel Number Enable This bit is the Enable for the DMA Channel Number value programmed in the field DMACH. When this bit is reset, the Routing does not occur based on VLAN Filter result. The frame is routed based on DA Based DMA Channel Routing. 0b - DMA Channel Number is disabled 1b - DMA Channel Number is enabled
23-21 —	Reserved
20 ERIVLT	Enable Inner VLAN Tag Comparison This bit is valid only when Double VLAN Tag Enable of the Filter is set. When this bit and the EDVLP field are set, the MAC receiver enables operation on the inner VLAN Tag (if present). When this bit is reset, the MAC receiver enables operation on the outer VLAN Tag (if present). 0b - Inner VLAN tag comparison is disabled 1b - Inner VLAN tag comparison is enabled
19 ERSVLM	Enable S-VLAN Match for received Frames This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC receiver enables filtering or matching for S-VLAN (Type = 0x88A8) packets. When this bit is reset, the MAC receiver enables filtering or matching for C-VLAN (Type = 0x8100) packets.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Receive S-VLAN Match is disabled 1b - Receive S-VLAN Match is enabled
18 DOVLTC	Disable VLAN Type Comparison This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC does not check whether the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit is of type S-VLAN or C-VLAN. When this bit is reset, the MAC filters or matches the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit only when VLAN Tag type is similar to the one specified by the Enable S-VLAN Match for received Frames bit. 0b - VLAN type comparison is enabled 1b - VLAN type comparison is disabled
17 ETV	12bits or 16bits VLAN comparison This bit is valid only when VEN of the Filter is set. When this bit is set, a 12-bit VLAN identifier is used for comparing and filtering instead of the complete 16-bit VLAN tag. Bits [11:0] of VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. 0b - 16 bit VLAN comparison 1b - 12 bit VLAN comparison
16 VEN	VLAN Tag Enable This bit is used to enable or disable the VLAN Tag. When this bit is set, the MAC compares the VLAN Tag of received packet with the VLAN Tag ID. When this bit is reset, no comparison is performed irrespective of the programming of the other fields. 0b - VLAN Tag is disabled 1b - VLAN Tag is enabled
15-0 VID	VLAN Tag ID This field holds the VLAN Tag value which is used by the MAC for perfect comparison. It is valid when VLAN Tag Enable is set.

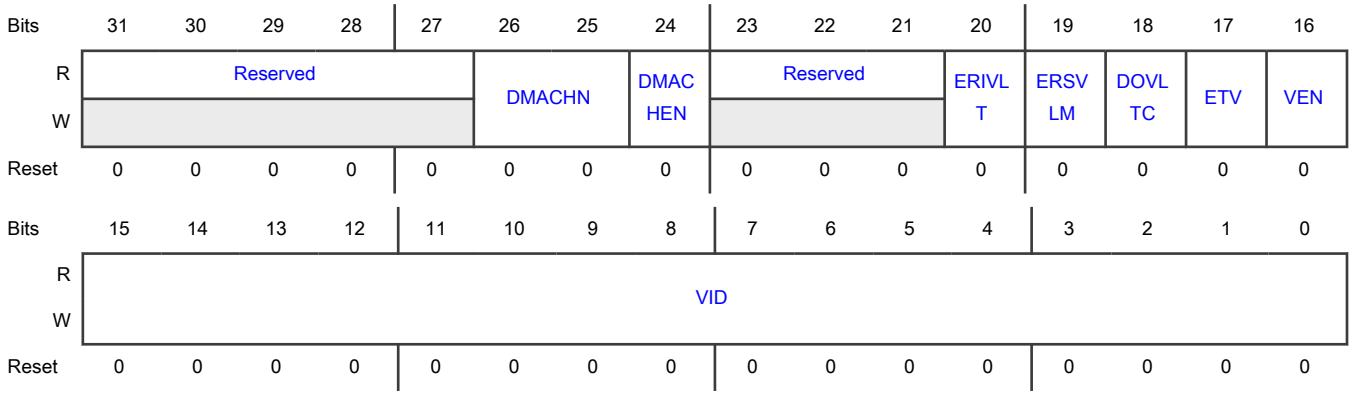
76.17.21 MAC VLAN Tag Filter 13 (MAC_VLAN_Tag_Filter13)

Offset

Register	Offset
MAC_VLAN_Tag_Filter13	54h

Function
This register contains VLAN Tag filter 13 control information.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-25 DMACHN	DMA Channel Number The DMA Channel number to which the VLAN Tagged Frame is to be routed if it passes this VLAN Tag Filter is programmed in this field. If the Routing based on VLAN Tag Filter is not necessary, this field need not be programmed.
24 DMACHEN	DMA Channel Number Enable This bit is the Enable for the DMA Channel Number value programmed in the field DMACH. When this bit is reset, the Routing does not occur based on VLAN Filter result. The frame is routed based on DA Based DMA Channel Routing. 0b - DMA Channel Number is disabled 1b - DMA Channel Number is enabled
23-21 —	Reserved
20 ERIVLT	Enable Inner VLAN Tag Comparison This bit is valid only when Double VLAN Tag Enable of the Filter is set. When this bit and the EDVLP field are set, the MAC receiver enables operation on the inner VLAN Tag (if present). When this bit is reset, the MAC receiver enables operation on the outer VLAN Tag (if present). 0b - Inner VLAN tag comparison is disabled 1b - Inner VLAN tag comparison is enabled
19 ERSVLM	Enable S-VLAN Match for received Frames This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC receiver enables filtering or matching for S-VLAN (Type = 0x88A8) packets. When this bit is reset, the MAC receiver enables filtering or matching for C-VLAN (Type = 0x8100) packets.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Receive S-VLAN Match is disabled 1b - Receive S-VLAN Match is enabled
18 DOVLTC	Disable VLAN Type Comparison This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC does not check whether the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit is of type S-VLAN or C-VLAN. When this bit is reset, the MAC filters or matches the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit only when VLAN Tag type is similar to the one specified by the Enable S-VLAN Match for received Frames bit. 0b - VLAN type comparison is enabled 1b - VLAN type comparison is disabled
17 ETV	12bits or 16bits VLAN comparison This bit is valid only when VEN of the Filter is set. When this bit is set, a 12-bit VLAN identifier is used for comparing and filtering instead of the complete 16-bit VLAN tag. Bits [11:0] of VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. 0b - 16 bit VLAN comparison 1b - 12 bit VLAN comparison
16 VEN	VLAN Tag Enable This bit is used to enable or disable the VLAN Tag. When this bit is set, the MAC compares the VLAN Tag of received packet with the VLAN Tag ID. When this bit is reset, no comparison is performed irrespective of the programming of the other fields. 0b - VLAN Tag is disabled 1b - VLAN Tag is enabled
15-0 VID	VLAN Tag ID This field holds the VLAN Tag value which is used by the MAC for perfect comparison. It is valid when VLAN Tag Enable is set.

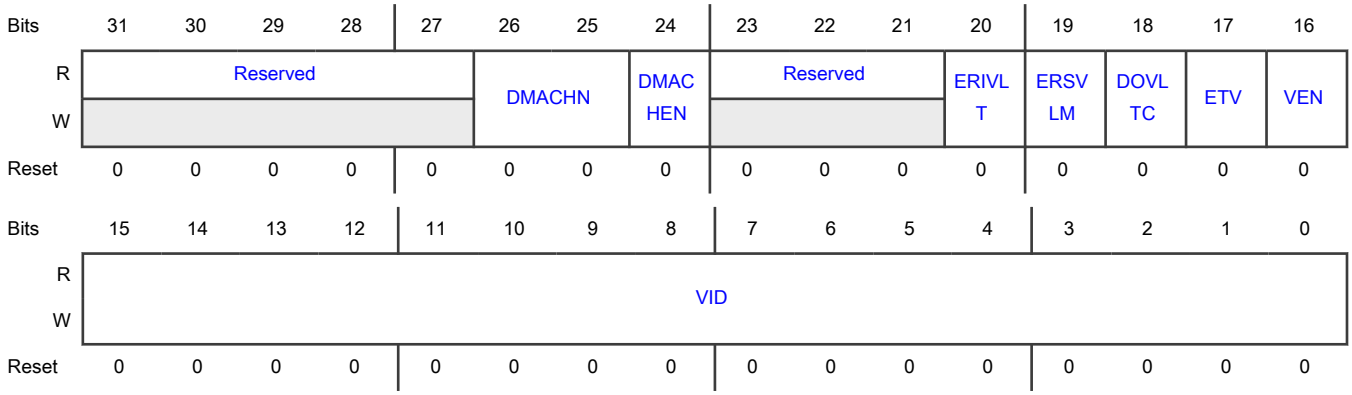
76.17.22 MAC VLAN Tag Filter 14 (MAC_VLAN_Tag_Filter14)

Offset

Register	Offset
MAC_VLAN_Tag_Filter14	54h

Function
This register contains VLAN Tag filter 14 control information.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-25 DMACHN	DMA Channel Number The DMA Channel number to which the VLAN Tagged Frame is to be routed if it passes this VLAN Tag Filter is programmed in this field. If the Routing based on VLAN Tag Filter is not necessary, this field need not be programmed.
24 DMACHEN	DMA Channel Number Enable This bit is the Enable for the DMA Channel Number value programmed in the field DMACH. When this bit is reset, the Routing does not occur based on VLAN Filter result. The frame is routed based on DA Based DMA Channel Routing. 0b - DMA Channel Number is disabled 1b - DMA Channel Number is enabled
23-21 —	Reserved
20 ERIVLT	Enable Inner VLAN Tag Comparison This bit is valid only when Double VLAN Tag Enable of the Filter is set. When this bit and the EDVLP field are set, the MAC receiver enables operation on the inner VLAN Tag (if present). When this bit is reset, the MAC receiver enables operation on the outer VLAN Tag (if present). 0b - Inner VLAN tag comparison is disabled 1b - Inner VLAN tag comparison is enabled
19 ERSVLM	Enable S-VLAN Match for received Frames This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC receiver enables filtering or matching for S-VLAN (Type = 0x88A8) packets. When this bit is reset, the MAC receiver enables filtering or matching for C-VLAN (Type = 0x8100) packets.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Receive S-VLAN Match is disabled 1b - Receive S-VLAN Match is enabled
18 DOVLTC	Disable VLAN Type Comparison This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC does not check whether the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit is of type S-VLAN or C-VLAN. When this bit is reset, the MAC filters or matches the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit only when VLAN Tag type is similar to the one specified by the Enable S-VLAN Match for received Frames bit. 0b - VLAN type comparison is enabled 1b - VLAN type comparison is disabled
17 ETV	12bits or 16bits VLAN comparison This bit is valid only when VEN of the Filter is set. When this bit is set, a 12-bit VLAN identifier is used for comparing and filtering instead of the complete 16-bit VLAN tag. Bits [11:0] of VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. 0b - 16 bit VLAN comparison 1b - 12 bit VLAN comparison
16 VEN	VLAN Tag Enable This bit is used to enable or disable the VLAN Tag. When this bit is set, the MAC compares the VLAN Tag of received packet with the VLAN Tag ID. When this bit is reset, no comparison is performed irrespective of the programming of the other fields. 0b - VLAN Tag is disabled 1b - VLAN Tag is enabled
15-0 VID	VLAN Tag ID This field holds the VLAN Tag value which is used by the MAC for perfect comparison. It is valid when VLAN Tag Enable is set.

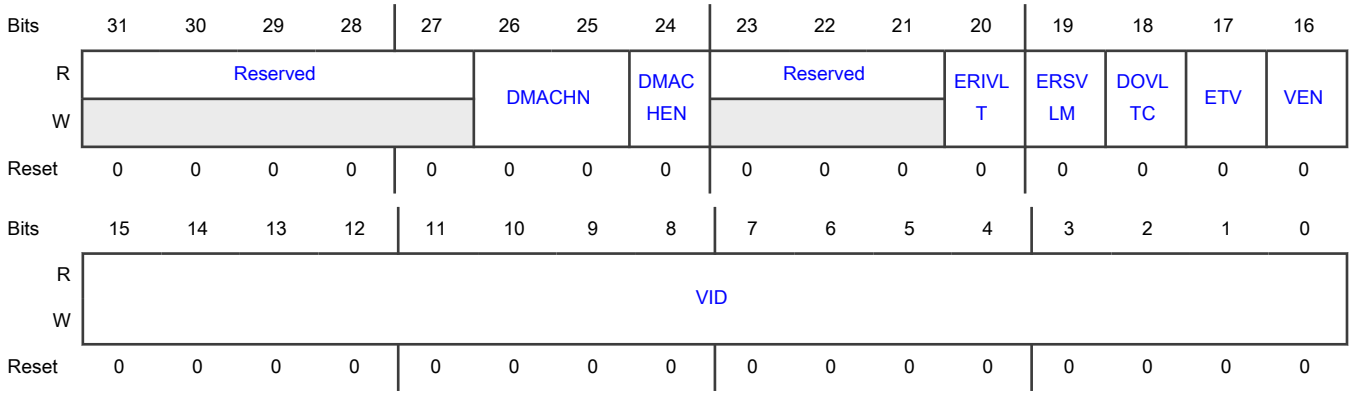
76.17.23 MAC VLAN Tag Filter 15 (MAC_VLAN_Tag_Filter15)

Offset

Register	Offset
MAC_VLAN_Tag_Filter15	54h

Function
This register contains VLAN Tag filter 15 control information.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-25 DMACHN	DMA Channel Number The DMA Channel number to which the VLAN Tagged Frame is to be routed if it passes this VLAN Tag Filter is programmed in this field. If the Routing based on VLAN Tag Filter is not necessary, this field need not be programmed.
24 DMACHEN	DMA Channel Number Enable This bit is the Enable for the DMA Channel Number value programmed in the field DMACH. When this bit is reset, the Routing does not occur based on VLAN Filter result. The frame is routed based on DA Based DMA Channel Routing. 0b - DMA Channel Number is disabled 1b - DMA Channel Number is enabled
23-21 —	Reserved
20 ERIVLT	Enable Inner VLAN Tag Comparison This bit is valid only when Double VLAN Tag Enable of the Filter is set. When this bit and the EDVLP field are set, the MAC receiver enables operation on the inner VLAN Tag (if present). When this bit is reset, the MAC receiver enables operation on the outer VLAN Tag (if present). 0b - Inner VLAN tag comparison is disabled 1b - Inner VLAN tag comparison is enabled
19 ERSVLM	Enable S-VLAN Match for received Frames This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC receiver enables filtering or matching for S-VLAN (Type = 0x88A8) packets. When this bit is reset, the MAC receiver enables filtering or matching for C-VLAN (Type = 0x8100) packets.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Receive S-VLAN Match is disabled 1b - Receive S-VLAN Match is enabled
18 DOVLTC	Disable VLAN Type Comparison This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC does not check whether the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit is of type S-VLAN or C-VLAN. When this bit is reset, the MAC filters or matches the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit only when VLAN Tag type is similar to the one specified by the Enable S-VLAN Match for received Frames bit. 0b - VLAN type comparison is enabled 1b - VLAN type comparison is disabled
17 ETV	12bits or 16bits VLAN comparison This bit is valid only when VEN of the Filter is set. When this bit is set, a 12-bit VLAN identifier is used for comparing and filtering instead of the complete 16-bit VLAN tag. Bits [11:0] of VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. 0b - 16 bit VLAN comparison 1b - 12 bit VLAN comparison
16 VEN	VLAN Tag Enable This bit is used to enable or disable the VLAN Tag. When this bit is set, the MAC compares the VLAN Tag of received packet with the VLAN Tag ID. When this bit is reset, no comparison is performed irrespective of the programming of the other fields. 0b - VLAN Tag is disabled 1b - VLAN Tag is enabled
15-0 VID	VLAN Tag ID This field holds the VLAN Tag value which is used by the MAC for perfect comparison. It is valid when VLAN Tag Enable is set.

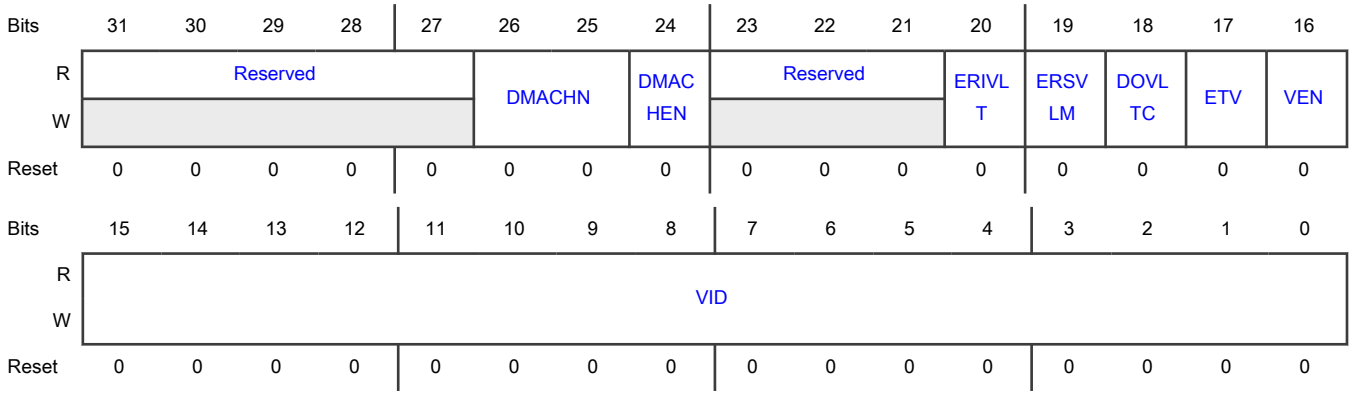
76.17.24 MAC VLAN Tag Filter 16 (MAC_VLAN_Tag_Filter16)

Offset

Register	Offset
MAC_VLAN_Tag_Filter16	54h

Function
This register contains VLAN Tag filter 16 control information.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-25 DMACHN	DMA Channel Number The DMA Channel number to which the VLAN Tagged Frame is to be routed if it passes this VLAN Tag Filter is programmed in this field. If the Routing based on VLAN Tag Filter is not necessary, this field need not be programmed.
24 DMACHEN	DMA Channel Number Enable This bit is the Enable for the DMA Channel Number value programmed in the field DMACH. When this bit is reset, the Routing does not occur based on VLAN Filter result. The frame is routed based on DA Based DMA Channel Routing. 0b - DMA Channel Number is disabled 1b - DMA Channel Number is enabled
23-21 —	Reserved
20 ERIVLT	Enable Inner VLAN Tag Comparison This bit is valid only when Double VLAN Tag Enable of the Filter is set. When this bit and the EDVLP field are set, the MAC receiver enables operation on the inner VLAN Tag (if present). When this bit is reset, the MAC receiver enables operation on the outer VLAN Tag (if present). 0b - Inner VLAN tag comparison is disabled 1b - Inner VLAN tag comparison is enabled
19 ERSVLM	Enable S-VLAN Match for received Frames This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC receiver enables filtering or matching for S-VLAN (Type = 0x88A8) packets. When this bit is reset, the MAC receiver enables filtering or matching for C-VLAN (Type = 0x8100) packets.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Receive S-VLAN Match is disabled 1b - Receive S-VLAN Match is enabled
18 DOVLTC	Disable VLAN Type Comparison This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC does not check whether the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit is of type S-VLAN or C-VLAN. When this bit is reset, the MAC filters or matches the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit only when VLAN Tag type is similar to the one specified by the Enable S-VLAN Match for received Frames bit. 0b - VLAN type comparison is enabled 1b - VLAN type comparison is disabled
17 ETV	12bits or 16bits VLAN comparison This bit is valid only when VEN of the Filter is set. When this bit is set, a 12-bit VLAN identifier is used for comparing and filtering instead of the complete 16-bit VLAN tag. Bits [11:0] of VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. 0b - 16 bit VLAN comparison 1b - 12 bit VLAN comparison
16 VEN	VLAN Tag Enable This bit is used to enable or disable the VLAN Tag. When this bit is set, the MAC compares the VLAN Tag of received packet with the VLAN Tag ID. When this bit is reset, no comparison is performed irrespective of the programming of the other fields. 0b - VLAN Tag is disabled 1b - VLAN Tag is enabled
15-0 VID	VLAN Tag ID This field holds the VLAN Tag value which is used by the MAC for perfect comparison. It is valid when VLAN Tag Enable is set.

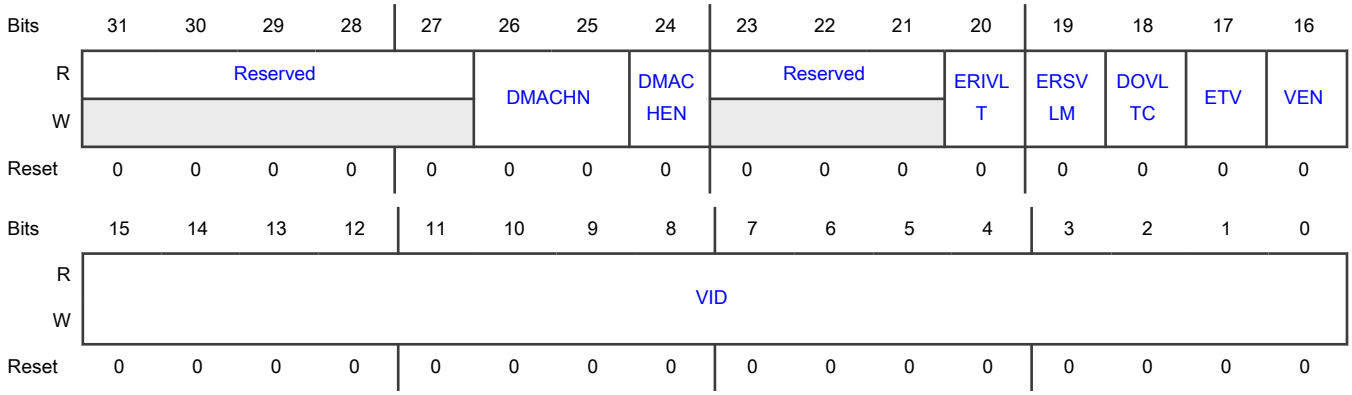
76.17.25 MAC VLAN Tag Filter 17 (MAC_VLAN_Tag_Filter17)

Offset

Register	Offset
MAC_VLAN_Tag_Filter17	54h

Function
This register contains VLAN Tag filter 17 control information.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-25 DMACHN	DMA Channel Number The DMA Channel number to which the VLAN Tagged Frame is to be routed if it passes this VLAN Tag Filter is programmed in this field. If the Routing based on VLAN Tag Filter is not necessary, this field need not be programmed.
24 DMACHEN	DMA Channel Number Enable This bit is the Enable for the DMA Channel Number value programmed in the field DMACH. When this bit is reset, the Routing does not occur based on VLAN Filter result. The frame is routed based on DA Based DMA Channel Routing. 0b - DMA Channel Number is disabled 1b - DMA Channel Number is enabled
23-21 —	Reserved
20 ERIVLT	Enable Inner VLAN Tag Comparison This bit is valid only when Double VLAN Tag Enable of the Filter is set. When this bit and the EDVLP field are set, the MAC receiver enables operation on the inner VLAN Tag (if present). When this bit is reset, the MAC receiver enables operation on the outer VLAN Tag (if present). 0b - Inner VLAN tag comparison is disabled 1b - Inner VLAN tag comparison is enabled
19 ERSVLM	Enable S-VLAN Match for received Frames This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC receiver enables filtering or matching for S-VLAN (Type = 0x88A8) packets. When this bit is reset, the MAC receiver enables filtering or matching for C-VLAN (Type = 0x8100) packets.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Receive S-VLAN Match is disabled 1b - Receive S-VLAN Match is enabled
18 DOVLTC	Disable VLAN Type Comparison This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC does not check whether the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit is of type S-VLAN or C-VLAN. When this bit is reset, the MAC filters or matches the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit only when VLAN Tag type is similar to the one specified by the Enable S-VLAN Match for received Frames bit. 0b - VLAN type comparison is enabled 1b - VLAN type comparison is disabled
17 ETV	12bits or 16bits VLAN comparison This bit is valid only when VEN of the Filter is set. When this bit is set, a 12-bit VLAN identifier is used for comparing and filtering instead of the complete 16-bit VLAN tag. Bits [11:0] of VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. 0b - 16 bit VLAN comparison 1b - 12 bit VLAN comparison
16 VEN	VLAN Tag Enable This bit is used to enable or disable the VLAN Tag. When this bit is set, the MAC compares the VLAN Tag of received packet with the VLAN Tag ID. When this bit is reset, no comparison is performed irrespective of the programming of the other fields. 0b - VLAN Tag is disabled 1b - VLAN Tag is enabled
15-0 VID	VLAN Tag ID This field holds the VLAN Tag value which is used by the MAC for perfect comparison. It is valid when VLAN Tag Enable is set.

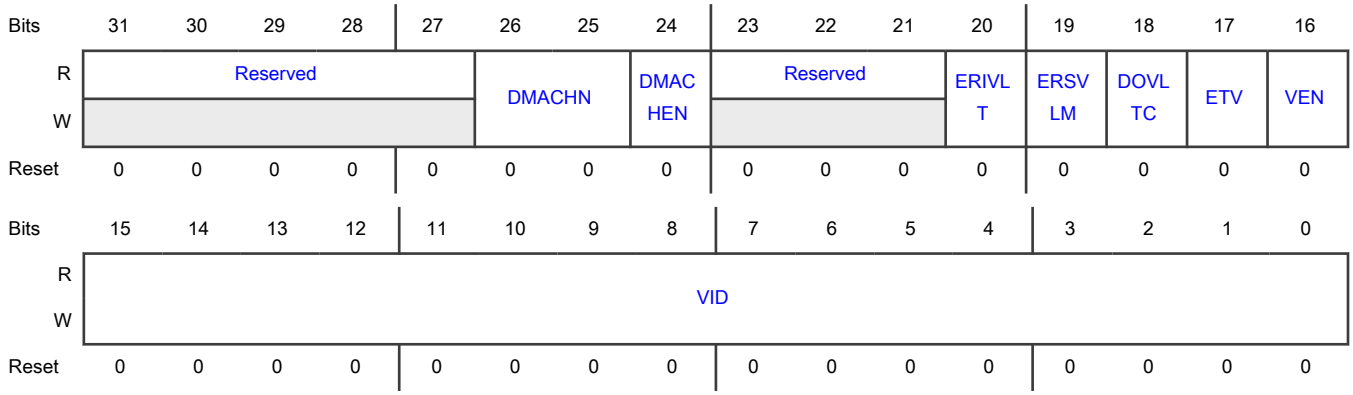
76.17.26 MAC VLAN Tag Filter 18 (MAC_VLAN_Tag_Filter18)

Offset

Register	Offset
MAC_VLAN_Tag_Filter18	54h

Function
This register contains VLAN Tag filter 18 control information.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-25 DMACHN	DMA Channel Number The DMA Channel number to which the VLAN Tagged Frame is to be routed if it passes this VLAN Tag Filter is programmed in this field. If the Routing based on VLAN Tag Filter is not necessary, this field need not be programmed.
24 DMACHEN	DMA Channel Number Enable This bit is the Enable for the DMA Channel Number value programmed in the field DMACH. When this bit is reset, the Routing does not occur based on VLAN Filter result. The frame is routed based on DA Based DMA Channel Routing. 0b - DMA Channel Number is disabled 1b - DMA Channel Number is enabled
23-21 —	Reserved
20 ERIVLT	Enable Inner VLAN Tag Comparison This bit is valid only when Double VLAN Tag Enable of the Filter is set. When this bit and the EDVLP field are set, the MAC receiver enables operation on the inner VLAN Tag (if present). When this bit is reset, the MAC receiver enables operation on the outer VLAN Tag (if present). 0b - Inner VLAN tag comparison is disabled 1b - Inner VLAN tag comparison is enabled
19 ERSVLM	Enable S-VLAN Match for received Frames This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC receiver enables filtering or matching for S-VLAN (Type = 0x88A8) packets. When this bit is reset, the MAC receiver enables filtering or matching for C-VLAN (Type = 0x8100) packets.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Receive S-VLAN Match is disabled 1b - Receive S-VLAN Match is enabled
18 DOVLTC	Disable VLAN Type Comparison This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC does not check whether the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit is of type S-VLAN or C-VLAN. When this bit is reset, the MAC filters or matches the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit only when VLAN Tag type is similar to the one specified by the Enable S-VLAN Match for received Frames bit. 0b - VLAN type comparison is enabled 1b - VLAN type comparison is disabled
17 ETV	12bits or 16bits VLAN comparison This bit is valid only when VEN of the Filter is set. When this bit is set, a 12-bit VLAN identifier is used for comparing and filtering instead of the complete 16-bit VLAN tag. Bits [11:0] of VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. 0b - 16 bit VLAN comparison 1b - 12 bit VLAN comparison
16 VEN	VLAN Tag Enable This bit is used to enable or disable the VLAN Tag. When this bit is set, the MAC compares the VLAN Tag of received packet with the VLAN Tag ID. When this bit is reset, no comparison is performed irrespective of the programming of the other fields. 0b - VLAN Tag is disabled 1b - VLAN Tag is enabled
15-0 VID	VLAN Tag ID This field holds the VLAN Tag value which is used by the MAC for perfect comparison. It is valid when VLAN Tag Enable is set.

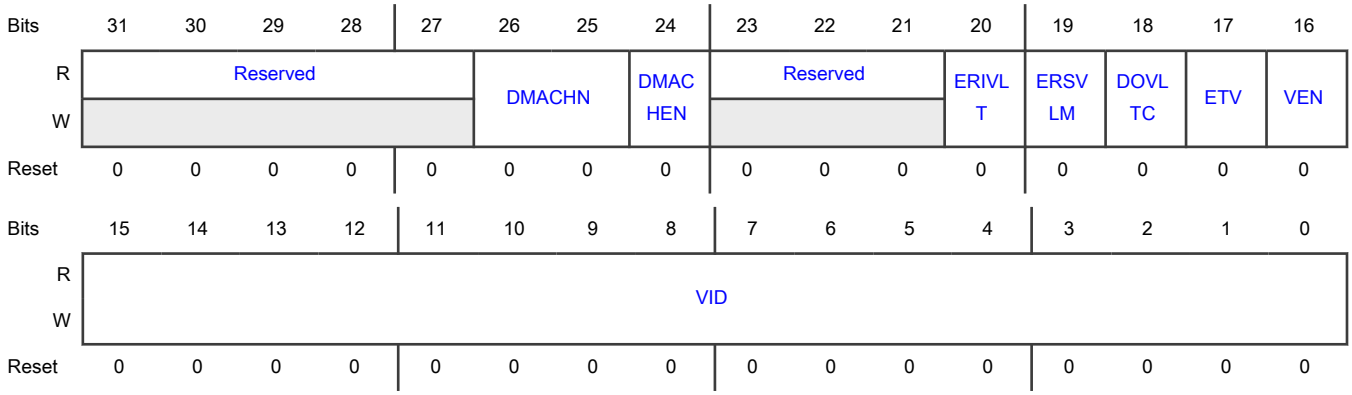
76.17.27 MAC VLAN Tag Filter 19 (MAC_VLAN_Tag_Filter19)

Offset

Register	Offset
MAC_VLAN_Tag_Filter19	54h

Function
This register contains VLAN Tag filter 19 control information.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-25 DMACHN	DMA Channel Number The DMA Channel number to which the VLAN Tagged Frame is to be routed if it passes this VLAN Tag Filter is programmed in this field. If the Routing based on VLAN Tag Filter is not necessary, this field need not be programmed.
24 DMACHEN	DMA Channel Number Enable This bit is the Enable for the DMA Channel Number value programmed in the field DMACH. When this bit is reset, the Routing does not occur based on VLAN Filter result. The frame is routed based on DA Based DMA Channel Routing. 0b - DMA Channel Number is disabled 1b - DMA Channel Number is enabled
23-21 —	Reserved
20 ERIVLT	Enable Inner VLAN Tag Comparison This bit is valid only when Double VLAN Tag Enable of the Filter is set. When this bit and the EDVLP field are set, the MAC receiver enables operation on the inner VLAN Tag (if present). When this bit is reset, the MAC receiver enables operation on the outer VLAN Tag (if present). 0b - Inner VLAN tag comparison is disabled 1b - Inner VLAN tag comparison is enabled
19 ERSVLM	Enable S-VLAN Match for received Frames This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC receiver enables filtering or matching for S-VLAN (Type = 0x88A8) packets. When this bit is reset, the MAC receiver enables filtering or matching for C-VLAN (Type = 0x8100) packets.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Receive S-VLAN Match is disabled 1b - Receive S-VLAN Match is enabled
18 DOVLTC	Disable VLAN Type Comparison This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC does not check whether the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit is of type S-VLAN or C-VLAN. When this bit is reset, the MAC filters or matches the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit only when VLAN Tag type is similar to the one specified by the Enable S-VLAN Match for received Frames bit. 0b - VLAN type comparison is enabled 1b - VLAN type comparison is disabled
17 ETV	12bits or 16bits VLAN comparison This bit is valid only when VEN of the Filter is set. When this bit is set, a 12-bit VLAN identifier is used for comparing and filtering instead of the complete 16-bit VLAN tag. Bits [11:0] of VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. 0b - 16 bit VLAN comparison 1b - 12 bit VLAN comparison
16 VEN	VLAN Tag Enable This bit is used to enable or disable the VLAN Tag. When this bit is set, the MAC compares the VLAN Tag of received packet with the VLAN Tag ID. When this bit is reset, no comparison is performed irrespective of the programming of the other fields. 0b - VLAN Tag is disabled 1b - VLAN Tag is enabled
15-0 VID	VLAN Tag ID This field holds the VLAN Tag value which is used by the MAC for perfect comparison. It is valid when VLAN Tag Enable is set.

76.17.28 MAC VLAN Tag Filter 2 (MAC_VLAN_Tag_Filter2)

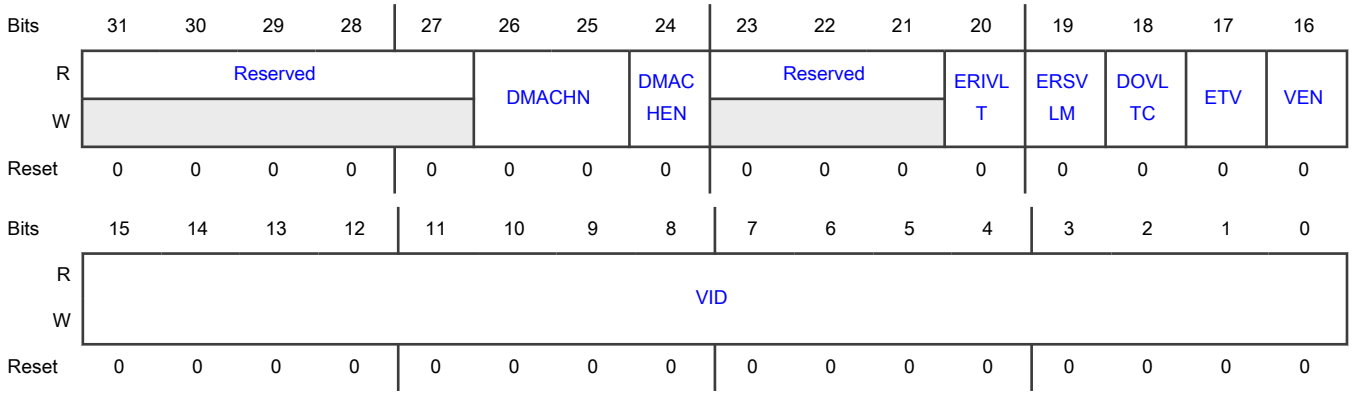
Offset

Register	Offset
MAC_VLAN_Tag_Filter2	54h

Function

This register contains VLAN Tag filter 2 control information.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-25 DMACHN	DMA Channel Number The DMA Channel number to which the VLAN Tagged Frame is to be routed if it passes this VLAN Tag Filter is programmed in this field. If the Routing based on VLAN Tag Filter is not necessary, this field need not be programmed.
24 DMACHEN	DMA Channel Number Enable This bit is the Enable for the DMA Channel Number value programmed in the field DMACH. When this bit is reset, the Routing does not occur based on VLAN Filter result. The frame is routed based on DA Based DMA Channel Routing. 0b - DMA Channel Number is disabled 1b - DMA Channel Number is enabled
23-21 —	Reserved
20 ERIVLT	Enable Inner VLAN Tag Comparison This bit is valid only when Double VLAN Tag Enable of the Filter is set. When this bit and the EDVLP field are set, the MAC receiver enables operation on the inner VLAN Tag (if present). When this bit is reset, the MAC receiver enables operation on the outer VLAN Tag (if present). 0b - Inner VLAN tag comparison is disabled 1b - Inner VLAN tag comparison is enabled
19 ERSVLM	Enable S-VLAN Match for received Frames This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC receiver enables filtering or matching for S-VLAN (Type = 0x88A8) packets. When this bit is reset, the MAC receiver enables filtering or matching for C-VLAN (Type = 0x8100) packets.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Receive S-VLAN Match is disabled 1b - Receive S-VLAN Match is enabled
18 DOVLTC	Disable VLAN Type Comparison This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC does not check whether the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit is of type S-VLAN or C-VLAN. When this bit is reset, the MAC filters or matches the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit only when VLAN Tag type is similar to the one specified by the Enable S-VLAN Match for received Frames bit. 0b - VLAN type comparison is enabled 1b - VLAN type comparison is disabled
17 ETV	12bits or 16bits VLAN comparison This bit is valid only when VEN of the Filter is set. When this bit is set, a 12-bit VLAN identifier is used for comparing and filtering instead of the complete 16-bit VLAN tag. Bits [11:0] of VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. 0b - 16 bit VLAN comparison 1b - 12 bit VLAN comparison
16 VEN	VLAN Tag Enable This bit is used to enable or disable the VLAN Tag. When this bit is set, the MAC compares the VLAN Tag of received packet with the VLAN Tag ID. When this bit is reset, no comparison is performed irrespective of the programming of the other fields. 0b - VLAN Tag is disabled 1b - VLAN Tag is enabled
15-0 VID	VLAN Tag ID This field holds the VLAN Tag value which is used by the MAC for perfect comparison. It is valid when VLAN Tag Enable is set.

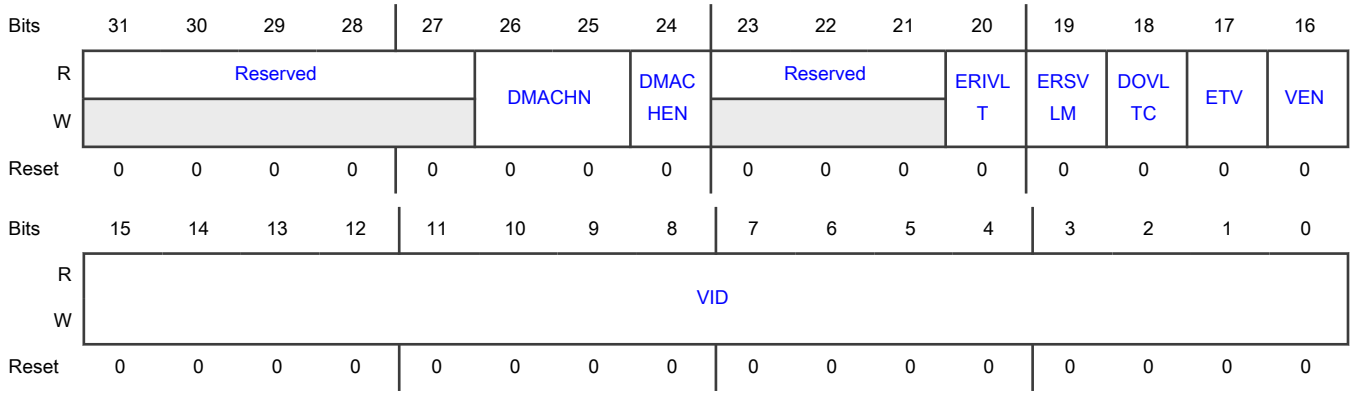
76.17.29 MAC VLAN Tag Filter 20 (MAC_VLAN_Tag_Filter20)

Offset

Register	Offset
MAC_VLAN_Tag_Filter20	54h

Function
This register contains VLAN Tag filter 20 control information.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-25 DMACHN	DMA Channel Number The DMA Channel number to which the VLAN Tagged Frame is to be routed if it passes this VLAN Tag Filter is programmed in this field. If the Routing based on VLAN Tag Filter is not necessary, this field need not be programmed.
24 DMACHEN	DMA Channel Number Enable This bit is the Enable for the DMA Channel Number value programmed in the field DMACH. When this bit is reset, the Routing does not occur based on VLAN Filter result. The frame is routed based on DA Based DMA Channel Routing. 0b - DMA Channel Number is disabled 1b - DMA Channel Number is enabled
23-21 —	Reserved
20 ERIVLT	Enable Inner VLAN Tag Comparison This bit is valid only when Double VLAN Tag Enable of the Filter is set. When this bit and the EDVLP field are set, the MAC receiver enables operation on the inner VLAN Tag (if present). When this bit is reset, the MAC receiver enables operation on the outer VLAN Tag (if present). 0b - Inner VLAN tag comparison is disabled 1b - Inner VLAN tag comparison is enabled
19 ERSVLM	Enable S-VLAN Match for received Frames This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC receiver enables filtering or matching for S-VLAN (Type = 0x88A8) packets. When this bit is reset, the MAC receiver enables filtering or matching for C-VLAN (Type = 0x8100) packets.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Receive S-VLAN Match is disabled 1b - Receive S-VLAN Match is enabled
18 DOVLTC	Disable VLAN Type Comparison This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC does not check whether the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit is of type S-VLAN or C-VLAN. When this bit is reset, the MAC filters or matches the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit only when VLAN Tag type is similar to the one specified by the Enable S-VLAN Match for received Frames bit. 0b - VLAN type comparison is enabled 1b - VLAN type comparison is disabled
17 ETV	12bits or 16bits VLAN comparison This bit is valid only when VEN of the Filter is set. When this bit is set, a 12-bit VLAN identifier is used for comparing and filtering instead of the complete 16-bit VLAN tag. Bits [11:0] of VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. 0b - 16 bit VLAN comparison 1b - 12 bit VLAN comparison
16 VEN	VLAN Tag Enable This bit is used to enable or disable the VLAN Tag. When this bit is set, the MAC compares the VLAN Tag of received packet with the VLAN Tag ID. When this bit is reset, no comparison is performed irrespective of the programming of the other fields. 0b - VLAN Tag is disabled 1b - VLAN Tag is enabled
15-0 VID	VLAN Tag ID This field holds the VLAN Tag value which is used by the MAC for perfect comparison. It is valid when VLAN Tag Enable is set.

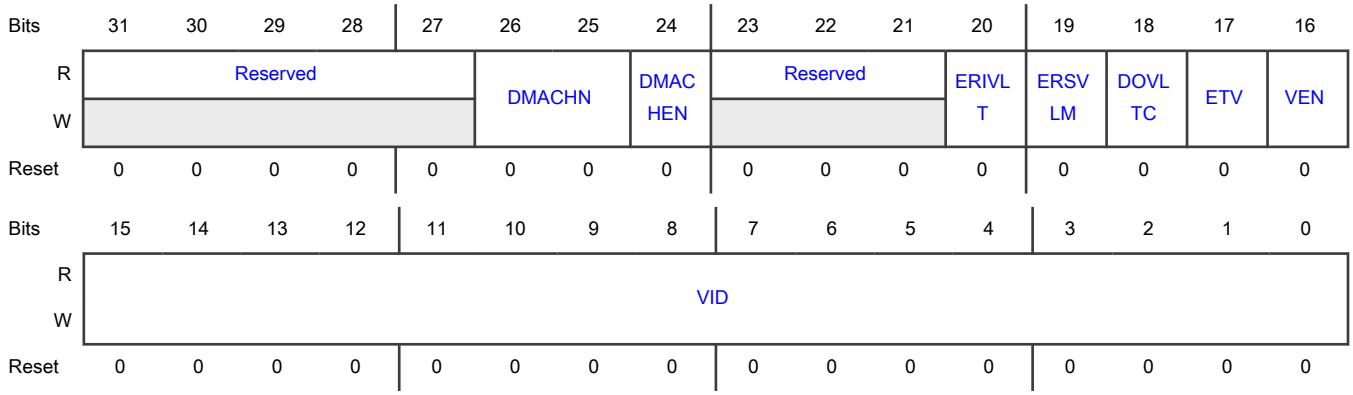
76.17.30 MAC VLAN Tag Filter 21 (MAC_VLAN_Tag_Filter21)

Offset

Register	Offset
MAC_VLAN_Tag_Filter21	54h

Function
This register contains VLAN Tag filter 21 control information.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-25 DMACHN	DMA Channel Number The DMA Channel number to which the VLAN Tagged Frame is to be routed if it passes this VLAN Tag Filter is programmed in this field. If the Routing based on VLAN Tag Filter is not necessary, this field need not be programmed.
24 DMACHEN	DMA Channel Number Enable This bit is the Enable for the DMA Channel Number value programmed in the field DMACH. When this bit is reset, the Routing does not occur based on VLAN Filter result. The frame is routed based on DA Based DMA Channel Routing. 0b - DMA Channel Number is disabled 1b - DMA Channel Number is enabled
23-21 —	Reserved
20 ERIVLT	Enable Inner VLAN Tag Comparison This bit is valid only when Double VLAN Tag Enable of the Filter is set. When this bit and the EDVLP field are set, the MAC receiver enables operation on the inner VLAN Tag (if present). When this bit is reset, the MAC receiver enables operation on the outer VLAN Tag (if present). 0b - Inner VLAN tag comparison is disabled 1b - Inner VLAN tag comparison is enabled
19 ERSVLM	Enable S-VLAN Match for received Frames This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC receiver enables filtering or matching for S-VLAN (Type = 0x88A8) packets. When this bit is reset, the MAC receiver enables filtering or matching for C-VLAN (Type = 0x8100) packets.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Receive S-VLAN Match is disabled 1b - Receive S-VLAN Match is enabled
18 DOVLTC	Disable VLAN Type Comparison This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC does not check whether the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit is of type S-VLAN or C-VLAN. When this bit is reset, the MAC filters or matches the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit only when VLAN Tag type is similar to the one specified by the Enable S-VLAN Match for received Frames bit. 0b - VLAN type comparison is enabled 1b - VLAN type comparison is disabled
17 ETV	12bits or 16bits VLAN comparison This bit is valid only when VEN of the Filter is set. When this bit is set, a 12-bit VLAN identifier is used for comparing and filtering instead of the complete 16-bit VLAN tag. Bits [11:0] of VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. 0b - 16 bit VLAN comparison 1b - 12 bit VLAN comparison
16 VEN	VLAN Tag Enable This bit is used to enable or disable the VLAN Tag. When this bit is set, the MAC compares the VLAN Tag of received packet with the VLAN Tag ID. When this bit is reset, no comparison is performed irrespective of the programming of the other fields. 0b - VLAN Tag is disabled 1b - VLAN Tag is enabled
15-0 VID	VLAN Tag ID This field holds the VLAN Tag value which is used by the MAC for perfect comparison. It is valid when VLAN Tag Enable is set.

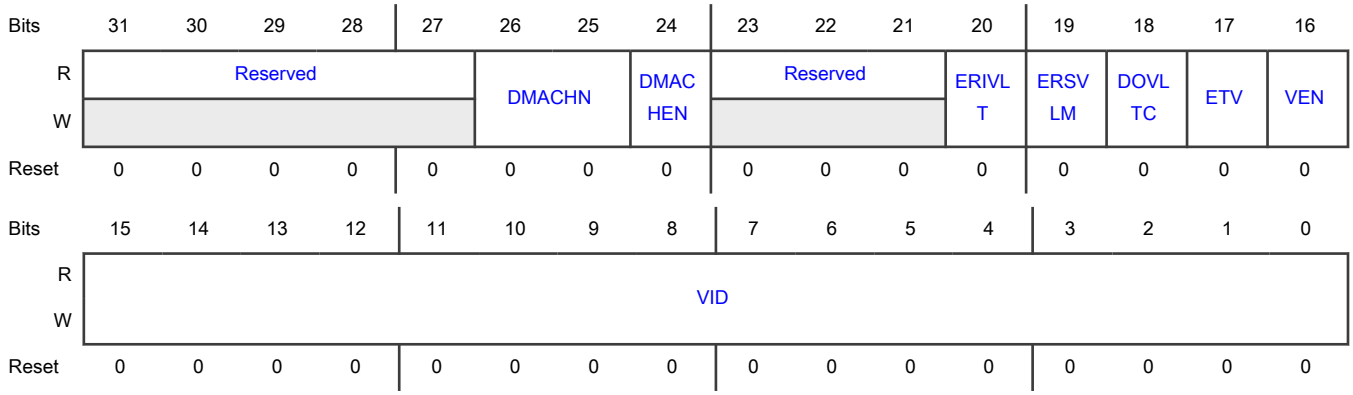
76.17.31 MAC VLAN Tag Filter 22 (MAC_VLAN_Tag_Filter22)

Offset

Register	Offset
MAC_VLAN_Tag_Filter22	54h

Function
This register contains VLAN Tag filter 22 control information.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-25 DMACHN	DMA Channel Number The DMA Channel number to which the VLAN Tagged Frame is to be routed if it passes this VLAN Tag Filter is programmed in this field. If the Routing based on VLAN Tag Filter is not necessary, this field need not be programmed.
24 DMACHEN	DMA Channel Number Enable This bit is the Enable for the DMA Channel Number value programmed in the field DMACH. When this bit is reset, the Routing does not occur based on VLAN Filter result. The frame is routed based on DA Based DMA Channel Routing. 0b - DMA Channel Number is disabled 1b - DMA Channel Number is enabled
23-21 —	Reserved
20 ERIVLT	Enable Inner VLAN Tag Comparison This bit is valid only when Double VLAN Tag Enable of the Filter is set. When this bit and the EDVLP field are set, the MAC receiver enables operation on the inner VLAN Tag (if present). When this bit is reset, the MAC receiver enables operation on the outer VLAN Tag (if present). 0b - Inner VLAN tag comparison is disabled 1b - Inner VLAN tag comparison is enabled
19 ERSVLM	Enable S-VLAN Match for received Frames This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC receiver enables filtering or matching for S-VLAN (Type = 0x88A8) packets. When this bit is reset, the MAC receiver enables filtering or matching for C-VLAN (Type = 0x8100) packets.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Receive S-VLAN Match is disabled 1b - Receive S-VLAN Match is enabled
18 DOVLTC	Disable VLAN Type Comparison This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC does not check whether the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit is of type S-VLAN or C-VLAN. When this bit is reset, the MAC filters or matches the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit only when VLAN Tag type is similar to the one specified by the Enable S-VLAN Match for received Frames bit. 0b - VLAN type comparison is enabled 1b - VLAN type comparison is disabled
17 ETV	12bits or 16bits VLAN comparison This bit is valid only when VEN of the Filter is set. When this bit is set, a 12-bit VLAN identifier is used for comparing and filtering instead of the complete 16-bit VLAN tag. Bits [11:0] of VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. 0b - 16 bit VLAN comparison 1b - 12 bit VLAN comparison
16 VEN	VLAN Tag Enable This bit is used to enable or disable the VLAN Tag. When this bit is set, the MAC compares the VLAN Tag of received packet with the VLAN Tag ID. When this bit is reset, no comparison is performed irrespective of the programming of the other fields. 0b - VLAN Tag is disabled 1b - VLAN Tag is enabled
15-0 VID	VLAN Tag ID This field holds the VLAN Tag value which is used by the MAC for perfect comparison. It is valid when VLAN Tag Enable is set.

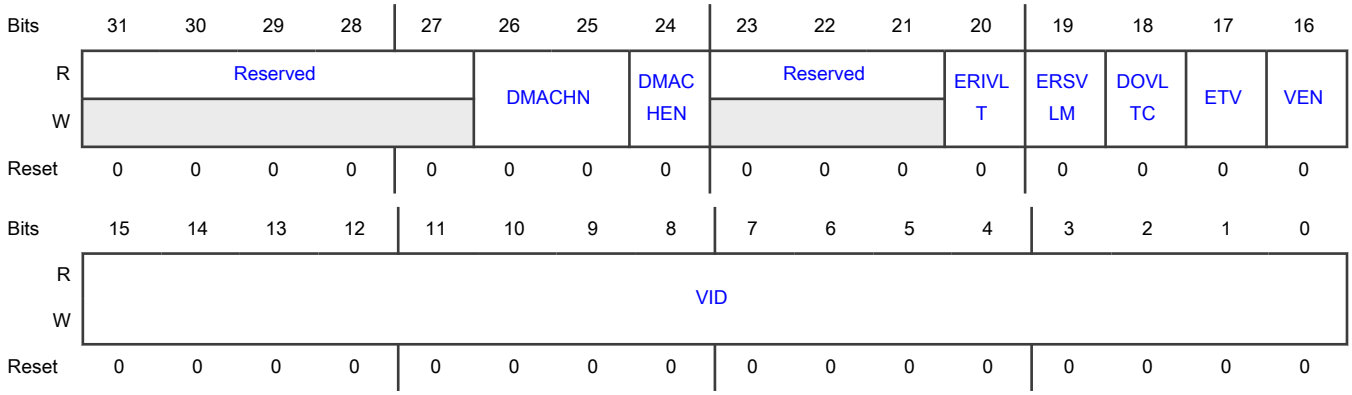
76.17.32 MAC VLAN Tag Filter 23 (MAC_VLAN_Tag_Filter23)

Offset

Register	Offset
MAC_VLAN_Tag_Filter23	54h

Function
This register contains VLAN Tag filter 23 control information.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-25 DMACHN	DMA Channel Number The DMA Channel number to which the VLAN Tagged Frame is to be routed if it passes this VLAN Tag Filter is programmed in this field. If the Routing based on VLAN Tag Filter is not necessary, this field need not be programmed.
24 DMACHEN	DMA Channel Number Enable This bit is the Enable for the DMA Channel Number value programmed in the field DMACH. When this bit is reset, the Routing does not occur based on VLAN Filter result. The frame is routed based on DA Based DMA Channel Routing. 0b - DMA Channel Number is disabled 1b - DMA Channel Number is enabled
23-21 —	Reserved
20 ERIVLT	Enable Inner VLAN Tag Comparison This bit is valid only when Double VLAN Tag Enable of the Filter is set. When this bit and the EDVLP field are set, the MAC receiver enables operation on the inner VLAN Tag (if present). When this bit is reset, the MAC receiver enables operation on the outer VLAN Tag (if present). 0b - Inner VLAN tag comparison is disabled 1b - Inner VLAN tag comparison is enabled
19 ERSVLM	Enable S-VLAN Match for received Frames This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC receiver enables filtering or matching for S-VLAN (Type = 0x88A8) packets. When this bit is reset, the MAC receiver enables filtering or matching for C-VLAN (Type = 0x8100) packets.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Receive S-VLAN Match is disabled 1b - Receive S-VLAN Match is enabled
18 DOVLTC	Disable VLAN Type Comparison This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC does not check whether the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit is of type S-VLAN or C-VLAN. When this bit is reset, the MAC filters or matches the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit only when VLAN Tag type is similar to the one specified by the Enable S-VLAN Match for received Frames bit. 0b - VLAN type comparison is enabled 1b - VLAN type comparison is disabled
17 ETV	12bits or 16bits VLAN comparison This bit is valid only when VEN of the Filter is set. When this bit is set, a 12-bit VLAN identifier is used for comparing and filtering instead of the complete 16-bit VLAN tag. Bits [11:0] of VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. 0b - 16 bit VLAN comparison 1b - 12 bit VLAN comparison
16 VEN	VLAN Tag Enable This bit is used to enable or disable the VLAN Tag. When this bit is set, the MAC compares the VLAN Tag of received packet with the VLAN Tag ID. When this bit is reset, no comparison is performed irrespective of the programming of the other fields. 0b - VLAN Tag is disabled 1b - VLAN Tag is enabled
15-0 VID	VLAN Tag ID This field holds the VLAN Tag value which is used by the MAC for perfect comparison. It is valid when VLAN Tag Enable is set.

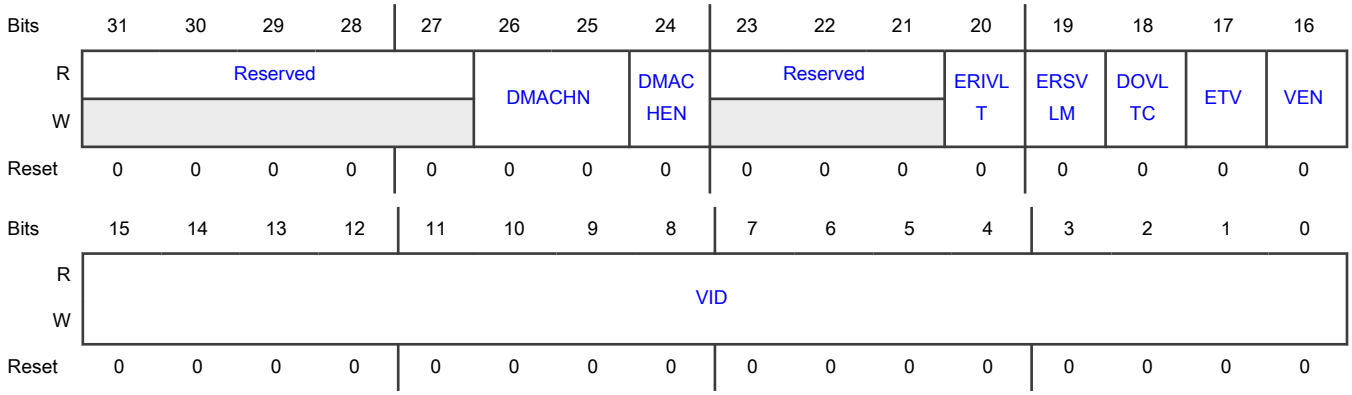
76.17.33 MAC VLAN Tag Filter 24 (MAC_VLAN_Tag_Filter24)

Offset

Register	Offset
MAC_VLAN_Tag_Filter24	54h

Function
This register contains VLAN Tag filter 24 control information.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-25 DMACHN	DMA Channel Number The DMA Channel number to which the VLAN Tagged Frame is to be routed if it passes this VLAN Tag Filter is programmed in this field. If the Routing based on VLAN Tag Filter is not necessary, this field need not be programmed.
24 DMACHEN	DMA Channel Number Enable This bit is the Enable for the DMA Channel Number value programmed in the field DMACH. When this bit is reset, the Routing does not occur based on VLAN Filter result. The frame is routed based on DA Based DMA Channel Routing. 0b - DMA Channel Number is disabled 1b - DMA Channel Number is enabled
23-21 —	Reserved
20 ERIVLT	Enable Inner VLAN Tag Comparison This bit is valid only when Double VLAN Tag Enable of the Filter is set. When this bit and the EDVLP field are set, the MAC receiver enables operation on the inner VLAN Tag (if present). When this bit is reset, the MAC receiver enables operation on the outer VLAN Tag (if present). 0b - Inner VLAN tag comparison is disabled 1b - Inner VLAN tag comparison is enabled
19 ERSVLM	Enable S-VLAN Match for received Frames This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC receiver enables filtering or matching for S-VLAN (Type = 0x88A8) packets. When this bit is reset, the MAC receiver enables filtering or matching for C-VLAN (Type = 0x8100) packets.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Receive S-VLAN Match is disabled 1b - Receive S-VLAN Match is enabled
18 DOVLTC	Disable VLAN Type Comparison This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC does not check whether the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit is of type S-VLAN or C-VLAN. When this bit is reset, the MAC filters or matches the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit only when VLAN Tag type is similar to the one specified by the Enable S-VLAN Match for received Frames bit. 0b - VLAN type comparison is enabled 1b - VLAN type comparison is disabled
17 ETV	12bits or 16bits VLAN comparison This bit is valid only when VEN of the Filter is set. When this bit is set, a 12-bit VLAN identifier is used for comparing and filtering instead of the complete 16-bit VLAN tag. Bits [11:0] of VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. 0b - 16 bit VLAN comparison 1b - 12 bit VLAN comparison
16 VEN	VLAN Tag Enable This bit is used to enable or disable the VLAN Tag. When this bit is set, the MAC compares the VLAN Tag of received packet with the VLAN Tag ID. When this bit is reset, no comparison is performed irrespective of the programming of the other fields. 0b - VLAN Tag is disabled 1b - VLAN Tag is enabled
15-0 VID	VLAN Tag ID This field holds the VLAN Tag value which is used by the MAC for perfect comparison. It is valid when VLAN Tag Enable is set.

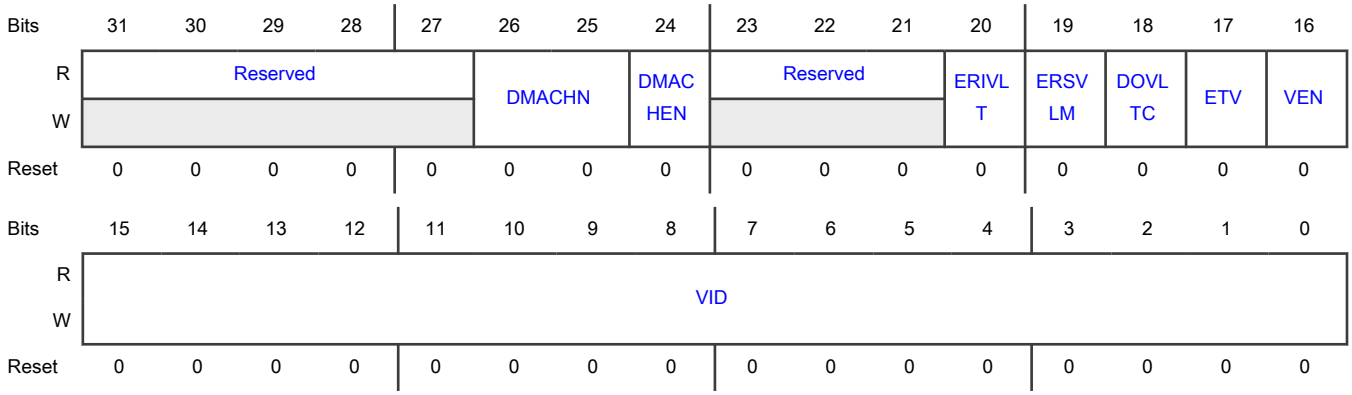
76.17.34 MAC VLAN Tag Filter 25 (MAC_VLAN_Tag_Filter25)

Offset

Register	Offset
MAC_VLAN_Tag_Filter25	54h

Function
This register contains VLAN Tag filter 25 control information.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-25 DMACHN	DMA Channel Number The DMA Channel number to which the VLAN Tagged Frame is to be routed if it passes this VLAN Tag Filter is programmed in this field. If the Routing based on VLAN Tag Filter is not necessary, this field need not be programmed.
24 DMACHEN	DMA Channel Number Enable This bit is the Enable for the DMA Channel Number value programmed in the field DMACH. When this bit is reset, the Routing does not occur based on VLAN Filter result. The frame is routed based on DA Based DMA Channel Routing. 0b - DMA Channel Number is disabled 1b - DMA Channel Number is enabled
23-21 —	Reserved
20 ERIVLT	Enable Inner VLAN Tag Comparison This bit is valid only when Double VLAN Tag Enable of the Filter is set. When this bit and the EDVLP field are set, the MAC receiver enables operation on the inner VLAN Tag (if present). When this bit is reset, the MAC receiver enables operation on the outer VLAN Tag (if present). 0b - Inner VLAN tag comparison is disabled 1b - Inner VLAN tag comparison is enabled
19 ERSVLM	Enable S-VLAN Match for received Frames This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC receiver enables filtering or matching for S-VLAN (Type = 0x88A8) packets. When this bit is reset, the MAC receiver enables filtering or matching for C-VLAN (Type = 0x8100) packets.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Receive S-VLAN Match is disabled 1b - Receive S-VLAN Match is enabled
18 DOVLTC	Disable VLAN Type Comparison This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC does not check whether the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit is of type S-VLAN or C-VLAN. When this bit is reset, the MAC filters or matches the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit only when VLAN Tag type is similar to the one specified by the Enable S-VLAN Match for received Frames bit. 0b - VLAN type comparison is enabled 1b - VLAN type comparison is disabled
17 ETV	12bits or 16bits VLAN comparison This bit is valid only when VEN of the Filter is set. When this bit is set, a 12-bit VLAN identifier is used for comparing and filtering instead of the complete 16-bit VLAN tag. Bits [11:0] of VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. 0b - 16 bit VLAN comparison 1b - 12 bit VLAN comparison
16 VEN	VLAN Tag Enable This bit is used to enable or disable the VLAN Tag. When this bit is set, the MAC compares the VLAN Tag of received packet with the VLAN Tag ID. When this bit is reset, no comparison is performed irrespective of the programming of the other fields. 0b - VLAN Tag is disabled 1b - VLAN Tag is enabled
15-0 VID	VLAN Tag ID This field holds the VLAN Tag value which is used by the MAC for perfect comparison. It is valid when VLAN Tag Enable is set.

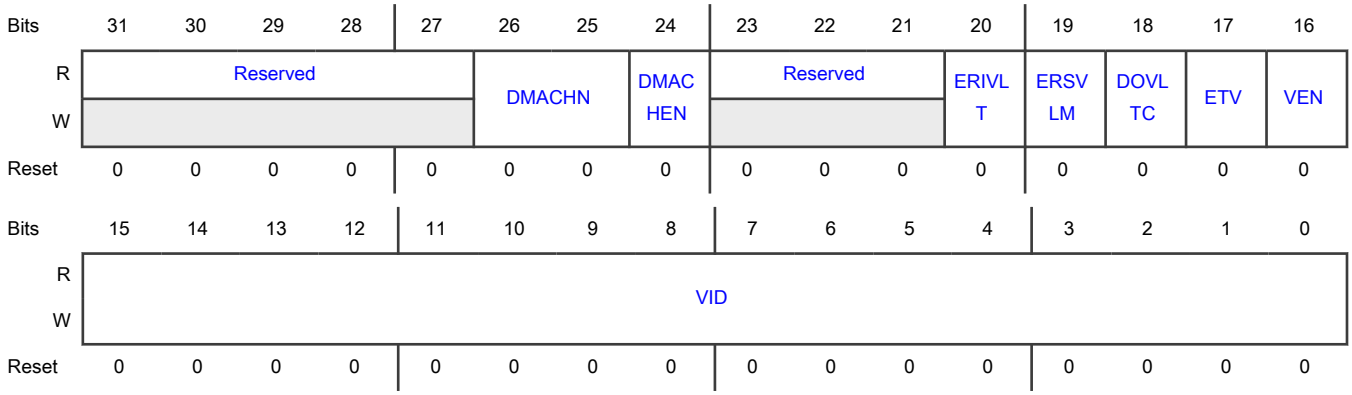
76.17.35 MAC VLAN Tag Filter 26 (MAC_VLAN_Tag_Filter26)

Offset

Register	Offset
MAC_VLAN_Tag_Filter26	54h

Function
This register contains VLAN Tag filter 26 control information.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-25 DMACHN	DMA Channel Number The DMA Channel number to which the VLAN Tagged Frame is to be routed if it passes this VLAN Tag Filter is programmed in this field. If the Routing based on VLAN Tag Filter is not necessary, this field need not be programmed.
24 DMACHEN	DMA Channel Number Enable This bit is the Enable for the DMA Channel Number value programmed in the field DMACH. When this bit is reset, the Routing does not occur based on VLAN Filter result. The frame is routed based on DA Based DMA Channel Routing. 0b - DMA Channel Number is disabled 1b - DMA Channel Number is enabled
23-21 —	Reserved
20 ERIVLT	Enable Inner VLAN Tag Comparison This bit is valid only when Double VLAN Tag Enable of the Filter is set. When this bit and the EDVLP field are set, the MAC receiver enables operation on the inner VLAN Tag (if present). When this bit is reset, the MAC receiver enables operation on the outer VLAN Tag (if present). 0b - Inner VLAN tag comparison is disabled 1b - Inner VLAN tag comparison is enabled
19 ERSVLM	Enable S-VLAN Match for received Frames This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC receiver enables filtering or matching for S-VLAN (Type = 0x88A8) packets. When this bit is reset, the MAC receiver enables filtering or matching for C-VLAN (Type = 0x8100) packets.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Receive S-VLAN Match is disabled 1b - Receive S-VLAN Match is enabled
18 DOVLTC	Disable VLAN Type Comparison This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC does not check whether the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit is of type S-VLAN or C-VLAN. When this bit is reset, the MAC filters or matches the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit only when VLAN Tag type is similar to the one specified by the Enable S-VLAN Match for received Frames bit. 0b - VLAN type comparison is enabled 1b - VLAN type comparison is disabled
17 ETV	12bits or 16bits VLAN comparison This bit is valid only when VEN of the Filter is set. When this bit is set, a 12-bit VLAN identifier is used for comparing and filtering instead of the complete 16-bit VLAN tag. Bits [11:0] of VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. 0b - 16 bit VLAN comparison 1b - 12 bit VLAN comparison
16 VEN	VLAN Tag Enable This bit is used to enable or disable the VLAN Tag. When this bit is set, the MAC compares the VLAN Tag of received packet with the VLAN Tag ID. When this bit is reset, no comparison is performed irrespective of the programming of the other fields. 0b - VLAN Tag is disabled 1b - VLAN Tag is enabled
15-0 VID	VLAN Tag ID This field holds the VLAN Tag value which is used by the MAC for perfect comparison. It is valid when VLAN Tag Enable is set.

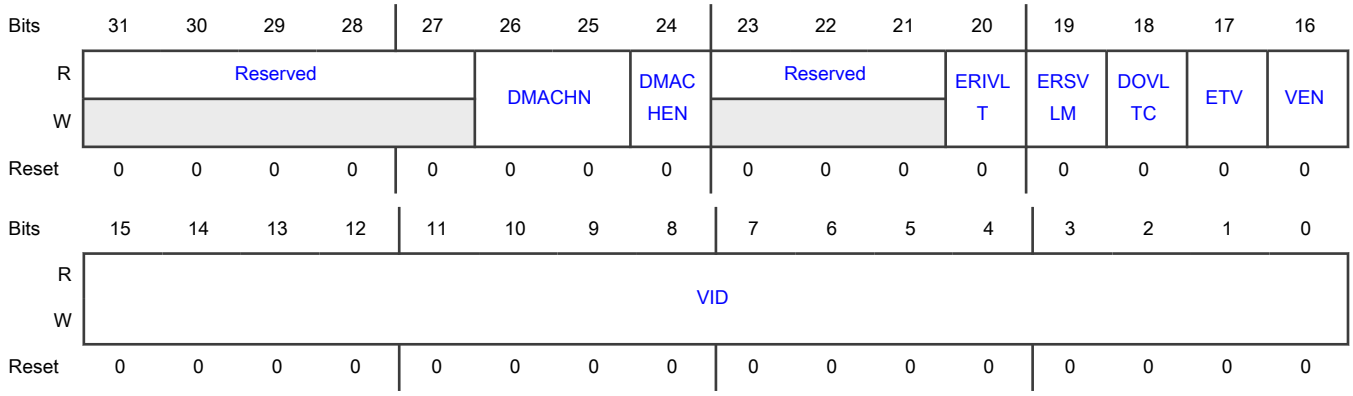
76.17.36 MAC VLAN Tag Filter 27 (MAC_VLAN_Tag_Filter27)

Offset

Register	Offset
MAC_VLAN_Tag_Filter27	54h

Function
This register contains VLAN Tag filter 27 control information.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-25 DMACHN	DMA Channel Number The DMA Channel number to which the VLAN Tagged Frame is to be routed if it passes this VLAN Tag Filter is programmed in this field. If the Routing based on VLAN Tag Filter is not necessary, this field need not be programmed.
24 DMACHEN	DMA Channel Number Enable This bit is the Enable for the DMA Channel Number value programmed in the field DMACH. When this bit is reset, the Routing does not occur based on VLAN Filter result. The frame is routed based on DA Based DMA Channel Routing. 0b - DMA Channel Number is disabled 1b - DMA Channel Number is enabled
23-21 —	Reserved
20 ERIVLT	Enable Inner VLAN Tag Comparison This bit is valid only when Double VLAN Tag Enable of the Filter is set. When this bit and the EDVLP field are set, the MAC receiver enables operation on the inner VLAN Tag (if present). When this bit is reset, the MAC receiver enables operation on the outer VLAN Tag (if present). 0b - Inner VLAN tag comparison is disabled 1b - Inner VLAN tag comparison is enabled
19 ERSVLM	Enable S-VLAN Match for received Frames This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC receiver enables filtering or matching for S-VLAN (Type = 0x88A8) packets. When this bit is reset, the MAC receiver enables filtering or matching for C-VLAN (Type = 0x8100) packets.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Receive S-VLAN Match is disabled 1b - Receive S-VLAN Match is enabled
18 DOVLTC	Disable VLAN Type Comparison This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC does not check whether the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit is of type S-VLAN or C-VLAN. When this bit is reset, the MAC filters or matches the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit only when VLAN Tag type is similar to the one specified by the Enable S-VLAN Match for received Frames bit. 0b - VLAN type comparison is enabled 1b - VLAN type comparison is disabled
17 ETV	12bits or 16bits VLAN comparison This bit is valid only when VEN of the Filter is set. When this bit is set, a 12-bit VLAN identifier is used for comparing and filtering instead of the complete 16-bit VLAN tag. Bits [11:0] of VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. 0b - 16 bit VLAN comparison 1b - 12 bit VLAN comparison
16 VEN	VLAN Tag Enable This bit is used to enable or disable the VLAN Tag. When this bit is set, the MAC compares the VLAN Tag of received packet with the VLAN Tag ID. When this bit is reset, no comparison is performed irrespective of the programming of the other fields. 0b - VLAN Tag is disabled 1b - VLAN Tag is enabled
15-0 VID	VLAN Tag ID This field holds the VLAN Tag value which is used by the MAC for perfect comparison. It is valid when VLAN Tag Enable is set.

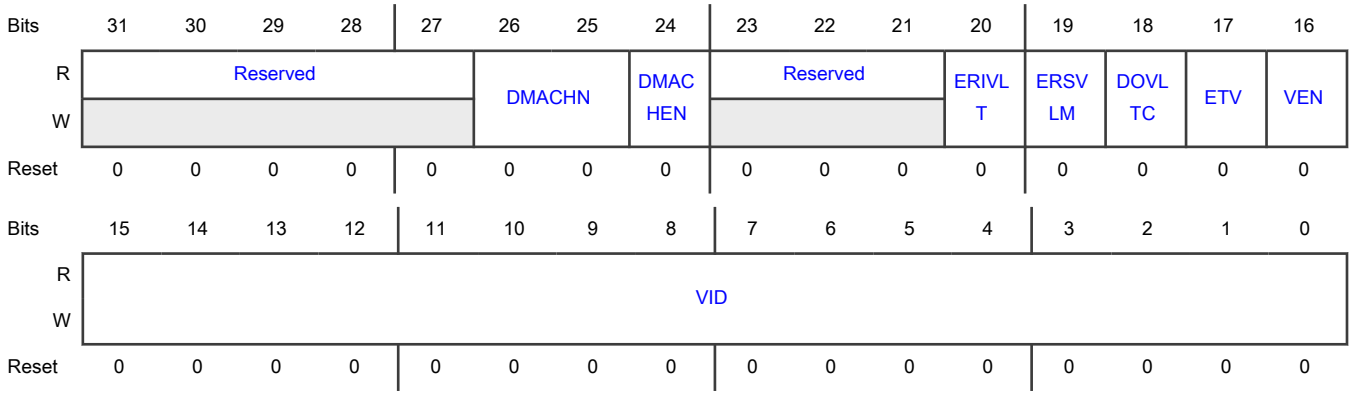
76.17.37 MAC VLAN Tag Filter 28 (MAC_VLAN_Tag_Filter28)

Offset

Register	Offset
MAC_VLAN_Tag_Filter28	54h

Function
This register contains VLAN Tag filter 28 control information.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-25 DMACHN	DMA Channel Number The DMA Channel number to which the VLAN Tagged Frame is to be routed if it passes this VLAN Tag Filter is programmed in this field. If the Routing based on VLAN Tag Filter is not necessary, this field need not be programmed.
24 DMACHEN	DMA Channel Number Enable This bit is the Enable for the DMA Channel Number value programmed in the field DMACH. When this bit is reset, the Routing does not occur based on VLAN Filter result. The frame is routed based on DA Based DMA Channel Routing. 0b - DMA Channel Number is disabled 1b - DMA Channel Number is enabled
23-21 —	Reserved
20 ERIVLT	Enable Inner VLAN Tag Comparison This bit is valid only when Double VLAN Tag Enable of the Filter is set. When this bit and the EDVLP field are set, the MAC receiver enables operation on the inner VLAN Tag (if present). When this bit is reset, the MAC receiver enables operation on the outer VLAN Tag (if present). 0b - Inner VLAN tag comparison is disabled 1b - Inner VLAN tag comparison is enabled
19 ERSVLM	Enable S-VLAN Match for received Frames This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC receiver enables filtering or matching for S-VLAN (Type = 0x88A8) packets. When this bit is reset, the MAC receiver enables filtering or matching for C-VLAN (Type = 0x8100) packets.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Receive S-VLAN Match is disabled 1b - Receive S-VLAN Match is enabled
18 DOVLTC	Disable VLAN Type Comparison This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC does not check whether the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit is of type S-VLAN or C-VLAN. When this bit is reset, the MAC filters or matches the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit only when VLAN Tag type is similar to the one specified by the Enable S-VLAN Match for received Frames bit. 0b - VLAN type comparison is enabled 1b - VLAN type comparison is disabled
17 ETV	12bits or 16bits VLAN comparison This bit is valid only when VEN of the Filter is set. When this bit is set, a 12-bit VLAN identifier is used for comparing and filtering instead of the complete 16-bit VLAN tag. Bits [11:0] of VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. 0b - 16 bit VLAN comparison 1b - 12 bit VLAN comparison
16 VEN	VLAN Tag Enable This bit is used to enable or disable the VLAN Tag. When this bit is set, the MAC compares the VLAN Tag of received packet with the VLAN Tag ID. When this bit is reset, no comparison is performed irrespective of the programming of the other fields. 0b - VLAN Tag is disabled 1b - VLAN Tag is enabled
15-0 VID	VLAN Tag ID This field holds the VLAN Tag value which is used by the MAC for perfect comparison. It is valid when VLAN Tag Enable is set.

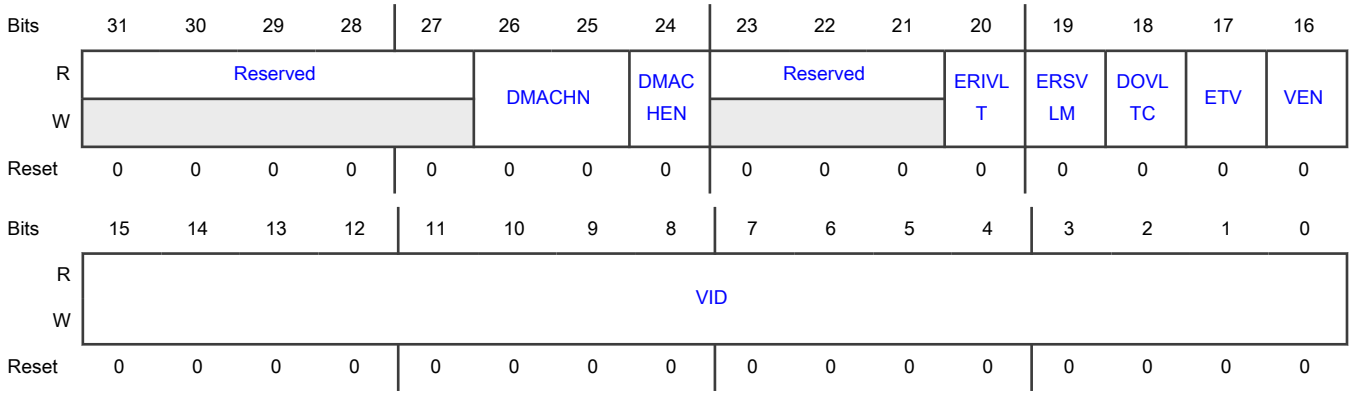
76.17.38 MAC VLAN Tag Filter 29 (MAC_VLAN_Tag_Filter29)

Offset

Register	Offset
MAC_VLAN_Tag_Filter29	54h

Function
This register contains VLAN Tag filter 29 control information.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-25 DMACHN	DMA Channel Number The DMA Channel number to which the VLAN Tagged Frame is to be routed if it passes this VLAN Tag Filter is programmed in this field. If the Routing based on VLAN Tag Filter is not necessary, this field need not be programmed.
24 DMACHEN	DMA Channel Number Enable This bit is the Enable for the DMA Channel Number value programmed in the field DMACH. When this bit is reset, the Routing does not occur based on VLAN Filter result. The frame is routed based on DA Based DMA Channel Routing. 0b - DMA Channel Number is disabled 1b - DMA Channel Number is enabled
23-21 —	Reserved
20 ERIVLT	Enable Inner VLAN Tag Comparison This bit is valid only when Double VLAN Tag Enable of the Filter is set. When this bit and the EDVLP field are set, the MAC receiver enables operation on the inner VLAN Tag (if present). When this bit is reset, the MAC receiver enables operation on the outer VLAN Tag (if present). 0b - Inner VLAN tag comparison is disabled 1b - Inner VLAN tag comparison is enabled
19 ERSVLM	Enable S-VLAN Match for received Frames This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC receiver enables filtering or matching for S-VLAN (Type = 0x88A8) packets. When this bit is reset, the MAC receiver enables filtering or matching for C-VLAN (Type = 0x8100) packets.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Receive S-VLAN Match is disabled 1b - Receive S-VLAN Match is enabled
18 DOVLTC	Disable VLAN Type Comparison This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC does not check whether the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit is of type S-VLAN or C-VLAN. When this bit is reset, the MAC filters or matches the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit only when VLAN Tag type is similar to the one specified by the Enable S-VLAN Match for received Frames bit. 0b - VLAN type comparison is enabled 1b - VLAN type comparison is disabled
17 ETV	12bits or 16bits VLAN comparison This bit is valid only when VEN of the Filter is set. When this bit is set, a 12-bit VLAN identifier is used for comparing and filtering instead of the complete 16-bit VLAN tag. Bits [11:0] of VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. 0b - 16 bit VLAN comparison 1b - 12 bit VLAN comparison
16 VEN	VLAN Tag Enable This bit is used to enable or disable the VLAN Tag. When this bit is set, the MAC compares the VLAN Tag of received packet with the VLAN Tag ID. When this bit is reset, no comparison is performed irrespective of the programming of the other fields. 0b - VLAN Tag is disabled 1b - VLAN Tag is enabled
15-0 VID	VLAN Tag ID This field holds the VLAN Tag value which is used by the MAC for perfect comparison. It is valid when VLAN Tag Enable is set.

76.17.39 MAC VLAN Tag Filter 3 (MAC_VLAN_Tag_Filter3)

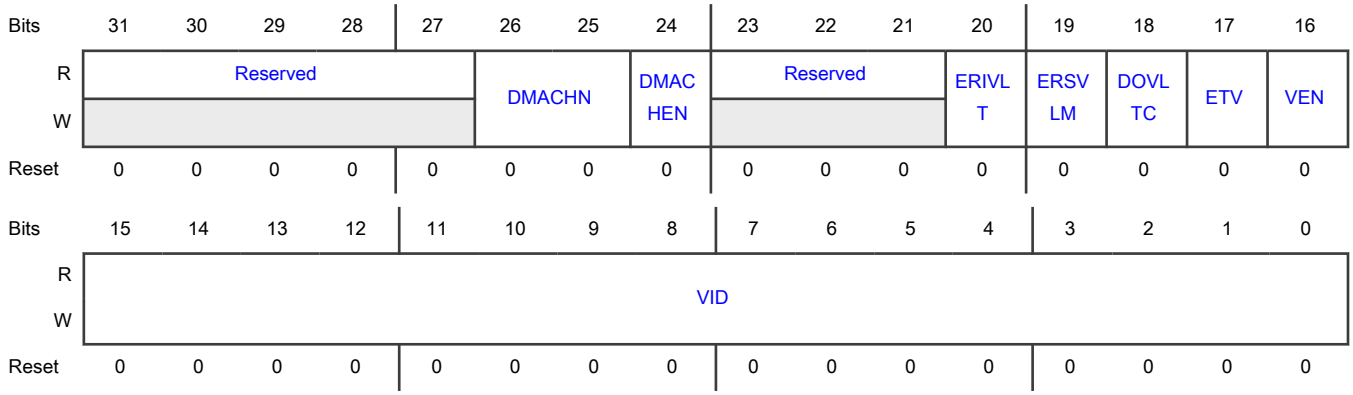
Offset

Register	Offset
MAC_VLAN_Tag_Filter3	54h

Function

This register contains VLAN Tag filter 3 control information.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-25 DMACHN	DMA Channel Number The DMA Channel number to which the VLAN Tagged Frame is to be routed if it passes this VLAN Tag Filter is programmed in this field. If the Routing based on VLAN Tag Filter is not necessary, this field need not be programmed.
24 DMACHEN	DMA Channel Number Enable This bit is the Enable for the DMA Channel Number value programmed in the field DMACH. When this bit is reset, the Routing does not occur based on VLAN Filter result. The frame is routed based on DA Based DMA Channel Routing. 0b - DMA Channel Number is disabled 1b - DMA Channel Number is enabled
23-21 —	Reserved
20 ERIVLT	Enable Inner VLAN Tag Comparison This bit is valid only when Double VLAN Tag Enable of the Filter is set. When this bit and the EDVLP field are set, the MAC receiver enables operation on the inner VLAN Tag (if present). When this bit is reset, the MAC receiver enables operation on the outer VLAN Tag (if present). 0b - Inner VLAN tag comparison is disabled 1b - Inner VLAN tag comparison is enabled
19 ERSVLM	Enable S-VLAN Match for received Frames This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC receiver enables filtering or matching for S-VLAN (Type = 0x88A8) packets. When this bit is reset, the MAC receiver enables filtering or matching for C-VLAN (Type = 0x8100) packets.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Receive S-VLAN Match is disabled 1b - Receive S-VLAN Match is enabled
18 DOVLTC	Disable VLAN Type Comparison This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC does not check whether the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit is of type S-VLAN or C-VLAN. When this bit is reset, the MAC filters or matches the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit only when VLAN Tag type is similar to the one specified by the Enable S-VLAN Match for received Frames bit. 0b - VLAN type comparison is enabled 1b - VLAN type comparison is disabled
17 ETV	12bits or 16bits VLAN comparison This bit is valid only when VEN of the Filter is set. When this bit is set, a 12-bit VLAN identifier is used for comparing and filtering instead of the complete 16-bit VLAN tag. Bits [11:0] of VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. 0b - 16 bit VLAN comparison 1b - 12 bit VLAN comparison
16 VEN	VLAN Tag Enable This bit is used to enable or disable the VLAN Tag. When this bit is set, the MAC compares the VLAN Tag of received packet with the VLAN Tag ID. When this bit is reset, no comparison is performed irrespective of the programming of the other fields. 0b - VLAN Tag is disabled 1b - VLAN Tag is enabled
15-0 VID	VLAN Tag ID This field holds the VLAN Tag value which is used by the MAC for perfect comparison. It is valid when VLAN Tag Enable is set.

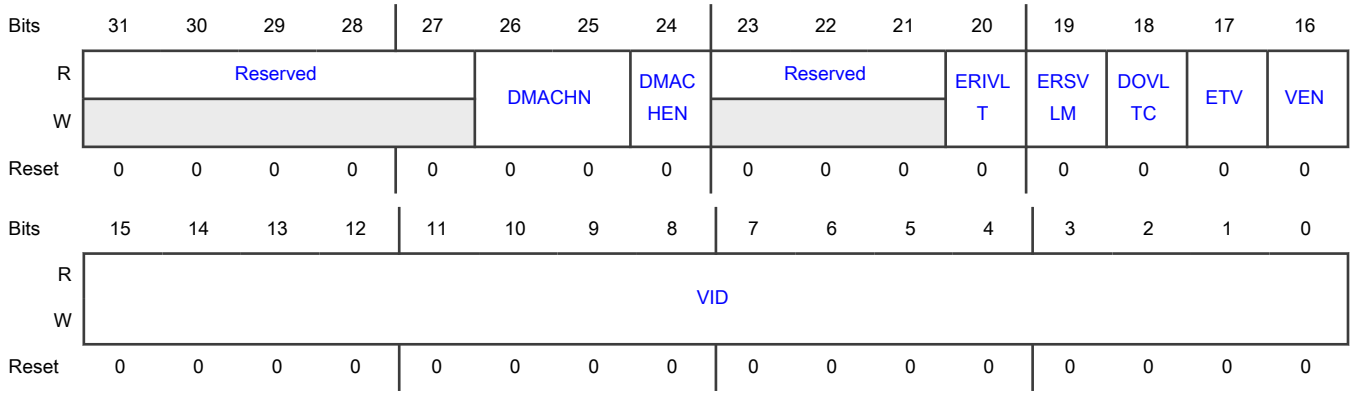
76.17.40 MAC VLAN Tag Filter 30 (MAC_VLAN_Tag_Filter30)

Offset

Register	Offset
MAC_VLAN_Tag_Filter30	54h

Function
This register contains VLAN Tag filter 30 control information.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-25 DMACHN	DMA Channel Number The DMA Channel number to which the VLAN Tagged Frame is to be routed if it passes this VLAN Tag Filter is programmed in this field. If the Routing based on VLAN Tag Filter is not necessary, this field need not be programmed.
24 DMACHEN	DMA Channel Number Enable This bit is the Enable for the DMA Channel Number value programmed in the field DMACH. When this bit is reset, the Routing does not occur based on VLAN Filter result. The frame is routed based on DA Based DMA Channel Routing. 0b - DMA Channel Number is disabled 1b - DMA Channel Number is enabled
23-21 —	Reserved
20 ERIVLT	Enable Inner VLAN Tag Comparison This bit is valid only when Double VLAN Tag Enable of the Filter is set. When this bit and the EDVLP field are set, the MAC receiver enables operation on the inner VLAN Tag (if present). When this bit is reset, the MAC receiver enables operation on the outer VLAN Tag (if present). 0b - Inner VLAN tag comparison is disabled 1b - Inner VLAN tag comparison is enabled
19 ERSVLM	Enable S-VLAN Match for received Frames This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC receiver enables filtering or matching for S-VLAN (Type = 0x88A8) packets. When this bit is reset, the MAC receiver enables filtering or matching for C-VLAN (Type = 0x8100) packets.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Receive S-VLAN Match is disabled 1b - Receive S-VLAN Match is enabled
18 DOVLTC	Disable VLAN Type Comparison This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC does not check whether the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit is of type S-VLAN or C-VLAN. When this bit is reset, the MAC filters or matches the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit only when VLAN Tag type is similar to the one specified by the Enable S-VLAN Match for received Frames bit. 0b - VLAN type comparison is enabled 1b - VLAN type comparison is disabled
17 ETV	12bits or 16bits VLAN comparison This bit is valid only when VEN of the Filter is set. When this bit is set, a 12-bit VLAN identifier is used for comparing and filtering instead of the complete 16-bit VLAN tag. Bits [11:0] of VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. 0b - 16 bit VLAN comparison 1b - 12 bit VLAN comparison
16 VEN	VLAN Tag Enable This bit is used to enable or disable the VLAN Tag. When this bit is set, the MAC compares the VLAN Tag of received packet with the VLAN Tag ID. When this bit is reset, no comparison is performed irrespective of the programming of the other fields. 0b - VLAN Tag is disabled 1b - VLAN Tag is enabled
15-0 VID	VLAN Tag ID This field holds the VLAN Tag value which is used by the MAC for perfect comparison. It is valid when VLAN Tag Enable is set.

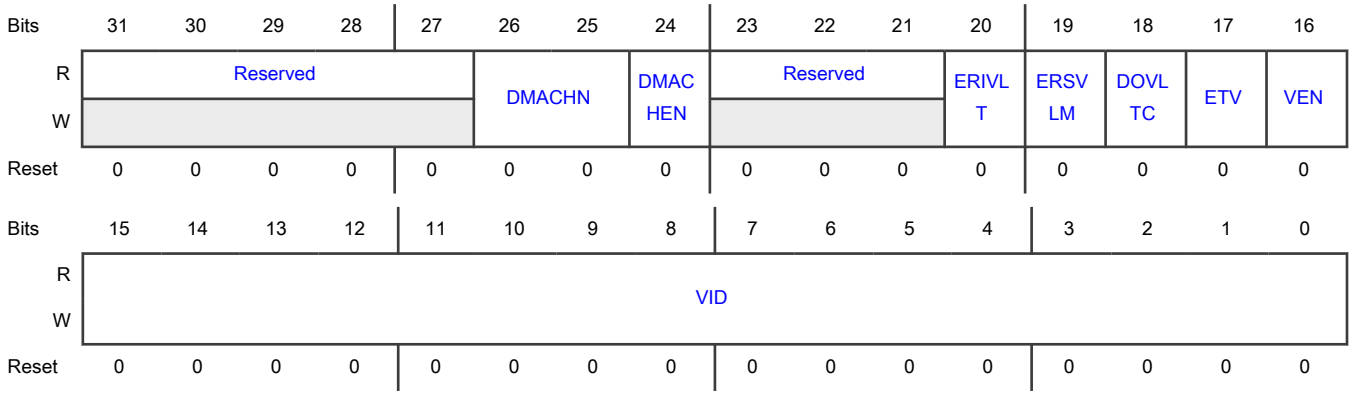
76.17.41 MAC VLAN Tag Filter 31 (MAC_VLAN_Tag_Filter31)

Offset

Register	Offset
MAC_VLAN_Tag_Filter31	54h

Function
This register contains VLAN Tag filter 31 control information.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-25 DMACHN	DMA Channel Number The DMA Channel number to which the VLAN Tagged Frame is to be routed if it passes this VLAN Tag Filter is programmed in this field. If the Routing based on VLAN Tag Filter is not necessary, this field need not be programmed.
24 DMACHEN	DMA Channel Number Enable This bit is the Enable for the DMA Channel Number value programmed in the field DMACH. When this bit is reset, the Routing does not occur based on VLAN Filter result. The frame is routed based on DA Based DMA Channel Routing. 0b - DMA Channel Number is disabled 1b - DMA Channel Number is enabled
23-21 —	Reserved
20 ERIVLT	Enable Inner VLAN Tag Comparison This bit is valid only when Double VLAN Tag Enable of the Filter is set. When this bit and the EDVLP field are set, the MAC receiver enables operation on the inner VLAN Tag (if present). When this bit is reset, the MAC receiver enables operation on the outer VLAN Tag (if present). 0b - Inner VLAN tag comparison is disabled 1b - Inner VLAN tag comparison is enabled
19 ERSVLM	Enable S-VLAN Match for received Frames This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC receiver enables filtering or matching for S-VLAN (Type = 0x88A8) packets. When this bit is reset, the MAC receiver enables filtering or matching for C-VLAN (Type = 0x8100) packets.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Receive S-VLAN Match is disabled 1b - Receive S-VLAN Match is enabled
18 DOVLTC	Disable VLAN Type Comparison This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC does not check whether the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit is of type S-VLAN or C-VLAN. When this bit is reset, the MAC filters or matches the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit only when VLAN Tag type is similar to the one specified by the Enable S-VLAN Match for received Frames bit. 0b - VLAN type comparison is enabled 1b - VLAN type comparison is disabled
17 ETV	12bits or 16bits VLAN comparison This bit is valid only when VEN of the Filter is set. When this bit is set, a 12-bit VLAN identifier is used for comparing and filtering instead of the complete 16-bit VLAN tag. Bits [11:0] of VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. 0b - 16 bit VLAN comparison 1b - 12 bit VLAN comparison
16 VEN	VLAN Tag Enable This bit is used to enable or disable the VLAN Tag. When this bit is set, the MAC compares the VLAN Tag of received packet with the VLAN Tag ID. When this bit is reset, no comparison is performed irrespective of the programming of the other fields. 0b - VLAN Tag is disabled 1b - VLAN Tag is enabled
15-0 VID	VLAN Tag ID This field holds the VLAN Tag value which is used by the MAC for perfect comparison. It is valid when VLAN Tag Enable is set.

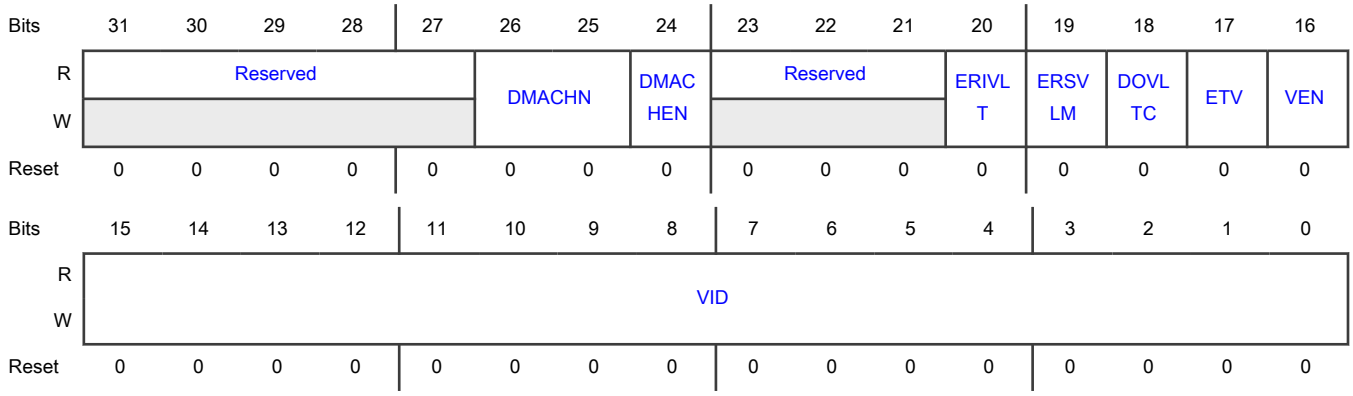
76.17.42 MAC VLAN Tag Filter 4 (MAC_VLAN_Tag_Filter4)

Offset

Register	Offset
MAC_VLAN_Tag_Filter4	54h

Function
This register contains VLAN Tag filter 4 control information.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-25 DMACHN	DMA Channel Number The DMA Channel number to which the VLAN Tagged Frame is to be routed if it passes this VLAN Tag Filter is programmed in this field. If the Routing based on VLAN Tag Filter is not necessary, this field need not be programmed.
24 DMACHEN	DMA Channel Number Enable This bit is the Enable for the DMA Channel Number value programmed in the field DMACH. When this bit is reset, the Routing does not occur based on VLAN Filter result. The frame is routed based on DA Based DMA Channel Routing. 0b - DMA Channel Number is disabled 1b - DMA Channel Number is enabled
23-21 —	Reserved
20 ERIVLT	Enable Inner VLAN Tag Comparison This bit is valid only when Double VLAN Tag Enable of the Filter is set. When this bit and the EDVLP field are set, the MAC receiver enables operation on the inner VLAN Tag (if present). When this bit is reset, the MAC receiver enables operation on the outer VLAN Tag (if present). 0b - Inner VLAN tag comparison is disabled 1b - Inner VLAN tag comparison is enabled
19 ERSVLM	Enable S-VLAN Match for received Frames This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC receiver enables filtering or matching for S-VLAN (Type = 0x88A8) packets. When this bit is reset, the MAC receiver enables filtering or matching for C-VLAN (Type = 0x8100) packets.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Receive S-VLAN Match is disabled 1b - Receive S-VLAN Match is enabled
18 DOVLTC	Disable VLAN Type Comparison This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC does not check whether the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit is of type S-VLAN or C-VLAN. When this bit is reset, the MAC filters or matches the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit only when VLAN Tag type is similar to the one specified by the Enable S-VLAN Match for received Frames bit. 0b - VLAN type comparison is enabled 1b - VLAN type comparison is disabled
17 ETV	12bits or 16bits VLAN comparison This bit is valid only when VEN of the Filter is set. When this bit is set, a 12-bit VLAN identifier is used for comparing and filtering instead of the complete 16-bit VLAN tag. Bits [11:0] of VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. 0b - 16 bit VLAN comparison 1b - 12 bit VLAN comparison
16 VEN	VLAN Tag Enable This bit is used to enable or disable the VLAN Tag. When this bit is set, the MAC compares the VLAN Tag of received packet with the VLAN Tag ID. When this bit is reset, no comparison is performed irrespective of the programming of the other fields. 0b - VLAN Tag is disabled 1b - VLAN Tag is enabled
15-0 VID	VLAN Tag ID This field holds the VLAN Tag value which is used by the MAC for perfect comparison. It is valid when VLAN Tag Enable is set.

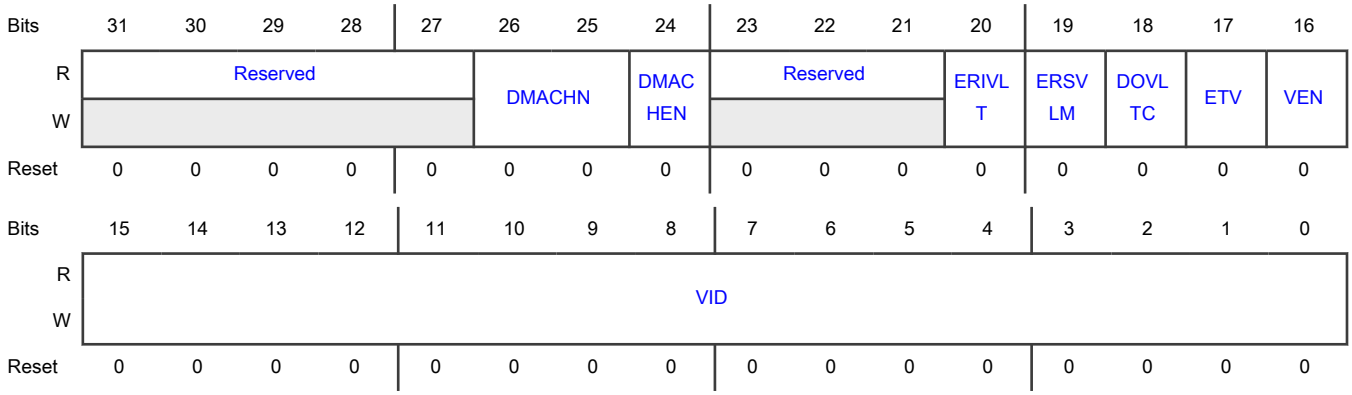
76.17.43 MAC VLAN Tag Filter 5 (MAC_VLAN_Tag_Filter5)

Offset

Register	Offset
MAC_VLAN_Tag_Filter5	54h

Function
This register contains VLAN Tag filter 5 control information.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-25 DMACHN	DMA Channel Number The DMA Channel number to which the VLAN Tagged Frame is to be routed if it passes this VLAN Tag Filter is programmed in this field. If the Routing based on VLAN Tag Filter is not necessary, this field need not be programmed.
24 DMACHEN	DMA Channel Number Enable This bit is the Enable for the DMA Channel Number value programmed in the field DMACH. When this bit is reset, the Routing does not occur based on VLAN Filter result. The frame is routed based on DA Based DMA Channel Routing. 0b - DMA Channel Number is disabled 1b - DMA Channel Number is enabled
23-21 —	Reserved
20 ERIVLT	Enable Inner VLAN Tag Comparison This bit is valid only when Double VLAN Tag Enable of the Filter is set. When this bit and the EDVLP field are set, the MAC receiver enables operation on the inner VLAN Tag (if present). When this bit is reset, the MAC receiver enables operation on the outer VLAN Tag (if present). 0b - Inner VLAN tag comparison is disabled 1b - Inner VLAN tag comparison is enabled
19 ERSVLM	Enable S-VLAN Match for received Frames This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC receiver enables filtering or matching for S-VLAN (Type = 0x88A8) packets. When this bit is reset, the MAC receiver enables filtering or matching for C-VLAN (Type = 0x8100) packets.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Receive S-VLAN Match is disabled 1b - Receive S-VLAN Match is enabled
18 DOVLTC	Disable VLAN Type Comparison This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC does not check whether the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit is of type S-VLAN or C-VLAN. When this bit is reset, the MAC filters or matches the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit only when VLAN Tag type is similar to the one specified by the Enable S-VLAN Match for received Frames bit. 0b - VLAN type comparison is enabled 1b - VLAN type comparison is disabled
17 ETV	12bits or 16bits VLAN comparison This bit is valid only when VEN of the Filter is set. When this bit is set, a 12-bit VLAN identifier is used for comparing and filtering instead of the complete 16-bit VLAN tag. Bits [11:0] of VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. 0b - 16 bit VLAN comparison 1b - 12 bit VLAN comparison
16 VEN	VLAN Tag Enable This bit is used to enable or disable the VLAN Tag. When this bit is set, the MAC compares the VLAN Tag of received packet with the VLAN Tag ID. When this bit is reset, no comparison is performed irrespective of the programming of the other fields. 0b - VLAN Tag is disabled 1b - VLAN Tag is enabled
15-0 VID	VLAN Tag ID This field holds the VLAN Tag value which is used by the MAC for perfect comparison. It is valid when VLAN Tag Enable is set.

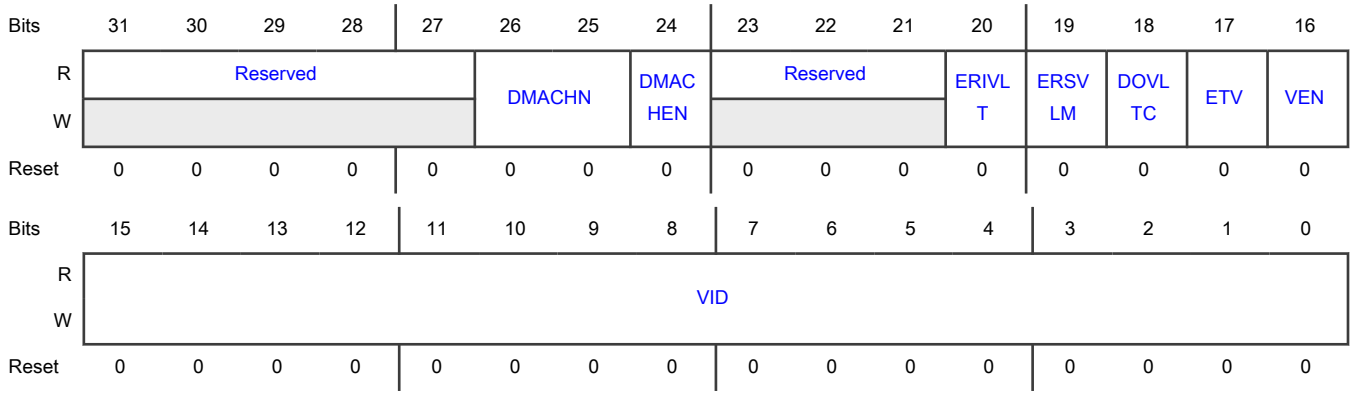
76.17.44 MAC VLAN Tag Filter 6 (MAC_VLAN_Tag_Filter6)

Offset

Register	Offset
MAC_VLAN_Tag_Filter6	54h

Function
This register contains VLAN Tag filter 6 control information.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-25 DMACHN	DMA Channel Number The DMA Channel number to which the VLAN Tagged Frame is to be routed if it passes this VLAN Tag Filter is programmed in this field. If the Routing based on VLAN Tag Filter is not necessary, this field need not be programmed.
24 DMACHEN	DMA Channel Number Enable This bit is the Enable for the DMA Channel Number value programmed in the field DMACH. When this bit is reset, the Routing does not occur based on VLAN Filter result. The frame is routed based on DA Based DMA Channel Routing. 0b - DMA Channel Number is disabled 1b - DMA Channel Number is enabled
23-21 —	Reserved
20 ERIVLT	Enable Inner VLAN Tag Comparison This bit is valid only when Double VLAN Tag Enable of the Filter is set. When this bit and the EDVLP field are set, the MAC receiver enables operation on the inner VLAN Tag (if present). When this bit is reset, the MAC receiver enables operation on the outer VLAN Tag (if present). 0b - Inner VLAN tag comparison is disabled 1b - Inner VLAN tag comparison is enabled
19 ERSVLM	Enable S-VLAN Match for received Frames This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC receiver enables filtering or matching for S-VLAN (Type = 0x88A8) packets. When this bit is reset, the MAC receiver enables filtering or matching for C-VLAN (Type = 0x8100) packets.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Receive S-VLAN Match is disabled 1b - Receive S-VLAN Match is enabled
18 DOVLTC	Disable VLAN Type Comparison This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC does not check whether the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit is of type S-VLAN or C-VLAN. When this bit is reset, the MAC filters or matches the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit only when VLAN Tag type is similar to the one specified by the Enable S-VLAN Match for received Frames bit. 0b - VLAN type comparison is enabled 1b - VLAN type comparison is disabled
17 ETV	12bits or 16bits VLAN comparison This bit is valid only when VEN of the Filter is set. When this bit is set, a 12-bit VLAN identifier is used for comparing and filtering instead of the complete 16-bit VLAN tag. Bits [11:0] of VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. 0b - 16 bit VLAN comparison 1b - 12 bit VLAN comparison
16 VEN	VLAN Tag Enable This bit is used to enable or disable the VLAN Tag. When this bit is set, the MAC compares the VLAN Tag of received packet with the VLAN Tag ID. When this bit is reset, no comparison is performed irrespective of the programming of the other fields. 0b - VLAN Tag is disabled 1b - VLAN Tag is enabled
15-0 VID	VLAN Tag ID This field holds the VLAN Tag value which is used by the MAC for perfect comparison. It is valid when VLAN Tag Enable is set.

76.17.45 MAC VLAN Tag Filter 7 (MAC_VLAN_Tag_Filter7)

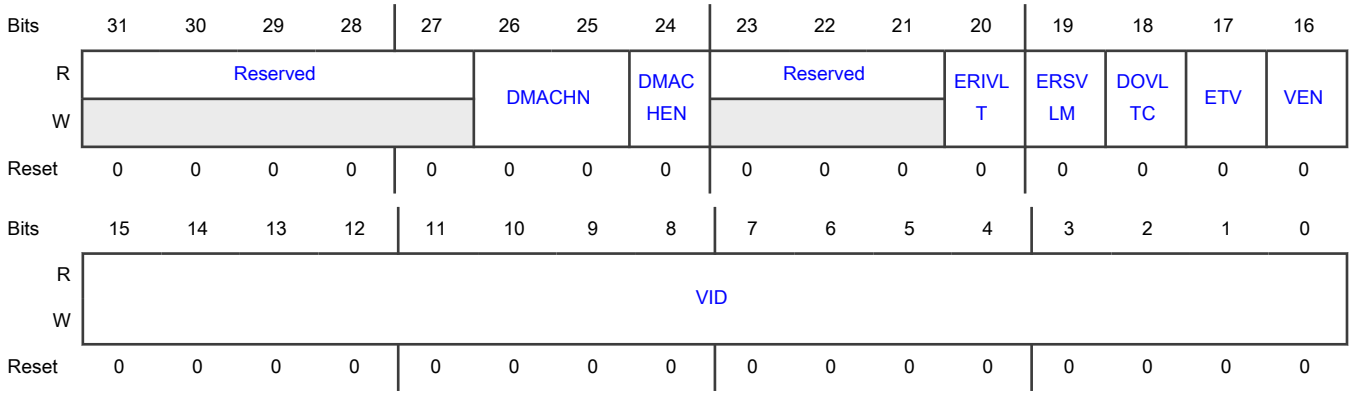
Offset

Register	Offset
MAC_VLAN_Tag_Filter7	54h

Function

This register contains VLAN Tag filter 7 control information.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-25 DMACHN	DMA Channel Number The DMA Channel number to which the VLAN Tagged Frame is to be routed if it passes this VLAN Tag Filter is programmed in this field. If the Routing based on VLAN Tag Filter is not necessary, this field need not be programmed.
24 DMACHEN	DMA Channel Number Enable This bit is the Enable for the DMA Channel Number value programmed in the field DMACH. When this bit is reset, the Routing does not occur based on VLAN Filter result. The frame is routed based on DA Based DMA Channel Routing. 0b - DMA Channel Number is disabled 1b - DMA Channel Number is enabled
23-21 —	Reserved
20 ERIVLT	Enable Inner VLAN Tag Comparison This bit is valid only when Double VLAN Tag Enable of the Filter is set. When this bit and the EDVLP field are set, the MAC receiver enables operation on the inner VLAN Tag (if present). When this bit is reset, the MAC receiver enables operation on the outer VLAN Tag (if present). 0b - Inner VLAN tag comparison is disabled 1b - Inner VLAN tag comparison is enabled
19 ERSVLM	Enable S-VLAN Match for received Frames This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC receiver enables filtering or matching for S-VLAN (Type = 0x88A8) packets. When this bit is reset, the MAC receiver enables filtering or matching for C-VLAN (Type = 0x8100) packets.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Receive S-VLAN Match is disabled 1b - Receive S-VLAN Match is enabled
18 DOVLTC	Disable VLAN Type Comparison This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC does not check whether the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit is of type S-VLAN or C-VLAN. When this bit is reset, the MAC filters or matches the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit only when VLAN Tag type is similar to the one specified by the Enable S-VLAN Match for received Frames bit. 0b - VLAN type comparison is enabled 1b - VLAN type comparison is disabled
17 ETV	12bits or 16bits VLAN comparison This bit is valid only when VEN of the Filter is set. When this bit is set, a 12-bit VLAN identifier is used for comparing and filtering instead of the complete 16-bit VLAN tag. Bits [11:0] of VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. 0b - 16 bit VLAN comparison 1b - 12 bit VLAN comparison
16 VEN	VLAN Tag Enable This bit is used to enable or disable the VLAN Tag. When this bit is set, the MAC compares the VLAN Tag of received packet with the VLAN Tag ID. When this bit is reset, no comparison is performed irrespective of the programming of the other fields. 0b - VLAN Tag is disabled 1b - VLAN Tag is enabled
15-0 VID	VLAN Tag ID This field holds the VLAN Tag value which is used by the MAC for perfect comparison. It is valid when VLAN Tag Enable is set.

76.17.46 MAC VLAN Tag Filter 8 (MAC_VLAN_Tag_Filter8)

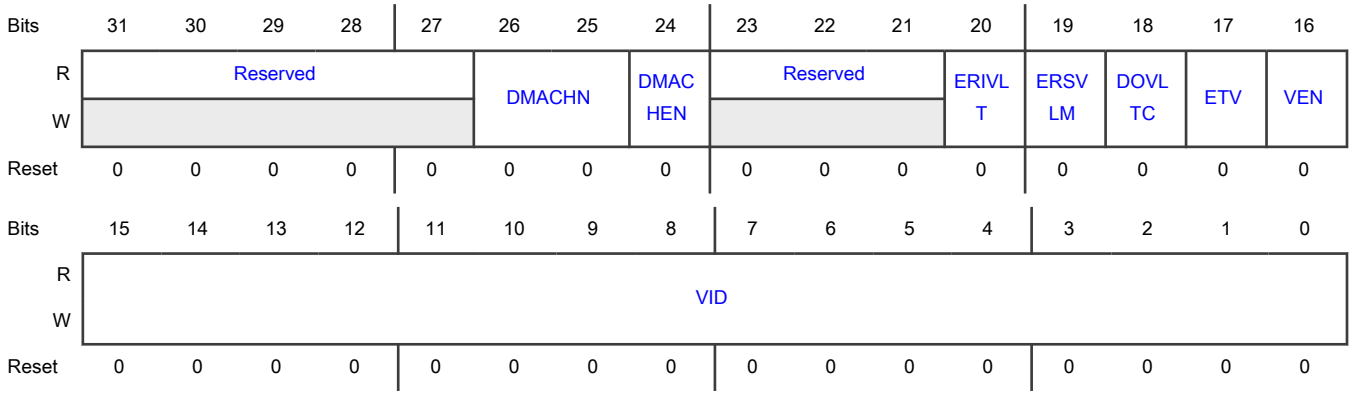
Offset

Register	Offset
MAC_VLAN_Tag_Filter8	54h

Function

This register contains VLAN Tag filter 8 control information.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-25 DMACHN	DMA Channel Number The DMA Channel number to which the VLAN Tagged Frame is to be routed if it passes this VLAN Tag Filter is programmed in this field. If the Routing based on VLAN Tag Filter is not necessary, this field need not be programmed.
24 DMACHEN	DMA Channel Number Enable This bit is the Enable for the DMA Channel Number value programmed in the field DMACH. When this bit is reset, the Routing does not occur based on VLAN Filter result. The frame is routed based on DA Based DMA Channel Routing. 0b - DMA Channel Number is disabled 1b - DMA Channel Number is enabled
23-21 —	Reserved
20 ERIVLT	Enable Inner VLAN Tag Comparison This bit is valid only when Double VLAN Tag Enable of the Filter is set. When this bit and the EDVLP field are set, the MAC receiver enables operation on the inner VLAN Tag (if present). When this bit is reset, the MAC receiver enables operation on the outer VLAN Tag (if present). 0b - Inner VLAN tag comparison is disabled 1b - Inner VLAN tag comparison is enabled
19 ERSVLM	Enable S-VLAN Match for received Frames This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC receiver enables filtering or matching for S-VLAN (Type = 0x88A8) packets. When this bit is reset, the MAC receiver enables filtering or matching for C-VLAN (Type = 0x8100) packets.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Receive S-VLAN Match is disabled 1b - Receive S-VLAN Match is enabled
18 DOVLTC	Disable VLAN Type Comparison This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC does not check whether the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit is of type S-VLAN or C-VLAN. When this bit is reset, the MAC filters or matches the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit only when VLAN Tag type is similar to the one specified by the Enable S-VLAN Match for received Frames bit. 0b - VLAN type comparison is enabled 1b - VLAN type comparison is disabled
17 ETV	12bits or 16bits VLAN comparison This bit is valid only when VEN of the Filter is set. When this bit is set, a 12-bit VLAN identifier is used for comparing and filtering instead of the complete 16-bit VLAN tag. Bits [11:0] of VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. 0b - 16 bit VLAN comparison 1b - 12 bit VLAN comparison
16 VEN	VLAN Tag Enable This bit is used to enable or disable the VLAN Tag. When this bit is set, the MAC compares the VLAN Tag of received packet with the VLAN Tag ID. When this bit is reset, no comparison is performed irrespective of the programming of the other fields. 0b - VLAN Tag is disabled 1b - VLAN Tag is enabled
15-0 VID	VLAN Tag ID This field holds the VLAN Tag value which is used by the MAC for perfect comparison. It is valid when VLAN Tag Enable is set.

76.17.47 MAC VLAN Tag Filter 9 (MAC_VLAN_Tag_Filter9)

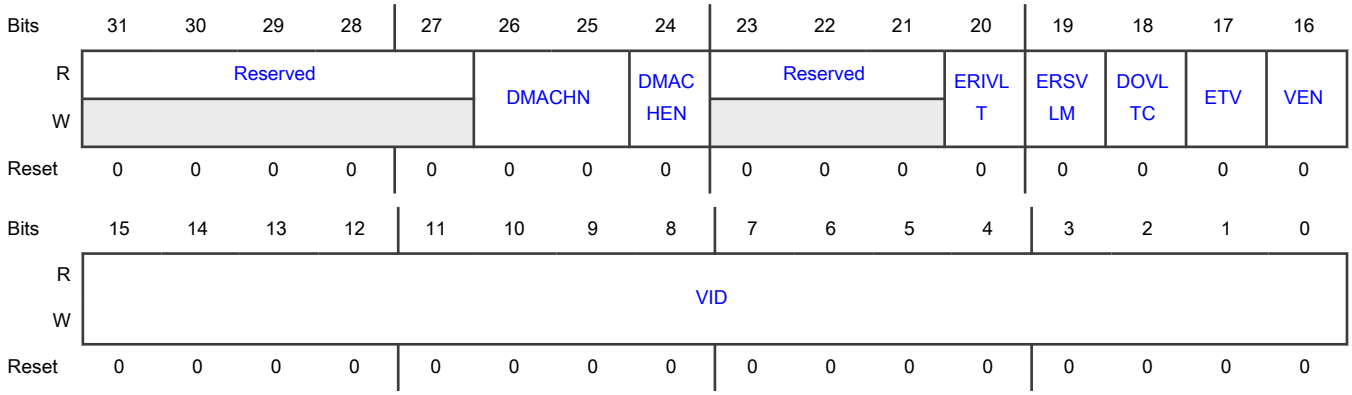
Offset

Register	Offset
MAC_VLAN_Tag_Filter9	54h

Function

This register contains VLAN Tag filter 9 control information.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-25 DMACHN	DMA Channel Number The DMA Channel number to which the VLAN Tagged Frame is to be routed if it passes this VLAN Tag Filter is programmed in this field. If the Routing based on VLAN Tag Filter is not necessary, this field need not be programmed.
24 DMACHEN	DMA Channel Number Enable This bit is the Enable for the DMA Channel Number value programmed in the field DMACH. When this bit is reset, the Routing does not occur based on VLAN Filter result. The frame is routed based on DA Based DMA Channel Routing. 0b - DMA Channel Number is disabled 1b - DMA Channel Number is enabled
23-21 —	Reserved
20 ERIVLT	Enable Inner VLAN Tag Comparison This bit is valid only when Double VLAN Tag Enable of the Filter is set. When this bit and the EDVLP field are set, the MAC receiver enables operation on the inner VLAN Tag (if present). When this bit is reset, the MAC receiver enables operation on the outer VLAN Tag (if present). 0b - Inner VLAN tag comparison is disabled 1b - Inner VLAN tag comparison is enabled
19 ERSVLM	Enable S-VLAN Match for received Frames This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC receiver enables filtering or matching for S-VLAN (Type = 0x88A8) packets. When this bit is reset, the MAC receiver enables filtering or matching for C-VLAN (Type = 0x8100) packets.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Receive S-VLAN Match is disabled 1b - Receive S-VLAN Match is enabled
18 DOVLTC	Disable VLAN Type Comparison This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC does not check whether the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit is of type S-VLAN or C-VLAN. When this bit is reset, the MAC filters or matches the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit only when VLAN Tag type is similar to the one specified by the Enable S-VLAN Match for received Frames bit. 0b - VLAN type comparison is enabled 1b - VLAN type comparison is disabled
17 ETV	12bits or 16bits VLAN comparison This bit is valid only when VEN of the Filter is set. When this bit is set, a 12-bit VLAN identifier is used for comparing and filtering instead of the complete 16-bit VLAN tag. Bits [11:0] of VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. 0b - 16 bit VLAN comparison 1b - 12 bit VLAN comparison
16 VEN	VLAN Tag Enable This bit is used to enable or disable the VLAN Tag. When this bit is set, the MAC compares the VLAN Tag of received packet with the VLAN Tag ID. When this bit is reset, no comparison is performed irrespective of the programming of the other fields. 0b - VLAN Tag is disabled 1b - VLAN Tag is enabled
15-0 VID	VLAN Tag ID This field holds the VLAN Tag value which is used by the MAC for perfect comparison. It is valid when VLAN Tag Enable is set.

76.17.48 MAC VLAN Hash Table (MAC_VLAN_Hash_Table)

Offset

Register	Offset
MAC_VLAN_Hash_Table	58h

Function

When VTHM bit of the MAC_VLAN_Tag register is set, the 16-bit VLAN Hash Table register is used for group address filtering based on the VLAN tag. For hash filtering, the content of the 16-bit VLAN tag or 12-bit VLAN ID (based on the ETV bit of MAC_VLAN_Tag Register) in the incoming packet is passed through the CRC logic. The upper four bits of the calculated CRC value are used to index the contents of the VLAN Hash table. For example, hash value of 4b'1000 selects Bit 8 of the VLAN Hash table. The hash value of the destination address is calculated in the following way: - Calculate the 32-bit CRC for the

VLAN tag or ID (For steps to calculate CRC32, see Section 3.2.8 of IEEE 802.3). - Perform bitwise reversal for the value obtained in step 1. - Take the upper four bits from the value obtained in step 2. If the VLAN hash Table register is configured to be double-synchronized to the (G)MII clock domain, the synchronization is triggered only when Bits[15:8] (in little-endian mode) or Bits[7:0] (in big-endian mode) of this register are written. - If double-synchronization is enabled, consecutive writes to this register should be performed after at least four clock cycles in the destination clock domain.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	VLHT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-0 VLHT	VLAN Hash Table This field contains the 16-bit VLAN Hash Table.

76.17.49 MAC VLAN Inclusion Or Replacement (MAC_VLAN_Incl)

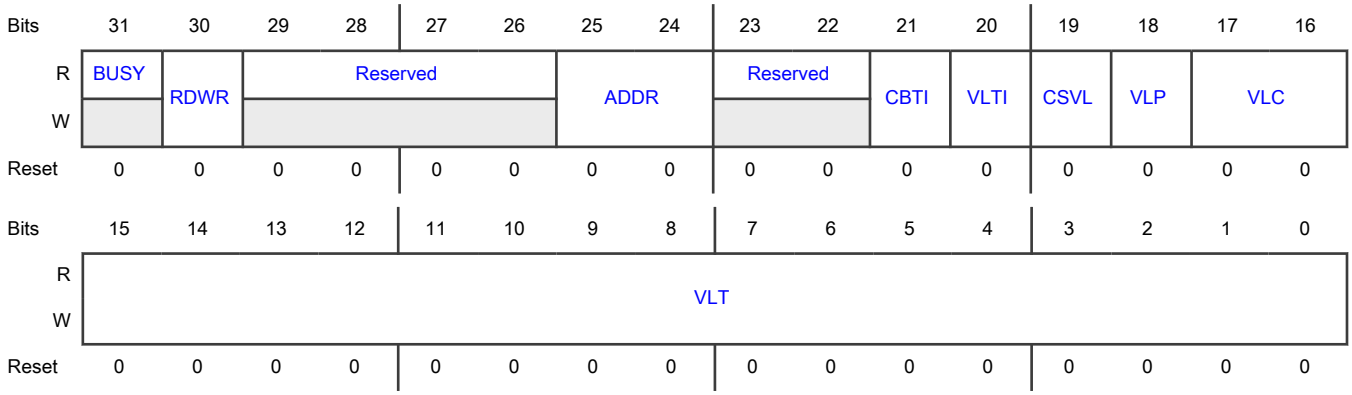
Offset

Register	Offset
MAC_VLAN_Incl	60h

Function

The VLAN Tag Inclusion or Replacement register contains the VLAN tag for insertion or replacement in the Transmit packets. It also contains the VLAN tag insertion controls.

Diagram



Fields

Field	Function
31 BUSY	Busy This bit indicates the status of the read/write operation of indirect access to the queue/channel specific VLAN inclusion register. For write operation write to a register is complete when this bit is reset. For read operation the read data is valid when the bit is reset. The application must make sure that this bit is reset before attempting subsequent access to this register. **NOTE** This field is not testable by an automated register test. 0b - Busy status not detected 1b - Busy status detected
30 RDWR	Read write control This bit controls the read or write operation for indirectly accessing the queue/channel specific VLAN Inclusion register. When set indicates write operation and when reset indicates read operation. This does not have any effect when CBTI is reset. 0b - Read operation of indirect access 1b - Write operation of indirect access
29-26 —	Reserved
25-24 ADDR	Address This field selects one of the queue/channel specific VLAN Inclusion register for read/write access. This does not have any effect when CBTI is reset.
23-22 —	Reserved
21	Channel based tag insertion

Table continues on the next page...

Table continued from the previous page...

Field	Function
CBTI	When this bit is set, outer VLAN tag is inserted for every packets transmitted by the MAC. The tag value is taken from the queue/channel specific VLAN tag register. The VLTI, VLP, VLC, and VLT fields of this register are ignored when this bit is set. When this bit is set, a write operation to byte 3 of this register initiates the read/write access to the indirect register. When reset, outer VLAN operation is based on the setting of VLTI, VLP, VLC and VLT fields of this register. 0b - Channel based tag insertion is disabled 1b - Channel based tag insertion is enabled
20 VLTI	VLAN Tag Input When this bit is set, it indicates that the VLAN tag to be inserted or replaced in Tx packet should be taken from: - The Tx descriptor 0b - VLAN Tag Input is disabled 1b - VLAN Tag Input is enabled
19 CSVL	C-VLAN or S-VLAN When this bit is set, S-VLAN type (0x88A8) is inserted or replaced in the 13th and 14th bytes of transmitted packets. When this bit is reset, C-VLAN type (0x8100) is inserted or replaced in the 13th and 14th bytes of transmitted packets. 0b - C-VLAN type (0x8100) is inserted or replaced 1b - S-VLAN type (0x88A8) is inserted or replaced
18 VLP	VLAN Priority Control When this bit is set, the control bits[17:16] are used for VLAN deletion, insertion, or replacement. When this bit is reset, the mti_vlan_ctrl_i control input is used and bits[17:16] are ignored. 0b - VLAN Priority Control is disabled 1b - VLAN Priority Control is enabled
17-16 VLC	VLAN Tag Control in Transmit Packets - 2'b00: No VLAN tag deletion, insertion, or replacement - 2'b01: VLAN tag deletion The MAC removes the VLAN type (bytes 13 and 14) and VLAN tag (bytes 15 and 16) of all transmitted packets with VLAN tags. - 2'b10: VLAN tag insertion The MAC inserts VLT in bytes 15 and 16 of the packet after inserting the Type value (0x8100 or 0x88a8) in bytes 13 and 14. This operation is performed on all transmitted packets, irrespective of whether they already have a VLAN tag. - 2'b11: VLAN tag replacement The MAC replaces VLT in bytes 15 and 16 of all VLAN-type transmitted packets (Bytes 13 and 14 are 0x8100 or 0x88a8). Note: Changes to this field take effect only on the start of a packet. If you write this register field when a packet is being transmitted, only the subsequent packet can use the updated value, that is, the current packet does not use the updated value. 00b - No VLAN tag deletion, insertion, or replacement 01b - VLAN tag deletion 10b - VLAN tag insertion 11b - VLAN tag replacement

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-0 VLT	VLAN Tag for Transmit Packets This field contains the value of the VLAN tag to be inserted or replaced. The value must only be changed when the transmit lines are inactive or during the initialization phase. Bits[15:13] are the User Priority field, Bit 12 is the CFI/DEI field, and Bits[11:0] are the VID field in the VLAN tag. The following list describes the bits of this field: - Bits[15:13]: User Priority - Bit 12: Canonical Format Indicator (CFI) or Drop Eligible Indicator (DEI) - Bits[11:0]: VLAN Identifier (VID) field of VLAN tag

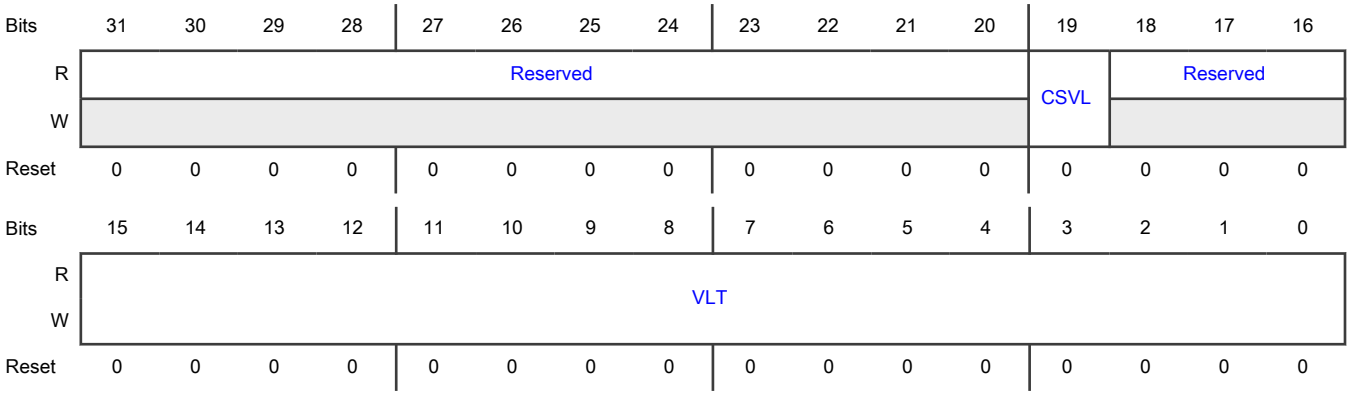
76.17.50 MAC VLAN Inclusion 0 (MAC_VLAN_Incl0)

Offset

Register	Offset
MAC_VLAN_Incl0	60h

Function
Contains the VLAN tag for insertion in the Transmit packets from Tx Queue 0. It also contains the VLAN tag insertion controls.

Diagram



Fields

Field	Function
31-20 —	Reserved
19 CSVL	C-VLAN or S-VLAN When this bit is set, S-VLAN type (0x88A8) is inserted in the 13th and 14th bytes of transmitted packets. When this bit is reset, C-VLAN type (0x8100) is inserted in the 13th and 14th bytes of transmitted packets.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - C-VLAN type (0x8100) is inserted 1b - S-VLAN type (0x88A8) is inserted
18-16 —	Reserved
15-0 VLT	VLAN Tag for Transmit Packets This field contains the value of the VLAN tag to be inserted. The value must only be changed when the transmit lines are inactive or during the initialization phase. Bits[15:13] are the User Priority field, Bit 12 is the CFI/DEI field, and Bits[11:0] are the VID field in the VLAN tag. The following list describes the bits of this field: - Bits[15:13]: User Priority - Bit 12: Canonical Format Indicator (CFI) or Drop Eligible Indicator (DEI) - Bits[11:0]: VLAN Identifier (VID) field of VLAN tag

76.17.51 MAC VLAN Inclusion 1 (MAC_VLAN_Incl1)

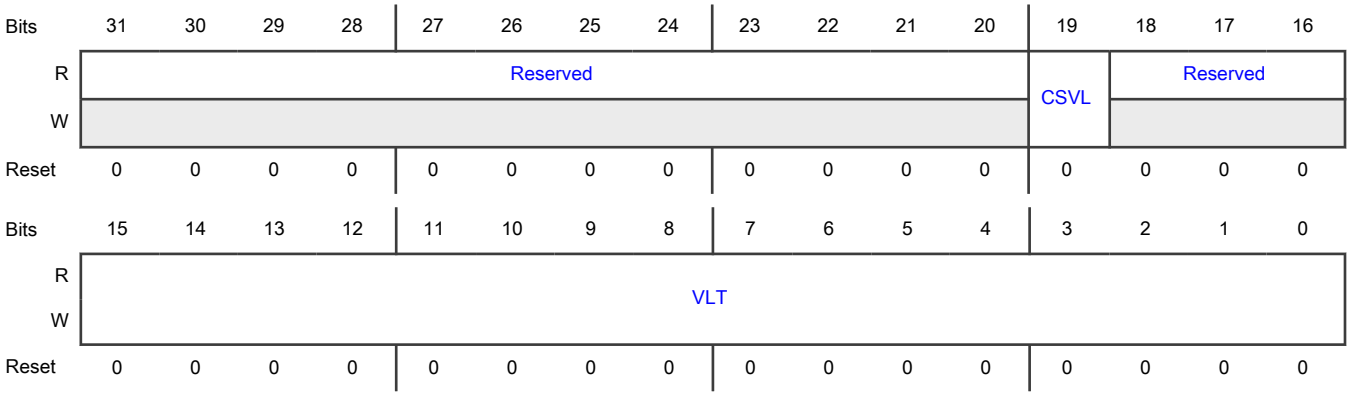
Offset

Register	Offset
MAC_VLAN_Incl1	60h

Function

Contains the VLAN tag for insertion in the Transmit packets from Tx Queue 1. It also contains the VLAN tag insertion controls.

Diagram



Fields

Field	Function
31-20 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
19 CSVL	C-VLAN or S-VLAN When this bit is set, S-VLAN type (0x88A8) is inserted in the 13th and 14th bytes of transmitted packets. When this bit is reset, C-VLAN type (0x8100) is inserted in the 13th and 14th bytes of transmitted packets. 0b - C-VLAN type (0x8100) is inserted 1b - S-VLAN type (0x88A8) is inserted
18-16 —	Reserved
15-0 VLT	VLAN Tag for Transmit Packets This field contains the value of the VLAN tag to be inserted. The value must only be changed when the transmit lines are inactive or during the initialization phase. Bits[15:13] are the User Priority field, Bit 12 is the CFI/DEI field, and Bits[11:0] are the VID field in the VLAN tag. The following list describes the bits of this field: - Bits[15:13]: User Priority - Bit 12: Canonical Format Indicator (CFI) or Drop Eligible Indicator (DEI) - Bits[11:0]: VLAN Identifier (VID) field of VLAN tag

76.17.52 MAC VLAN Inclusion 2 (MAC_VLAN_Incl2)

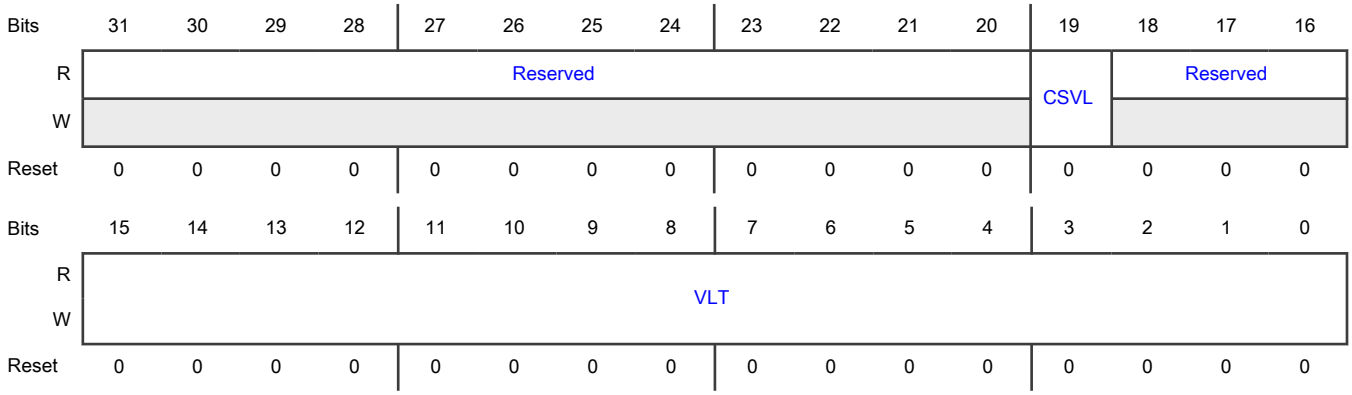
Offset

Register	Offset
MAC_VLAN_Incl2	60h

Function

Contains the VLAN tag for insertion in the Transmit packets from Tx Queue 2. It also contains the VLAN tag insertion controls.

Diagram



Fields

Field	Function
31-20 —	Reserved
19 CSVL	C-VLAN or S-VLAN When this bit is set, S-VLAN type (0x88A8) is inserted in the 13th and 14th bytes of transmitted packets. When this bit is reset, C-VLAN type (0x8100) is inserted in the 13th and 14th bytes of transmitted packets. 0b - C-VLAN type (0x8100) is inserted 1b - S-VLAN type (0x88A8) is inserted
18-16 —	Reserved
15-0 VLT	VLAN Tag for Transmit Packets This field contains the value of the VLAN tag to be inserted. The value must only be changed when the transmit lines are inactive or during the initialization phase. Bits[15:13] are the User Priority field, Bit 12 is the CFI/DEI field, and Bits[11:0] are the VID field in the VLAN tag. The following list describes the bits of this field: - Bits[15:13]: User Priority - Bit 12: Canonical Format Indicator (CFI) or Drop Eligible Indicator (DEI) - Bits[11:0]: VLAN Identifier (VID) field of VLAN tag

76.17.53 MAC VLAN Inclusion 3 (MAC_VLAN_Incl3)

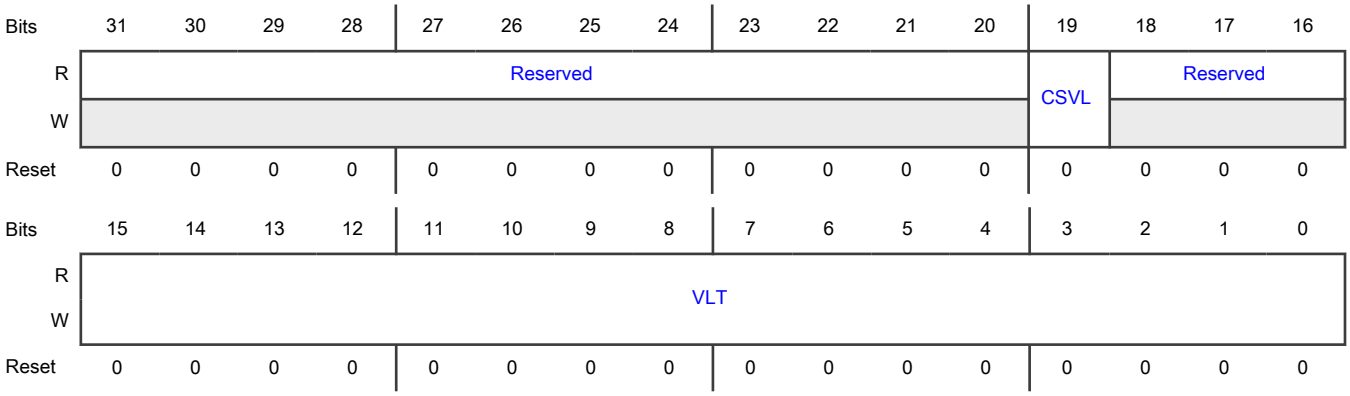
Offset

Register	Offset
MAC_VLAN_Incl3	60h

Function

Contains the VLAN tag for insertion in the Transmit packets from Tx Queue 3. It also contains the VLAN tag insertion controls.

Diagram



Fields

Field	Function
31-20 —	Reserved
19 CSVL	C-VLAN or S-VLAN When this bit is set, S-VLAN type (0x88A8) is inserted in the 13th and 14th bytes of transmitted packets. When this bit is reset, C-VLAN type (0x8100) is inserted in the 13th and 14th bytes of transmitted packets. 0b - C-VLAN type (0x8100) is inserted 1b - S-VLAN type (0x88A8) is inserted
18-16 —	Reserved
15-0 VLT	VLAN Tag for Transmit Packets This field contains the value of the VLAN tag to be inserted. The value must only be changed when the transmit lines are inactive or during the initialization phase. Bits[15:13] are the User Priority field, Bit 12 is the CFI/DEI field, and Bits[11:0] are the VID field in the VLAN tag. The following list describes the bits of this field: - Bits[15:13]: User Priority - Bit 12: Canonical Format Indicator (CFI) or Drop Eligible Indicator (DEI) - Bits[11:0]: VLAN Identifier (VID) field of VLAN tag

76.17.54 MAC VLAN Inclusion 4 (MAC_VLAN_Incl4)

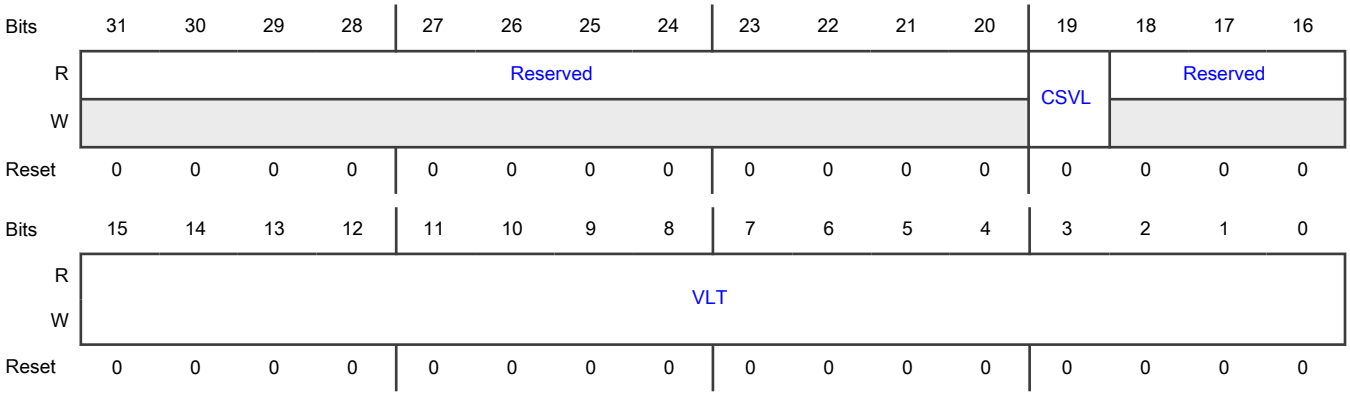
Offset

Register	Offset
MAC_VLAN_Incl4	60h

Function

Contains the VLAN tag for insertion in the Transmit packets from Tx Queue 4. It also contains the VLAN tag insertion controls.

Diagram



Fields

Field	Function
31-20 —	Reserved
19 CSVL	C-VLAN or S-VLAN When this bit is set, S-VLAN type (0x88A8) is inserted in the 13th and 14th bytes of transmitted packets. When this bit is reset, C-VLAN type (0x8100) is inserted in the 13th and 14th bytes of transmitted packets. 0b - C-VLAN type (0x8100) is inserted 1b - S-VLAN type (0x88A8) is inserted
18-16 —	Reserved
15-0 VLT	VLAN Tag for Transmit Packets This field contains the value of the VLAN tag to be inserted. The value must only be changed when the transmit lines are inactive or during the initialization phase. Bits[15:13] are the User Priority field, Bit 12 is the CFI/DEI field, and Bits[11:0] are the VID field in the VLAN tag. The following list describes the bits of this field: - Bits[15:13]: User Priority - Bit 12: Canonical Format Indicator (CFI) or Drop Eligible Indicator (DEI) - Bits[11:0]: VLAN Identifier (VID) field of VLAN tag

76.17.55 MAC VLAN Inclusion 5 (MAC_VLAN_Incl5)

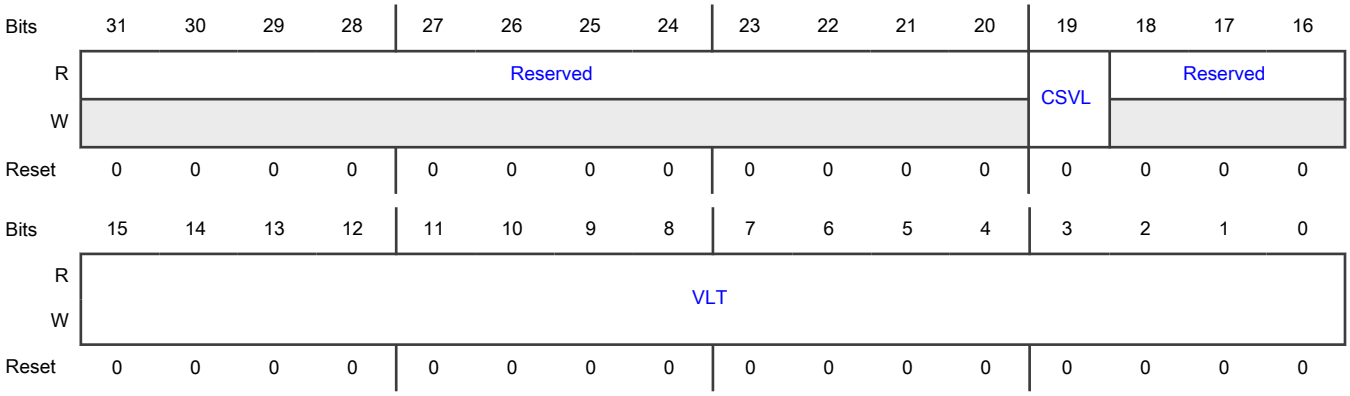
Offset

Register	Offset
MAC_VLAN_Incl5	60h

Function

Contains the VLAN tag for insertion in the Transmit packets from Tx Queue 5. It also contains the VLAN tag insertion controls.

Diagram



Fields

Field	Function
31-20 —	Reserved
19 CSVL	C-VLAN or S-VLAN When this bit is set, S-VLAN type (0x88A8) is inserted in the 13th and 14th bytes of transmitted packets. When this bit is reset, C-VLAN type (0x8100) is inserted in the 13th and 14th bytes of transmitted packets. 0b - C-VLAN type (0x8100) is inserted 1b - S-VLAN type (0x88A8) is inserted
18-16 —	Reserved
15-0 VLT	VLAN Tag for Transmit Packets This field contains the value of the VLAN tag to be inserted. The value must only be changed when the transmit lines are inactive or during the initialization phase. Bits[15:13] are the User Priority field, Bit 12 is the CFI/DEI field, and Bits[11:0] are the VID field in the VLAN tag. The following list describes the bits of this field: - Bits[15:13]: User Priority - Bit 12: Canonical Format Indicator (CFI) or Drop Eligible Indicator (DEI) - Bits[11:0]: VLAN Identifier (VID) field of VLAN tag

76.17.56 MAC VLAN Inclusion 6 (MAC_VLAN_Incl6)

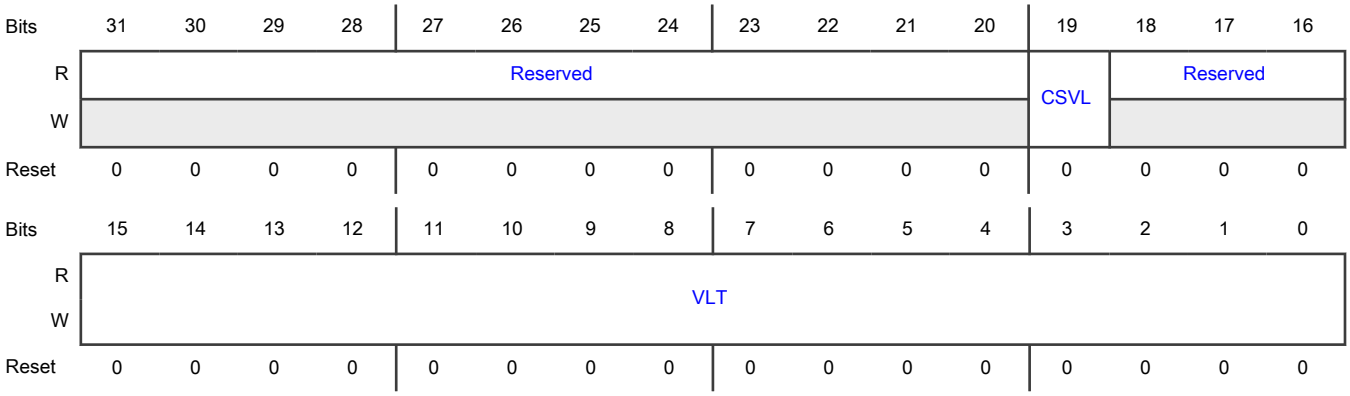
Offset

Register	Offset
MAC_VLAN_Incl6	60h

Function

Contains the VLAN tag for insertion in the Transmit packets from Tx Queue 6. It also contains the VLAN tag insertion controls.

Diagram



Fields

Field	Function
31-20 —	Reserved
19 CSVL	C-VLAN or S-VLAN When this bit is set, S-VLAN type (0x88A8) is inserted in the 13th and 14th bytes of transmitted packets. When this bit is reset, C-VLAN type (0x8100) is inserted in the 13th and 14th bytes of transmitted packets. 0b - C-VLAN type (0x8100) is inserted 1b - S-VLAN type (0x88A8) is inserted
18-16 —	Reserved
15-0 VLT	VLAN Tag for Transmit Packets This field contains the value of the VLAN tag to be inserted. The value must only be changed when the transmit lines are inactive or during the initialization phase. Bits[15:13] are the User Priority field, Bit 12 is the CFI/DEI field, and Bits[11:0] are the VID field in the VLAN tag. The following list describes the bits of this field: - Bits[15:13]: User Priority - Bit 12: Canonical Format Indicator (CFI) or Drop Eligible Indicator (DEI) - Bits[11:0]: VLAN Identifier (VID) field of VLAN tag

76.17.57 MAC VLAN Inclusion 7 (MAC_VLAN_Incl7)

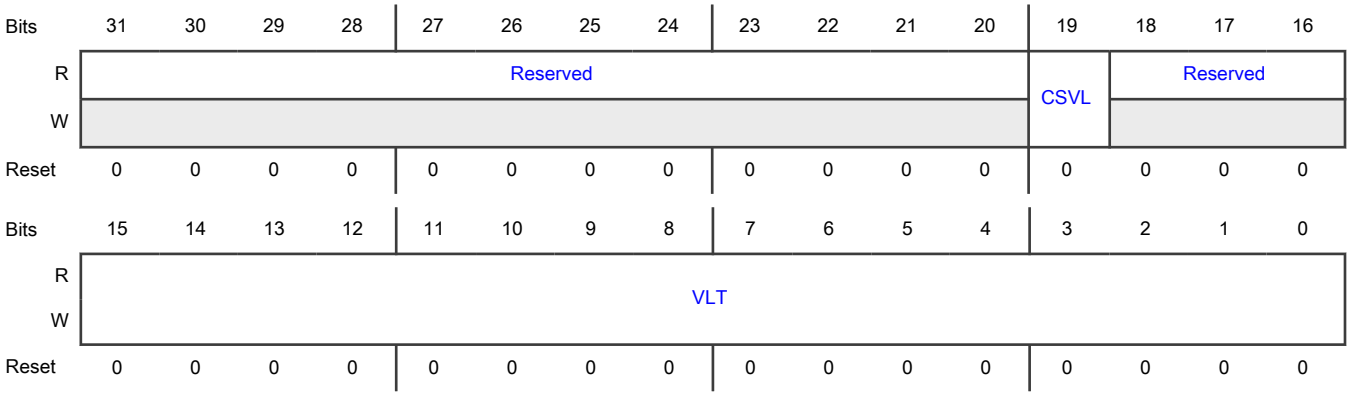
Offset

Register	Offset
MAC_VLAN_Incl7	60h

Function

Contains the VLAN tag for insertion in the Transmit packets from Tx Queue 7. It also contains the VLAN tag insertion controls.

Diagram



Fields

Field	Function
31-20 —	Reserved
19 CSVL	C-VLAN or S-VLAN When this bit is set, S-VLAN type (0x88A8) is inserted in the 13th and 14th bytes of transmitted packets. When this bit is reset, C-VLAN type (0x8100) is inserted in the 13th and 14th bytes of transmitted packets. 0b - C-VLAN type (0x8100) is inserted 1b - S-VLAN type (0x88A8) is inserted
18-16 —	Reserved
15-0 VLT	VLAN Tag for Transmit Packets This field contains the value of the VLAN tag to be inserted. The value must only be changed when the transmit lines are inactive or during the initialization phase. Bits[15:13] are the User Priority field, Bit 12 is the CFI/DEI field, and Bits[11:0] are the VID field in the VLAN tag. The following list describes the bits of this field: - Bits[15:13]: User Priority - Bit 12: Canonical Format Indicator (CFI) or Drop Eligible Indicator (DEI) - Bits[11:0]: VLAN Identifier (VID) field of VLAN tag

76.17.58 Inner VLAN Tag Inclusion Or Replacement (MAC_Inner_VLAN_Incl)

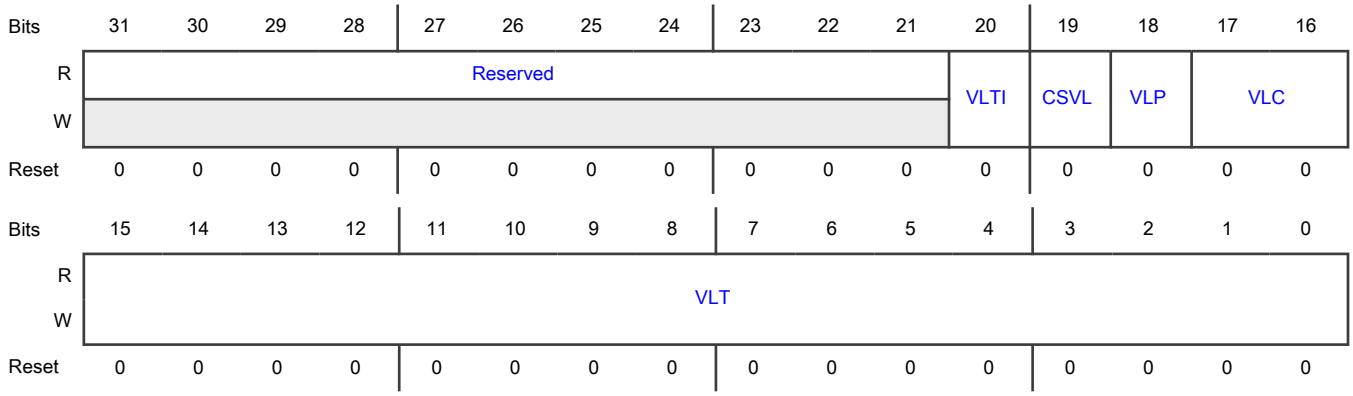
Offset

Register	Offset
MAC_Inner_VLAN_Incl	64h

Function

The Inner VLAN Tag Inclusion or Replacement register contains the inner VLAN tag to be inserted or replaced in the Transmit packet. It also contains the inner VLAN tag insertion controls.

Diagram



Fields

Field	Function
31-21 —	Reserved
20 VLTI	VLAN Tag Input When this bit is set, it indicates that the VLAN tag to be inserted or replaced in Tx packet should be taken from: - The Tx descriptor 0b - VLAN Tag Input is disabled 1b - VLAN Tag Input is enabled
19 CSVL	C-VLAN or S-VLAN When this bit is set, S-VLAN type (0x88A8) is inserted or replaced in the 17th and 18th bytes of transmitted packets. When this bit is reset, C-VLAN type (0x8100) is inserted or replaced in the 17th and 18th bytes of transmitted packets. 0b - C-VLAN type (0x8100) is inserted 1b - S-VLAN type (0x88A8) is inserted
18 VLP	VLAN Priority Control When this bit is set, the VLC field is used for VLAN deletion, insertion, or replacement. When this bit is reset, the mti_vlan_ctrl_i control input is used and the VLC field is ignored. 0b - VLAN Priority Control is disabled 1b - VLAN Priority Control is enabled
17-16 VLC	VLAN Tag Control in Transmit Packets - 2'b00: No VLAN tag deletion, insertion, or replacement - 2'b01: VLAN tag deletion The MAC removes the VLAN type (bytes 17 and 18) and VLAN tag (bytes 19 and 20) of all transmitted packets with VLAN tags. - 2'b10: VLAN tag insertion The MAC inserts VLT in bytes 19 and 20 of the packet after inserting the Type value (0x8100 or 0x88a8) in bytes 17 and 18. This operation is performed on all transmitted packets, irrespective of whether they already have a VLAN tag. - 2'b11: VLAN tag replacement The MAC replaces VLT in bytes 19 and 20 of all VLAN-type transmitted packets (Bytes 17 and 18 are 0x8100 or 0x88a8). Note: Changes to this field take effect only on the start of a packet. If you write this register field when a packet is being transmitted, only the subsequent packet can use the updated value, that is, the current packet does not use the updated value. 00b - No VLAN tag deletion, insertion, or replacement 01b - VLAN tag deletion 10b - VLAN tag insertion 11b - VLAN tag replacement
15-0 VLT	VLAN Tag for Transmit Packets This field contains the value of the VLAN tag to be inserted or replaced. The value must only be changed when the transmit lines are inactive or during the initialization phase. Bits[15:13] are the User Priority field, Bit 12 is the CFI/DEI field, and Bits[11:0] are the VID field in the VLAN tag. The following list describes the bits of this field: - Bits[15:13]: User Priority - Bit 12: Canonical Format Indicator (CFI) or Drop Eligible Indicator (DEI) - Bits[11:0]: VLAN Identifier (VID) field of VLAN tag

76.17.59 MAC Q0 Tx Flow Control (MAC_Q0_Tx_Flow_Ctrl)

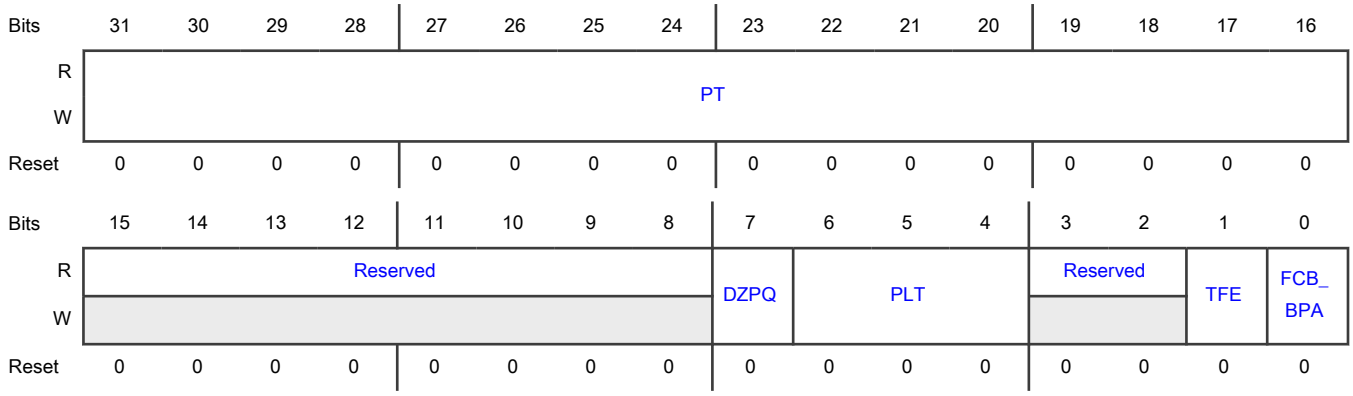
Offset

Register	Offset
MAC_Q0_Tx_Flow_Ctrl	70h

Function

The Flow Control register controls the generation and reception of the Control (Pause Command) packets by the Flow control module of the MAC. A Write to a register with the Busy bit set to 1 triggers the Flow Control block to generate a Pause packet. The fields of the control packet are selected as specified in the 802.3x specification, and the Pause Time value from this register is used in the Pause Time field of the control packet. The Busy bit remains set until the control packet is transferred onto the cable. The application must make sure that the Busy bit is cleared before writing to the register. When the PFCE bit in the MAC_Rx_Flow_Ctrl register is enabled, this register controls the generation of Priority Flow Control (PFC) frames with priorities mapped according to PSRQ0 in the MAC_RxQ_Ctrl2 register.

Diagram



Fields

Field	Function
31-16 PT	Pause Time This field holds the value to be used in the Pause Time field in the Tx control packet. If the Pause Time bits are configured to be double-synchronized to the (G)MII clock domain, consecutive writes to this register should be performed only after at least four clock cycles in the destination clock domain.
15-8 —	Reserved
7 DZPQ	Disable Zero-Quanta Pause When this bit is set, it disables the automatic generation of the zero-quanta Pause packets on de-assertion of the flow-control signal from the FIFO layer (MTL or external sideband flow control signal sbd_flowctrl_i or mti_flowctrl_i). When this bit is reset, normal operation with automatic zero-quanta Pause packet generation is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Zero-Quanta Pause packet generation is enabled 1b - Zero-Quanta Pause packet generation is disabled
6-4 PLT	Pause Low Threshold This field configures the threshold of the Pause timer at which the input flow control signal mti_flowctrl_i (or sbd_flowctrl_i) is checked for automatic retransmission of the Pause packet. The threshold values should be always less than the Pause Time configured in Bits[31:16]. For example, if PT = 100H (256 slot times), and PLT = 001, a second Pause packet is automatically transmitted if the mti_flowctrl_i signal is asserted at 228 (256-28) slot times after the first Pause packet is transmitted. The following list provides the threshold values for different values. The slot time is defined as the time taken to transmit 512 bits (64 bytes) on the GMII or MII interface. This (approximate) computation is based on the packet size (64, 1518, 2000, 9018, 16384, or 32768) + 2 Pause Packet Size + IPG in Slot Times. 000b - Pause Time minus 4 Slot Times (PT -4 slot times) 001b - Pause Time minus 28 Slot Times (PT -28 slot times) 010b - Pause Time minus 36 Slot Times (PT -36 slot times) 011b - Pause Time minus 144 Slot Times (PT -144 slot times) 100b - Pause Time minus 256 Slot Times (PT -256 slot times) 101b - Pause Time minus 512 Slot Times (PT -512 slot times) 110b - Reserved
3-2 —	Reserved
1 TFE	Transmit Flow Control Enable Full-Duplex Mode: In the full-duplex mode, when this bit is set, the MAC enables the flow control operation to Tx Pause packets. When this bit is reset, the flow control operation in the MAC is disabled, and the MAC does not transmit any Pause packets. Half-Duplex Mode: In the half-duplex mode, when this bit is set, the MAC enables the backpressure operation. When this bit is reset, the backpressure feature is disabled. 0b - Transmit Flow Control is disabled 1b - Transmit Flow Control is enabled
0 FCB_BPA	Flow Control Busy or Backpressure Activate This bit initiates a Pause packet in the full-duplex mode and activates the backpressure function in the half-duplex mode if the TFE bit is set. Full-Duplex Mode: In the full-duplex mode, this bit should be read as 1'b0 before writing to this register. To initiate a Pause packet, the application must set this bit to 1'b1. During Control packet transfer, this bit continues to be set to indicate that a packet transmission is in progress. When Pause packet transmission is complete, the MAC resets this bit to 1'b0. You should not write to this register until this bit is cleared. Half-Duplex Mode: When this bit is set (and TFE bit is set) in the half-duplex mode, the MAC asserts the backpressure. During backpressure, when the MAC receives a new packet, the transmitter starts sending a JAM pattern resulting in a collision. This control register bit is logically ORed with the mti_flowctrl_i input signal for the backpressure function. When the MAC is

Table continues on the next page...

Table continued from the previous page...

Field	Function
	configured for the full-duplex mode, the BPA is automatically disabled. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. **NOTE** This field is not testable by an automated register test. 0b - Flow Control Busy or Backpressure Activate is disabled 1b - Flow Control Busy or Backpressure Activate is enabled

76.17.60 MAC Q1 Tx Flow Control (MAC_Q1_Tx_Flow_Ctrl)

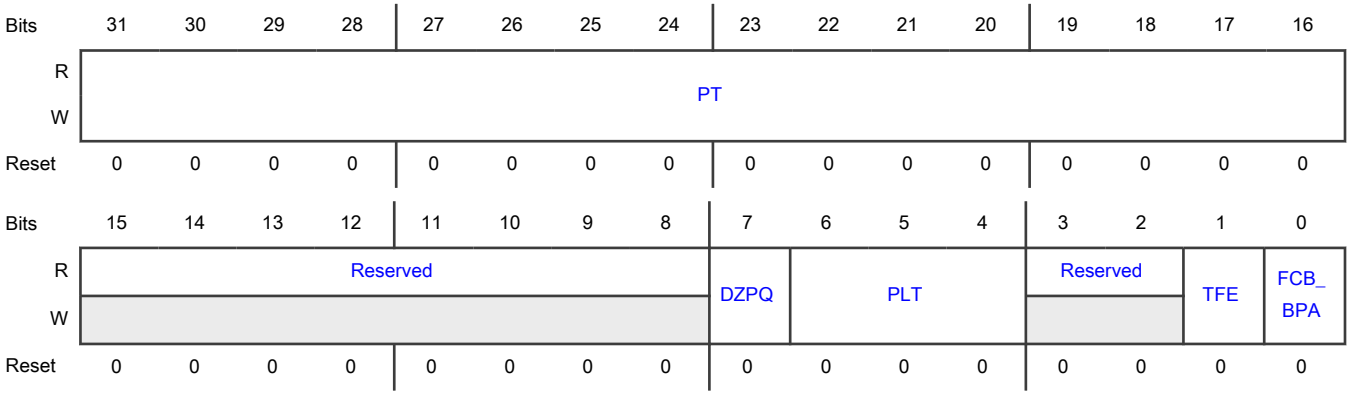
Offset

Register	Offset
MAC_Q1_Tx_Flow_Ctrl	74h

Function

This register controls the generation of PFC Control packets of priorities mapped as per the PSRQi field in the MAC_RxQ_Ctrl2/MAC_RxQ_Ctrl3 registers.

Diagram



Fields

Field	Function
31-16 PT	Pause Time This field holds the value to be used in the Pause Time field in the Tx control packet. If the Pause Time bits are configured to be double-synchronized to the (G)MII clock domain, consecutive writes to this register should be performed only after at least four clock cycles in the destination clock domain.
15-8	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
7 DZPQ	Disable Zero-Quanta Pause When this bit is set, it disables the automatic generation of the zero-quanta Pause packets on de-assertion of the flow-control signal from the FIFO layer (MTL or external sideband flow control signal sbd_flowctrl_i or mti_flowctrl_i). When this bit is reset, normal operation with automatic zero-quanta Pause packet generation is enabled. 0b - Zero-Quanta Pause packet generation is enabled 1b - Zero-Quanta Pause packet generation is disabled
6-4 PLT	Pause Low Threshold This field configures the threshold of the Pause timer at which the input flow control signal mti_flowctrl_i (or sbd_flowctrl_i) is checked for automatic retransmission of the Pause packet. The threshold values should be always less than the Pause Time configured in Bits[31:16]. For example, if PT = 100H (256 slot times), and PLT = 001, a second Pause packet is automatically transmitted if the mti_flowctrl_i signal is asserted at 228 (256-28) slot times after the first Pause packet is transmitted. The following list provides the threshold values for different values. The slot time is defined as the time taken to transmit 512 bits (64 bytes) on the GMII or MII interface. This (approximate) computation is based on the packet size (64, 1518, 2000, 9018, 16384, or 32768) + 2 Pause Packet Size + IPG in Slot Times. 000b - Pause Time minus 4 Slot Times (PT -4 slot times) 001b - Pause Time minus 28 Slot Times (PT -28 slot times) 010b - Pause Time minus 36 Slot Times (PT -36 slot times) 011b - Pause Time minus 144 Slot Times (PT -144 slot times) 100b - Pause Time minus 256 Slot Times (PT -256 slot times) 101b - Pause Time minus 512 Slot Times (PT -512 slot times) 110b - Reserved
3-2 —	Reserved
1 TFE	Transmit Flow Control Enable When this bit is set in full-duplex mode, the MAC enables the flow control operation to Tx pause packets. When this bit is reset, the flow control operation in the MAC is disabled, and the MAC does not transmit any pause packets. 0b - Transmit Flow Control is disabled 1b - Transmit Flow Control is enabled
0 FCB_BPA	Flow Control Busy This bit initiates a PFC packet if the TFE bit is set. To initiate a PFC packet, the application must set this bit to 1'b1. During Control packet transfer, this bit continues to be set to indicate that a packet transmission is in progress. When PFC packet transmission is complete, the MAC resets this bit to 1'b0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	You should not write to this register until this bit is cleared. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. **NOTE** This field is not testable by an automated register test. 0b - Flow Control Busy or Backpressure Activate is disabled 1b - Flow Control Busy or Backpressure Activate is enabled

76.17.61 MAC Q2 Tx Flow Control (MAC_Q2_Tx_Flow_Ctrl)

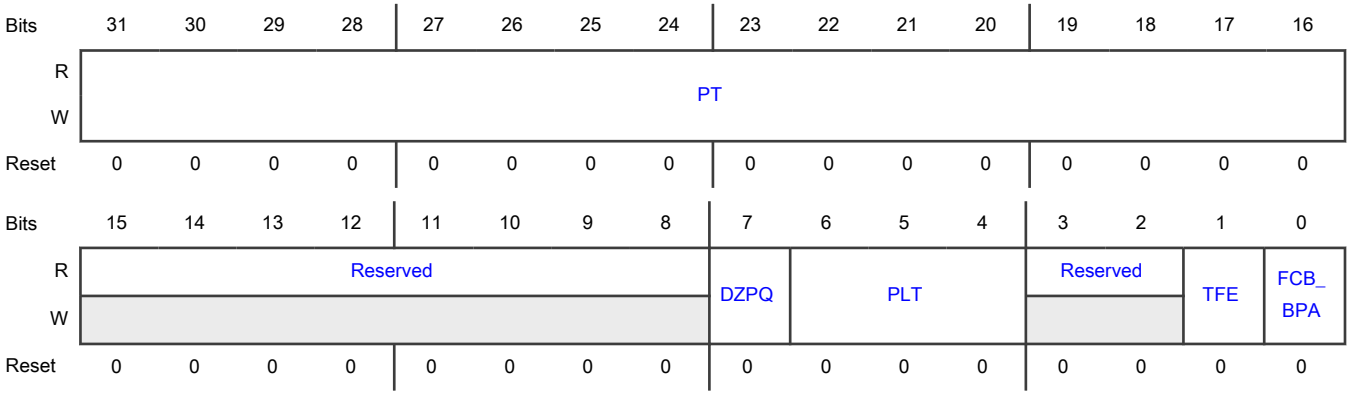
Offset

Register	Offset
MAC_Q2_Tx_Flow_Ctrl	78h

Function

This register controls the generation of PFC Control packets of priorities mapped as per the PSRQi field in the MAC_RxQ_Ctrl2/MAC_RxQ_Ctrl3 registers.

Diagram



Fields

Field	Function
31-16 PT	Pause Time This field holds the value to be used in the Pause Time field in the Tx control packet. If the Pause Time bits are configured to be double-synchronized to the (G)MII clock domain, consecutive writes to this register should be performed only after at least four clock cycles in the destination clock domain.
15-8	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
7 DZPQ	Disable Zero-Quanta Pause When this bit is set, it disables the automatic generation of the zero-quanta Pause packets on de-assertion of the flow-control signal from the FIFO layer (MTL or external sideband flow control signal sbd_flowctrl_i or mti_flowctrl_i). When this bit is reset, normal operation with automatic zero-quanta Pause packet generation is enabled. 0b - Zero-Quanta Pause packet generation is enabled 1b - Zero-Quanta Pause packet generation is disabled
6-4 PLT	Pause Low Threshold This field configures the threshold of the Pause timer at which the input flow control signal mti_flowctrl_i (or sbd_flowctrl_i) is checked for automatic retransmission of the Pause packet. The threshold values should be always less than the Pause Time configured in Bits[31:16]. For example, if PT = 100H (256 slot times), and PLT = 001, a second Pause packet is automatically transmitted if the mti_flowctrl_i signal is asserted at 228 (256-28) slot times after the first Pause packet is transmitted. The following list provides the threshold values for different values. The slot time is defined as the time taken to transmit 512 bits (64 bytes) on the GMII or MII interface. This (approximate) computation is based on the packet size (64, 1518, 2000, 9018, 16384, or 32768) + 2 Pause Packet Size + IPG in Slot Times. 000b - Pause Time minus 4 Slot Times (PT -4 slot times) 001b - Pause Time minus 28 Slot Times (PT -28 slot times) 010b - Pause Time minus 36 Slot Times (PT -36 slot times) 011b - Pause Time minus 144 Slot Times (PT -144 slot times) 100b - Pause Time minus 256 Slot Times (PT -256 slot times) 101b - Pause Time minus 512 Slot Times (PT -512 slot times) 110b - Reserved
3-2 —	Reserved
1 TFE	Transmit Flow Control Enable When this bit is set in full-duplex mode, the MAC enables the flow control operation to Tx pause packets. When this bit is reset, the flow control operation in the MAC is disabled, and the MAC does not transmit any pause packets. 0b - Transmit Flow Control is disabled 1b - Transmit Flow Control is enabled
0 FCB_BPA	Flow Control Busy This bit initiates a PFC packet if the TFE bit is set. To initiate a PFC packet, the application must set this bit to 1'b1. During Control packet transfer, this bit continues to be set to indicate that a packet transmission is in progress. When PFC packet transmission is complete, the MAC resets this bit to 1'b0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	You should not write to this register until this bit is cleared. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. **NOTE** This field is not testable by an automated register test. 0b - Flow Control Busy or Backpressure Activate is disabled 1b - Flow Control Busy or Backpressure Activate is enabled

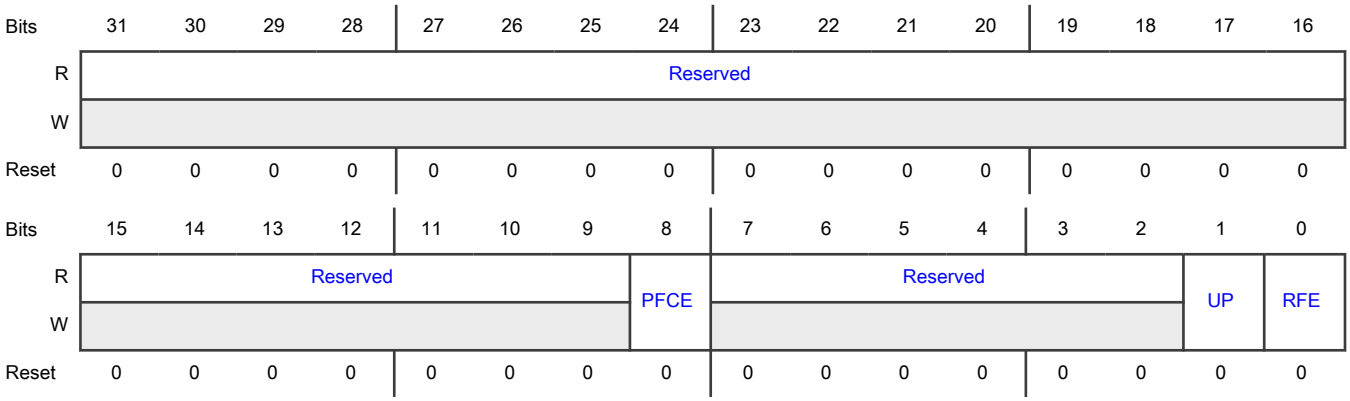
76.17.62 MAC Receive Flow Control (MAC_Rx_Flow_Ctrl)

Offset

Register	Offset
MAC_Rx_Flow_Ctrl	90h

Function
The Receive Flow Control register controls the pausing of MAC Transmit based on the received Pause packet.

Diagram



Fields

Field	Function
31-9 —	Reserved
8 PFCE	Priority Based Flow Control Enable When this bit is set, it enables generation and reception of priority-based flow control (PFC) packets. When this bit is reset, it enables generation and reception of 802.3x Pause control packets.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Priority Based Flow Control is disabled 1b - Priority Based Flow Control is enabled
7-2 —	Reserved
1 UP	Unicast Pause Packet Detect A pause packet is processed when it has the unique multicast address specified in the IEEE 802.3. When this bit is set, the MAC can also detect Pause packets with unicast address of the station. This unicast address should be as specified in MAC_Address0_High and MAC_Address0_Low. When this bit is reset, the MAC only detects Pause packets with unique multicast address. Note: The MAC does not process a Pause packet if the multicast address is different from the unique multicast address. This is also applicable to the received PFC packet when the Priority Flow Control (PFC) is enabled. The unique multicast address (0x01_80_C2_00_00_01) is as specified in IEEE 802.1 Qbb-2011. 0b - Unicast Pause Packet Detect disabled 1b - Unicast Pause Packet Detect enabled
0 RFE	Receive Flow Control Enable When this bit is set and the MAC is operating in full-duplex mode, the MAC decodes the received Pause packet and disables its transmitter for a specified (Pause) time. When this bit is reset or the MAC is operating in half-duplex mode, the decode function of the Pause packet is disabled. When PFC is enabled, flow control is enabled for PFC packets. The MAC decodes the received PFC packet and disables the Transmit queue, with matching priorities, for a duration of received Pause time. 0b - Receive Flow Control is disabled 1b - Receive Flow Control is enabled

76.17.63 MAC Rx Queue Control 4 (MAC_RxQ_Ctrl4)

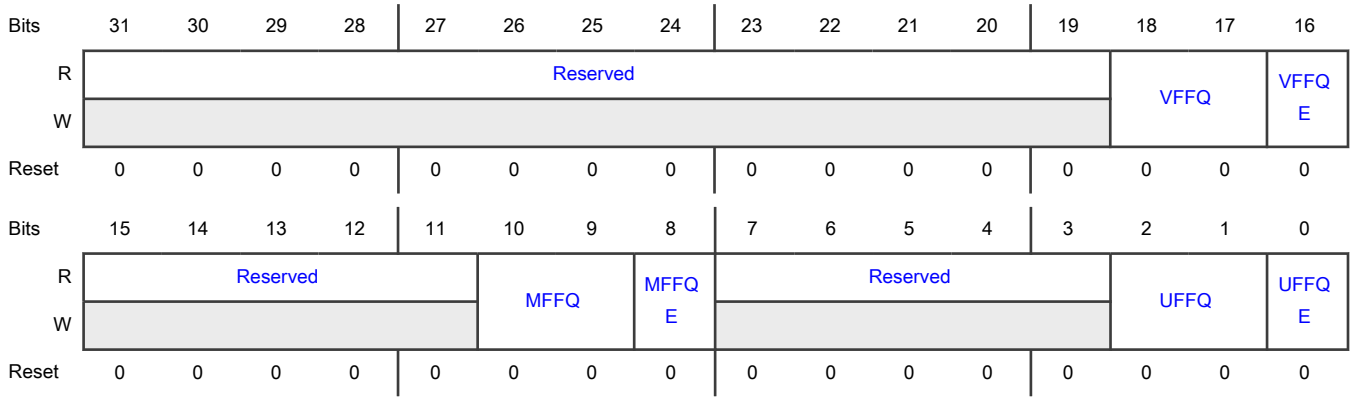
Offset

Register	Offset
MAC_RxQ_Ctrl4	94h

Function

The Receive Queue Control 4 register controls the routing of unicast and multicast packets that fail the Destination or Source address filter to the Rx queues.

Diagram



Fields

Field	Function
31-19 —	Reserved
18-17 VFFQ	VLAN Tag Filter Fail Packets Queue This field holds the Rx queue number to which the tagged packets failing the Destination or Source Address filter (and UFFQE/MFFQE not enabled) or failing the VLAN tag filter must be routed to. This field is valid only when the VFFQE bit is set.
16 VFFQE	VLAN Tag Filter Fail Packets Queuing Enable When this bit is set, the tagged packets which fail the Destination or Source address filter or fail the VLAN tag filter, are routed to the Rx Queue Number programmed in the VFFQ. When this bit is reset, the tagged packets which fail the Destination or Source address filter or fail the VLAN tag filter are routed based on other routing options. This bit is valid only when the RA bit of the MAC_Packet_Filter register is set. 0b - VLAN tag Filter Fail Packets Queuing is disabled 1b - VLAN tag Filter Fail Packets Queuing is enabled
15-11 —	Reserved
10-9 MFFQ	Multicast Address Filter Fail Packets Queue. This field holds the Rx queue number to which the Multicast packets failing the Destination or Source Address filter are routed to. This field is valid only when the MFFQE bit is set.
8 MFFQE	Multicast Address Filter Fail Packets Queuing Enable. When this bit is set, the Multicast packets which fail the Destination or Source address filter is routed to the Rx Queue Number programmed in the MFFQ. When this bit is reset, the Multicast packets which fail the Destination or Source address filter is routed based on other routing options. This bit is valid only when the RA bit of the MAC_Packet_Filter register is set.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Multicast Address Filter Fail Packets Queuing is disabled 1b - Multicast Address Filter Fail Packets Queuing is enabled
7-3 —	Reserved
2-1 UFFQ	Unicast Address Filter Fail Packets Queue. This field holds the Rx queue number to which the Unicast packets failing the Destination or Source Address filter are routed to. This field is valid only when the UFFQE bit is set.
0 UFFQE	Unicast Address Filter Fail Packets Queuing Enable. When this bit is set, the Unicast packets which fail the Destination or Source address filter is routed to the Rx Queue Number programmed in the UFFQ. When this bit is reset, the Unicast packets which fail the Destination or Source address filter is routed based on other routing options. This bit is valid only when the RA bit of the MAC_Packet_Filter register is set. 0b - Unicast Address Filter Fail Packets Queuing is disabled 1b - Unicast Address Filter Fail Packets Queuing is enabled

76.17.64 MAC Tx Queue Parity Map 0 (MAC_TxQ_Prty_Map0)

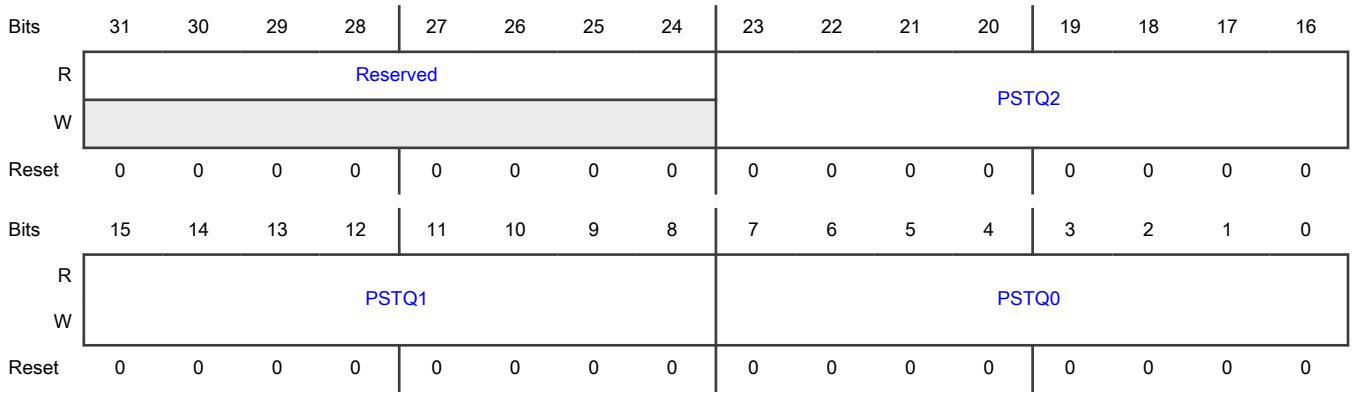
Offset

Register	Offset
MAC_TxQ_Prty_Map0	98h

Function

The Transmit Queue Priority Mapping 0 register contains the priority values assigned to Tx Queue 0 through Tx Queue 3.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-16 PSTQ2	Priorities Selected in Transmit Queue 2 This bit is similar to the PSTQ0 bit.
15-8 PSTQ1	Priorities Selected in Transmit Queue 1 This bit is similar to the PSTQ0 bit.
7-0 PSTQ0	Priorities Selected in Transmit Queue 0 This field holds the priorities assigned to Tx Queue 0 by the software. This field determines if Tx Queue 0 should be blocked from transmitting specified pause time when a PFC packet is received with priorities matching the priorities programmed in this field. If the content of this field is not mutually exclusive to the corresponding fields of other Transmit queues, that is, same priority is mapped to multiple Tx queues, the MAC blocks all queues with matching priority, for the specified time.

76.17.65 MAC Rx Queue Control 0 (MAC_RxQ_Ctrl0)

Offset

Register	Offset
MAC_RxQ_Ctrl0	A0h

Function

The Receive Queue Control 0 register controls the queue management in the MAC Receiver. Note: In multiple Rx queues configuration, all the queues are disabled by default. Enable the Rx queue by programming the corresponding field in this register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved		Reserved		Reserved		Reserved		Reserved		RXQ2EN		RXQ1EN		RXQ0EN	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-14 —	Reserved
13-12 —	Reserved
11-10 —	Reserved
9-8 —	Reserved
7-6 —	Reserved
5-4 RXQ2EN	Receive Queue 2 Enable This field is similar to the RXQ0EN field. 00b - Queue not enabled 01b - Queue enabled for AV 10b - Queue enabled for DCB/Generic 11b - Reserved
3-2 RXQ1EN	Receive Queue 1 Enable This field is similar to the RXQ0EN field. 00b - Queue not enabled 01b - Queue enabled for AV 10b - Queue enabled for DCB/Generic 11b - Reserved
1-0 RXQ0EN	Receive Queue 0 Enable This field indicates whether Rx Queue 0 is enabled for AV or DCB. 00b - Queue not enabled 01b - Queue enabled for AV 10b - Queue enabled for DCB/Generic 11b - Reserved

76.17.66 MAC Rx Queue Control 1 (MAC_RxQ_Ctrl1)

Offset

Register	Offset
MAC_RxQ_Ctrl1	A4h

Function

The Receive Queue Control 1 register controls the routing of multicast, broadcast, AV, DCB, and untagged packets to the Rx queues.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved		TBRQE	OMCBCQ	Reserved	FPRQ			TPQC		TACPQE	MCBCQEN	Reserved	MCBCQ		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	UPQ			Reserved	DCBCPQ			Reserved	PTPQ			Reserved	AVCPQ		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29 TBRQE	Type Field Based Rx Queuing Enable When this bit is set, it enables Type Field based Rx Queuing where the Type field of received packet is compared with programmed TYP field in MAC_TMRQ_Regs(#i) and if a match occurs the packet is routed to the corresponding TMRQ field.
28 OMCBCQ	Over-riding MC-BC queue priority select - 1: Priority of MCBCQ is reduced and the received packet is first routed to PTPQ, AVCPQ, DCBCPQ depending on packet type. - 0: Received Multicast/Broadcast packet is routed to MCBCQ. 0b - overriding MCBCQ priority disabled 1b - overriding MCBCQ priority enabled
27 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
26-24 FPRQ	Frame Preemption Residue Queue This field holds the Rx queue number to which the residual preemption frames must be forwarded. Preemption frames that are tagged and pass the SA/DA/VLAN filtering are routed based on PSRQ and all other frames are treated as residual frames and is routed to the queue number mentioned in this field. The Queue-0 is used as a default queue for express frames, so this field cannot be programmed to a value 0.
23-22 TPQC	Tagged PTP over Ethernet Packets Queuing Control. This field controls the routing of the VLAN Tagged PTPoE packets. If DWC_EQOS_AV_ENABLE is selected in the configuration, the following programmable options are allowed. - 2'b00: VLAN Tagged PTPoE packets are routed as generic VLAN Tagged packet (based on PSRQ for only non-AV enabled Rx Queues). - 2'b01: VLAN Tagged PTPoE packets are routed to Rx Queue specified by PTPQ field (That Rx Queue can be enabled for AV or non-AV traffic). - 2'b10: VLAN Tagged PTPoE packets are routed to only AV enabled Rx Queues based on PSRQ. - 2'b11: Reserved If DWC_EQOS_AV_ENABLE is not selected in the configuration, the following programmable options are allowed. - 1'b0: VLAN Tagged PTPoE packets are routed as generic VLAN Tagged packet (based on PSRQ for DCB/Generic enabled Rx Queues). - 1'b1: VLAN Tagged PTPoE packets are routed to Rx Queues specified by PTPQ field.
21 TACPQE	Tagged AV Control Packets Queuing Enable. When set, the MAC routes the received Tagged AV Control packets to the Rx queue specified by AVCPQ field. When reset, the MAC routes the received Tagged AV Control packets based on the tag priority matching the PSRQ fields in MAC_RxQ_Ctrl2 and MAC_RxQ_Ctrl3 registers. 0b - Tagged AV Control Packets Queuing is disabled 1b - Tagged AV Control Packets Queuing is enabled
20 MCBCQEN	Multicast and Broadcast Queue Enable This bit specifies that Multicast or Broadcast packets routing to the Rx Queue is enabled and the Multicast or Broadcast packets must be routed to Rx Queue specified in MCBCQ field. 0b - Multicast and Broadcast Queue is disabled 1b - Multicast and Broadcast Queue is enabled
19 —	Reserved
18-16 MCBCQ	Multicast and Broadcast Queue This field specifies the Rx Queue onto which Multicast or Broadcast Packets are routed. Any Rx Queue enabled for Generic/DCB/AV traffic can be used to route the Multicast or Broadcast Packets. 000b - Receive Queue 0 001b - Receive Queue 1 010b - Receive Queue 2 011b - Receive Queue 3

Table continues on the next page...

Table continued from the previous page...

Field	Function
	100b - Receive Queue 4 101b - Receive Queue 5 110b - Receive Queue 6 111b - Receive Queue 7
15 —	Reserved
14-12 UPQ	Untagged Packet Queue This field indicates the Rx Queue to which Untagged Packets are to be routed. Any Rx Queue enabled for Generic/DCB/AV traffic can be used to route the Untagged Packets. 000b - Receive Queue 0 001b - Receive Queue 1 010b - Receive Queue 2 011b - Receive Queue 3 100b - Receive Queue 4 101b - Receive Queue 5 110b - Receive Queue 6 111b - Receive Queue 7
11 —	Reserved
10-8 DCBCPQ	DCB Control Packets Queue This field specifies the Rx queue on which the received DCB control packets are routed. The DCB data packets are routed based on the PSRQ field of the Transmit Flow Control Register of corresponding queue. 000b - Receive Queue 0 001b - Receive Queue 1 010b - Receive Queue 2 011b - Receive Queue 3 100b - Receive Queue 4 101b - Receive Queue 5 110b - Receive Queue 6 111b - Receive Queue 7
7 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
6-4 PTPQ	PTP Packets Queue This field specifies the Rx queue on which the PTP packets sent over the Ethernet payload (not over IPv4 or IPv6) are routed. When the AV8021ASMEN bit of MAC_Timestamp_Control register is set, only untagged PTP over Ethernet packets are routed on an Rx Queue. If the bit is not set, then based on programming of TPQC field, both tagged and untagged PTPoE packets can be routed to this Rx Queue. 000b - Receive Queue 0 001b - Receive Queue 1 010b - Receive Queue 2 011b - Receive Queue 3 100b - Receive Queue 4 101b - Receive Queue 5 110b - Receive Queue 6 111b - Receive Queue 7
3 —	Reserved
2-0 AVCPQ	AV Untagged Control Packets Queue This field specifies the Receive queue on which the received AV tagged and untagged control packets are routed. The AV tagged (when TACPQE bit is set) and untagged control packets are routed to Receive queue specified by this field. 000b - Receive Queue 0 001b - Receive Queue 1 010b - Receive Queue 2 011b - Receive Queue 3 100b - Receive Queue 4 101b - Receive Queue 5 110b - Receive Queue 6 111b - Receive Queue 7

76.17.67 MAC Rx Queue Control 2 (MAC_RxQ_Ctrl2)

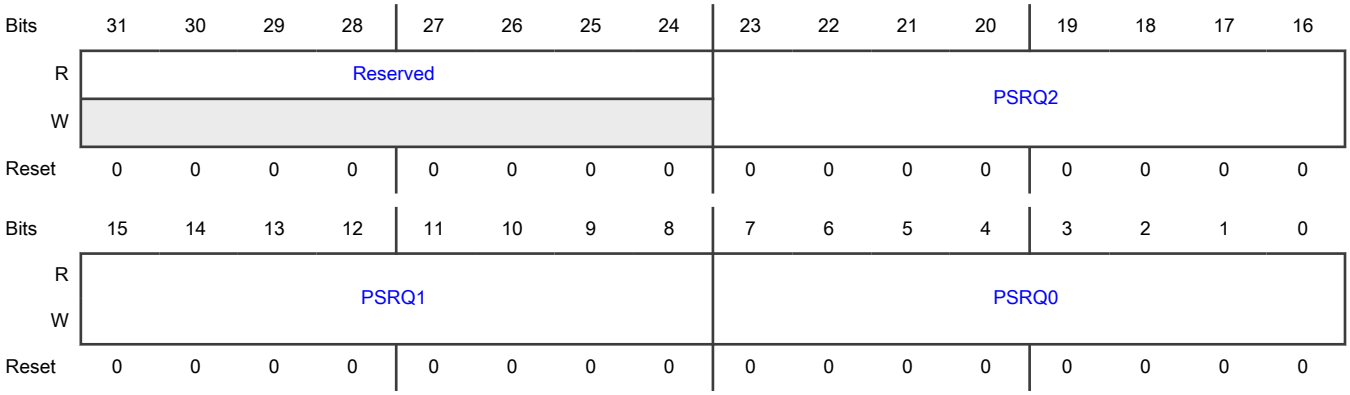
Offset

Register	Offset
MAC_RxQ_Ctrl2	A8h

Function

This register controls the routing of tagged packets based on the USP (user Priority) field of the received packets to the RxQueues 0 to 3.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-16 PSRQ2	Priorities Selected in the Receive Queue 2 This field decides the priorities assigned to Rx Queue 2. All packets with priorities that match the values set in this field are routed to Rx Queue 2. For example, if PSRQ2[1, 0] are set, packets with USP field equal to 1 or 0 are routed to Rx Queue 2. The software must ensure that the content of this field is mutually exclusive to the PSRQ fields for other queues, that is, the same priority is not mapped to multiple Rx queues. this field also determines the priorities to be included in the PFC packet sent to remote station when Rx Queue 2 crosses the flow control threshold settings.
15-8 PSRQ1	Priorities Selected in the Receive Queue 1 This field decides the priorities assigned to Rx Queue 1. All packets with priorities that match the values set in this field are routed to Rx Queue 1. For example, if PSRQ1[4] is set, packets with USP field equal to 4 are routed to Rx Queue 1. The software must ensure that the content of this field is mutually exclusive to the PSRQ fields for other queues, that is, the same priority is not mapped to multiple Rx queues. this field also determines the priorities to be included in the PFC packet sent to remote station when Rx Queue 1 crosses the flow control threshold settings.
7-0 PSRQ0	Priorities Selected in the Receive Queue 0 This field decides the priorities assigned to Rx Queue 0. All packets with priorities that match the values set in this field are routed to Rx Queue 0. For example, if PSRQ0[5] is set, packets with USP field equal to 5 are routed to Rx Queue 0. The software must ensure that the content of this field is mutually exclusive to the PSRQ fields for other queues, that is, the same priority is not mapped to multiple Rx queues. this field also determines the priorities to be included in the PFC packet sent to remote station when Rx Queue 0 crosses the flow control threshold settings.

76.17.68 MAC Interrupt Status (MAC_Interrupt_Status)

Offset

Register	Offset
MAC_Interrupt_Status	B0h

Function

The Interrupt Status register contains the status of interrupts.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved											MFRIS	MFTIS	MDIOIS	FPEIS	Reserved
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	RXSTSIS	TXSTSIS	TSIS	Reserved	MMCTXIS	MMCRXIS	MMCIS	Reserved		Reserved	Reserved	PHYIS	Reserved	Reserved	RGSMIIIS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-21 —	Reserved
20 MFRIS	MMC FPE Receive Interrupt Status This bit is set high when an interrupt is generated in the MMC FPE Receive Interrupt Register. This bit is cleared when all bits in this interrupt register are cleared. This bit is valid only when you select the Enable MAC Management Counters (MMC) option along with FPE support. 0b - MMC FPE Receive Interrupt status not active 1b - MMC FPE Receive Interrupt status active
19 MFTIS	MMC FPE Transmit Interrupt Status This bit is set high when an interrupt is generated in the MMC FPE Transmit Interrupt Register. This bit is cleared when all bits in this interrupt register are cleared. This bit is valid only when you select the Enable MAC Management Counters (MMC) option along with FPE support. 0b - MMC FPE Transmit Interrupt status not active 1b - MMC FPE Transmit Interrupt status active

Table continues on the next page...

Table continued from the previous page...

Field	Function
18 MDIOIS	MDIO Interrupt Status This bit indicates an interrupt event after the completion of MDIO operation. To reset this bit, the application has to read this bit/Write 1 to this bit when RCWE bit of MAC_CSR_SW_Ctrl register is set. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0b - MDIO Interrupt status not active 1b - MDIO Interrupt status active
17 FPEIS	Frame Preemption Interrupt Status This bit indicates an interrupt event during the operation of Frame Preemption (Bits[19:16] of MAC_FPE_CTRL_STS register is set). To reset this bit, the application must clear the event in MAC_FPE_CTRL_STS that has caused the Interrupt. 0b - Frame Preemption Interrupt status not active 1b - Frame Preemption Interrupt status active
16 —	Reserved
15 —	Reserved
14 RXSTSIS	Receive Status Interrupt This bit indicates the status of received packets. This bit is set when the RWT bit is set in the MAC_Rx_Tx_Status register. This bit is cleared when the corresponding interrupt source bit is read (or corresponding interrupt source bit is written to 1 when RCWE bit of MAC_CSR_SW_Ctrl register is set) in the MAC_Rx_Tx_Status register. 0b - Receive Interrupt status not active 1b - Receive Interrupt status active
13 TXSTSIS	Transmit Status Interrupt This bit indicates the status of transmitted packets. This bit is set when any of the following bits is set in the MAC_Rx_Tx_Status register: - Excessive Collision (EXCOL) - Late Collision (LCOL) - Excessive Deferral (EXDEF) - Loss of Carrier (LCARR) - No Carrier (NCARR) - Jabber Timeout (TJT) This bit is cleared when the corresponding interrupt source bit is read (or corresponding interrupt source bit is written to 1 when RCWE bit of MAC_CSR_SW_Ctrl register is set) in the MAC_Rx_Tx_Status register. 0b - Transmit Interrupt status not active 1b - Transmit Interrupt status active
12 TSIS	Timestamp Interrupt Status If the Timestamp feature is enabled, this bit is set when any of the following conditions is true: - The system time value is equal to or exceeds the value specified in the Target Time High and Low registers. - There is an overflow in the Seconds register. - The Target Time Error occurred, that

Table continues on the next page...

Table continued from the previous page...

Field	Function
	is, programmed target time already elapsed. In configurations other than EQOS_CORE, when drop transmit status is enabled in MTL, this bit is set when the captured transmit timestamp is updated in the MAC_Tx_Timestamp_Status_Nanoseconds and Mac_TxTimestamp_Status_Seconds registers. This bit is cleared when the corresponding interrupt source bit is read (or corresponding interrupt source bit is written to 1 when RCWE bit of MAC_CSR_SW_Ctrl register is set) in the MAC_Timestamp_Status register. 0b - Timestamp Interrupt status not active 1b - Timestamp Interrupt status active
11 —	Reserved
10 MMCTXIS	MMC Transmit Interrupt Status This bit is set high when an interrupt is generated in the MMC Transmit Interrupt Register. This bit is cleared when all bits in this interrupt register are cleared. This bit is valid only when you select the Enable MAC Management Counters (MMC) option. 0b - MMC Transmit Interrupt status not active 1b - MMC Transmit Interrupt status active
9 MMCRXIS	MMC Receive Interrupt Status This bit is set high when an interrupt is generated in the MMC Receive Interrupt Register. This bit is cleared when all bits in this interrupt register are cleared. This bit is valid only when you select the Enable MAC Management Counters (MMC) option. 0b - MMC Receive Interrupt status not active 1b - MMC Receive Interrupt status active
8 MMCIS	MMC Interrupt Status This bit is set high when Bit 11, Bit 10, or Bit 9 is set high. This bit is cleared only when all these bits are low. This bit is valid only when you select the Enable MAC Management Counters (MMC) option. 0b - MMC Interrupt status not active 1b - MMC Interrupt status active
7-6 —	Reserved
5 —	Reserved
4 —	Reserved
3	PHY Interrupt

Table continues on the next page...

Table continued from the previous page...

Field	Function
PHYIS	This bit is set when rising edge is detected on the phy_intr_i input. This bit is cleared when this register is read (or this bit is written to 1 when RCWE bit of MAC_CSR_SW_Ctrl register is set). 0b - PHY Interrupt not detected 1b - PHY Interrupt detected
2 —	Reserved
1 —	Reserved
0 RGSMIIIS	RGMII or SMII Interrupt Status This bit is set because of any change in value of the Link Status of RGMII or SMII interface (LNKSTS bit in MAC_PHYIF_Control_Status register). This bit is cleared when the MAC_PHYIF_Control_Status register is read (or LNKSTS bit of MAC_PHYIF_Control_Status register is written to 1 when RCWE bit of MAC_CSR_SW_Ctrl register is set). This bit is valid only when you select the optional RGMII or SMII PHY interface. **NOTE** This field is not testable by an automated register test. 0b - RGMII or SMII Interrupt Status is not active 1b - RGMII or SMII Interrupt Status is active

76.17.69 MAC Interrupt Enable (MAC_Interrupt_Enable)

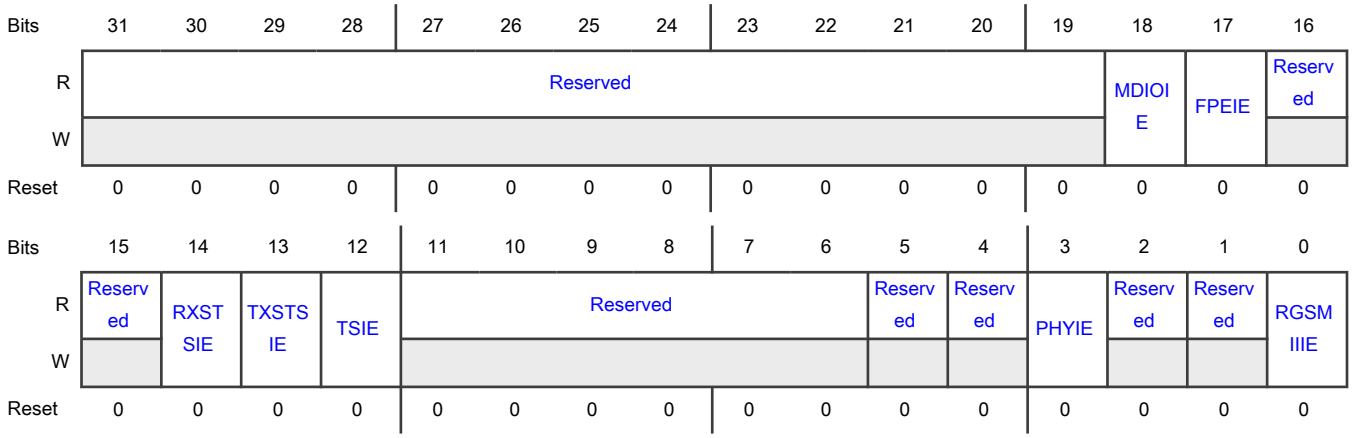
Offset

Register	Offset
MAC_Interrupt_Enable	B4h

Function

The Interrupt Enable register contains the masks for generating the interrupts.

Diagram



Fields

Field	Function
31-19 —	Reserved
18 MDIOIE	MDIO Interrupt Enable When this bit is set, it enables the assertion of the interrupt when MDIOIS field is set in the MAC_Interrupt_Status register. 0b - MDIO Interrupt is disabled 1b - MDIO Interrupt is enabled
17 FPEIE	Frame Preemption Interrupt Enable When this bit is set, it enables the assertion of the interrupt when FPEIS field is set in the MAC_Interrupt_Status register. 0b - Frame Preemption Interrupt is disabled 1b - Frame Preemption Interrupt is enabled
16 —	Reserved
15 —	Reserved
14 RXSTSIE	Receive Status Interrupt Enable When this bit is set, it enables the assertion of the interrupt signal because of the setting of RXSTSIS bit in the MAC_Interrupt_Status register. 0b - Receive Status Interrupt is disabled 1b - Receive Status Interrupt is enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
13 TXSTSIE	Transmit Status Interrupt Enable When this bit is set, it enables the assertion of the interrupt signal because of the setting of TXSTSIS bit in the MAC_Interrupt_Status register. 0b - Timestamp Status Interrupt is disabled 1b - Timestamp Status Interrupt is enabled
12 TSIE	Timestamp Interrupt Enable When this bit is set, it enables the assertion of the interrupt signal because of the setting of TSIS bit in MAC_Interrupt_Status register. 0b - Timestamp Interrupt is disabled 1b - Timestamp Interrupt is enabled
11-6 —	Reserved
5 —	Reserved
4 —	Reserved
3 PHYIE	PHY Interrupt Enable When this bit is set, it enables the assertion of the interrupt signal because of the setting of PHYIS bit in MAC_Interrupt_Status register. 0b - PHY Interrupt is disabled 1b - PHY Interrupt is enabled
2 —	Reserved
1 —	Reserved
0 RGSMIIIE	RGMII or SMII Interrupt Enable When this bit is set, it enables the assertion of the interrupt signal because of the setting of RGSMIIIS bit in MAC_Interrupt_Status register. 0b - RGMII or SMII Interrupt is disabled 1b - RGMII or SMII Interrupt is enabled

76.17.70 MAC Rx Transmit Status (MAC_Rx_Tx_Status)

Offset

Register	Offset
MAC_Rx_Tx_Status	B8h

Function

The Receive Transmit Status register contains the Receive and Transmit Error status.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved							RWT	Reserved		EXCOL	LCOL	EXDEF	LCARR	NCARR	TJT
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-9 —	Reserved
8 RWT	Receive Watchdog Timeout This bit is set when a packet with length greater than 2,048 bytes is received (10, 240 bytes when Jumbo Packet mode is enabled) and the WD bit is reset in the MAC_Configuration register. This bit is set when a packet with length greater than 16,383 bytes is received and the WD bit is set in the MAC_Configuration register. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0b - No receive watchdog timeout 1b - Receive watchdog timed out
7-6 —	Reserved
5 EXCOL	Excessive Collisions When the DTXSTS bit is set in the MTL_Operation_Mode register, this bit indicates that the transmission aborted after 16 successive collisions while attempting to transmit the current packet. If the DR bit is set

Table continues on the next page...

Table continued from the previous page...

Field	Function
	in the MAC_Configuration register, this bit is set after the first collision and the packet transmission is aborted. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0b - No collision 1b - Excessive collision is sensed
4 LCOL	Late Collision When the DTXSTS bit is set in the MTL_Operation_Mode register, this bit indicates that the packet transmission aborted because a collision occurred after the collision window (64 bytes including Preamble in MII mode; 512 bytes including Preamble and Carrier Extension in GMII mode). This bit is not valid if the Underflow error occurs. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0b - No collision 1b - Late collision is sensed
3 EXDEF	Excessive Deferral When the DTXSTS bit is set in the MTL_Operation_Mode register and the DC bit is set in the MAC_Configuration register, this bit indicates that the transmission ended because of excessive deferral of over 24,288 bit times (155,680 in 1000/2500 Mbps mode or when Jumbo packet is enabled). Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0b - No Excessive deferral 1b - Excessive deferral
2 LCARR	Loss of Carrier When the DTXSTS bit is set in the MTL_Operation_Mode register, this bit indicates that the loss of carrier occurred during packet transmission, that is, the PHY_CRS_I signal was inactive for one or more transmission clock periods during packet transmission. This bit is valid only for packets transmitted without collision. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0b - Carrier is present 1b - Loss of carrier
1 NCARR	No Carrier When the DTXSTS bit is set in the MTL_Operation_Mode register, this bit indicates that the carrier signal from the PHY is not present at the end of preamble transmission. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0b - Carrier is present 1b - No carrier
0 TJT	Transmit Jabber Timeout

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This bit indicates that the Transmit Jabber Timer expired which happens when the packet size exceeds 2,048 bytes (10,240 bytes when the Jumbo packet is enabled) and JD bit is reset in the MAC_Configuration register. This bit is set when the packet size exceeds 16,383 bytes and the JD bit is set in the MAC_Configuration register. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0b - No Transmit Jabber Timeout 1b - Transmit Jabber Timeout occurred

76.17.71 MAC Physical Interface Control Status (MAC_PHYIF_Control_Status)

Offset

Register	Offset
MAC_PHYIF_Control_Status	F8h

Function

The PHY Interface Control and Status register indicates the status signals received by the SGMII, RGMII, or SMII interface (selected at reset) from the PHY. This register is optional.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved										Reserved	Reserved	LNKSTS	LNKSPEED		LNKMOD
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved											Reserved	Reserved	Reserved	LUD	TC
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-22 —	Reserved
21	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
20 —	Reserved
19 LNKSTS	Link Status This bit indicates whether the link is up (1'b1) or down (1'b0). **NOTE** This field is not testable by an automated register test. 0b - Link down 1b - Link up
18-17 LNKSPEED	Link Speed This bit indicates the current speed of the link. **NOTE** This field is not testable by an automated register test. 00b - 2.5 MHz 01b - 25 MHz 10b - 125 MHz 11b - Reserved
16 LNKMOD	Link Mode This bit indicates the current mode of operation of the link. **NOTE** This field is not testable by an automated register test. 0b - Half-duplex mode 1b - Full-duplex mode
15-5 —	Reserved
4 —	Reserved
3 —	Reserved
2	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
1 LUD	Link Up or Down This bit indicates whether the link is up or down during transmission of configuration in the RGMII, SGMII, or SMII interface. 0b - Link down 1b - Link up
0 TC	Transmit Configuration in RGMII, SGMII, or SMII When set, this bit enables the transmission of duplex mode, link speed, and link up or down information to the PHY in the RGMII, SMII, or SGMII port. When this bit is reset, no such information is driven to the PHY. The details of this feature are provided in the following sections: - "Reduced Gigabit Media Independent Interface" - "Serial Media Independent Interface" - "Serial Gigabit Media Independent Interface" 0b - Disable Transmit Configuration in RGMII, SGMII, or SMII 1b - Enable Transmit Configuration in RGMII, SGMII, or SMII

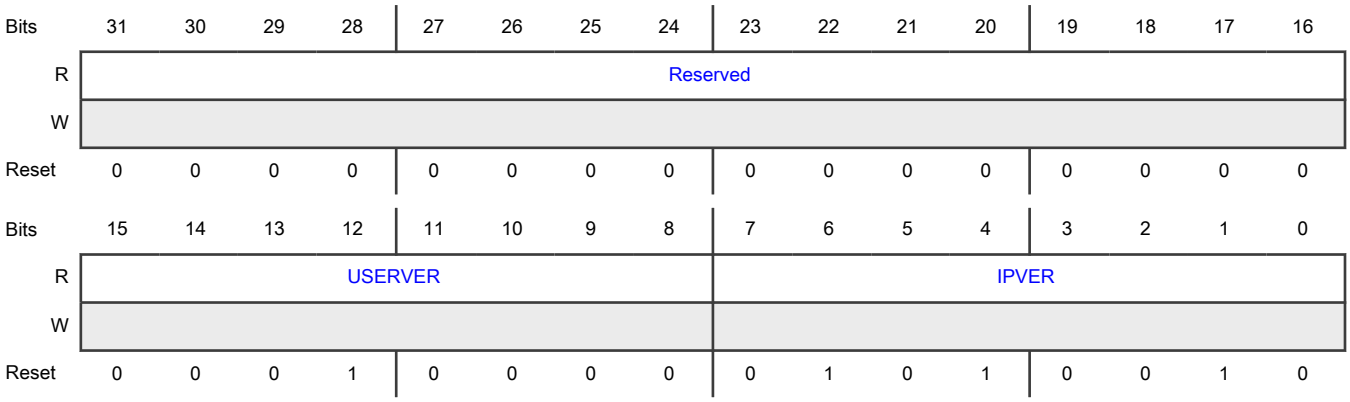
76.17.72 MAC Version (MAC_Version)

Offset

Register	Offset
MAC_Version	110h

Function
Identifies the version of the module.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-8 USERVER	User-defined Version (configured with coreConsultant) User-defined Version (configured with coreConsultant)
7-0 IPVER	IP version

76.17.73 MAC Debug (MAC_Debug)

Offset

Register	Offset
MAC_Debug	114h

Function
The Debug register provides the debug status of various MAC blocks.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved													TFCSTS		TPESTS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved													RFCFCSTS		RPESTS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-19 —	Reserved
18-17	MAC Transmit Packet Controller Status

Table continues on the next page...

Table continued from the previous page...

Field	Function
TFCSTS	This field indicates the state of the MAC Transmit Packet Controller module. 00b - Idle state 01b - Waiting for one of the following: Status of the previous packet OR IPG or back off period to be over 10b - Generating and transmitting a Pause control packet (in full-duplex mode) 11b - Transferring input packet for transmission
16 TPESTS	MAC GMII or MII Transmit Protocol Engine Status When this bit is set, it indicates that the MAC GMII or MII transmit protocol engine is actively transmitting data, and it is not in the Idle state. 0b - MAC GMII or MII Transmit Protocol Engine Status not detected 1b - MAC GMII or MII Transmit Protocol Engine Status detected
15-3 —	Reserved
2-1 RFCFCSTS	MAC Receive Packet Controller FIFO Status When this bit is set, this field indicates the active state of the small FIFO Read and Write controllers of the MAC Receive Packet Controller module.
0 RPESTS	MAC GMII or MII Receive Protocol Engine Status When this bit is set, it indicates that the MAC GMII or MII receive protocol engine is actively receiving data, and it is not in the Idle state. 0b - MAC GMII or MII Receive Protocol Engine Status not detected 1b - MAC GMII or MII Receive Protocol Engine Status detected

76.17.74 MAC Hardware Feature 0 (MAC_HW_Feature0)

Offset

Register	Offset
MAC_HW_Feature0	11Ch

Function

This register indicates the presence of first set of the optional features or functions of the DWC_ether_qos. The software driver can use this register to dynamically enable or disable the programs related to the optional blocks. Note: All bits are set or reset according to the features selected while configuring the IP in coreConsultant.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved	ACTPHYSEL			SAVLANI...	TSSTSSEL		MACADR6...	MACADR3...	ADDMACADRSEL					Reserved	RXCOESEL
W																
Reset	0	0	0	0	1	1	1	0	0	0	0	0	1	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	TXCOESEL	EEESEL	TSSEL	Reserved		ARPOFFS...	MMCSEL	MGKSEL	RWKSEL	SMASEL	VLHASH	PCSSEL	HDSEL	GMIISEL	MIISEL
W																
Reset	0	1	0	1	0	0	1	1	0	0	1	1	0	1	1	1

Fields

Field	Function
31 —	Reserved
30-28 ACTPHYSEL	Active PHY Selected When you have multiple PHY interfaces in your configuration, this field indicates the sampled value of PHY_INTF_SEL_I during reset de-assertion. 000b - GMII or MII 001b - RGMII 010b - SGMII 011b - TBI 100b - RMII 101b - RTBI 110b - SMII 111b - RevMII
27 SAVLANINS	Source Address or VLAN Insertion Enable This bit is set to 1 when the Enable SA and VLAN Insertion on Tx option is selected 0b - Source Address or VLAN Insertion Enable option is not selected 1b - Source Address or VLAN Insertion Enable option is selected
26-25 TSSTSSEL	Timestamp System Time Source This bit indicates the source of the Timestamp system time: This bit is set to 1 when the Enable IEEE 1588 Timestamp Support option is selected 00b - Internal 01b - External

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10b - Both 11b - Reserved
24 MACADR64SEL	MAC Addresses 64-127 Selected This bit is set to 1 when the Enable Additional 64 MAC Address Registers (64-127) option is selected 0b - MAC Addresses 64-127 Select option is not selected 1b - MAC Addresses 64-127 Select option is selected
23 MACADR32SEL	MAC Addresses 32-63 Selected This bit is set to 1 when the Enable Additional 32 MAC Address Registers (32-63) option is selected 0b - MAC Addresses 32-63 Select option is not selected 1b - MAC Addresses 32-63 Select option is selected
22-18 ADDMACADRSEL	MAC Addresses 1-31 Selected This bit is set to 1 when the non-zero value is selected for Enable Additional 1-31 MAC Address Registers option
17 —	Reserved
16 RXCOESEL	Receive Checksum Offload Enabled This bit is set to 1 when the Enable Receive TCP/IP Checksum Check option is selected 0b - Receive Checksum Offload Enable option is not selected 1b - Receive Checksum Offload Enable option is selected
15 —	Reserved
14 TXCOESEL	Transmit Checksum Offload Enabled This bit is set to 1 when the Enable Transmit TCP/IP Checksum Insertion option is selected 0b - Transmit Checksum Offload Enable option is not selected 1b - Transmit Checksum Offload Enable option is selected
13 EEESEL	Energy Efficient Ethernet Enabled This bit is set to 1 when the Enable Energy Efficient Ethernet (EEE) option is selected 0b - Energy Efficient Ethernet Enable option is not selected 1b - Energy Efficient Ethernet Enable option is selected
12 TSSEL	IEEE 1588-2008 Timestamp Enabled This bit is set to 1 when the Enable IEEE 1588 Timestamp Support option is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - IEEE 1588-2008 Timestamp Enable option is not selected 1b - IEEE 1588-2008 Timestamp Enable option is selected
11-10 —	Reserved
9 ARPOFFSEL	ARP Offload Enabled This bit is set to 1 when the Enable IPv4 ARP Offload option is selected 0b - ARP Offload Enable option is not selected 1b - ARP Offload Enable option is selected
8 MMCSEL	RMON Module Enable This bit is set to 1 when the Enable MAC Management Counters (MMC) option is selected 0b - RMON Module Enable option is not selected 1b - RMON Module Enable option is selected
7 MGKSEL	PMT Magic Packet Enable This bit is set to 1 when the Enable Magic Packet Detection option is selected 0b - PMT Magic Packet Enable option is not selected 1b - PMT Magic Packet Enable option is selected
6 RWKSEL	PMT Remote Wake-up Packet Enable This bit is set to 1 when the Enable Remote Wake-Up Packet Detection option is selected 0b - PMT Remote Wake-up Packet Enable option is not selected 1b - PMT Remote Wake-up Packet Enable option is selected
5 SMASEL	SMA (MDIO) Interface This bit is set to 1 when the Enable Station Management (MDIO Interface) option is selected 0b - SMA (MDIO) Interface not selected 1b - SMA (MDIO) Interface selected
4 VLHASH	VLAN Hash Filter Selected This bit is set to 1 when the Enable VLAN Hash Table Based Filtering option is selected 0b - VLAN Hash Filter not selected 1b - VLAN Hash Filter selected
3 PCSSEL	PCS Registers (TBI, SGMII, or RTBI PHY interface) This bit is set to 1 when the TBI, SGMII, or RTBI PHY interface option is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No PCS Registers (TBI, SGMII, or RTBI PHY interface) 1b - PCS Registers (TBI, SGMII, or RTBI PHY interface)
2 HDSEL	Half-duplex Support This bit is set to 1 when the half-duplex mode is selected 0b - No Half-duplex support 1b - Half-duplex support
1 GMIISEL	1000 Mbps Support This bit is set to 1 when 1000 Mbps is selected as the Mode of Operation 0b - No 1000 Mbps support 1b - 1000 Mbps support
0 MIISEL	10 or 100 Mbps Support This bit is set to 1 when 10/100 Mbps is selected as the Mode of Operation 0b - No 10 or 100 Mbps support 1b - 10 or 100 Mbps support

76.17.75 MAC Hardware Feature 1 (MAC_HW_Feature1)

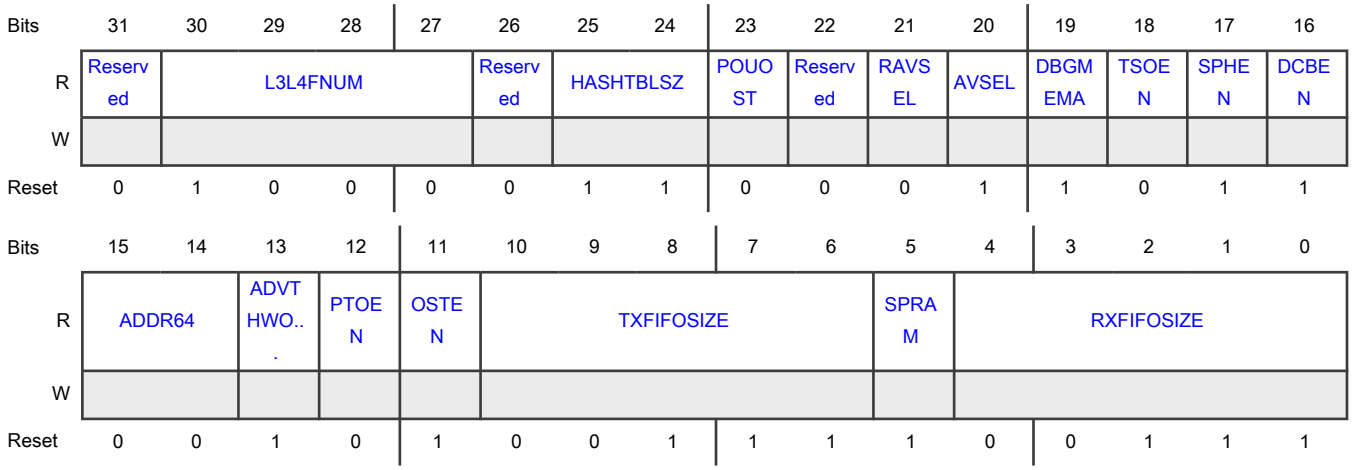
Offset

Register	Offset
MAC_HW_Feature1	120h

Function

This register indicates the presence of second set of the optional features or functions of the DWC_ether_qos. The software driver can use this register to dynamically enable or disable the programs related to the optional blocks. Note: All bits are set or reset according to the features selected while configuring the IP in coreConsultant.

Diagram



Fields

Field	Function
31 —	Reserved
30-27 L3L4FNUM	Total number of L3 or L4 Filters This field indicates the total number of L3 or L4 filters: 0000b - No L3 or L4 Filter 0001b - 1 L3 or L4 Filter 0010b - 2 L3 or L4 Filters 0011b - 3 L3 or L4 Filters 0100b - 4 L3 or L4 Filters 0101b - 5 L3 or L4 Filters 0110b - 6 L3 or L4 Filters 0111b - 7 L3 or L4 Filters 1000b - 8 L3 or L4 Filters
26 —	Reserved
25-24 HASHTBLSZ	Hash Table Size This field indicates the size of the hash table: 00b - No hash table 01b - 64 10b - 128 11b - 256

Table continues on the next page...

Table continued from the previous page...

Field	Function
23 POUOST	One Step for PTP over UDP/IP Feature Enable This bit is set to 1 when the Enable One step timestamp for PTP over UDP/IP feature is selected. 0b - One Step for PTP over UDP/IP Feature is not selected 1b - One Step for PTP over UDP/IP Feature is selected
22 —	Reserved
21 RAVSEL	Rx Side Only AV Feature Enable This bit is set to 1 when the Enable Audio Video Bridging option on Rx Side Only is selected. 0b - Rx Side Only AV Feature is not selected 1b - Rx Side Only AV Feature is selected
20 AVSEL	AV Feature Enable This bit is set to 1 when the Enable Audio Video Bridging option is selected. 0b - AV Feature is not selected 1b - AV Feature is selected
19 DBGMEMA	DMA Debug Registers Enable This bit is set to 1 when the Debug Mode Enable option is selected 0b - DMA Debug Registers option is not selected 1b - DMA Debug Registers option is selected
18 TSOEN	TCP Segmentation Offload Enable This bit is set to 1 when the Enable TCP Segmentation Offloading for TCP/IP Packets option is selected 0b - TCP Segmentation Offload Feature is not selected 1b - TCP Segmentation Offload Feature is selected
17 SPHEN	Split Header Feature Enable This bit is set to 1 when the Enable Split Header Structure option is selected 0b - Split Header Feature is not selected 1b - Split Header Feature is selected
16 DCBEN	DCB Feature Enable This bit is set to 1 when the Enable Data Center Bridging option is selected 0b - DCB Feature is not selected 1b - DCB Feature is selected
15-14	Address Width.

Table continues on the next page...

Table continued from the previous page...

Field	Function
ADDR64	This field indicates the configured address width: 00b - 32 01b - 40 10b - 48 11b - Reserved
13 ADVTHWORD	IEEE 1588 High Word Register Enable This bit is set to 1 when the Add IEEE 1588 Higher Word Register option is selected 0b - IEEE 1588 High Word Register option is not selected 1b - IEEE 1588 High Word Register option is selected
12 PTOEN	PTP Offload Enable This bit is set to 1 when the Enable PTP Timestamp Offload Feature is selected. 0b - PTP Offload feature is not selected 1b - PTP Offload feature is selected
11 OSTEN	One-Step Timestamping Enable This bit is set to 1 when the Enable One-Step Timestamp Feature is selected. 0b - One-Step Timestamping feature is not selected 1b - One-Step Timestamping feature is selected
10-6 TXFIFOSIZE	MTL Transmit FIFO Size This field contains the configured value of MTL Tx FIFO in bytes expressed as Log to base 2 minus 7, that is, Log2(TXFIFO_SIZE) -7: 0_0000b - 128 bytes 0_0001b - 256 bytes 0_0010b - 512 bytes 0_0011b - 1024 bytes 0_0100b - 2048 bytes 0_0101b - 4096 bytes 0_0110b - 8192 bytes 0_0111b - 16384 bytes 0_1000b - 32 KB 0_1001b - 64 KB 0_1010b - 128 KB 0_1011b - Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
5 SPRAM	Single Port RAM Enable This bit is set to 1 when the Use single port RAM Feature is selected. 0b - Single Port RAM feature is not selected 1b - Single Port RAM feature is selected
4-0 RXFIFOSIZE	MTL Receive FIFO Size This field contains the configured value of MTL Rx FIFO in bytes expressed as Log to base 2 minus 7, that is, Log2(RXFIFO_SIZE) -7: 0_0000b - 128 bytes 0_0001b - 256 bytes 0_0010b - 512 bytes 0_0011b - 1024 bytes 0_0100b - 2048 bytes 0_0101b - 4096 bytes 0_0110b - 8192 bytes 0_0111b - 16384 bytes 0_1000b - 32 KB 0_1001b - 64 KB 0_1010b - 128 KB 0_1011b - 256 KB 0_1100b - Reserved

76.17.76 MAC Hardware Feature 2 (MAC_HW_Feature2)

Offset

Register	Offset
MAC_HW_Feature2	124h

Function

This register indicates the presence of third set of the optional features or functions of the DWC_ether_qos. The software driver can use this register to dynamically enable or disable the programs related to the optional blocks.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved	AUXSNAPNUM			Reserved	PPSOUTNUM			TDCSZ		TXCHCNT				RDCSZ	
W																
Reset	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXCHCNT				Reserved		TXQCNT				Reserved		RXQCNT			
W																
Reset	0	0	1	0	0	0	0	0	1	0	0	0	0	0	1	0

Fields

Field	Function
31 —	Reserved
30-28 AUXSNAPNUM	Number of Auxiliary Snapshot Inputs This field indicates the number of auxiliary snapshot inputs: 000b - No auxiliary input 001b - 1 auxiliary input 010b - 2 auxiliary input 011b - 3 auxiliary input 100b - 4 auxiliary input 101b - Reserved
27 —	Reserved
26-24 PPSOUTNUM	Number of PPS Outputs This field indicates the number of PPS outputs: 000b - No PPS output 001b - 1 PPS output 010b - 2 PPS output 011b - 3 PPS output 100b - 4 PPS output 101b - Reserved
23-22	Tx DMA Descriptor Cache Size in terms of 16 bytes descriptors:

Table continues on the next page...

Table continued from the previous page...

Field	Function
TDCSZ	00 - Cache Not configured 01 - 4 16 bytes descriptor 10 - 8 16 bytes descriptor 11 - 16 16 bytes descriptor
21-18 TXCHCNT	Number of DMA Transmit Channels This field indicates the number of DMA Transmit channels: 0000b - 1 MTL Tx Channel 0001b - 4 0010b - 8 0011b - 16 0100b - 5 MTL Tx Channels 0101b - 6 MTL Tx Channels 0110b - 7 MTL Tx Channels 0111b - 8 MTL Tx Channels
17-16 RDCSZ	Rx DMA Descriptor Cache Size in terms of 16 bytes descriptors: 00 - Cache Not configured 01 - 4 16 bytes descriptor 10 - 8 16 bytes descriptor 11 - 16 16 bytes descriptor
15-12 RXCHCNT	Number of DMA Receive Channels This field indicates the number of DMA Receive channels: 0000b - 1 MTL Rx Channel 0001b - 4 0010b - 8 0011b - 16 0100b - 5 MTL Rx Channels 0101b - 6 MTL Rx Channels 0110b - 7 MTL Rx Channels 0111b - 8 MTL Rx Channels
11-10 —	Reserved
9-6 TXQCNT	Number of MTL Transmit Queues This field indicates the number of MTL Transmit queues: 0000b - 1 MTL Tx Queue 0001b - 2 MTL Tx Queues 0010b - 3 MTL Tx Queues

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0011b - 4 MTL Tx Queues 0100b - 5 MTL Tx Queues 0101b - 6 MTL Tx Queues 0110b - 7 MTL Tx Queues 0111b - 8 MTL Tx Queues
5-4 —	Reserved
3-0 RXQCNT	Number of MTL Receive Queues This field indicates the number of MTL Receive queues: 0000b - 1 MTL Rx Queue 0001b - 2 MTL Rx Queues 0010b - 3 MTL Rx Queues 0011b - 4 MTL Rx Queues 0100b - 5 MTL Rx Queues 0101b - 6 MTL Rx Queues 0110b - 7 MTL Rx Queues 0111b - 8 MTL Rx Queues

76.17.77 MAC Hardware Feature 3 (MAC_HW_Feature3)

Offset

Register	Offset
MAC_HW_Feature3	128h

Function

This register indicates the presence of fourth set the optional features or functions of the DWC_ether_qos. The software driver can use this register to dynamically enable or disable the programs related to the optional blocks.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved		ASP		TBSSEL	FPESEL	Reserved				ESTWID		ESTDEP			ESTSEL
W																
Reset	0	0	1	0	1	1	0	0	0	0	1	1	0	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	FRPES		FRPBS		FRPSEL	PDUPSEL	Reserved			DVLAN	CBTISEL	Reserved	NRVF		
W																
Reset	0	0	0	1	0	1	1	0	0	0	1	1	0	1	0	1

Fields

Field	Function
31-30 —	Reserved
29-28 ASP	Automotive Safety Package Following are the encoding for the different Safety features 00b - No Safety features selected 01b - Only "ECC protection for external memory" feature is selected 10b - All the Automotive Safety features are selected without the "Parity Port Enable for external interface" feature 11b - All the Automotive Safety features are selected with the "Parity Port Enable for external interface" feature
27 TBSSEL	Time Based Scheduling Enable This bit is set to 1 when the Time Based Scheduling feature is selected. 0b - Time Based Scheduling Enable feature is not selected 1b - Time Based Scheduling Enable feature is selected
26 FPESEL	Frame Preemption Enable This bit is set to 1 when the Enable Frame preemption feature is selected. 0b - Frame Preemption Enable feature is not selected 1b - Frame Preemption Enable feature is selected
25-22 —	Reserved
21-20	Width of the Time Interval field in the Gate Control List

Table continues on the next page...

Table continued from the previous page...

Field	Function
ESTWID	This field indicates the width of the Configured Time Interval Field 00b - Width not configured 01b - 16 10b - 20 11b - 24
19-17 ESTDEP	Depth of the Gate Control List This field indicates the depth of Gate Control list expressed as Log2(DWC_EQOS_EST_DEP)-5 000b - No Depth configured 001b - 64 010b - 128 011b - 256 100b - 512 101b - 1024 110b - Reserved
16 ESTSEL	Enhancements to Scheduled Traffic Enable This bit is set to 1 when the Enable Enhancements to Scheduling Traffic feature is selected. 0b - Enable Enhancements to Scheduling Traffic feature is not selected 1b - Enable Enhancements to Scheduling Traffic feature is selected
15 —	Reserved
14-13 FRPES	Flexible Receive Parser Table Entries size This field indicates the Max Number of Parser Entries supported by Flexible Receive Parser. 00b - 64 Entries 01b - 128 Entries 10b - 256 Entries 11b - Reserved
12-11 FRPBS	Flexible Receive Parser Buffer size This field indicates the supported Max Number of bytes of the packet data to be Parsed by Flexible Receive Parser. 00b - 64 Bytes 01b - 128 Bytes 10b - 256 Bytes

Table continues on the next page...

Table continued from the previous page...

Field	Function
	11b - Reserved
10 FRPSEL	Flexible Receive Parser Selected This bit is set to 1 when the Enable Flexible Programmable Receive Parser option is selected. 0b - Flexible Receive Parser feature is not selected 1b - Flexible Receive Parser feature is selected
9 PDUPSEL	Broadcast/Multicast Packet Duplication This bit is set to 1 when the Broadcast/Multicast Packet Duplication feature is selected. 0b - Broadcast/Multicast Packet Duplication feature is not selected 1b - Broadcast/Multicast Packet Duplication feature is selected
8-6 —	Reserved
5 DVLAN	Double VLAN Tag Processing Selected This bit is set to 1 when the Enable Double VLAN Processing Feature is selected. 0b - Double VLAN option is not selected 1b - Double VLAN option is selected
4 CBTISEL	Queue/Channel based VLAN tag insertion on Tx Enable This bit is set to 1 when the Enable Queue/Channel based VLAN tag insertion on Tx Feature is selected. 0b - Enable Queue/Channel based VLAN tag insertion on Tx feature is not selected 1b - Enable Queue/Channel based VLAN tag insertion on Tx feature is selected
3 —	Reserved
2-0 NRVF	Number of Extended VLAN Tag Filters Enabled This field indicates the Number of Extended VLAN Tag Filters selected: 000b - No Extended Rx VLAN Filters 001b - 4 Extended Rx VLAN Filters 010b - 8 Extended Rx VLAN Filters 011b - 16 Extended Rx VLAN Filters 100b - 24 Extended Rx VLAN Filters 101b - 32 Extended Rx VLAN Filters 110b - Reserved

76.17.78 MAC DPP FSM Interrupt Status (MAC_DPP_FSM_Interrupt_Status)

Offset

Register	Offset
MAC_DPP_FSM_Interrupt_Status	140h

Function

This register contains the status of Automotive Safety related Data Path Parity Errors, Interface Timeout Errors, FSM State Parity Errors and FSM State Timeout Errors. All the non-Reserved bits are cleared on read.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved			Reserved	MRWCPES	MTFCPES	MTBUPES	FSMPES	Reserved						Reserved	MSTTES
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	Reserved	Reserved	PTES	ATES	Reserved	RTES	TTES	Reserved	Reserved	ARPES	MTSPES	MPES	RDPES	Reserved	Reserved
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	Reserved
28 —	Reserved
27 MRWCPES	MTL RWC data path Parity checker Error Status This filed when set indicates that, parity error is detected on the MTL RWC data interface (or at PC12 as shown in Recieve data path parity protection diagram). **NOTE** This field is not testable by an automated register test. 0b - MTL RWC data path Parity checker Error Status not detected 1b - MTL RWC data path Parity checker Error Status detected
26	MAC TFC data path Parity checker Error Status

Table continues on the next page...

Table continued from the previous page...

Field	Function
MTFCPES	This filed when set indicates that, parity error is detected on the MAC TFC data interface (or at PC11 as shown in Transmit data path parity protection diagram). **NOTE** This field is not testable by an automated register test. 0b - MAC TFC data path Parity checker Error Status not detected 1b - MAC TFC data path Parity checker Error Status detected
25 MTBUPES	MAC TBU data path Parity checker Error Status This filed when set indicates that, parity error is detected on the MAC TBU data interface (or at PC10 as shown in Transmit data path parity protection diagram). **NOTE** This field is not testable by an automated register test. 0b - MAC TBU data path Parity checker Error Status not detected 1b - MAC TBU data path Parity checker Error Status detected
24 FSMPES	FSM State Parity Error Status This field when set indicates one of the FSMs State registers has a parity error detected. **NOTE** This field is not testable by an automated register test. 0b - FSM State Parity Error Status not detected 1b - FSM State Parity Error Status detected
23-18 —	Reserved
17 —	Reserved
16 MSTTES	Master Read/Write Timeout Error Status This field when set indicates that an Application/CSR Timeout has occurred on the master (AXI/AHB/ARI/ATI) interface. **NOTE** This field is not testable by an automated register test. 0b - Master Read/Write Timeout Error Status not detected 1b - Master Read/Write Timeout Error Status detected
15 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
14 —	Reserved
13 —	Reserved
12 PTES	PTP FSM Timeout Error Status This field when set indicates that one of the PTP FSM Timeout has occurred. **NOTE** This field is not testable by an automated register test. 0b - PTP FSM Timeout Error Status not detected 1b - PTP FSM Timeout Error Status detected
11 ATES	APP FSM Timeout Error Status This field when set indicates that one of the APP FSM Timeout has occurred. **NOTE** This field is not testable by an automated register test. 0b - APP FSM Timeout Error Status not detected 1b - APP FSM Timeout Error Status detected
10 —	Reserved
9 RTES	Rx FSM Timeout Error Status This field when set indicates that one of the Rx FSM Timeout has occurred. **NOTE** This field is not testable by an automated register test. 0b - Rx FSM Timeout Error Status not detected 1b - Rx FSM Timeout Error Status detected
8 TTES	Tx FSM Timeout Error Status This field when set indicates that one of the Tx FSM Timeout has occurred. **NOTE** This field is not testable by an automated register test. 0b - Tx FSM Timeout Error Status not detected 1b - Tx FSM Timeout Error Status detected

Table continues on the next page...

Table continued from the previous page...

Field	Function
7 —	Reserved
6 —	Reserved
5 ARPES	Application Receive interface data path Parity Error Status This bit when set indicates that a parity error is detected at the following checkers based on the system configuration: - In MTL configuration (DWC_EQOS_SYS=1), parity checker (PC6 as shown in Receive Data path Parity protection diagram) at ARI interface. - In DMA configuration (DWC_EQOS_SYS=2), parity checker (PC6 as shown in Receive Data path Parity protection diagram) at DMA application interface. - In AHB configuration (DWC_EQOS_SYS=3), parity checker (PC6 as shown in Receive Data path Parity protection diagram) at AHB master interface. - In AXI configuration (DWC_EQOS_SYS=4), parity checker (PC6 as shown in Receive Data path Parity protection diagram) at AXI master interface. **NOTE** This field is not testable by an automated register test. 0b - Application Receive interface data path Parity Error Status not detected 1b - Application Receive interface data path Parity Error Status detected
4 MTSPES	MTL TX Status data path Parity checker Error Status This filed when set indicates that, parity error is detected on the MTL TX Status data on ati interface (or at PC5 as shown in Transmit data path parity protection diagram). **NOTE** This field is not testable by an automated register test. 0b - MTL TX Status data path Parity checker Error Status not detected 1b - MTL TX Status data path Parity checker Error Status detected
3 MPES	MTL data path Parity checker Error Status This bit when set indicates that a parity error is detected at the MTL transmit write controller parity checker (or at PC4 as shown in Transmit data path parity protection diagram). **NOTE** This field is not testable by an automated register test. 0b - MTL data path Parity checker Error Status not detected 1b - MTL data path Parity checker Error Status detected
2 RDPES	Read Descriptor Parity checker Error Status This bit when set indicates that a parity error is detected at the DMA Read descriptor parity checker (or at PC3 as shown in Transmit data path parity protection diagram).

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field is not testable by an automated register test. 0b - Read Descriptor Parity checker Error Status not detected 1b - Read Descriptor Parity checker Error Status detected
1 —	Reserved
0 —	Reserved

76.17.79 MAC FSM Control (MAC_FSM_Control)

Offset

Register	Offset
MAC_FSM_Control	148h

Function

This register is used to control the FSM State parity and timeout error injection in Debug mode.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved	Reserved	Reserved	PLGRNML	ALGRNML	Reserved	RLGRNML	TLGRNML	Reserved	Reserved	Reserved	PPEIN	APEIN	Reserved	RPEIN	TPEIN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	Reserved	Reserved	PTEIN	ATEIN	Reserved	RTEIN	TTEIN	Reserved						PRTYEN	TMOUTEN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
30 —	Reserved
29 —	Reserved
28 PLGRNML	PTP Large/Normal Mode Select This field when set indicates that large mode tic generation is used for PTP domain, else normal mode tic generation is used. 0b - normal mode tic generation is used for PTP domain 1b - large mode tic generation is used for PTP domain
27 ALGRNML	APP Large/Normal Mode Select This field when set indicates that large mode tic generation is used for APP domain, else normal mode tic generation is used. 0b - normal mode tic generation is used for APP domain 1b - large mode tic generation is used for APP domain
26 —	Reserved
25 RLGRNML	Rx Large/Normal Mode Select This field when set indicates that large mode tic generation is used for Rx domain, else normal mode tic generation is used. 0b - normal mode tic generation is used for Rx domain 1b - large mode tic generation is used for Rx domain
24 TLGRNML	Tx Large/Normal Mode Select This field when set indicates that large mode tic generation is used for Tx domain, else normal mode tic generation is used. 0b - normal mode tic generation is used for Tx domain 1b - large mode tic generation is used for Tx domain
23 —	Reserved
22 —	Reserved
21	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
20 PPEIN	PTP FSM Parity Error Injection This field when set indicates that Error Injection for PTP FSM Parity is enabled. 0b - PTP FSM Parity Error Injection is disabled 1b - PTP FSM Parity Error Injection is enabled
19 APEIN	APP FSM Parity Error Injection This field when set indicates that Error Injection for APP FSM Parity is enabled. 0b - APP FSM Parity Error Injection is disabled 1b - APP FSM Parity Error Injection is enabled
18 —	Reserved
17 RPEIN	Rx FSM Parity Error Injection This field when set indicates that Error Injection for RX FSM Parity is enabled. 0b - Rx FSM Parity Error Injection is disabled 1b - Rx FSM Parity Error Injection is enabled
16 TPEIN	Tx FSM Parity Error Injection This field when set indicates that Error Injection for TX FSM Parity is enabled. 0b - Tx FSM Parity Error Injection is disabled 1b - Tx FSM Parity Error Injection is enabled
15 —	Reserved
14 —	Reserved
13 —	Reserved
12 PTEIN	PTP FSM Timeout Error Injection This field when set indicates that Error Injection for PTP FSM timeout is enabled. 0b - PTP FSM Timeout Error Injection is disabled 1b - PTP FSM Timeout Error Injection is enabled
11	APP FSM Timeout Error Injection

Table continues on the next page...

Table continued from the previous page...

Field	Function
ATEIN	This field when set indicates that Error Injection for APP FSM timeout is enabled. 0b - APP FSM Timeout Error Injection is disabled 1b - APP FSM Timeout Error Injection is enabled
10 —	Reserved
9 RTEIN	Rx FSM Timeout Error Injection This field when set indicates that Error Injection for RX FSM timeout is enabled. 0b - Rx FSM Timeout Error Injection is disabled 1b - Rx FSM Timeout Error Injection is enabled
8 TTEIN	Tx FSM Timeout Error Injection This field when set indicates that Error Injection for TX FSM timeout is enabled. 0b - Tx FSM Timeout Error Injection is disabled 1b - Tx FSM Timeout Error Injection is enabled
7-2 —	Reserved
1 PRTYEN	This bit when set indicates that the FSM parity feature is enabled. This bit when set indicates that the FSM parity feature is enabled. 0b - FSM Parity feature is disabled 1b - FSM Parity feature is enabled
0 TMOUTEN	This bit when set indicates that the FSM timeout feature is enabled. This bit when set indicates that the FSM timeout feature is enabled. 0b - FSM timeout feature is disabled 1b - FSM timeout feature is enabled

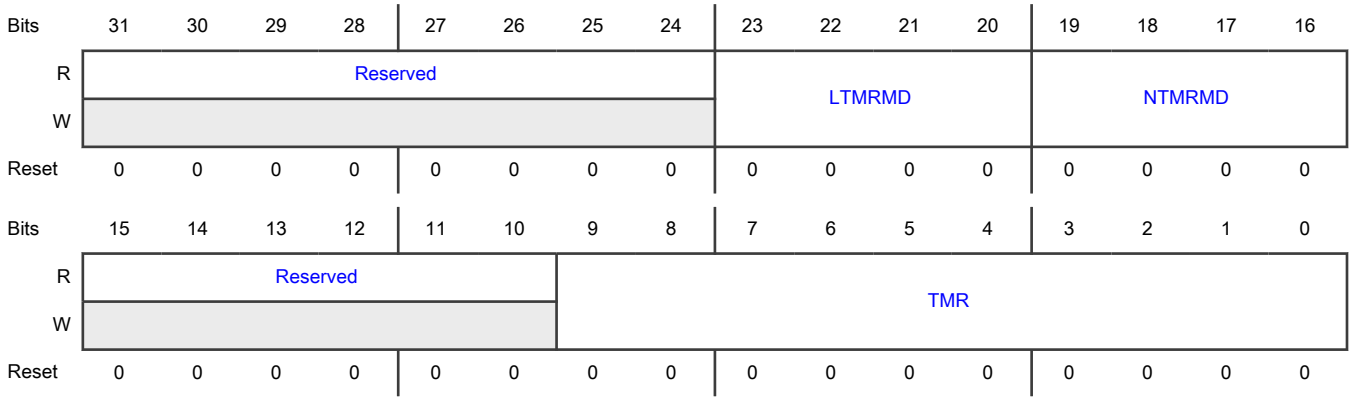
76.17.80 MAC FSM ACT Timer (MAC_FSM_ACT_Timer)

Offset

Register	Offset
MAC_FSM_ACT_Timer	14Ch

Function
This register is used to select the FSM and Interface Timeout values.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-20 LTMRMD	Large Mode Timeout Value This field provides the mode value to be used for large mode FSM and other interface time outs. The timeout duration is based on the mode value. 0000b - Timer disabled 0001b - 1us 0010b - 1.024ms (~4ms) 0011b - 16.384ms (~16ms) 0100b - 65.536ms (~64ms) 0101b - 262.144ms (~256ms) 0110b - 1.048sec (~1sec) 0111b - 4.194sec (~4sec) 1000b - 16.777sec (~16sec) 1001b - 33.554sec (~32sec) 1010b - 67.108sec (~64sec) 1011b - Reserved
19-16 NTMRMD	Normal Mode Timeout Value This field provides the value to be used for normal mode FSM and other interface time outs. The timeout duration is based on the mode value. 0000b - Timer disabled 0001b - 1us 0010b - 1.024ms (~4ms)

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0011b - 16.384ms (~16ms) 0100b - 65.536ms (~64ms) 0101b - 262.144ms (~256ms) 0110b - 1.048sec (~1sec) 0111b - 4.194sec (~4sec) 1000b - 16.777sec (~16sec) 1001b - 33.554sec (~32sec) 1010b - 67.108sec (~64sec) 1011b - Reserved
15-10 —	Reserved
9-0 TMR	CSR Clocks for 1us Tic This field indicates the number of CSR clocks required to generate 1us tic.

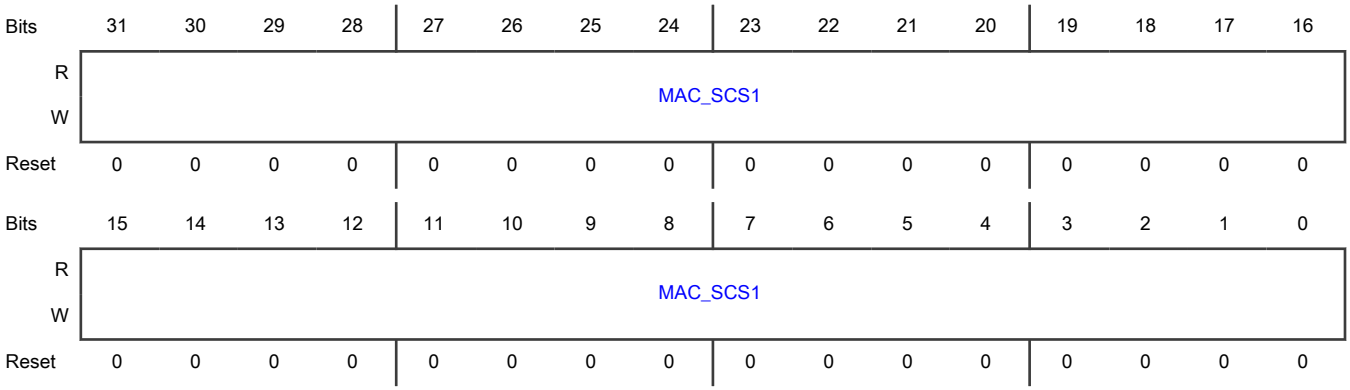
76.17.81 SCS REG 1 (SCS_REG1)

Offset

Register	Offset
SCS_REG1	150h

Function
NXP Reserved Register

Diagram



Fields

Field	Function
31-0 MAC_SCS1	MAC SCS 1 NXP Reserved, All the bits must be set to "0". This field is reserved for NXP Internal use, and must always be set to "0" unless instructed by NXP. Setting any bit to "1" might cause unexpected behavior in the IP.

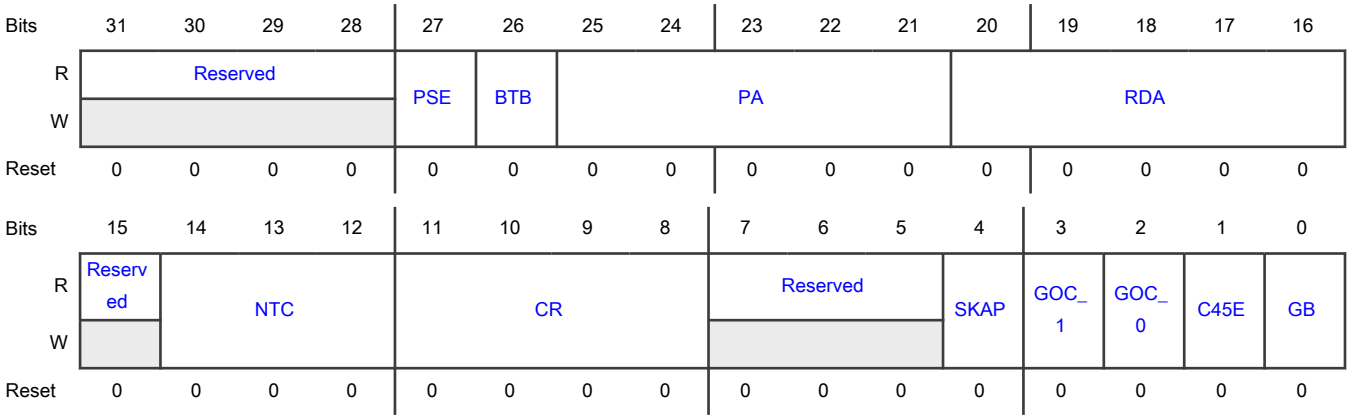
76.17.82 MAC MDIO Address (MAC_MDIO_Address)

Offset

Register	Offset
MAC_MDIO_Address	200h

Function
The MDIO Address register controls the management cycles to external PHY through a management interface.

Diagram



Fields

Field	Function
31-28 —	Reserved
27 PSE	Preamble Suppression Enable When this bit is set, the SMA suppresses the 32-bit preamble and transmits MDIO frames with only 1 preamble bit. When this bit is 0, the MDIO frame always has 32 bits of preamble as defined in the IEEE specifications. 0b - Preamble Suppression disabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Preamble Suppression enabled
26 BTB	Back to Back transactions When this bit is set and the NTC has value greater than 0, then the MAC informs the completion of a read or write command at the end of frame transfer (before the trailing clocks are transmitted). The software can thus initiate the next command which is executed immediately irrespective of the number trailing clocks generated for the previous frame. When this bit is reset, then the read/write command completion (GB is cleared) only after the trailing clocks are generated. In this mode, it is ensured that the NTC is always generated after each frame. This bit must not be set when NTC=0. 0b - Back to Back transactions disabled 1b - Back to Back transactions enabled
25-21 PA	Physical Layer Address This field indicates which Clause 22 PHY devices (out of 32 devices) the MAC is accessing. This field indicates which Clause 45 capable PHYs (out of 32 PHYs) the MAC is accessing. **NOTE** This field is not testable by an automated register test.
20-16 RDA	Register/Device Address These bits select the PHY register in selected Clause 22 PHY device. These bits select the Device (MMD) in selected Clause 45 capable PHY.
15 —	Reserved
14-12 NTC	Number of Trailing Clocks This field controls the number of trailing clock cycles generated on GMII_MDC_O (MDC) after the end of transmission of MDIO frame. The valid values can be from 0 to 7. Programming the value to 3'h3 indicates that there are additional three clock cycles on the MDC line after the end of MDIO frame transfer. 000b - Loopback is disabled 001b - Loopback is enabled
11-8 CR	CSR Clock Range The CSR Clock Range selection determines the frequency of the MDC clock according to the CSR clock frequency used in your design: - 0000: CSR clock = 60-100 MHz; MDC clock = CSR clock/42 - 0001: CSR clock = 100-150 MHz; MDC clock = CSR clock/62 - 0010: CSR clock = 20-35 MHz; MDC clock = CSR clock/16 - 0011: CSR clock = 35-60 MHz; MDC clock = CSR clock/26 - 0100: CSR clock = 150-250 MHz; MDC clock = CSR clock/102 - 0101: CSR clock = 250-300 MHz; MDC clock = CSR clock/124 - 0110: CSR clock = 300-500 MHz; MDC clock = CSR clock/204 - 0111: CSR clock = 500-800 MHz; MDC clock = CSR clock/324 The suggested range of CSR clock frequency applicable for each value (when Bit 11 = 0) ensures that the MDC clock is approximately between 1.0 MHz to 2.5 MHz frequency range. When Bit 11 is set, you can achieve a higher frequency of the MDC clock than the frequency limit of 2.5 MHz (specified in the IEEE 802.3) and program a clock divider of lower value. For example, when

Table continues on the next page...

Table continued from the previous page...

Field	Function
	CSR clock is of 100 MHz frequency and you program these bits as 1010, the resultant MDC clock is of 12.5 MHz which is beyond the range specified in IEEE 802.3. Program the following values only if the interfacing chips support faster MDC clocks: - 1000: CSR clock/4 - 1001: CSR clock/6 - 1010: CSR clock/8 - 1011: CSR clock/10 - 1100: CSR clock/12 - 1101: CSR clock/14 - 1110: CSR clock/16 - 1111: CSR clock/18 These bits are not used for accessing RevMII. These bits are read-only if the RevMII interface is selected as single PHY interface.
7-5 —	Reserved
4 SKAP	Skip Address Packet When this bit is set, the SMA does not send the address packets before read, write, or post-read increment address packets. This bit is valid only when C45E is set. 0b - Skip Address Packet is disabled 1b - Skip Address Packet is enabled
3 GOC_1	GMII Operation Command 1 This bit is higher bit of the operation command to the PHY or RevMII, GOC_1 and GOC_O is encoded as follows: - 00: Reserved - 01: Write - 10: Post Read Increment Address for Clause 45 PHY - 11: Read When Clause 22 PHY or RevMII is enabled, only Write and Read commands are valid. **NOTE** This field is not testable by an automated register test. 0b - GMII Operation Command 1 is disabled 1b - GMII Operation Command 1 is enabled
2 GOC_0	GMII Operation Command 0 This is the lower bit of the operation command to the PHY or RevMII. When in SMA mode (MDIO master) this bit along with GOC_1 determines the operation to be performed to the PHY. When only RevMII is selected in configuration this bit is read-only and tied to 1. 0b - GMII Operation Command 0 is disabled 1b - GMII Operation Command 0 is enabled
1 C45E	Clause 45 PHY Enable When this bit is set, Clause 45 capable PHY is connected to MDIO. When this bit is reset, Clause 22 capable PHY is connected to MDIO. 0b - Clause 45 PHY is disabled 1b - Clause 45 PHY is enabled
0 GB	GMII Busy The application sets this bit to instruct the SMA to initiate a Read or Write access to the MDIO slave. The MAC clears this bit after the MDIO frame transfer is completed. Hence the software must not write or change any of the fields in MAC_MDIO_Address and MAC_MDIO_Data registers as long as this bit is

Table continues on the next page...

Table continued from the previous page...

Field	Function
	set. For write transfers, the application must first write 16-bit data in the GDI field (and also RA field when C45E is set) in MAC_MDIO_Data register before setting this bit. When C45E is set, it should also write into the RA field of MAC_MDIO_Data register before initiating a read transfer. When a read transfer is completed (GB=0), the data read from the PHY register is valid in the GD field of the MAC_MDIO_Data register. Note: Even if the addressed PHY is not present, there is no change in the functionality of this bit. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. **NOTE** This field is not testable by an automated register test. 0b - GMII Busy is disabled 1b - GMII Busy is enabled

76.17.83 MAC MDIO Data (MAC_MDIO_Data)

Offset

Register	Offset
MAC_MDIO_Data	204h

Function

The MDIO Data register stores the Write data to be written to the PHY register located at the address specified in MAC_MDIO_Address. This register also stores the Read data from the PHY register located at the address specified by MDIO Address register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	RA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	GD															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16	Register Address

Table continues on the next page...

Table continued from the previous page...

Field	Function
RA	This field is valid only when C45E is set. It contains the Register Address in the PHY to which the MDIO frame is intended for. **NOTE** This field is not testable by an automated register test.
15-0 GD	GMII Data This field contains the 16-bit data value read from the PHY or RevMII after a Management Read operation or the 16-bit data value to be written to the PHY or RevMII before a Management Write operation. **NOTE** This field is not testable by an automated register test.

76.17.84 MAC ARP Address (MAC_ARP_Address)

Offset

Register	Offset
MAC_ARP_Address	210h

Function

The ARP Address register contains the IPv4 Destination Address of the MAC.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	ARPPA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ARPPA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0	ARP Protocol Address

Table continues on the next page...

Field	Function
ARPPA	This field contains the IPv4 Destination Address of the MAC. This address is used for perfect match with the Protocol Address of Target field in the received ARP packet. This field is available only when the Enable IPv4 ARP Offload option is selected.

76.17.85 MAC CSR Software Control (MAC_CSR_SW_Ctrl)

Offset

Register	Offset
MAC_CSR_SW_Ctrl	230h

Function

This register contains software programmable controls for changing the CSR access response and status bits clearing.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved							SEEN	Reserved							RCWE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-9 —	Reserved
8 SEEN	Slave Error Response Enable When this bit is set, the MAC responds with Slave Error for accesses to reserved registers in CSR space. When this bit is reset, the MAC responds with Okay response to any register accessed from CSR space. 0b - Slave Error Response is disabled 1b - Slave Error Response is enabled
7-1	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
0 RCWE	Register Clear on Write 1 Enable When this bit is set, the access mode of some register fields changes to Clear on Write 1, the application needs to set that respective bit to 1 to clear it. When this bit is reset, the access mode of these register fields remain as Clear on Read. 0b - Register Clear on Write 1 is disabled 1b - Register Clear on Write 1 is enabled

76.17.86 MAC FPE Control STS (MAC_FPE_CTRL_STS)

Offset

Register	Offset
MAC_FPE_CTRL_STS	234h

Function

This register controls the operation of Frame Preemption.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved												TRSP	TVER	RRSP	RVER
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												S1_SET_0	SRSP	SVER	EFPE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-20 —	Reserved
19 TRSP	Transmitted Respond Frame

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Set when a Respond mPacket is transmitted (triggered by setting SRSP field). An interrupt can be generated for this event if FPEIE bit of MAC_Interrupt_Enable is set. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0b - Not transmitted Respond Frame 1b - transmitted Respond Frame
18 TVER	Transmitted Verify Frame Set when a Verify mPacket is transmitted (triggered by setting SVER field). An interrupt can be generated for this event if FPEIE bit of MAC_Interrupt_Enable is set. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0b - Not transmitted Verify Frame 1b - transmitted Verify Frame
17 RRSP	Received Respond Frame Set when a Respond mPacket is received. An interrupt can be generated for this event if FPEIE bit of MAC_Interrupt_Enable is set. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0b - Not received Respond Frame 1b - Received Respond Frame
16 RVER	Received Verify Frame Set when a Verify mPacket is received. An interrupt can be generated for this event if FPEIE bit of MAC_Interrupt_Enable is set. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0b - Not received Verify Frame 1b - Received Verify Frame
15-4 —	Reserved
3 S1_SET_0	S1 SET 0 NXP Reserved, Must be set to "0". This field is reserved for NXP Internal use, and must always be set to "0" unless instructed by NXP. Setting to "1" might cause unexpected behavior in the IP.
2 SRSP	Send Respond mPacket When set indicates hardware to send a Respond mPacket. Reset by hardware after sending the Respond mPacket. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. 0b - Send Respond mPacket is disabled 1b - Send Respond mPacket is enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 SVER	Send Verify mPacket When set indicates hardware to send a verify mPacket. Reset by hardware after sending the Verify mPacket. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. 0b - Send Verify mPacket is disabled 1b - Send Verify mPacket is enabled
0 EFPE	Enable Tx Frame Preemption When set Frame Preemption Tx functionality is enabled. 0b - Tx Frame Preemption is disabled 1b - Tx Frame Preemption is enabled

76.17.87 MAC Extended Configuration 1 (MAC_Ext_Cfg1)

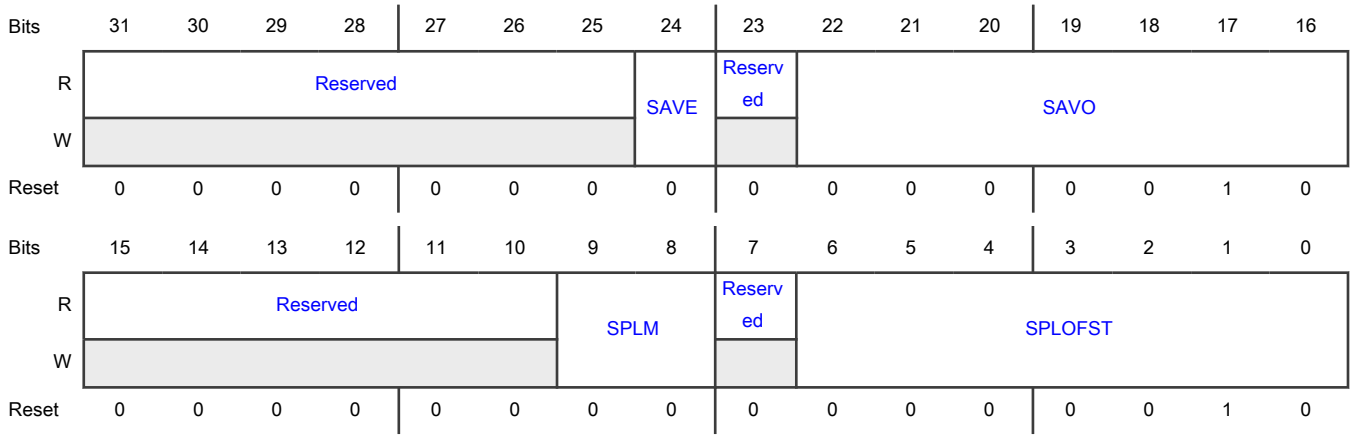
Offset

Register	Offset
MAC_Ext_Cfg1	238h

Function

This register contains Split mode control field and offset field for Split Header feature.

Diagram



Fields

Field	Function
31-25	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
24 SAVE	Split AV Enable - When this bit is set to 1, and the received packet is an AV Type packet, the header is split at SAVO bytes from the beginning of Length/Type field of the packet, for L2 Split. - When this bit is set to 0, header is split at SPLOFST bytes from the beginning of Length/Type field of the frame, for L2 Split.
23 —	Reserved
22-16 SAVO	Split AV Offset When SAVE bit is set to 1, and the received packet is an AV Type packet, these bits indicate the value of the offset from the beginning of Length/Type field at which header should be split, when appropriate SPLM is selected. The reset value of this field is 2 bytes, indicating a split at L2 header. Value is in terms of bytes.
15-10 —	Reserved
9-8 SPLM	Split Mode These bits indicate the mode of splitting the incoming Rx packets. They are 00b - Split at L3/L4 header 01b - Split at L2 header with an offset. Always Split at SPLOFST bytes from the beginning of Length/Type field of the Frame 10b - Combination mode: Split similar to SPLM=00 for IP packets that are untagged or tagged and VLAN stripped 11b - Reserved
7 —	Reserved
6-0 SPLOFST	Split Offset These bits indicate the value of offset from the beginning of Length/Type field at which header split should take place when the appropriate SPLM is selected. The reset value of this field is 2 bytes indicating a split at L2 header. Value is in terms of bytes.

76.17.88 MAC Presentation Time (MAC_Presn_Time_ns)

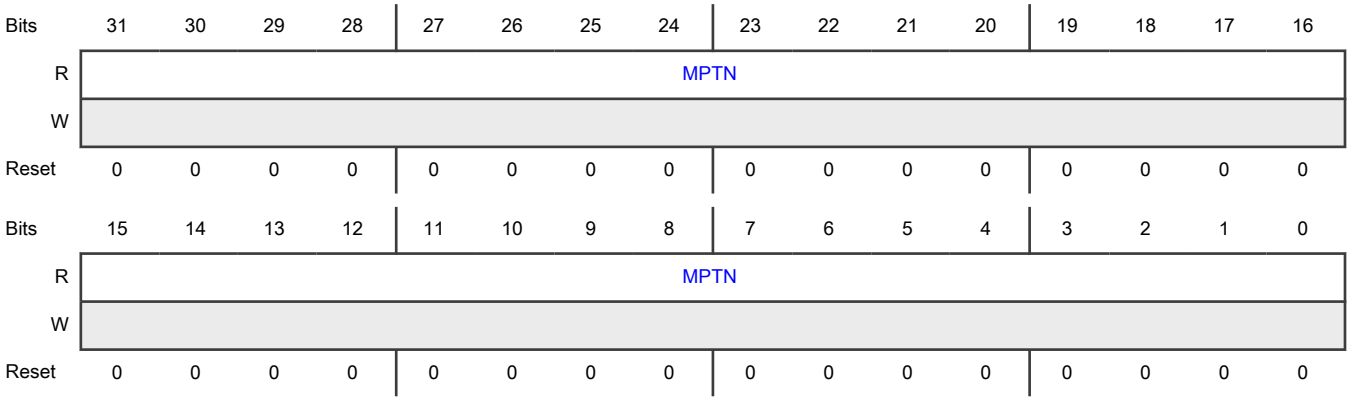
Offset

Register	Offset
MAC_Presn_Time_ns	240h

Function

This register contains the 32-bit binary rollover equivalent time of the PTP System Time in ns Exists when DWC_EQOS_FLEXI_PPS_OUT_EN is configured

Diagram



Fields

Field	Function
31-0 MPTN	MAC 1722 Presentation Time in ns These bits indicate the value of the 32-bit binary rollover equivalent time of the PTP System Time in ns

76.17.89 MAC Presentation Time Update (MAC_Presn_Time_Updt)

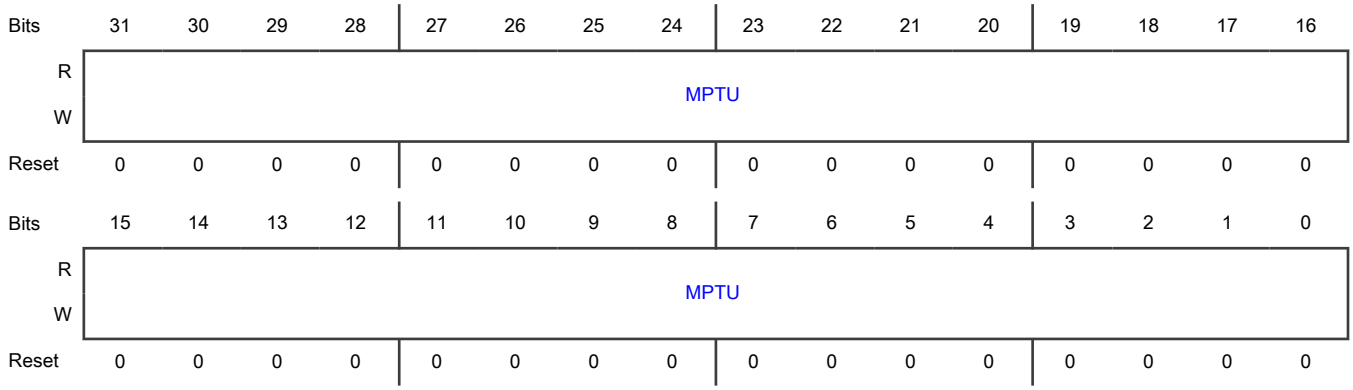
Offset

Register	Offset
MAC_Presn_Time_Updt	244h

Function

This field holds the 32-bit value of MAC 1722 Presentation Time in ns, that should be added to the Current Presentation Time Counter value. Init happens when TSINIT is set, and update happens when the TSUPDT bit is set (TSINIT and TSINIT defined in MAC_Timestamp_ Control register). Exists when DWC_EQOS_FLEXI_PPS_OUT_EN is configured

Diagram



Fields

Field	Function
31-0 MPTU	MAC 1722 Presentation Time Update This field holds the init value or the update value for the presentation time. When used for update, this field holds the 32-bit value in ns, that should be added to the Current Presentation Time Counter value. Init happens when TSINIT is set, and update happens when the TSUPDT bit is set (TSINIT and TSINIT defined in MAC_Timestamp_ Control register). When ADDSUB field of MAC_System_Time_Nanoseconds_Update is set, this value is directly used for subtraction

76.17.90 MAC Address 0 High (MAC_Address0_High)

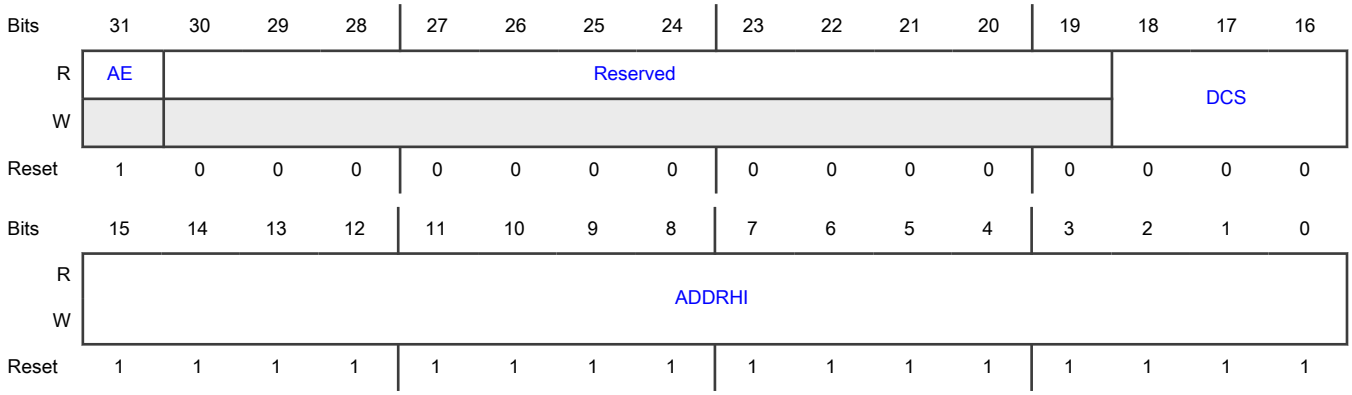
Offset

Register	Offset
MAC_Address0_High	300h

Function

The MAC Address0 High register holds the upper 16 bits of the first 6-byte MAC address of the station. The first DA byte that is received on the (G)MII interface corresponds to the LS byte (Bits [7:0]) of the MAC Address Low register. For example, if 0x112233445566 is received (0x11 in lane 0 of the first column) on the (G)MII as the destination address, then the MacAddress0 Register [47:0] is compared with 0x665544332211. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC Address0 Low Register are written. For proper synchronization updates, the consecutive writes to this Address Low Register should be performed after at least four clock cycles in the destination clock domain.

Diagram



Fields

Field	Function
31 AE	Address Enable This bit is always set to 1. 0b - INVALID : This bit must be always set to 1 1b - This bit is always set to 1
30-19 —	Reserved
18-16 DCS	DMA Channel Select If the PDC bit of MAC_Ext_Configuration register is not set: This field contains the binary representation of the DMA Channel number to which an Rx packet whose DA matches the MAC Address0 content is routed. If the PDC bit of MAC_Ext_Configuration register is set: This field contains the one-hot representation of one or more DMA Channel numbers to which an Rx packet whose DA matches the MAC Address0 content is routed.
15-0 ADDRHI	MAC Address0[47:32] This field contains the upper 16 bits [47:32] of the first 6-byte MAC address. The MAC uses this field for filtering the received packets and inserting the MAC address in the Transmit Flow Control (Pause) Packets.

76.17.91 MAC Address 0 Low (MAC_Address0_Low)

Offset

Register	Offset
MAC_Address0_Low	304h

Function

The MAC Address0 Low register holds the lower 32 bits of the 6-byte first MAC address of the station.

Diagram



Fields

Field	Function
31-0 ADDRLO	MAC Address0[31:0] This field contains the lower 32 bits of the first 6-byte MAC address. The MAC uses this field for filtering the received packets and inserting the MAC address in the Transmit Flow Control (Pause) Packets.

76.17.92 MAC Address 1 High (MAC_Address1_High)

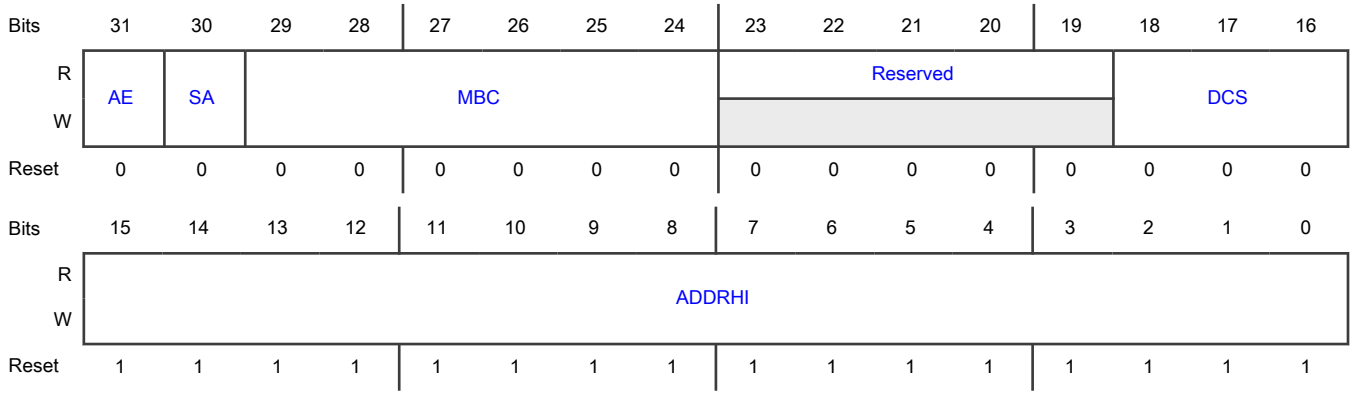
Offset

Register	Offset
MAC_Address1_High	308h

Function

The MAC Address1 High register holds the upper 16 bits of the second 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC Address1 Low Register are written. For proper synchronization updates, the consecutive writes to this Address Low Register should be performed after at least four clock cycles in the destination clock domain.

Diagram



Fields

Field	Function
31 AE	Address Enable When this bit is set, the address filter module uses the second MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. 0b - Address is ignored 1b - Address is enabled
30 SA	Source Address When this bit is set, the MAC Address1[47:0] is used to compare with the SA fields of the received packet. When this bit is reset, the MAC Address1[47:0] is used to compare with the DA fields of the received packet. 0b - Compare with Destination Address 1b - Compare with Source Address
29-24 MBC	Mask Byte Control These bits are mask control bits for comparing each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address1 registers. Each bit controls the masking of the bytes as follows: - Bit 29: MAC_Address 1 High[15:8] - Bit 28: MAC_Address 1 High[7:0] - Bit 27: MAC_Address 1 Low[31:24] - .. - Bit 24: MAC_Address 1 Low[7:0] You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address.
23-19 —	Reserved
18-16 DCS	DMA Channel Select If the PDC bit of MAC_Ext_Configuration register is not set: This field contains the binary representation of the DMA Channel number to which an Rx packet whose DA matches the MAC Address(#i) content is routed. If the PDC bit of MAC_Ext_Configuration register is set: This field contains the one-hot representation of one or more DMA Channel numbers to which an Rx packet whose DA matches the MAC Address(#i) content is routed.
15-0 ADDRHI	MAC Address1 [47:32] This field contains the upper 16 bits[47:32] of the second 6-byte MAC address.

76.17.93 MAC Address 1 Low (MAC_Address1_Low)

Offset

Register	Offset
MAC_Address1_Low	30Ch

Function
The MAC Address1 Low register holds the lower 32 bits of the second 6-byte MAC address of the station.

Diagram



Fields

Field	Function
31-0 ADDRLO	MAC Address1 [31:0] This field contains the lower 32 bits of second 6-byte MAC address. The content of this field is undefined until loaded by the application after the initialization process.

76.17.94 MAC Address 2 High (MAC_Address2_High)

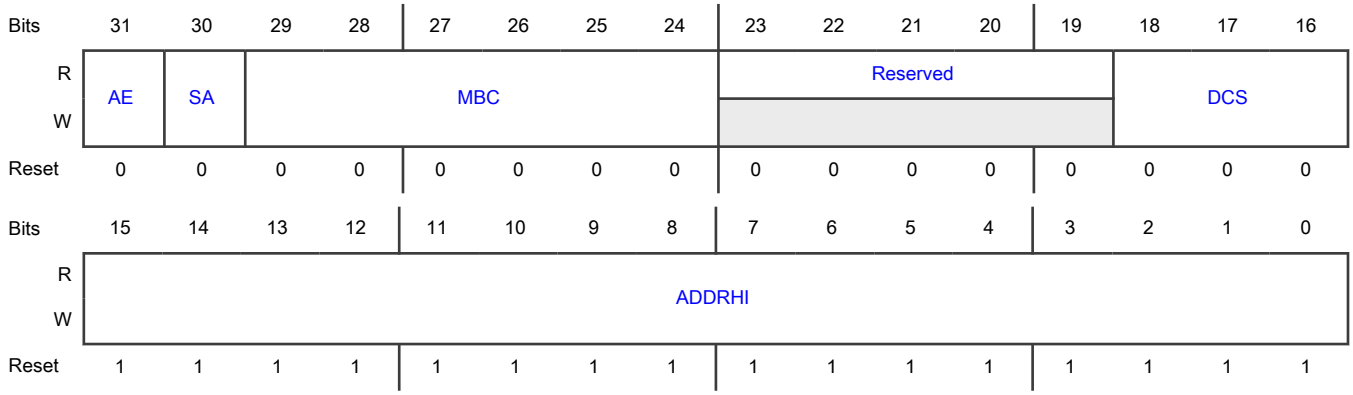
Offset

Register	Offset
MAC_Address2_High	310h

Function

The MAC Address1 High register holds the upper 16 bits of the second 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC Address1 Low Register are written. For proper synchronization updates, the consecutive writes to this Address Low Register should be performed after at least four clock cycles in the destination clock domain.

Diagram



Fields

Field	Function
31 AE	Address Enable When this bit is set, the address filter module uses the second MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. 0b - Address is ignored 1b - Address is enabled
30 SA	Source Address When this bit is set, the MAC Address1[47:0] is used to compare with the SA fields of the received packet. When this bit is reset, the MAC Address1[47:0] is used to compare with the DA fields of the received packet. 0b - Compare with Destination Address 1b - Compare with Source Address
29-24 MBC	Mask Byte Control These bits are mask control bits for comparing each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address1 registers. Each bit controls the masking of the bytes as follows: - Bit 29: MAC_Address 1 High[15:8] - Bit 28: MAC_Address 1 High[7:0] - Bit 27: MAC_Address 1 Low[31:24] - .. - Bit 24: MAC_Address 1 Low[7:0] You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address.
23-19 —	Reserved
18-16 DCS	DMA Channel Select If the PDC bit of MAC_Ext_Configuration register is not set: This field contains the binary representation of the DMA Channel number to which an Rx packet whose DA matches the MAC Address(#i) content is routed. If the PDC bit of MAC_Ext_Configuration register is set: This field contains the one-hot representation of one or more DMA Channel numbers to which an Rx packet whose DA matches the MAC Address(#i) content is routed.
15-0 ADDRHI	MAC Address1 [47:32] This field contains the upper 16 bits[47:32] of the second 6-byte MAC address.

76.17.95 MAC Address 2 Low (MAC_Address2_Low)

Offset

Register	Offset
MAC_Address2_Low	314h

Function
The MAC Address1 Low register holds the lower 32 bits of the second 6-byte MAC address of the station.

Diagram



Fields

Field	Function
31-0 ADDRLO	MAC Address1 [31:0] This field contains the lower 32 bits of second 6-byte MAC address. The content of this field is undefined until loaded by the application after the initialization process.

76.17.96 MMC Control (MMC_Control)

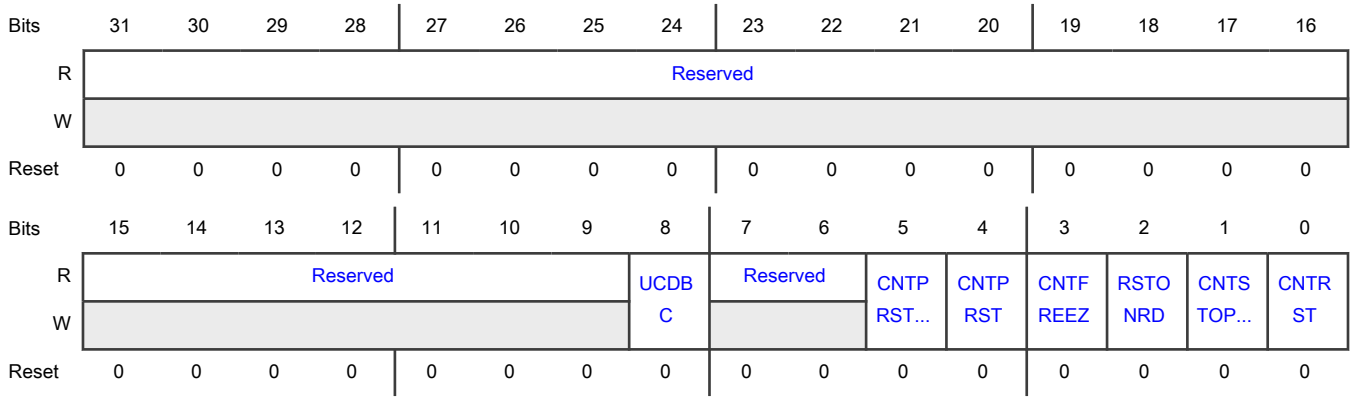
Offset

Register	Offset
MMC_Control	700h

Function

This register establishes the operating mode of MMC.

Diagram



Fields

Field	Function
31-9 —	Reserved
8 UCDBC	Update MMC Counters for Dropped Broadcast Packets Note: The CNTRST bit has a higher priority than the CNTPRST bit. Therefore, when the software tries to set both bits in the same write cycle, all counters are cleared and the CNTPRST bit is not set. When set, the MAC updates all related MMC Counters for Broadcast packets that are dropped because of the setting of the DBF bit of MAC_Packet_Filter register. When reset, the MMC Counters are not updated for dropped Broadcast packets. 0b - Update MMC Counters for Dropped Broadcast Packets is disabled 1b - Update MMC Counters for Dropped Broadcast Packets is enabled
7-6 —	Reserved
5 CNTPRSTLVL	Full-Half Preset When this bit is low and the CNTPRST bit is set, all MMC counters get preset to almost-half value. All octet counters get preset to 0x7FFF_F800 (Half 2KBytes) and all packet-counters gets preset to 0x7FFF_FFF0 (Half 16). When this bit is high and the CNTPRST bit is set, all MMC counters get preset to almost-full value. All octet counters get preset to 0xFFFF_F800 (Full 2KBytes) and all packet-counters gets preset to 0xFFFF_FFF0 (Full 16). For 16-bit counters, the almost-half preset values are 0x7800 and 0x7FF0 for the respective octet and packet counters. Similarly, the almost-full preset values for the 16-bit counters are 0xF800 and 0xFFF0. 0b - Full-Half Preset is disabled 1b - Full-Half Preset is enabled
4 CNTPRST	Counters Preset When this bit is set, all counters are initialized or preset to almost full or almost half according to the CNTPRSTLVL bit. This bit is cleared automatically after 1 clock cycle. This bit, along with the CNTPRSTLVL bit, is useful for debugging and testing the assertion of interrupts because of MMC counter becoming half-full or full. Access restriction applies. Self-cleared. Setting 0 clears. Setting 1 sets. **NOTE** This field is not testable by an automated register test. 0b - Counters Preset is disabled 1b - Counters Preset is enabled
3 CNTFREEZ	MMC Counter Freeze When this bit is set, it freezes all MMC counters to their current value. Until this bit is reset to 0, no MMC counter is updated because of any transmitted or received packet. If any MMC counter is read with the Reset on Read bit set, then that counter is also cleared in this mode. 0b - MMC Counter Freeze is disabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - MMC Counter Freeze is enabled
2 RSTONRD	Reset on Read When this bit is set, the MMC counters are reset to zero after Read (self-clearing after reset). The counters are cleared when the least significant byte lane (Bits[7:0]) is read. 0b - Reset on Read is disabled 1b - Reset on Read is enabled
1 CNTSTOPRO	Counter Stop Rollover When this bit is set, the counter does not roll over to zero after reaching the maximum value. 0b - Counter Stop Rollover is disabled 1b - Counter Stop Rollover is enabled
0 CNTRST	Counters Reset When this bit is set, all counters are reset. This bit is cleared automatically after 1 clock cycle. Access restriction applies. Self-cleared. Setting 0 clears. Setting 1 sets. **NOTE** This field is not testable by an automated register test. 0b - Counters are not reset 1b - All counters are reset

76.17.97 MMC Receive Interrupt (MMC_Rx_Interrupt)

Offset

Register	Offset
MMC_Rx_Interrupt	704h

Function

This register maintains the interrupts generated from all Receive statistics counters. The MMC Receive Interrupt register maintains the interrupts that are generated when the following occur: - Receive statistic counters reach half of their maximum values (0x8000_0000 for 32 bit counter and 0x8000 for 16 bit counter). - Receive statistic counters cross their maximum values (0xFFFF_FFFF for 32 bit counter and 0xFFFF for 16 bit counter). When the Counter Stop Rollover is set, interrupts are set but the counter remains at all-ones. The MMC Receive Interrupt register is a 32 bit register. An interrupt bit is cleared when the respective MMC counter that caused the interrupt is read. The least significant byte lane (Bits[7:0]) of the respective counter must be read to clear the interrupt bit. Note: R_SS_RC means that this register bit is set internally, and it is cleared when the Counter register is read.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved				Reserved	Reserved	RXCTRLP...	RXRCVER...	RXWDOGP...	RXVLANG...	RXFOVPIS	RXPAUSP...	RXORANG...	RXLENER...	RXUCGPIS	RX1024T...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RX512T1...	RX256T5...	RX128T2...	RX65T12...	RX64OCT...	RXOSIZE...	RXUSIZE...	RXJABER...	RXRUNTP...	RXALGNE...	RXCRCER...	RXMCGPIS	RXBCGPIS	RXGOCTIS	RXGBOCT...	RXGBPKT...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved
27 —	Reserved
26 —	Reserved
25 RXCTRLPIS	MMC Receive Control Packet Counter Interrupt Status This bit is set when the rxctrlpackets_g counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - MMC Receive Control Packet Counter Interrupt Status not detected 1b - MMC Receive Control Packet Counter Interrupt Status detected
24 RXRCVERRPIS	MMC Receive Error Packet Counter Interrupt Status This bit is set when the rxrcverror counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - MMC Receive Error Packet Counter Interrupt Status not detected 1b - MMC Receive Error Packet Counter Interrupt Status detected
23 RXWDOGPIS	MMC Receive Watchdog Error Packet Counter Interrupt Status This bit is set when the rxwatchdog error counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - MMC Receive Watchdog Error Packet Counter Interrupt Status not detected 1b - MMC Receive Watchdog Error Packet Counter Interrupt Status detected

Table continues on the next page...

Table continued from the previous page...

Field	Function
22 RXVLANGBPIS	MMC Receive VLAN Good Bad Packet Counter Interrupt Status This bit is set when the rxvlanpackets_gb counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - MMC Receive VLAN Good Bad Packet Counter Interrupt Status not detected 1b - MMC Receive VLAN Good Bad Packet Counter Interrupt Status detected
21 RXFOVPIS	MMC Receive FIFO Overflow Packet Counter Interrupt Status This bit is set when the rxfifooverflow counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - MMC Receive FIFO Overflow Packet Counter Interrupt Status not detected 1b - MMC Receive FIFO Overflow Packet Counter Interrupt Status detected
20 RXPAUSPIS	MMC Receive Pause Packet Counter Interrupt Status This bit is set when the rxpausepackets counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - MMC Receive Pause Packet Counter Interrupt Status not detected 1b - MMC Receive Pause Packet Counter Interrupt Status detected
19 RXORANGEPIS	MMC Receive Out Of Range Error Packet Counter Interrupt Status. This bit is set when the rxoutofrangetype counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - MMC Receive Out Of Range Error Packet Counter Interrupt Status not detected 1b - MMC Receive Out Of Range Error Packet Counter Interrupt Status detected
18 RXLENERPIS	MMC Receive Length Error Packet Counter Interrupt Status This bit is set when the rxlengtherror counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - MMC Receive Length Error Packet Counter Interrupt Status not detected 1b - MMC Receive Length Error Packet Counter Interrupt Status detected
17 RXUCGPIS	MMC Receive Unicast Good Packet Counter Interrupt Status This bit is set when the rxunicastpackets_g counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - MMC Receive Unicast Good Packet Counter Interrupt Status not detected 1b - MMC Receive Unicast Good Packet Counter Interrupt Status detected
16 RX1024TMAXOCTGBPIS	MMC Receive 1024 to Maximum Octet Good Bad Packet Counter Interrupt Status This bit is set when the rx1024tomaxoctets_gb counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - MMC Receive 1024 to Maximum Octet Good Bad Packet Counter Interrupt Status not detected 1b - MMC Receive 1024 to Maximum Octet Good Bad Packet Counter Interrupt Status detected
15 RX512T1023OCTGBPIS	MMC Receive 512 to 1023 Octet Good Bad Packet Counter Interrupt Status This bit is set when the rx512to1023octets_gb counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - MMC Receive 512 to 1023 Octet Good Bad Packet Counter Interrupt Status not detected 1b - MMC Receive 512 to 1023 Octet Good Bad Packet Counter Interrupt Status detected
14 RX256T511OCTGBPIS	MMC Receive 256 to 511 Octet Good Bad Packet Counter Interrupt Status This bit is set when the rx256to511octets_gb counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - MMC Receive 256 to 511 Octet Good Bad Packet Counter Interrupt Status not detected 1b - MMC Receive 256 to 511 Octet Good Bad Packet Counter Interrupt Status detected
13 RX128T255OCTGBPIS	MMC Receive 128 to 255 Octet Good Bad Packet Counter Interrupt Status This bit is set when the rx128to255octets_gb counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - MMC Receive 128 to 255 Octet Good Bad Packet Counter Interrupt Status not detected 1b - MMC Receive 128 to 255 Octet Good Bad Packet Counter Interrupt Status detected
12 RX65T127OCTGBPIS	MMC Receive 65 to 127 Octet Good Bad Packet Counter Interrupt Status This bit is set when the rx65to127octets_gb counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - MMC Receive 65 to 127 Octet Good Bad Packet Counter Interrupt Status not detected 1b - MMC Receive 65 to 127 Octet Good Bad Packet Counter Interrupt Status detected
11 RX64OCTGBPIS	MMC Receive 64 Octet Good Bad Packet Counter Interrupt Status This bit is set when the rx64octets_gb counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - MMC Receive 64 Octet Good Bad Packet Counter Interrupt Status not detected 1b - MMC Receive 64 Octet Good Bad Packet Counter Interrupt Status detected
10 RXOSIZEGPIS	MMC Receive Oversize Good Packet Counter Interrupt Status This bit is set when the rxoversize_g counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - MMC Receive Oversize Good Packet Counter Interrupt Status not detected 1b - MMC Receive Oversize Good Packet Counter Interrupt Status detected

Table continues on the next page...

Table continued from the previous page...

Field	Function
9 RXUSIZEGPIS	MMC Receive Undersize Good Packet Counter Interrupt Status This bit is set when the rxundersize_g counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - MMC Receive Undersize Good Packet Counter Interrupt Status not detected 1b - MMC Receive Undersize Good Packet Counter Interrupt Status detected
8 RXJABERPIS	MMC Receive Jabber Error Packet Counter Interrupt Status This bit is set when the rxjabbererror counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - MMC Receive Jabber Error Packet Counter Interrupt Status not detected 1b - MMC Receive Jabber Error Packet Counter Interrupt Status detected
7 RXRUNTPIS	MMC Receive Runt Packet Counter Interrupt Status This bit is set when the rxrunterror counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - MMC Receive Runt Packet Counter Interrupt Status not detected 1b - MMC Receive Runt Packet Counter Interrupt Status detected
6 RXALGNERPIS	MMC Receive Alignment Error Packet Counter Interrupt Status This bit is set when the rxalignmenterror counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - MMC Receive Alignment Error Packet Counter Interrupt Status not detected 1b - MMC Receive Alignment Error Packet Counter Interrupt Status detected
5 RXCRCERPIS	MMC Receive CRC Error Packet Counter Interrupt Status This bit is set when the rxcrcerror counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - MMC Receive CRC Error Packet Counter Interrupt Status not detected 1b - MMC Receive CRC Error Packet Counter Interrupt Status detected
4 RXMCGPIS	MMC Receive Multicast Good Packet Counter Interrupt Status This bit is set when the rxmulticastpackets_g counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - MMC Receive Multicast Good Packet Counter Interrupt Status not detected 1b - MMC Receive Multicast Good Packet Counter Interrupt Status detected
3 RXBCGPIS	MMC Receive Broadcast Good Packet Counter Interrupt Status This bit is set when the rxbroadcastpackets_g counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - MMC Receive Broadcast Good Packet Counter Interrupt Status not detected 1b - MMC Receive Broadcast Good Packet Counter Interrupt Status detected
2 RXGOCTIS	MMC Receive Good Octet Counter Interrupt Status This bit is set when the rxoctetcount_g counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - MMC Receive Good Octet Counter Interrupt Status not detected 1b - MMC Receive Good Octet Counter Interrupt Status detected
1 RXGBOCTIS	MMC Receive Good Bad Octet Counter Interrupt Status This bit is set when the rxoctetcount_gb counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - MMC Receive Good Bad Octet Counter Interrupt Status not detected 1b - MMC Receive Good Bad Octet Counter Interrupt Status detected
0 RXGBPKTIS	MMC Receive Good Bad Packet Counter Interrupt Status This bit is set when the rxpacketcount_gb counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - MMC Receive Good Bad Packet Counter Interrupt Status not detected 1b - MMC Receive Good Bad Packet Counter Interrupt Status detected

76.17.98 MMC Transmit Interrupt (MMC_Tx_Interrupt)

Offset

Register	Offset
MMC_Tx_Interrupt	708h

Function

This register maintains the interrupts generated from all Transmit statistics counters. The MMC Transmit Interrupt register maintains the interrupts generated when transmit statistic counters reach half their maximum values (0x8000_0000 for 32 bit counter and 0x8000 for 16 bit counter), and when they cross their maximum values (0xFFFF_FFFF for 32-bit counter and 0xFFFF for 16-bit counter). When Counter Stop Rollover is set, the interrupts are set but the counter remains at all-ones. The MMC Transmit Interrupt register is a 32 bit register. An interrupt bit is cleared when the respective MMC counter that caused the interrupt is read. The least significant byte lane (Bits[7:0]) of the respective counter must be read to clear the interrupt bit.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved				Reserved	Reserved	TXOSIZE...	TXVLANG...	TXPAUSP...	TXEXDEF...	TXGPKTIS	TXGOCTIS	TXCARER...	TXEXCOL...	TXLATCO...	TXDEFPIS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TXMCOLG...	TXSCOLG...	TXUFLOW...	TXBCGBP...	TXMCGBP...	TXUCGBP...	TX1024T...	TX512T1...	TX256T5...	TX128T2...	TX65T12...	TX64OCT...	TXMCGPIS	TXBCGPIS	TXGBPKT...	TXGBOCT...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved
27 —	Reserved
26 —	Reserved
25 TXOSIZEGPIS	MMC Transmit Oversize Good Packet Counter Interrupt Status This bit is set when the txoversize_g counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - MMC Transmit Oversize Good Packet Counter Interrupt Status not detected 1b - MMC Transmit Oversize Good Packet Counter Interrupt Status detected
24 TXVLANGPIS	MMC Transmit VLAN Good Packet Counter Interrupt Status This bit is set when the txvlanpackets_g counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - MMC Transmit VLAN Good Packet Counter Interrupt Status not detected 1b - MMC Transmit VLAN Good Packet Counter Interrupt Status detected
23 TXPAUSPIS	MMC Transmit Pause Packet Counter Interrupt Status This bit is set when the txpausepacketserror counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - MMC Transmit Pause Packet Counter Interrupt Status not detected 1b - MMC Transmit Pause Packet Counter Interrupt Status detected

Table continues on the next page...

Table continued from the previous page...

Field	Function
22 TXEXDEFPIS	MMC Transmit Excessive Deferral Packet Counter Interrupt Status This bit is set when the txexcessdef counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - MMC Transmit Excessive Deferral Packet Counter Interrupt Status not detected 1b - MMC Transmit Excessive Deferral Packet Counter Interrupt Status detected
21 TXGPKTIS	MMC Transmit Good Packet Counter Interrupt Status This bit is set when the txpacketcount_g counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - MMC Transmit Good Packet Counter Interrupt Status not detected 1b - MMC Transmit Good Packet Counter Interrupt Status detected
20 TXGOCTIS	MMC Transmit Good Octet Counter Interrupt Status This bit is set when the txoctetcount_g counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - MMC Transmit Good Octet Counter Interrupt Status not detected 1b - MMC Transmit Good Octet Counter Interrupt Status detected
19 TXCARERPIS	MMC Transmit Carrier Error Packet Counter Interrupt Status This bit is set when the txcarriererror counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - MMC Transmit Carrier Error Packet Counter Interrupt Status not detected 1b - MMC Transmit Carrier Error Packet Counter Interrupt Status detected
18 TXEXCOLPIS	MMC Transmit Excessive Collision Packet Counter Interrupt Status This bit is set when the txexesscol counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - MMC Transmit Excessive Collision Packet Counter Interrupt Status not detected 1b - MMC Transmit Excessive Collision Packet Counter Interrupt Status detected
17 TXLATCOLPIS	MMC Transmit Late Collision Packet Counter Interrupt Status This bit is set when the txlatecol counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - MMC Transmit Late Collision Packet Counter Interrupt Status not detected 1b - MMC Transmit Late Collision Packet Counter Interrupt Status detected
16 TXDEFPIS	MMC Transmit Deferred Packet Counter Interrupt Status This bit is set when the txdeferred counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - MMC Transmit Deferred Packet Counter Interrupt Status not detected 1b - MMC Transmit Deferred Packet Counter Interrupt Status detected
15 TXMCOLGPIS	MMC Transmit Multiple Collision Good Packet Counter Interrupt Status This bit is set when the txmulticol_g counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - MMC Transmit Multiple Collision Good Packet Counter Interrupt Status not detected 1b - MMC Transmit Multiple Collision Good Packet Counter Interrupt Status detected
14 TXSCOLGPIS	MMC Transmit Single Collision Good Packet Counter Interrupt Status This bit is set when the txsinglecol_g counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - MMC Transmit Single Collision Good Packet Counter Interrupt Status not detected 1b - MMC Transmit Single Collision Good Packet Counter Interrupt Status detected
13 TXUFLOWERPIS	MMC Transmit Underflow Error Packet Counter Interrupt Status This bit is set when the txunderflowerror counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - MMC Transmit Underflow Error Packet Counter Interrupt Status not detected 1b - MMC Transmit Underflow Error Packet Counter Interrupt Status detected
12 TXBCGBPIS	MMC Transmit Broadcast Good Bad Packet Counter Interrupt Status This bit is set when the txbroadcastpackets_gb counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - MMC Transmit Broadcast Good Bad Packet Counter Interrupt Status not detected 1b - MMC Transmit Broadcast Good Bad Packet Counter Interrupt Status detected
11 TXMCGBPIS	MMC Transmit Multicast Good Bad Packet Counter Interrupt Status The bit is set when the txmulticastpackets_gb counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - MMC Transmit Multicast Good Bad Packet Counter Interrupt Status not detected 1b - MMC Transmit Multicast Good Bad Packet Counter Interrupt Status detected
10 TXUCGBPIS	MMC Transmit Unicast Good Bad Packet Counter Interrupt Status This bit is set when the txunicastpackets_gb counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - MMC Transmit Unicast Good Bad Packet Counter Interrupt Status not detected 1b - MMC Transmit Unicast Good Bad Packet Counter Interrupt Status detected
9	MMC Transmit 1024 to Maximum Octet Good Bad Packet Counter Interrupt Status

Table continues on the next page...

Table continued from the previous page...

Field	Function
TX1024TMAXOCTGBPIS	This bit is set when the tx1024tomaxoctets_gb counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - MMC Transmit 1024 to Maximum Octet Good Bad Packet Counter Interrupt Status not detected 1b - MMC Transmit 1024 to Maximum Octet Good Bad Packet Counter Interrupt Status detected
8 TX512T1023OCTGBPIS	MMC Transmit 512 to 1023 Octet Good Bad Packet Counter Interrupt Status This bit is set when the tx512to1023octets_gb counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - MMC Transmit 512 to 1023 Octet Good Bad Packet Counter Interrupt Status not detected 1b - MMC Transmit 512 to 1023 Octet Good Bad Packet Counter Interrupt Status detected
7 TX256T511OCTGBPIS	MMC Transmit 256 to 511 Octet Good Bad Packet Counter Interrupt Status This bit is set when the tx256to511octets_gb counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - MMC Transmit 256 to 511 Octet Good Bad Packet Counter Interrupt Status not detected 1b - MMC Transmit 256 to 511 Octet Good Bad Packet Counter Interrupt Status detected
6 TX128T255OCTGBPIS	MMC Transmit 128 to 255 Octet Good Bad Packet Counter Interrupt Status This bit is set when the tx128to255octets_gb counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - MMC Transmit 128 to 255 Octet Good Bad Packet Counter Interrupt Status not detected 1b - MMC Transmit 128 to 255 Octet Good Bad Packet Counter Interrupt Status detected
5 TX65T127OCTGBPIS	MMC Transmit 65 to 127 Octet Good Bad Packet Counter Interrupt Status This bit is set when the tx65to127octets_gb counter reaches half the maximum value, and also when it reaches the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - MMC Transmit 65 to 127 Octet Good Bad Packet Counter Interrupt Status not detected 1b - MMC Transmit 65 to 127 Octet Good Bad Packet Counter Interrupt Status detected
4 TX64OCTGBPIS	MMC Transmit 64 Octet Good Bad Packet Counter Interrupt Status This bit is set when the tx64octets_gb counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - MMC Transmit 64 Octet Good Bad Packet Counter Interrupt Status not detected 1b - MMC Transmit 64 Octet Good Bad Packet Counter Interrupt Status detected
3 TXMCGPIS	MMC Transmit Multicast Good Packet Counter Interrupt Status This bit is set when the txmulticastpackets_g counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - MMC Transmit Multicast Good Packet Counter Interrupt Status not detected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - MMC Transmit Multicast Good Packet Counter Interrupt Status detected
2 TXBCGPIS	MMC Transmit Broadcast Good Packet Counter Interrupt Status This bit is set when the txbroadcastpackets_g counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - MMC Transmit Broadcast Good Packet Counter Interrupt Status not detected 1b - MMC Transmit Broadcast Good Packet Counter Interrupt Status detected
1 TXGBPKTIS	MMC Transmit Good Bad Packet Counter Interrupt Status This bit is set when the txpacketcount_gb counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - MMC Transmit Good Bad Packet Counter Interrupt Status not detected 1b - MMC Transmit Good Bad Packet Counter Interrupt Status detected
0 TXGBOCTIS	MMC Transmit Good Bad Octet Counter Interrupt Status This bit is set when the txoctetcount_gb counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - MMC Transmit Good Bad Octet Counter Interrupt Status not detected 1b - MMC Transmit Good Bad Octet Counter Interrupt Status detected

76.17.99 MMC Receive Interrupt Mask (MMC_Rx_Interrupt_Mask)

Offset

Register	Offset
MMC_Rx_Interrupt_Mask	70Ch

Function
This register maintains the masks for interrupts generated from all Receive statistics counters. The MMC Receive Interrupt Mask register maintains the masks for the interrupts generated when receive statistic counters reach half of their maximum value or the maximum values. This register is 32 bit wide.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved				Reserved	Reserved	RXCTRLP...	RXRCVER...	RXWDOGP...	RXVLANG...	RXFOVPIM	RXPAUSP...	RXORANG...	RXLENER...	RXUCGPIM	RX1024T...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RX512T1...	RX256T5...	RX128T2...	RX65T12...	RX64OCT...	RXOSIZE...	RXUSIZE...	RXJABER...	RXRUNTP...	RXALGNE...	RXCRCER...	RXMCGPIM	RXBCGPIM	RXGOCTIM	RXGBOCT...	RXGBPKT...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved
27 —	Reserved
26 —	Reserved
25 RXCTRLPIM	MMC Receive Control Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxctrlpackets_g counter reaches half of the maximum value or the maximum value. 0b - MMC Receive Control Packet Counter Interrupt Mask is disabled 1b - MMC Receive Control Packet Counter Interrupt Mask is enabled
24 RXRCVERRPIM	MMC Receive Error Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxrcverror counter reaches half of the maximum value or the maximum value. 0b - MMC Receive Error Packet Counter Interrupt Mask is disabled 1b - MMC Receive Error Packet Counter Interrupt Mask is enabled
23 RXWDOGPIM	MMC Receive Watchdog Error Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxwatchdog counter reaches half of the maximum value or the maximum value. 0b - MMC Receive Watchdog Error Packet Counter Interrupt Mask is disabled 1b - MMC Receive Watchdog Error Packet Counter Interrupt Mask is enabled
22	MMC Receive VLAN Good Bad Packet Counter Interrupt Mask

Table continues on the next page...

Table continued from the previous page...

Field	Function
RXVLANGBPIM	Setting this bit masks the interrupt when the rxvlanpackets_gb counter reaches half of the maximum value or the maximum value. 0b - MMC Receive VLAN Good Bad Packet Counter Interrupt Mask is disabled 1b - MMC Receive VLAN Good Bad Packet Counter Interrupt Mask is enabled
21 RXFOVPIM	MMC Receive FIFO Overflow Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxfifooverflow counter reaches half of the maximum value or the maximum value. 0b - MMC Receive FIFO Overflow Packet Counter Interrupt Mask is disabled 1b - MMC Receive FIFO Overflow Packet Counter Interrupt Mask is enabled
20 RXPAUSPIM	MMC Receive Pause Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxpausepackets counter reaches half of the maximum value or the maximum value. 0b - MMC Receive Pause Packet Counter Interrupt Mask is disabled 1b - MMC Receive Pause Packet Counter Interrupt Mask is enabled
19 RXORANGEPIM	MMC Receive Out Of Range Error Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxoutofrangetype counter reaches half of the maximum value or the maximum value. 0b - MMC Receive Out Of Range Error Packet Counter Interrupt Mask is disabled 1b - MMC Receive Out Of Range Error Packet Counter Interrupt Mask is enabled
18 RXLENERPIM	MMC Receive Length Error Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxlengtherror counter reaches half of the maximum value or the maximum value. 0b - MMC Receive Length Error Packet Counter Interrupt Mask is disabled 1b - MMC Receive Length Error Packet Counter Interrupt Mask is enabled
17 RXUCGPIM	MMC Receive Unicast Good Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxunicastpackets_g counter reaches half of the maximum value or the maximum value. 0b - MMC Receive Unicast Good Packet Counter Interrupt Mask is disabled 1b - MMC Receive Unicast Good Packet Counter Interrupt Mask is enabled
16 RX1024TMAXOCTGBPIM	MMC Receive 1024 to Maximum Octet Good Bad Packet Counter Interrupt Mask. Setting this bit masks the interrupt when the rx1024tomaxoctets_gb counter reaches half of the maximum value or the maximum value. 0b - MMC Receive 1024 to Maximum Octet Good Bad Packet Counter Interrupt Mask is disabled 1b - MMC Receive 1024 to Maximum Octet Good Bad Packet Counter Interrupt Mask is enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
15 RX512T1023OCTGBPIM	MMC Receive 512 to 1023 Octet Good Bad Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rx512to1023octets_gb counter reaches half of the maximum value or the maximum value. 0b - MMC Receive 512 to 1023 Octet Good Bad Packet Counter Interrupt Mask is disabled 1b - MMC Receive 512 to 1023 Octet Good Bad Packet Counter Interrupt Mask is enabled
14 RX256T511OCTGBPIM	MMC Receive 256 to 511 Octet Good Bad Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rx256to511octets_gb counter reaches half of the maximum value or the maximum value. 0b - MMC Receive 256 to 511 Octet Good Bad Packet Counter Interrupt Mask is disabled 1b - MMC Receive 256 to 511 Octet Good Bad Packet Counter Interrupt Mask is enabled
13 RX128T255OCTGBPIM	MMC Receive 128 to 255 Octet Good Bad Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rx128to255octets_gb counter reaches half of the maximum value or the maximum value. 0b - MMC Receive 128 to 255 Octet Good Bad Packet Counter Interrupt Mask is disabled 1b - MMC Receive 128 to 255 Octet Good Bad Packet Counter Interrupt Mask is enabled
12 RX65T127OCTGBPIM	MMC Receive 65 to 127 Octet Good Bad Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rx65to127octets_gb counter reaches half of the maximum value or the maximum value. 0b - MMC Receive 65 to 127 Octet Good Bad Packet Counter Interrupt Mask is disabled 1b - MMC Receive 65 to 127 Octet Good Bad Packet Counter Interrupt Mask is enabled
11 RX64OCTGBPIM	MMC Receive 64 Octet Good Bad Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rx64octets_gb counter reaches half of the maximum value or the maximum value. 0b - MMC Receive 64 Octet Good Bad Packet Counter Interrupt Mask is disabled 1b - MMC Receive 64 Octet Good Bad Packet Counter Interrupt Mask is enabled
10 RXOSIZEGPIM	MMC Receive Oversize Good Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxoversize_g counter reaches half of the maximum value or the maximum value. 0b - MMC Receive Oversize Good Packet Counter Interrupt Mask is disabled 1b - MMC Receive Oversize Good Packet Counter Interrupt Mask is enabled
9 RXUSIZEGPIM	MMC Receive Undersize Good Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxundersize_g counter reaches half of the maximum value or the maximum value.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - MMC Receive Undersize Good Packet Counter Interrupt Mask is disabled 1b - MMC Receive Undersize Good Packet Counter Interrupt Mask is enabled
8 RXJABERPIM	MMC Receive Jabber Error Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxjabbererror counter reaches half of the maximum value or the maximum value. 0b - MMC Receive Jabber Error Packet Counter Interrupt Mask is disabled 1b - MMC Receive Jabber Error Packet Counter Interrupt Mask is enabled
7 RXRUNTPIM	MMC Receive Runt Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxrunterror counter reaches half of the maximum value or the maximum value. 0b - MMC Receive Runt Packet Counter Interrupt Mask is disabled 1b - MMC Receive Runt Packet Counter Interrupt Mask is enabled
6 RXALGNERPIM	MMC Receive Alignment Error Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxalignmenterror counter reaches half of the maximum value or the maximum value. 0b - MMC Receive Alignment Error Packet Counter Interrupt Mask is disabled 1b - MMC Receive Alignment Error Packet Counter Interrupt Mask is enabled
5 RXCRCERPIM	MMC Receive CRC Error Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxcrcerror counter reaches half of the maximum value or the maximum value. 0b - MMC Receive CRC Error Packet Counter Interrupt Mask is disabled 1b - MMC Receive CRC Error Packet Counter Interrupt Mask is enabled
4 RXMCGPIM	MMC Receive Multicast Good Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxmulticastpackets_g counter reaches half of the maximum value or the maximum value. 0b - MMC Receive Multicast Good Packet Counter Interrupt Mask is disabled 1b - MMC Receive Multicast Good Packet Counter Interrupt Mask is enabled
3 RXBCGPIM	MMC Receive Broadcast Good Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxbroadcastpackets_g counter reaches half of the maximum value or the maximum value. 0b - MMC Receive Broadcast Good Packet Counter Interrupt Mask is disabled 1b - MMC Receive Broadcast Good Packet Counter Interrupt Mask is enabled
2	MMC Receive Good Octet Counter Interrupt Mask

Table continues on the next page...

Table continued from the previous page...

Field	Function
RXGOCTIM	Setting this bit masks the interrupt when the rxoctetcount_g counter reaches half of the maximum value or the maximum value. 0b - MMC Receive Good Octet Counter Interrupt Mask is disabled 1b - MMC Receive Good Octet Counter Interrupt Mask is enabled
1 RXGBOCTIM	MMC Receive Good Bad Octet Counter Interrupt Mask Setting this bit masks the interrupt when the rxoctetcount_gb counter reaches half of the maximum value or the maximum value. 0b - MMC Receive Good Bad Octet Counter Interrupt Mask is disabled 1b - MMC Receive Good Bad Octet Counter Interrupt Mask is enabled
0 RXGBPKTIM	MMC Receive Good Bad Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxpacketcount_gb counter reaches half of the maximum value or the maximum value. 0b - MMC Receive Good Bad Packet Counter Interrupt Mask is disabled 1b - MMC Receive Good Bad Packet Counter Interrupt Mask is enabled

76.17.100 MMC Transmit Interrupt Mask (MMC_Tx_Interrupt_Mask)

Offset

Register	Offset
MMC_Tx_Interrupt_Mask	710h

Function

This register maintains the masks for interrupts generated from all Transmit statistics counters. The MMC Transmit Interrupt Mask register maintains the masks for the interrupts generated when the transmit statistic counters reach half of their maximum value or the maximum values. This register is 32 bit wide.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved				Reserved	Reserved	TXOSIZE...	TXVLANG...	TXPAUSP...	TXEXDEF...	TXGPKTIM	TXGOCTIM	TXCARER...	TXEXCOL...	TXLATCO...	TXDEFPIM
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TXMCOLG...	TXSCOLG...	TXUFLOW...	TXBCGBP...	TXMCGBP...	TXUCGBP...	TX1024T...	TX512T1...	TX256T5...	TX128T2...	TX65T12...	TX64OCT...	TXMCGPIM	TXBCGPIM	TXGBPKT...	TXGBOCT...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved
27 —	Reserved
26 —	Reserved
25 TXOSIZEGPIM	MMC Transmit Oversize Good Packet Counter Interrupt Mask Setting this bit masks the interrupt when the txoversize_g counter reaches half of the maximum value or the maximum value. 0b - MMC Transmit Oversize Good Packet Counter Interrupt Mask is disabled 1b - MMC Transmit Oversize Good Packet Counter Interrupt Mask is enabled
24 TXVLANGPIM	MMC Transmit VLAN Good Packet Counter Interrupt Mask Setting this bit masks the interrupt when the txvlanpackets_g counter reaches half of the maximum value or the maximum value. 0b - MMC Transmit VLAN Good Packet Counter Interrupt Mask is disabled 1b - MMC Transmit VLAN Good Packet Counter Interrupt Mask is enabled
23 TXPAUSPIM	MMC Transmit Pause Packet Counter Interrupt Mask Setting this bit masks the interrupt when the txpausepackets counter reaches half of the maximum value or the maximum value. 0b - MMC Transmit Pause Packet Counter Interrupt Mask is disabled 1b - MMC Transmit Pause Packet Counter Interrupt Mask is enabled
22 TXEXDEFPIM	MMC Transmit Excessive Deferral Packet Counter Interrupt Mask Setting this bit masks the interrupt when the txexcessdef counter reaches half of the maximum value or the maximum value. 0b - MMC Transmit Excessive Deferral Packet Counter Interrupt Mask is disabled 1b - MMC Transmit Excessive Deferral Packet Counter Interrupt Mask is enabled
21 TXGPKTIM	MMC Transmit Good Packet Counter Interrupt Mask Setting this bit masks the interrupt when the txpacketcount_g counter reaches half of the maximum value or the maximum value. 0b - MMC Transmit Good Packet Counter Interrupt Mask is disabled 1b - MMC Transmit Good Packet Counter Interrupt Mask is enabled
20	MMC Transmit Good Octet Counter Interrupt Mask

Table continues on the next page...

Table continued from the previous page...

Field	Function
TXGOCTIM	Setting this bit masks the interrupt when the txoctetcount_g counter reaches half of the maximum value or the maximum value. 0b - MMC Transmit Good Octet Counter Interrupt Mask is disabled 1b - MMC Transmit Good Octet Counter Interrupt Mask is enabled
19 TXCARERPIM	MMC Transmit Carrier Error Packet Counter Interrupt Mask Setting this bit masks the interrupt when the txcarriererror counter reaches half of the maximum value or the maximum value. 0b - MMC Transmit Carrier Error Packet Counter Interrupt Mask is disabled 1b - MMC Transmit Carrier Error Packet Counter Interrupt Mask is enabled
18 TXEXCOLPIM	MMC Transmit Excessive Collision Packet Counter Interrupt Mask Setting this bit masks the interrupt when the txexcesscol counter reaches half of the maximum value or the maximum value. 0b - MMC Transmit Excessive Collision Packet Counter Interrupt Mask is disabled 1b - MMC Transmit Excessive Collision Packet Counter Interrupt Mask is enabled
17 TXLATCOLPIM	MMC Transmit Late Collision Packet Counter Interrupt Mask Setting this bit masks the interrupt when the txlatecol counter reaches half of the maximum value or the maximum value. 0b - MMC Transmit Late Collision Packet Counter Interrupt Mask is disabled 1b - MMC Transmit Late Collision Packet Counter Interrupt Mask is enabled
16 TXDEFPIM	MMC Transmit Deferred Packet Counter Interrupt Mask Setting this bit masks the interrupt when the txdeferred counter reaches half of the maximum value or the maximum value. 0b - MMC Transmit Deferred Packet Counter Interrupt Mask is disabled 1b - MMC Transmit Deferred Packet Counter Interrupt Mask is enabled
15 TXMCOLGPIM	MMC Transmit Multiple Collision Good Packet Counter Interrupt Mask Setting this bit masks the interrupt when the txmulticol_g counter reaches half of the maximum value or the maximum value. 0b - MMC Transmit Multiple Collision Good Packet Counter Interrupt Mask is disabled 1b - MMC Transmit Multiple Collision Good Packet Counter Interrupt Mask is enabled
14 TXSCOLGPIM	MMC Transmit Single Collision Good Packet Counter Interrupt Mask Setting this bit masks the interrupt when the txsinglecol_g counter reaches half of the maximum value or the maximum value. 0b - MMC Transmit Single Collision Good Packet Counter Interrupt Mask is disabled 1b - MMC Transmit Single Collision Good Packet Counter Interrupt Mask is enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
13 TXUFLOWERPIM	MMC Transmit Underflow Error Packet Counter Interrupt Mask Setting this bit masks the interrupt when the txunderflowerror counter reaches half of the maximum value or the maximum value. 0b - MMC Transmit Underflow Error Packet Counter Interrupt Mask is disabled 1b - MMC Transmit Underflow Error Packet Counter Interrupt Mask is enabled
12 TXBCGBPIM	MMC Transmit Broadcast Good Bad Packet Counter Interrupt Mask Setting this bit masks the interrupt when the txbroadcastpackets_gb counter reaches half of the maximum value or the maximum value. 0b - MMC Transmit Broadcast Good Bad Packet Counter Interrupt Mask is disabled 1b - MMC Transmit Broadcast Good Bad Packet Counter Interrupt Mask is enabled
11 TXMCGBPIM	MMC Transmit Multicast Good Bad Packet Counter Interrupt Mask Setting this bit masks the interrupt when the txmulticastpackets_gb counter reaches half of the maximum value or the maximum value. 0b - MMC Transmit Multicast Good Bad Packet Counter Interrupt Mask is disabled 1b - MMC Transmit Multicast Good Bad Packet Counter Interrupt Mask is enabled
10 TXUCGBPIM	MMC Transmit Unicast Good Bad Packet Counter Interrupt Mask Setting this bit masks the interrupt when the txunicastpackets_gb counter reaches half of the maximum value or the maximum value. 0b - MMC Transmit Unicast Good Bad Packet Counter Interrupt Mask is disabled 1b - MMC Transmit Unicast Good Bad Packet Counter Interrupt Mask is enabled
9 TX1024TMAXOCTGBPIM	MMC Transmit 1024 to Maximum Octet Good Bad Packet Counter Interrupt Mask Setting this bit masks the interrupt when the tx1024tomaxoctets_gb counter reaches half of the maximum value or the maximum value. 0b - MMC Transmit 1024 to Maximum Octet Good Bad Packet Counter Interrupt Mask is disabled 1b - MMC Transmit 1024 to Maximum Octet Good Bad Packet Counter Interrupt Mask is enabled
8 TX512T1023OCTGBPIM	MMC Transmit 512 to 1023 Octet Good Bad Packet Counter Interrupt Mask Setting this bit masks the interrupt when the tx512to1023octets_gb counter reaches half of the maximum value or the maximum value. 0b - MMC Transmit 512 to 1023 Octet Good Bad Packet Counter Interrupt Mask is disabled 1b - MMC Transmit 512 to 1023 Octet Good Bad Packet Counter Interrupt Mask is enabled
7 TX256T511OCTGBPIM	MMC Transmit 256 to 511 Octet Good Bad Packet Counter Interrupt Mask Setting this bit masks the interrupt when the tx256to511octets_gb counter reaches half of the maximum value or the maximum value.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - MMC Transmit 256 to 511 Octet Good Bad Packet Counter Interrupt Mask is disabled 1b - MMC Transmit 256 to 511 Octet Good Bad Packet Counter Interrupt Mask is enabled
6 TX128T255OCTGBPIM	MMC Transmit 128 to 255 Octet Good Bad Packet Counter Interrupt Mask Setting this bit masks the interrupt when the tx128to255octets_gb counter reaches half of the maximum value or the maximum value. 0b - MMC Transmit 128 to 255 Octet Good Bad Packet Counter Interrupt Mask is disabled 1b - MMC Transmit 128 to 255 Octet Good Bad Packet Counter Interrupt Mask is enabled
5 TX65T127OCTGBPIM	MMC Transmit 65 to 127 Octet Good Bad Packet Counter Interrupt Mask Setting this bit masks the interrupt when the tx65to127octets_gb counter reaches half of the maximum value or the maximum value. 0b - MMC Transmit 65 to 127 Octet Good Bad Packet Counter Interrupt Mask is disabled 1b - MMC Transmit 65 to 127 Octet Good Bad Packet Counter Interrupt Mask is enabled
4 TX64OCTGBPIM	MMC Transmit 64 Octet Good Bad Packet Counter Interrupt Mask Setting this bit masks the interrupt when the tx64octets_gb counter reaches half of the maximum value or the maximum value. 0b - MMC Transmit 64 Octet Good Bad Packet Counter Interrupt Mask is disabled 1b - MMC Transmit 64 Octet Good Bad Packet Counter Interrupt Mask is enabled
3 TXMCGPIM	MMC Transmit Multicast Good Packet Counter Interrupt Mask Setting this bit masks the interrupt when the txmulticastpackets_g counter reaches half of the maximum value or the maximum value. 0b - MMC Transmit Multicast Good Packet Counter Interrupt Mask is disabled 1b - MMC Transmit Multicast Good Packet Counter Interrupt Mask is enabled
2 TXBCGPIM	MMC Transmit Broadcast Good Packet Counter Interrupt Mask Setting this bit masks the interrupt when the txbroadcastpackets_g counter reaches half of the maximum value or the maximum value. 0b - MMC Transmit Broadcast Good Packet Counter Interrupt Mask is disabled 1b - MMC Transmit Broadcast Good Packet Counter Interrupt Mask is enabled
1 TXGBPKTIM	MMC Transmit Good Bad Packet Counter Interrupt Mask Setting this bit masks the interrupt when the txpacketcount_gb counter reaches half of the maximum value or the maximum value. 0b - MMC Transmit Good Bad Packet Counter Interrupt Mask is disabled 1b - MMC Transmit Good Bad Packet Counter Interrupt Mask is enabled
0	MMC Transmit Good Bad Octet Counter Interrupt Mask

Table continues on the next page...

Table continued from the previous page...

Field	Function
TXGBOCTIM	Setting this bit masks the interrupt when the txoctetcount_gb counter reaches half of the maximum value or the maximum value. 0b - MMC Transmit Good Bad Octet Counter Interrupt Mask is disabled 1b - MMC Transmit Good Bad Octet Counter Interrupt Mask is enabled

76.17.101 Transmit Octet Count Good Bad (Tx_Octet_Count_Good_Bad)

Offset

Register	Offset
Tx_Octet_Count_Good_Bad	714h

Function

This register provides the number of bytes transmitted by the DWC_ether_qos, exclusive of preamble and retried bytes, in good and bad packets.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TXOCTGB															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TXOCTGB															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TXOCTGB	Tx Octet Count Good Bad This field indicates the number of bytes transmitted, exclusive of preamble and retried bytes, in good and bad packets.

76.17.102 Transmit Packet Count Good Bad (Tx_Packet_Count_Good_Bad)

Offset

Register	Offset
Tx_Packet_Count_Good_Bad	718h

Function

This register provides the number of good and bad packets transmitted by DWC_ether_qos, exclusive of retried packets.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TXPKTGB															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TXPKTGB															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TXPKTGB	Tx Packet Count Good Bad This field indicates the number of good and bad packets transmitted, exclusive of retried packets.

76.17.103 Transmit Broadcast Packets Good (Tx_Broadcast_Packets_Good)

Offset

Register	Offset
Tx_Broadcast_Packets_Good	71Ch

Function

This register provides the number of good broadcast packets transmitted by DWC_ether_qos.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TXBCASTG															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TXBCASTG															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TXBCASTG	Tx Broadcast Packets Good This field indicates the number of good broadcast packets transmitted.

76.17.104 Transmit Multicast Packets Good (Tx_Multicast_Packets_Good)

Offset

Register	Offset
Tx_Multicast_Packets_Good	720h

Function

This register provides the number of good multicast packets transmitted by DWC_ether_qos.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TXMCASTG															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TXMCASTG															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TXMCASTG	Tx Multicast Packets Good This field indicates the number of good multicast packets transmitted.

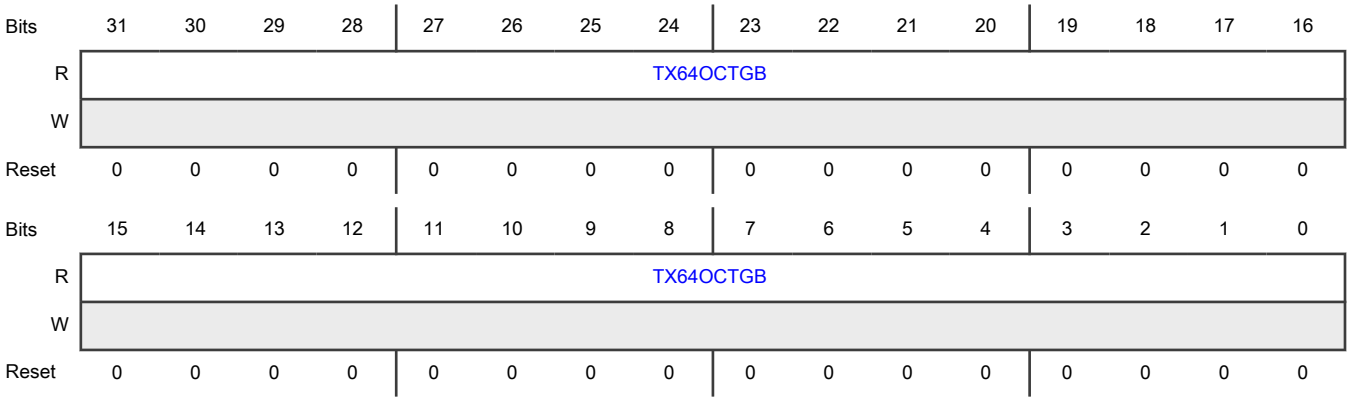
76.17.105 Transmit 64-Octet Packets Good Bad (Tx_64Octets_Packets_Good_Bad)

Offset

Register	Offset
Tx_64Octets_Packets_Good_Bad	724h

Function
This register provides the number of good and bad packets transmitted by DWC_ether_qos with length 64 bytes, exclusive of preamble and retried packets.

Diagram



Fields

Field	Function
31-0 TX64OCTGB	Tx 64Octets Packets Good_Bad This field indicates the number of good and bad packets transmitted with length 64 bytes, exclusive of preamble and retried packets.

76.17.106 Transmit 65 to 127 Octet Packets Good Bad (Tx_65To127Octets_Packets_Good_Bad)

Offset

Register	Offset
Tx_65To127Octets_Packets_Good_Bad	728h

Function

This register provides the number of good and bad packets transmitted by DWC_ether_qos with length between 65 and 127 (inclusive) bytes, exclusive of preamble and retried packets.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TX65_127OCTGB															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TX65_127OCTGB															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TX65_127OCTGB	Tx 65To127Octets Packets Good Bad This field indicates the number of good and bad packets transmitted with length between 65 and 127 (inclusive) bytes, exclusive of preamble and retried packets.

76.17.107 Transmit 128 to 255 Octet Packets Good Bad (Tx_128To255Octets_Packets_Good_Bad)

Offset

Register	Offset
Tx_128To255Octets_Packets_Good_Bad	72Ch

Function

This register provides the number of good and bad packets transmitted by DWC_ether_qos with length between 128 to 255 (inclusive) bytes, exclusive of preamble and retried packets.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TX128_255OCTGB															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TX128_255OCTGB															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TX128_255OCTGB	Tx 128To255Octets Packets Good Bad This field indicates the number of good and bad packets transmitted with length between 128 and 255 (inclusive) bytes, exclusive of preamble and retried packets.

76.17.108 Transmit 256 to 511 Octet Packets Good Bad (Tx_256To511Octets_Packets_Good_Bad)

Offset

Register	Offset
Tx_256To511Octets_Packets_Good_Bad	730h

Function

This register provides the number of good and bad packets transmitted by DWC_ether_qos with length between 256 to 511 (inclusive) bytes, exclusive of preamble and retried packets.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TX256_511OCTGB															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TX256_511OCTGB															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TX256_511OCTGB	Tx 256To511Octets Packets Good Bad This field indicates the number of good and bad packets transmitted with length between 256 and 511 (inclusive) bytes, exclusive of preamble and retried packets.

76.17.109 Transmit 512 to 1023 Octet Packets Good Bad (Tx_512To1023Octets_Packets_Good_Bad)

Offset

Register	Offset
Tx_512To1023Octets_Packets_Good_Bad	734h

Function
This register provides the number of good and bad packets transmitted by DWC_ether_qos with length 512 to 1023 (inclusive) bytes, exclusive of preamble and retried packets.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TX512_1023OCTGB															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TX512_1023OCTGB															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TX512_1023OCTGB	Tx 512To1023Octets Packets Good Bad This field indicates the number of good and bad packets transmitted with length between 512 and 1023 (inclusive) bytes, exclusive of preamble and retried packets.

76.17.110 Transmit 1024 to Max Octet Packets Good Bad (Tx_1024ToMaxOctets_Packets_Good_Bad)

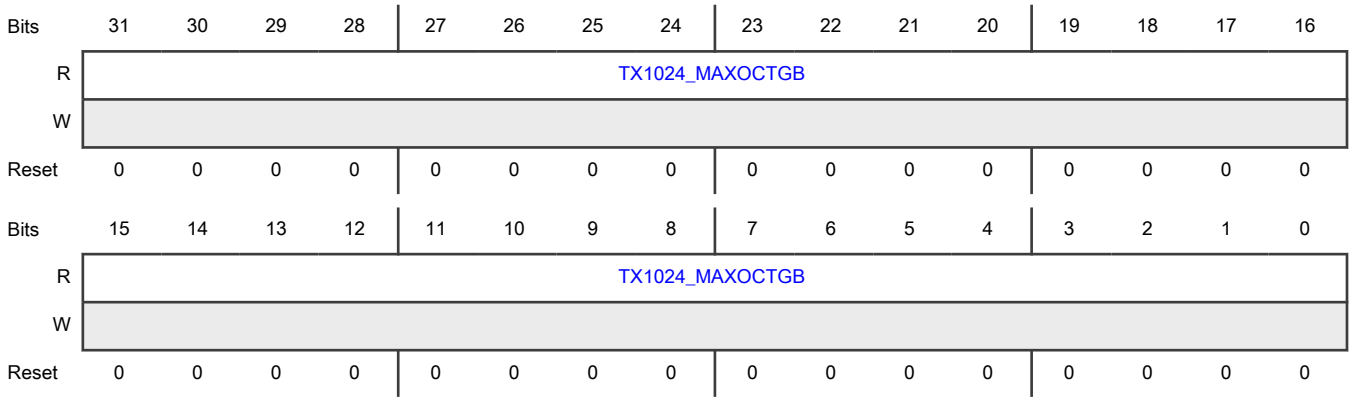
Offset

Register	Offset
Tx_1024ToMaxOctets_Packets_Good_Bad	738h

Function

This register provides the number of good and bad packets transmitted by DWC_ether_qos with length 1024 to maxsize (inclusive) bytes, exclusive of preamble and retried packets.

Diagram



Fields

Field	Function
31-0 TX1024_MAXOCTGB	Tx 1024ToMaxOctets Packets Good Bad This field indicates the number of good and bad packets transmitted with length between 1024 and maxsize (inclusive) bytes, exclusive of preamble and retried packets.

76.17.111 Transmit Unicast Packets Good Bad (Tx_Unicast_Packets_Good_Bad)

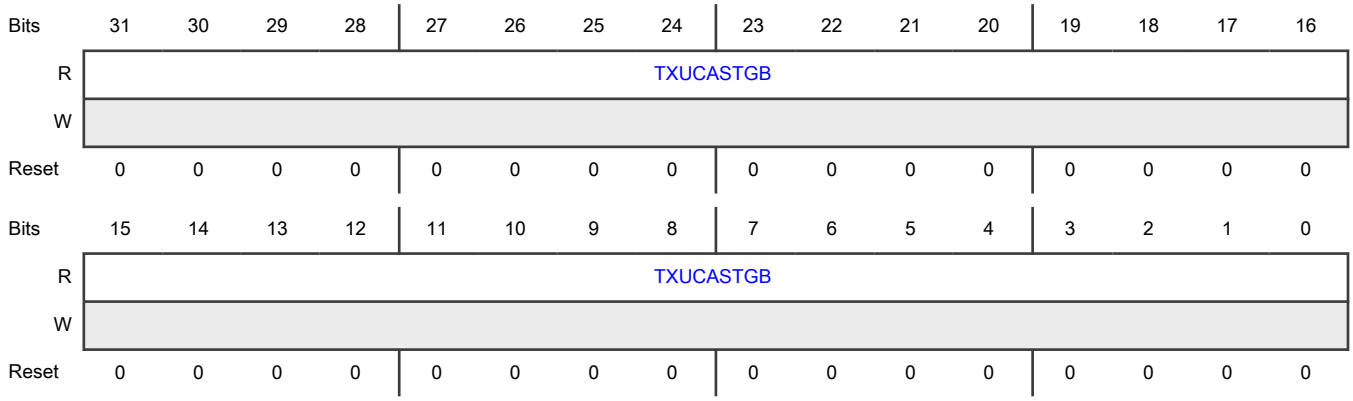
Offset

Register	Offset
Tx_Unicast_Packets_Good_Bad	73Ch

Function

This register provides the number of good and bad unicast packets transmitted by DWC_ether_qos.

Diagram



Fields

Field	Function
31-0 TXUCASTGB	Tx Unicast Packets Good Bad This field indicates the number of good and bad unicast packets transmitted.

76.17.112 Transmit Multicast Packets Good Bad (Tx_Multicast_Packets_Good_Bad)

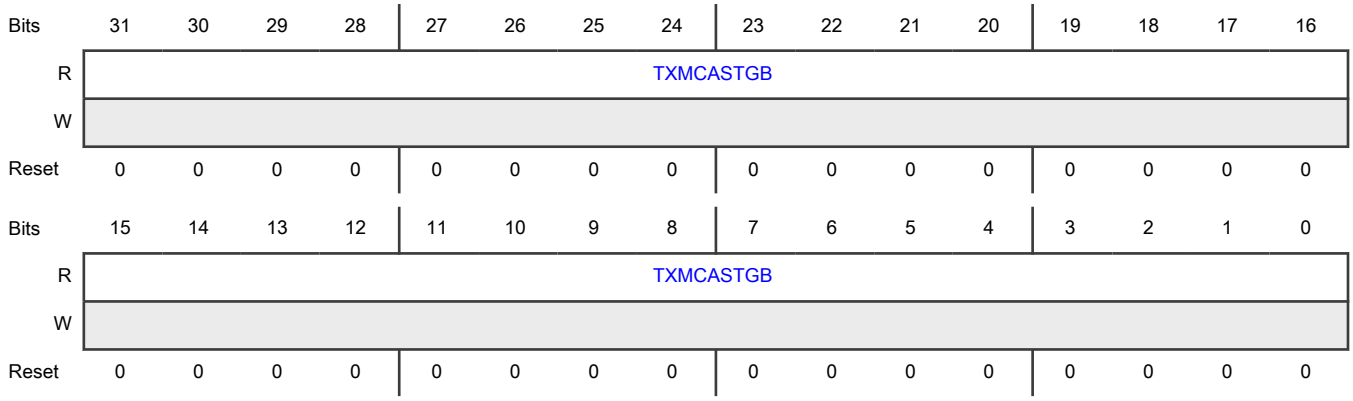
Offset

Register	Offset
Tx_Multicast_Packets_Good_Bad	740h

Function

This register provides the number of good and bad multicast packets transmitted by DWC_ether_qos.

Diagram



Fields

Field	Function
31-0 TXMCASTGB	Tx Multicast Packets Good Bad This field indicates the number of good and bad multicast packets transmitted.

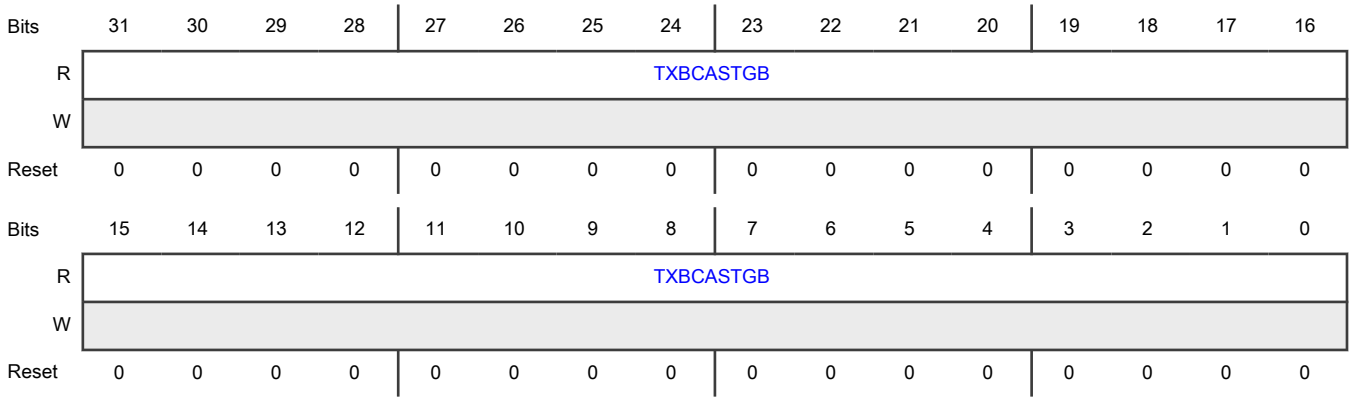
76.17.113 Transmit Broadcast Packets Good Bad (Tx_Broadcast_Packets_Good_Bad)

Offset

Register	Offset
Tx_Broadcast_Packets_Good_Bad	744h

Function
This register provides the number of good and bad broadcast packets transmitted by DWC_ether_qos.

Diagram



Fields

Field	Function
31-0 TXBCASTGB	Tx Broadcast Packets Good Bad This field indicates the number of good and bad broadcast packets transmitted.

76.17.114 Transmit Underflow Error Packets (Tx_Underflow_Error_Packets)

Offset

Register	Offset
Tx_Underflow_Error_Packets	748h

Function

This register provides the number of packets aborted by DWC_ether_qos because of packets underflow error.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TXUNDRFLW															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TXUNDRFLW															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TXUNDRFLW	Tx Underflow Error Packets This field indicates the number of packets aborted because of packets underflow error.

76.17.115 Transmit Single Collision Good Packets (Tx_Single_Collision_Good_Packets)

Offset

Register	Offset
Tx_Single_Collision_Good_Packets	74Ch

Function

This register provides the number of successfully transmitted packets by DWC_ether_qos after a single collision in the half-duplex mode.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TXSNGLCOLG															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TXSNGLCOLG															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TXSNGLCOLG	Tx Single Collision Good Packets This field indicates the number of successfully transmitted packets after a single collision in the half-duplex mode.

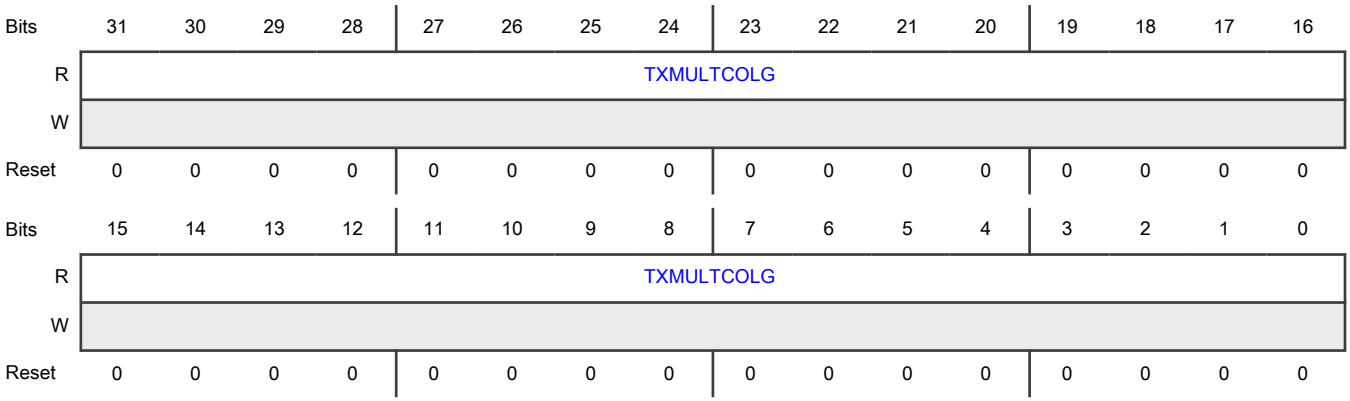
76.17.116 Transmit Multiple Collision Good Packets (Tx_Multiple_Collision_Good_Packets)

Offset

Register	Offset
Tx_Multiple_Collision_Good_Packets	750h

Function
This register provides the number of successfully transmitted packets by DWC_ether_qos after multiple collisions in the half-duplex mode.

Diagram



Fields

Field	Function
31-0 TXMULTCOLG	Tx Multiple Collision Good Packets This field indicates the number of successfully transmitted packets after multiple collisions in the half-duplex mode.

76.17.117 Transmit Deferred Packets (Tx_Deferred_Packets)

Offset

Register	Offset
Tx_Deferred_Packets	754h

Function
This register provides the number of successfully transmitted by DWC_ether_qos after a deferral in the half-duplex mode.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TXDEFRD															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TXDEFRD															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TXDEFRD	Tx Deferred Packets This field indicates the number of successfully transmitted after a deferral in the half-duplex mode.

76.17.118 Transmit Late Collision Packets (Tx_Late_Collision_Packets)

Offset

Register	Offset
Tx_Late_Collision_Packets	758h

Function
This register provides the number of packets aborted by DWC_ether_qos because of late collision error.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TXLATECOL															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TXLATECOL															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TXLATECOL	Tx Late Collision Packets This field indicates the number of packets aborted because of late collision error.

76.17.119 Transmit Excessive Collision Packets (Tx_Excessive_Collision_Packets)

Offset

Register	Offset
Tx_Excessive_Collision_Packets	75Ch

Function

This register provides the number of packets aborted by DWC_ether_qos because of excessive (16) collision errors.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TXEXSCOL															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TXEXSCOL															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TXEXSCOL	Tx Excessive Collision Packets This field indicates the number of packets aborted because of excessive (16) collision errors.

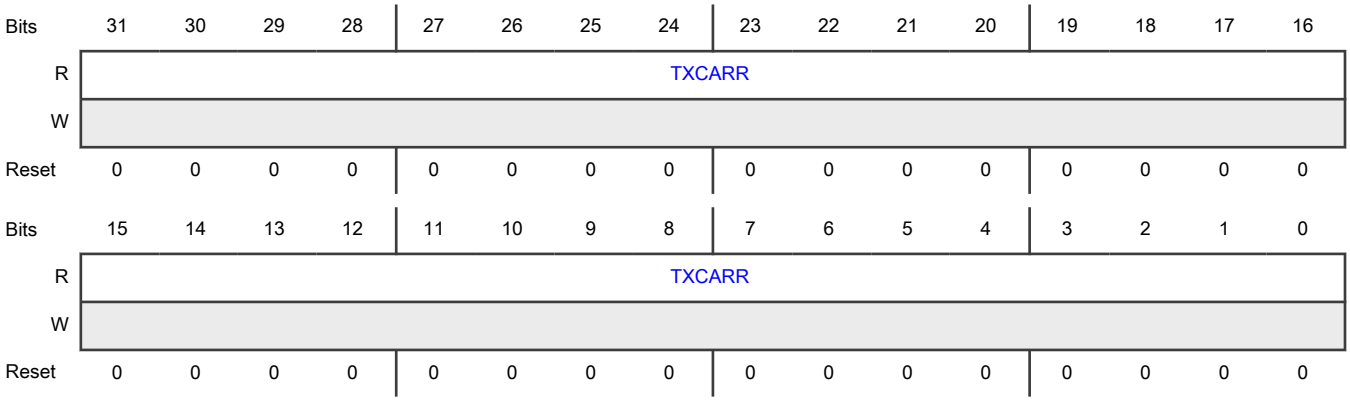
76.17.120 Transmit Carrier Error Packets (Tx_Carrier_Error_Packets)

Offset

Register	Offset
Tx_Carrier_Error_Packets	760h

Function
This register provides the number of packets aborted by DWC_ether_qos because of carrier sense error (no carrier or loss of carrier).

Diagram



Fields

Field	Function
31-0 TXCARR	Tx Carrier Error Packets This field indicates the number of packets aborted because of carrier sense error (no carrier or loss of carrier).

76.17.121 Transmit Octet Count Good (Tx_Octet_Count_Good)

Offset

Register	Offset
Tx_Octet_Count_Good	764h

Function

This register provides the number of bytes transmitted by DWC_ether_qos, exclusive of preamble, only in good packets.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TXOCTG															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TXOCTG															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TXOCTG	Tx Octet Count Good This field indicates the number of bytes transmitted, exclusive of preamble, only in good packets.

76.17.122 Transmit Packet Count Good (Tx_Packet_Count_Good)

Offset

Register	Offset
Tx_Packet_Count_Good	768h

Function

This register provides the number of good packets transmitted by DWC_ether_qos.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TXPKTG															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TXPKTG															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TXPKTG	Tx Packet Count Good This field indicates the number of good packets transmitted.

76.17.123 Transmit Excessive Deferral Error (Tx_Excessive_Deferral_Error)

Offset

Register	Offset
Tx_Excessive_Deferral_Error	76Ch

Function

This register provides the number of packets aborted by DWC_ether_qos because of excessive deferral error (deferred for more than two max-sized packet times).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TXEXSDEF															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TXEXSDEF															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TXEXSDEF	Tx Excessive Deferral Error This field indicates the number of packets aborted because of excessive deferral error (deferred for more than two max-sized packet times).

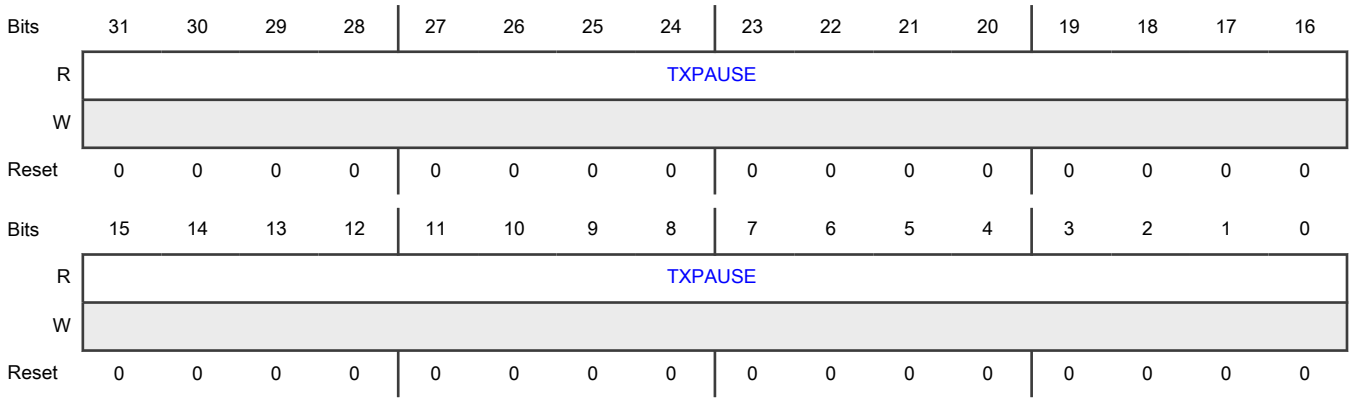
76.17.124 Transmit Pause Packets (Tx_Pause_Packets)

Offset

Register	Offset
Tx_Pause_Packets	770h

Function
This register provides the number of good Pause packets transmitted by DWC_ether_qos.

Diagram



Fields

Field	Function
31-0 TXPAUSE	Tx Pause Packets This field indicates the number of good Pause packets transmitted.

76.17.125 Transmit VLAN Packets Good (Tx_VLAN_Packets_Good)

Offset

Register	Offset
Tx_VLAN_Packets_Good	774h

Function

This register provides the number of good VLAN packets transmitted by DWC_ether_qos.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TXVLANG															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TXVLANG															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TXVLANG	Tx VLAN Packets Good This field provides the number of good VLAN packets transmitted.

76.17.126 Transmit O Size Packets Good (Tx_OSize_Packets_Good)

Offset

Register	Offset
Tx_OSize_Packets_Good	778h

Function

This register provides the number of packets transmitted by DWC_ether_qos without errors and with length greater than the maxsize (1,518 or 1,522 bytes for VLAN tagged packets; 2000 bytes if enabled in S2KP bit of the MAC_Configuration register).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TXOSIZG															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TXOSIZG															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TXOSIZG	Tx OSize Packets Good This field indicates the number of packets transmitted without errors and with length greater than the maxsize (1,518 or 1,522 bytes for VLAN tagged packets; 2000 bytes if enabled in S2KP bit of the MAC_Configuration register).

76.17.127 Receive Packets Count Good Bad (Rx_Packets_Count_Good_Bad)

Offset

Register	Offset
Rx_Packets_Count_Good_Bad	780h

Function

This register provides the number of good and bad packets received by DWC_ether_qos.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RXPKTGB															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXPKTGB															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 RXPKTGB	Rx Packets Count Good Bad This field indicates the number of good and bad packets received.

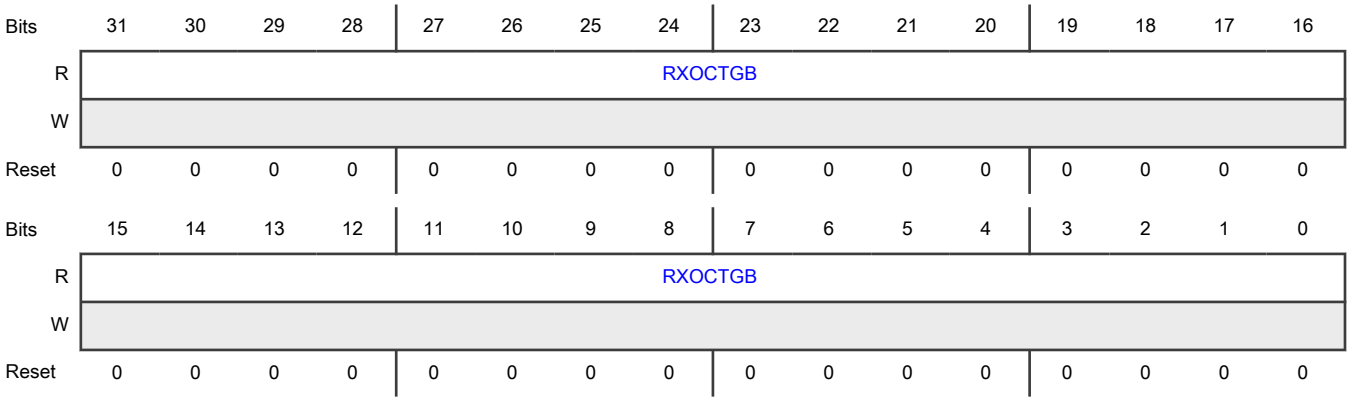
76.17.128 Receive Octet Count Good Bad (Rx_Octet_Count_Good_Bad)

Offset

Register	Offset
Rx_Octet_Count_Good_Bad	784h

Function
This register provides the number of bytes received by DWC_ther_qos, exclusive of preamble, in good and bad packets.

Diagram



Fields

Field	Function
31-0 RXOCTGB	Rx Octet Count Good Bad This field indicates the number of bytes received, exclusive of preamble, in good and bad packets.

76.17.129 Receive Octet Count Good (Rx_Octet_Count_Good)

Offset

Register	Offset
Rx_Octet_Count_Good	788h

Function

This register provides the number of bytes received by DWC_ether_qos, exclusive of preamble, only in good packets.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RXOCTG															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXOCTG															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 RXOCTG	Rx Octet Count Good This field indicates the number of bytes received, exclusive of preamble, only in good packets.

76.17.130 Receive Broadcast Packets Good (Rx_Broadcast_Packets_Good)

Offset

Register	Offset
Rx_Broadcast_Packets_Good	78Ch

Function

This register provides the number of good broadcast packets received by DWC_ether_qos.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RXBCASTG															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXBCASTG															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 RXBCASTG	Rx Broadcast Packets Good This field indicates the number of good broadcast packets received.

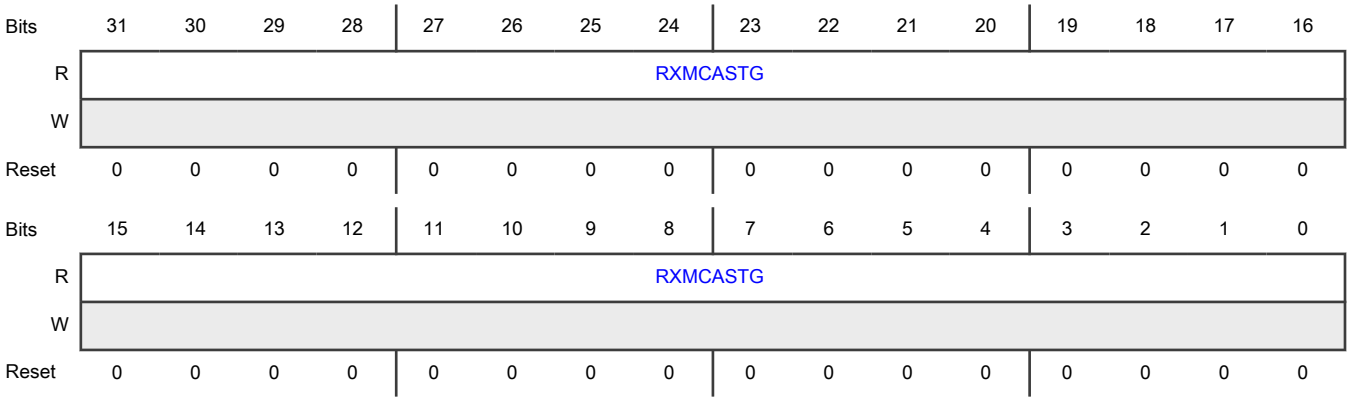
76.17.131 Receive Multicast Packets Good (Rx_Multicast_Packets_Good)

Offset

Register	Offset
Rx_Multicast_Packets_Good	790h

Function
This register provides the number of good multicast packets received by DWC_ether_qos.

Diagram



Fields

Field	Function
31-0 RXMCASTG	Rx Multicast Packets Good This field indicates the number of good multicast packets received.

76.17.132 Receive CRC Error Packets (Rx_CRC_Error_Packets)

Offset

Register	Offset
Rx_CRC_Error_Packets	794h

Function

This register provides the number of packets received by DWC_ether_qos with CRC error.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RXCRCERR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXCRCERR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 RXCRCERR	Rx CRC Error Packets This field indicates the number of packets received with CRC error.

76.17.133 Receive Alignment Error Packets (Rx_Alignment_Error_Packets)

Offset

Register	Offset
Rx_Alignment_Error_Packets	798h

Function

This register provides the number of packets received by DWC_ether_qos with alignment (dribble) error. It is valid only in 10/100 mode.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RXALGNERR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXALGNERR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 RXALGNERR	Rx Alignment Error Packets This field indicates the number of packets received with alignment (dribble) error. It is valid only in 10/100 mode.

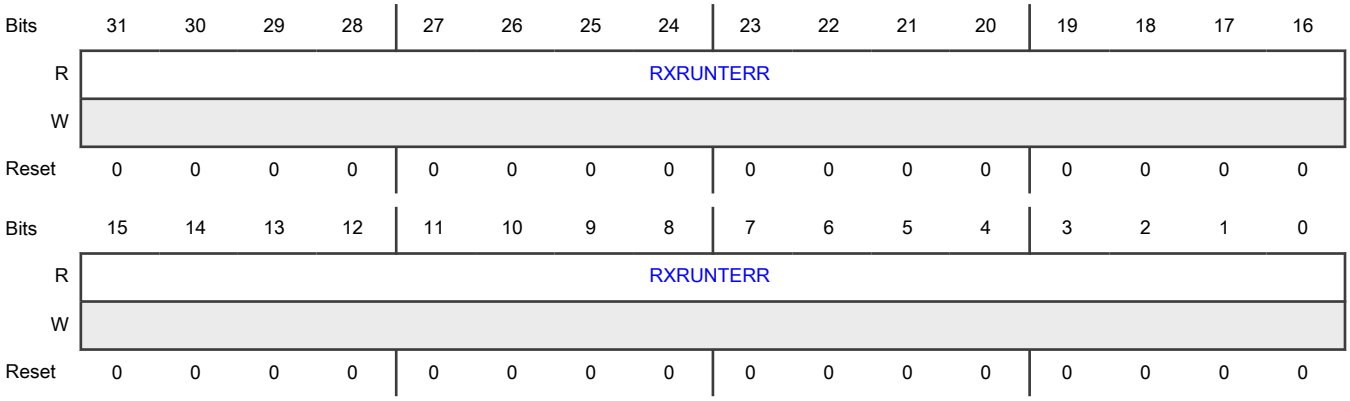
76.17.134 Receive Runt Error Packets (Rx_Runt_Error_Packets)

Offset

Register	Offset
Rx_Runt_Error_Packets	79Ch

Function
This register provides the number of packets received by DWC_ether_qos with runt (length less than 64 bytes and CRC error) error.

Diagram



Fields

Field	Function
31-0 RXRUNTERR	Rx Runt Error Packets This field indicates the number of packets received with runt (length less than 64 bytes and CRC error) error.

76.17.135 Receive Jabber Error Packets (Rx_Jabber_Error_Packets)

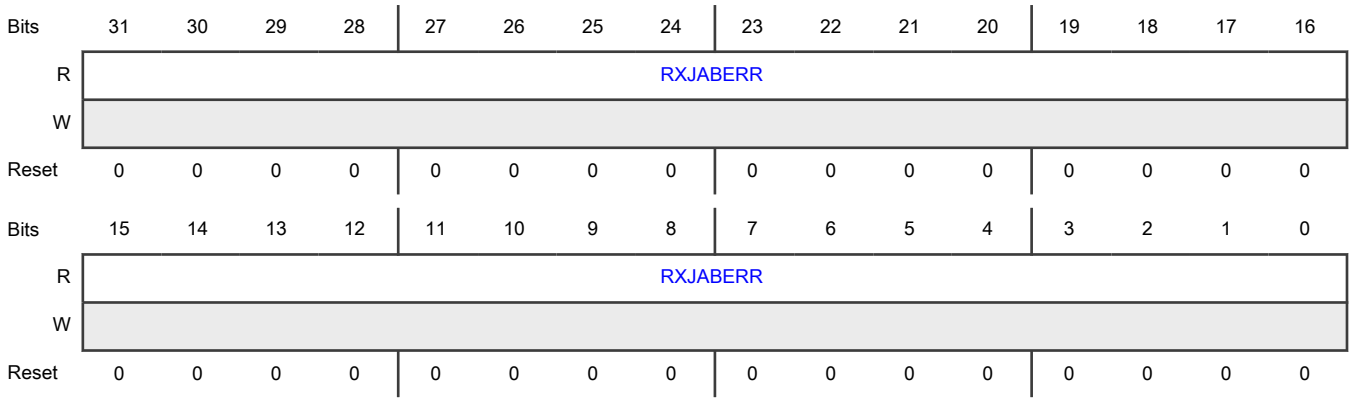
Offset

Register	Offset
Rx_Jabber_Error_Packets	7A0h

Function

This register provides the number of giant packets received by DWC_ether_qos with length (including CRC) greater than 1,518 bytes (1,522 bytes for VLAN tagged) and with CRC error. If Jumbo Packet mode is enabled, packets of length greater than 9,018 bytes (9,022 bytes for VLAN tagged) are considered as giant packets.

Diagram



Fields

Field	Function
31-0 RXJABERR	Rx Jabber Error Packets This field indicates the number of giant packets received with length (including CRC) greater than 1,518 bytes (1,522 bytes for VLAN tagged) and with CRC error. If Jumbo Packet mode is enabled, packets of length greater than 9,018 bytes (9,022 bytes for VLAN tagged) are considered as giant packets.

76.17.136 Receive Undersize Packets Good (Rx_Undersize_Packets_Good)

Offset

Register	Offset
Rx_Undersize_Packets_Good	7A4h

Function

This register provides the number of packets received by DWC_ether_qos with length less than 64 bytes, without any errors.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RXUNDERSZG															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXUNDERSZG															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 RXUNDERSZG	Rx Undersize Packets Good This field indicates the number of packets received with length less than 64 bytes, without any errors.

76.17.137 Receive Oversize Packets Good (Rx_Oversize_Packets_Good)

Offset

Register	Offset
Rx_Oversize_Packets_Good	7A8h

Function

This register provides the number of packets received by DWC_ether_qos without errors, with length greater than the maxsize (1,518 bytes or 1,522 bytes for VLAN tagged packets; 2000 bytes if enabled in the S2KP bit of the MAC_Configuration register).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RXOVERSZG															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXOVERSZG															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 RXOVERSZG	Rx Oversize Packets Good This field indicates the number of packets received without errors, with length greater than the maxsize (1,518 bytes or 1,522 bytes for VLAN tagged packets; 2000 bytes if enabled in the S2KP bit of the MAC_Configuration register).

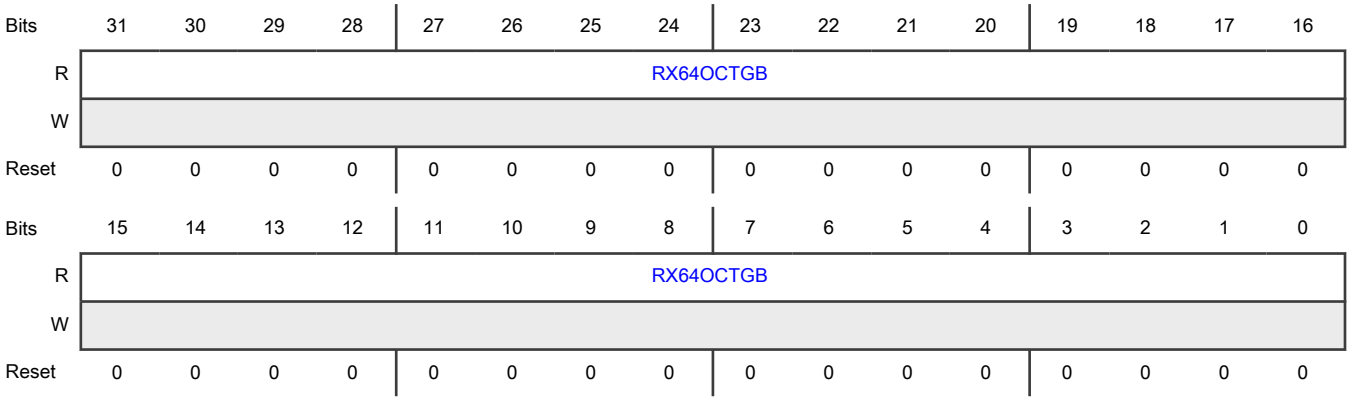
76.17.138 Receive 64 Octets Packets Good Bad (Rx_64Octets_Packets_Good_Bad)

Offset

Register	Offset
Rx_64Octets_Packets_Good_Bad	7ACh

Function
This register provides the number of good and bad packets received by DWC_ether_qos with length 64 bytes, exclusive of the preamble.

Diagram



Fields

Field	Function
31-0 RX64OCTGB	Rx 64 Octets Packets Good Bad This field indicates the number of good and bad packets received with length 64 bytes, exclusive of the preamble.

76.17.139 Receive 65 to 127 Octets Packets Good Bad (Rx_65To127Octets_Packets_Good_Bad)

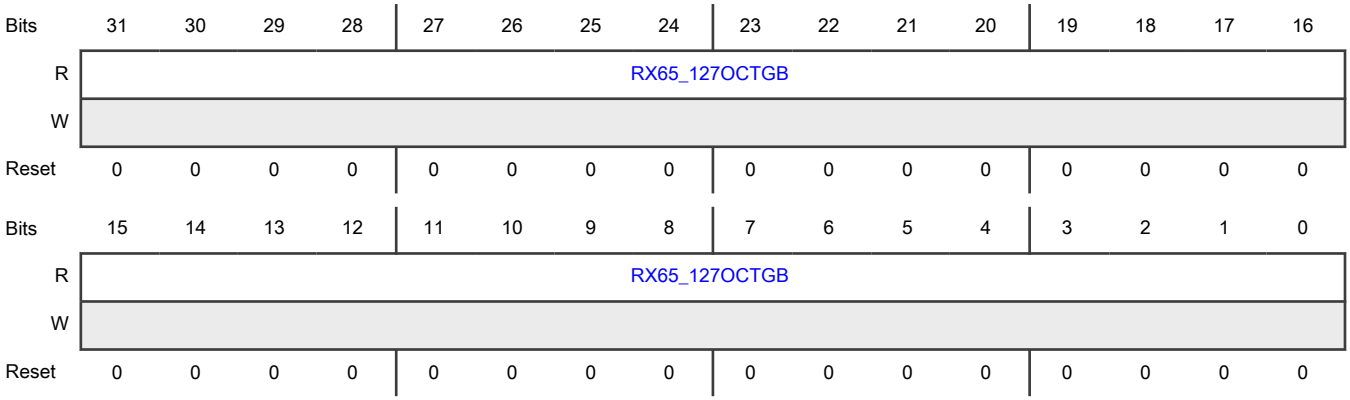
Offset

Register	Offset
Rx_65To127Octets_Packets_Good_Bad	7B0h

Function

This register provides the number of good and bad packets received by DWC_ether_qos with length between 65 and 127 (inclusive) bytes, exclusive of the preamble.

Diagram



Fields

Field	Function
31-0 RX65_127OCTGB	Rx 65-127 Octets Packets Good Bad This field indicates the number of good and bad packets received with length between 65 and 127 (inclusive) bytes, exclusive of the preamble.

76.17.140 Receive 128 to 255 Octets Packets Good Bad (Rx_128To255Octets_Packets_Good_Bad)

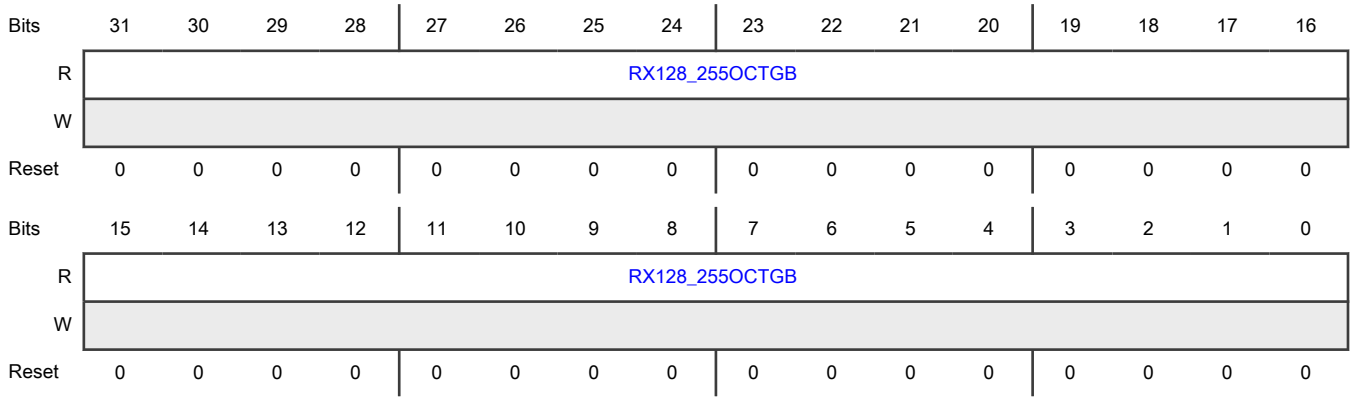
Offset

Register	Offset
Rx_128To255Octets_Packets_Good_Bad	7B4h

Function

This register provides the number of good and bad packets received by DWC_ether_qos with length between 128 and 255 (inclusive) bytes, exclusive of the preamble.

Diagram



Fields

Field	Function
31-0 RX128_255OCTGB	Rx 128-255 Octets Packets Good Bad This field indicates the number of good and bad packets received with length between 128 and 255 (inclusive) bytes, exclusive of the preamble.

76.17.141 Receive 256 to 511 Octets Packets Good Bad (Rx_256To511Octets_Packets_Good_Bad)

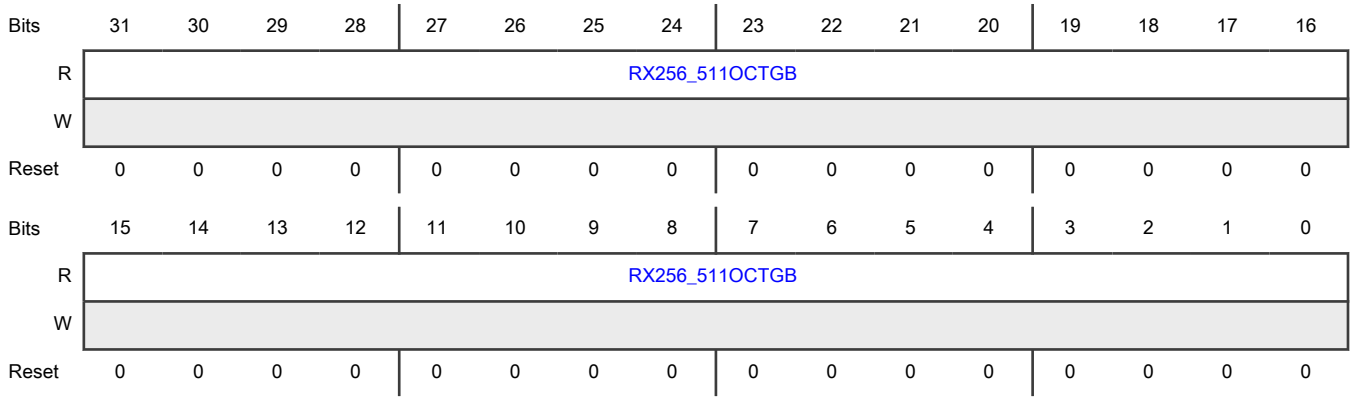
Offset

Register	Offset
Rx_256To511Octets_Packets_Good_Bad	7B8h

Function

This register provides the number of good and bad packets received by DWC_ether_qos with length between 256 and 511 (inclusive) bytes, exclusive of the preamble.

Diagram



Fields

Field	Function
31-0 RX256_511OCTGB	Rx 256-511 Octets Packets Good Bad This field indicates the number of good and bad packets received with length between 256 and 511 (inclusive) bytes, exclusive of the preamble.

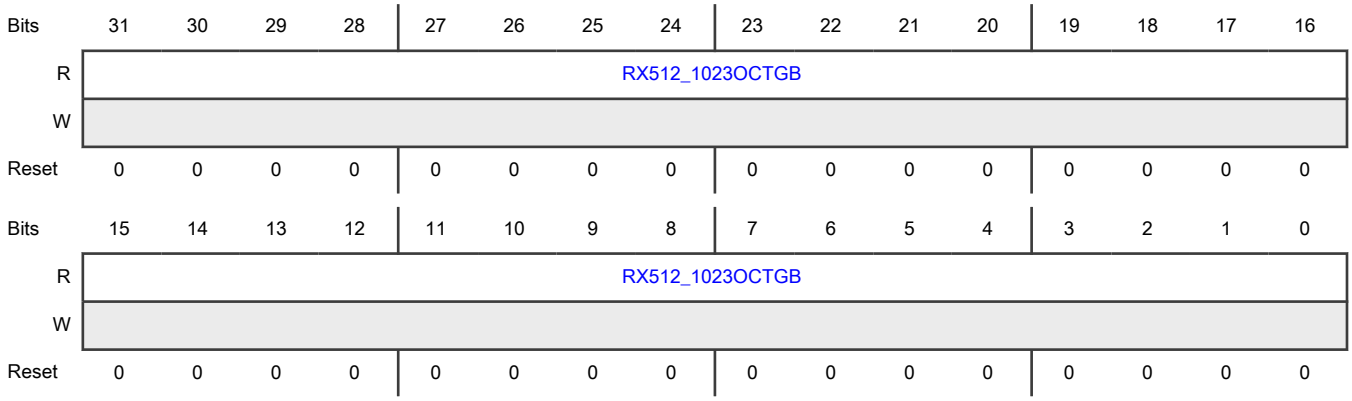
76.17.142 Receive 512 to 1023 Octets Packets Good Bad (Rx_512To1023Octets_Packets_Good_Bad)

Offset

Register	Offset
Rx_512To1023Octets_Packets_Good_Bad	7BCh

Function
This register provides the number of good and bad packets received by DWC_ether_qos with length between 512 and 1023 (inclusive) bytes, exclusive of the preamble.

Diagram



Fields

Field	Function
31-0 RX512_1023OCTGB	RX 512-1023 Octets Packets Good Bad This field indicates the number of good and bad packets received with length between 512 and 1023 (inclusive) bytes, exclusive of the preamble.

76.17.143 Receive 1024 to Max Octets Packets Good Bad (Rx_1024ToMaxOctets_Packets_Good_Bad)

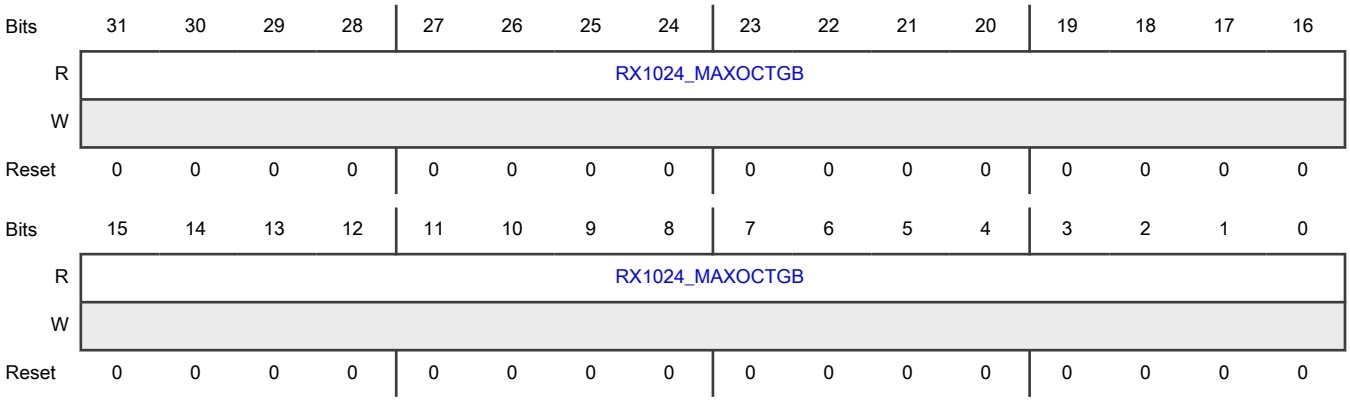
Offset

Register	Offset
Rx_1024ToMaxOctets_Packets_Good_Bad	7C0h

Function

This register provides the number of good and bad packets received by DWC_ether_qos with length between 1024 and maxsize (inclusive) bytes, exclusive of the preamble.

Diagram



Fields

Field	Function
31-0 RX1024_MAXOCTGB	Rx 1024-Max Octets Good Bad This field indicates the number of good and bad packets received with length between 1024 and maxsize (inclusive) bytes, exclusive of the preamble.

76.17.144 Receive Unicast Packets Good (Rx_Unicast_Packets_Good)

Offset

Register	Offset
Rx_Unicast_Packets_Good	7C4h

Function

This register provides the number of good unicast packets received by DWC_ether_qos.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RXUCASTG															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXUCASTG															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 RXUCASTG	Rx Unicast Packets Good This field indicates the number of good unicast packets received.

76.17.145 Receive Length Error Packets (Rx_Length_Error_Packets)

Offset

Register	Offset
Rx_Length_Error_Packets	7C8h

Function

This register provides the number of packets received by DWC_ether_qos with length error (Length Type field not equal to packet size), for all packets with valid length field.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RXLENERR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXLENERR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 RXLENERR	Rx Length Error Packets This field indicates the number of packets received with length error (Length Type field not equal to packet size), for all packets with valid length field.

76.17.146 Receive Out of Range Type Packet (Rx_Out_Of_Range_Type_Packets)

Offset

Register	Offset
Rx_Out_Of_Range_Type_Packets	7CCh

Function
This register provides the number of packets received by DWC_ether_qos with length field not equal to the valid packet size (greater than 1,500 but less than 1,536).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RXOUTOFRNG															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXOUTOFRNG															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 RXOUTOFRNG	Rx Out of Range Type Packet This field indicates the number of packets received with length field not equal to the valid packet size (greater than 1,500 but less than 1,536).

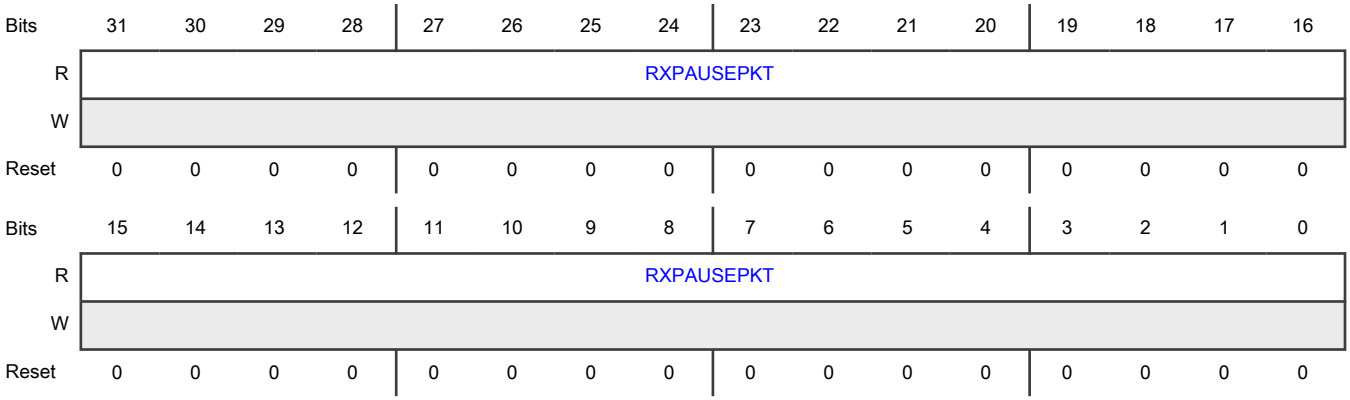
76.17.147 Receive Pause Packets (Rx_Pause_Packets)

Offset

Register	Offset
Rx_Pause_Packets	7D0h

Function
This register provides the number of good and valid Pause packets received by DWC_ether_qos.

Diagram



Fields

Field	Function
31-0 RXPAUSEPKT	Rx Pause Packets This field indicates the number of good and valid Pause packets received.

76.17.148 Receive FIFO Overflow Packets (Rx_FIFO_Overflow_Packets)

Offset

Register	Offset
Rx_FIFO_Overflow_Packets	7D4h

Function
This register provides the number of missed received packets because of FIFO overflow in DWC_ether_qos.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RXFIFOOVFL															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXFIFOOVFL															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 RXFIFOOVFL	Rx FIFO Overflow Packets This field indicates the number of missed received packets because of FIFO overflow.

76.17.149 Receive VLAN Packets Good Bad (Rx_VLAN_Packets_Good_Bad)

Offset

Register	Offset
Rx_VLAN_Packets_Good_Bad	7D8h

Function

This register provides the number of good and bad VLAN packets received by DWC_ether_qos.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RXVLANPKTGB															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXVLANPKTGB															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 RXVLANPKTGB	Rx VLAN Packets Good Bad This field indicates the number of good and bad VLAN packets received.

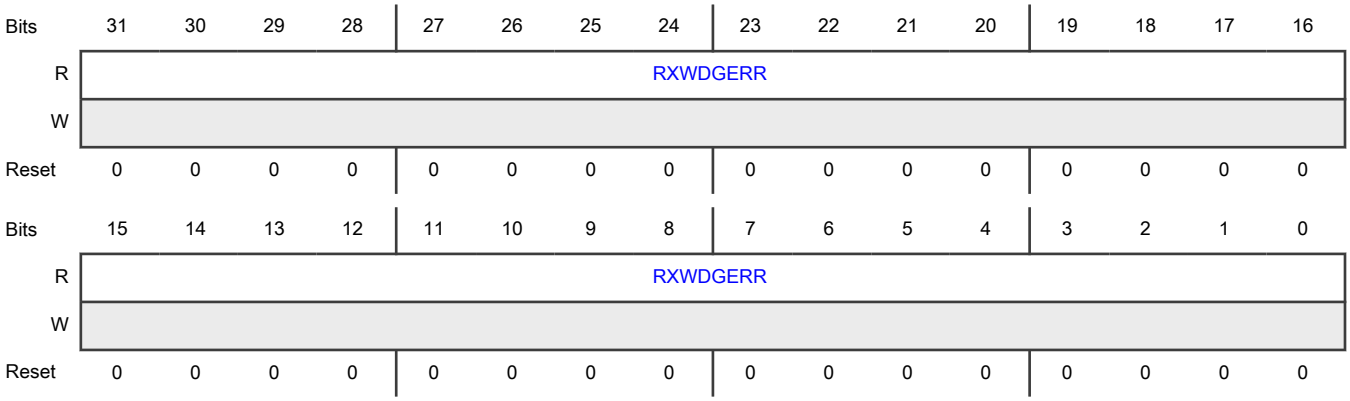
76.17.150 Receive Watchdog Error Packets (Rx_Watchdog_Error_Packets)

Offset

Register	Offset
Rx_Watchdog_Error_Packets	7DCh

Function
This register provides the number of packets received by DWC_ether_qos with error because of watchdog timeout error (packets with a data load larger than 2,048 bytes (when JE and WD bits are reset in MAC_Configuration register), 10,240 bytes (when JE bit is set and WD bit is reset in MAC_Configuration register), 16,384 bytes (when WD bit is set in MAC_Configuration register) or the value programmed in the MAC_Watchdog_Timeout register).

Diagram



Fields

Field	Function
31-0 RXWDGERR	Rx Watchdog Error Packets This field indicates the number of packets received with error because of watchdog timeout error (packets with a data load larger than 2,048 bytes (when JE and WD bits are reset in MAC_Configuration register), 10,240 bytes (when JE bit is set and WD bit is reset in MAC_Configuration register), 16,384 bytes (when WD bit is set in MAC_Configuration register) or the value programmed in the MAC_Watchdog_Timeout register).

76.17.151 Receive Error Packets (Rx_Receive_Error_Packets)

Offset

Register	Offset
Rx_Receive_Error_Packets	7E0h

Function

This register provides the number of packets received by DWC_ether_qos with Receive error or Packet Extension error on the GMII or MII interface.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RXRCVERR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXRCVERR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 RXRCVERR	Rx Receive Error Packets This field indicates the number of packets received with Receive error or Packet Extension error on the GMII or MII interface.

76.17.152 Receive Control Packets Good (Rx_Control_Packets_Good)

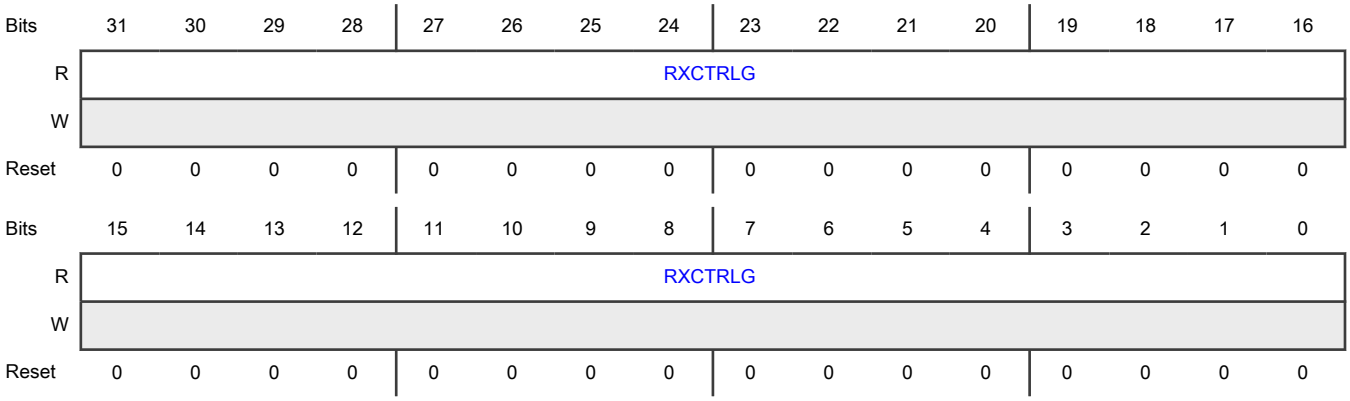
Offset

Register	Offset
Rx_Control_Packets_Good	7E4h

Function

This register provides the number of good control packets received by DWC_ether_qos.

Diagram



Fields

Field	Function
31-0 RXCTRLG	Rx Control Packets Good This field indicates the number of good control packets received.

76.17.153 MMC Transmit FPE Fragment Counter Interrupt Status (MMC_FPE_Tx_Interrupt)

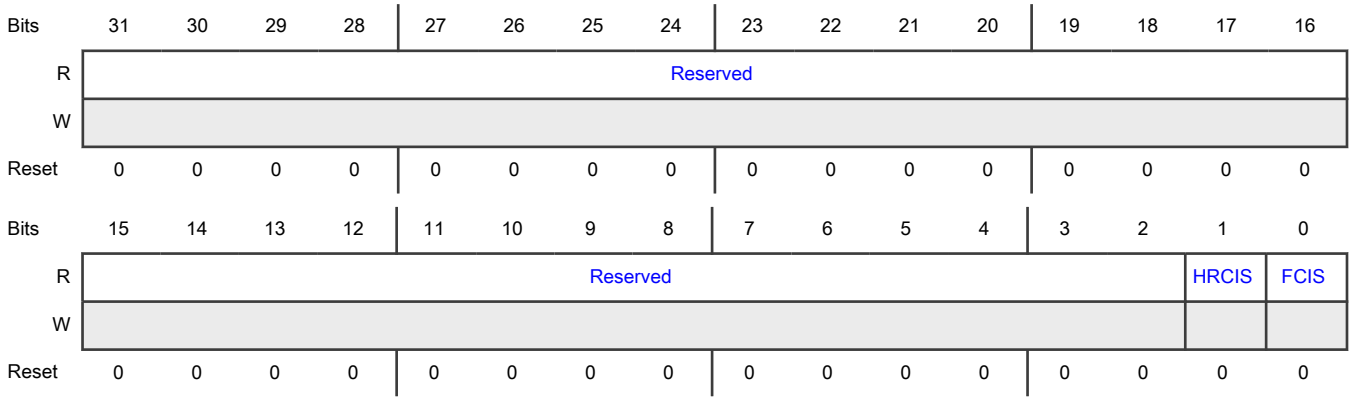
Offset

Register	Offset
MMC_FPE_Tx_Interrupt	8A0h

Function

This register maintains the interrupts generated from all FPE related Transmit statistics counters. The MMC FPE Transmit Interrupt register maintains the interrupts generated when transmit statistic counters reach half their maximum values (0x8000_0000 for 32 bit counter and 0x8000 for 16 bit counter), and when they cross their maximum values (0xFFFF_FFFF for 32-bit counter and 0xFFFF for 16-bit counter). When Counter Stop Rollover is set, the interrupts are set but the counter remains at all-ones. The MMC FPE Transmit Interrupt register is a 32 bit register. An interrupt bit is cleared when the respective MMC counter that caused the interrupt is read. The least significant byte lane (Bits[7:0]) of the respective counter must be read to clear the interrupt bit.

Diagram



Fields

Field	Function
31-2 —	Reserved
1 HRCIS	MMC Tx Hold Request Counter Interrupt Status This bit is set when the Tx_Hold_Req_Cntr counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. Exists when any one of the RX/TX MMC counters are enabled during FPE with AV_EST Enabled configuration. 0b - MMC Tx Hold Request Counter Interrupt Status not detected 1b - MMC Tx Hold Request Counter Interrupt Status detected
0 FCIS	MMC Tx FPE Fragment Counter Interrupt status This bit is set when the Tx_FPE_Fragment_Cntr counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. Exists when any one of the RX/TX MMC counters are enabled during FPE Enabled configuration. 0b - MMC Tx FPE Fragment Counter Interrupt status not detected 1b - MMC Tx FPE Fragment Counter Interrupt status detected

76.17.154 MMC FPE Transmit Interrupt Mask (MMC_FPE_Tx_Interrupt_Mask)

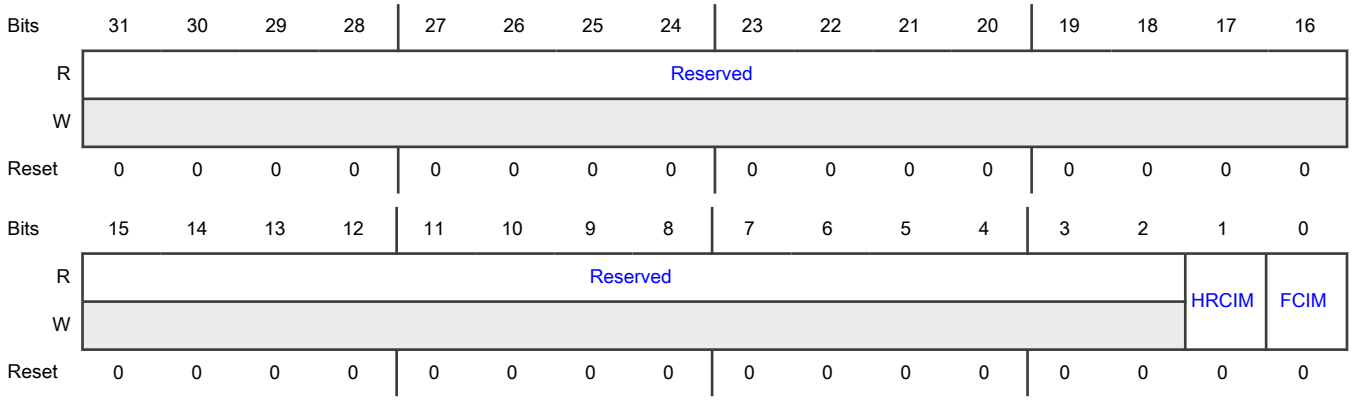
Offset

Register	Offset
MMC_FPE_Tx_Interrupt_Mask	8A4h

Function

This register maintains the masks for interrupts generated from all FPE related Transmit statistics counters. The MMC Receive Interrupt Mask register maintains the masks for the interrupts generated when FPE related receive statistic counters reach half of their maximum value or the maximum values. This register is 32 bit wide.

Diagram



Fields

Field	Function
31-2 —	Reserved
1 HRCIM	MMC Transmit Hold Request Counter Interrupt Mask Setting this bit masks the interrupt when the Tx_Hold_Req_Cntr counter reaches half of the maximum value or the maximum value. Exists when any one of the RX/TX MMC counters are enabled during FPE with AV_EST Enabled configuration. 0b - MMC Transmit Hold Request Counter Interrupt Mask is disabled 1b - MMC Transmit Hold Request Counter Interrupt Mask is enabled
0 FCIM	MMC Transmit Fragment Counter Interrupt Mask Setting this bit masks the interrupt when the Tx_FPE_Fragment_Cntr counter reaches half of the maximum value or the maximum value. Exists when any one of the RX/TX MMC counters are enabled during FPE Enabled configuration. 0b - MMC Transmit Fragment Counter Interrupt Mask is disabled 1b - MMC Transmit Fragment Counter Interrupt Mask is enabled

76.17.155 Transmit FPE Fragment Counter (MMC_Tx_FPE_Fragment_Cntr)

Offset

Register	Offset
MMC_Tx_FPE_Fragment_Cntr	8A8h

Function

This register provides the number of additional mPackets transmitted due to preemption.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TXFFC															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TXFFC															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TXFFC	Tx FPE Fragment counter This field indicates the number of additional mPackets that has been transmitted due to preemption Exists when any one of the RX/TX MMC counters are enabled during FPE Enabled configuration.

76.17.156 Transmit Hold Request Counter (MMC_Tx_Hold_Req_Cntr)

Offset

Register	Offset
MMC_Tx_Hold_Req_Cntr	8ACh

Function

This register provides the count of number of times a hold request is given to MAC

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TXHRC															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TXHRC															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TXHRC	Tx Hold Request Counter This field indicates count of number of a hold request is given to MAC. Exists when any one of the RX/TX MMC counters are enabled during FPE with AV_EST Enabled configuration.

76.17.157 MMC Receive Packet Assembly Error Counter Interrupt Status (MMC_FPE_Rx_Interrupt)

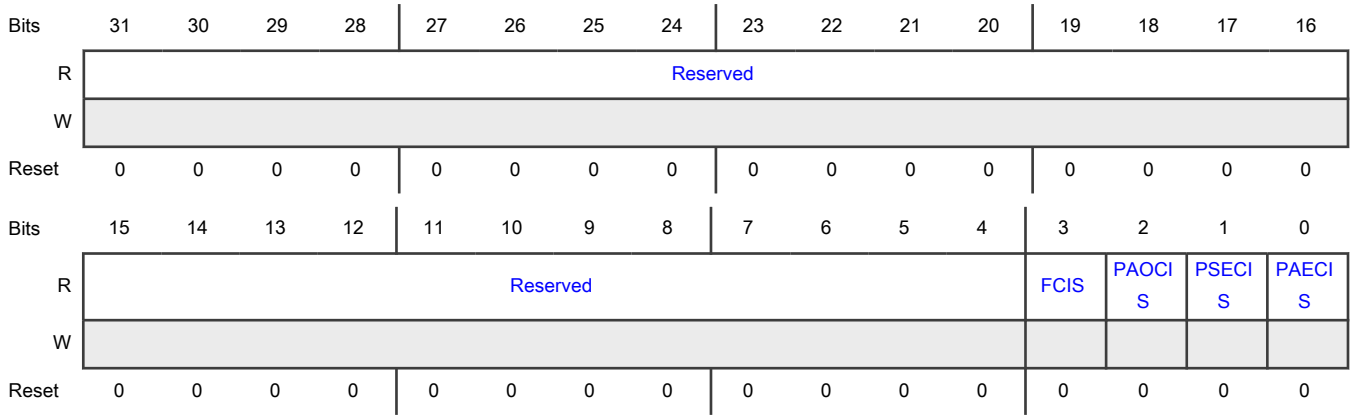
Offset

Register	Offset
MMC_FPE_Rx_Interrupt	8C0h

Function

This register maintains the interrupts generated from all FPE related Receive statistics counters. The MMC FPE Receive Interrupt register maintains the interrupts generated when transmit statistic counters reach half their maximum values (0x8000_0000 for 32 bit counter and 0x8000 for 16 bit counter), and when they cross their maximum values (0xFFFF_FFFF for 32-bit counter and 0xFFFF for 16-bit counter). When Counter Stop Rollover is set, the interrupts are set but the counter remains at all-ones. The MMC FPE Receive Interrupt register is a 32 bit register. An interrupt bit is cleared when the respective MMC counter that caused the interrupt is read. The least significant byte lane (Bits[7:0]) of the respective counter must be read to clear the interrupt bit.

Diagram



Fields

Field	Function
31-4 —	Reserved
3	MMC Rx FPE Fragment Counter Interrupt Status

Table continues on the next page...

Table continued from the previous page...

Field	Function
FCIS	This bit is set when the Rx_FPE_Fragment_Cntr counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. Exists when any one of the RX/TX MMC counters are enabled during FPE Enabled configuration. 0b - MMC Rx FPE Fragment Counter Interrupt Status not detected 1b - MMC Rx FPE Fragment Counter Interrupt Status detected
2 PAOCIS	MMC Rx Packet Assembly OK Counter Interrupt Status This bit is set when the Rx_Packet_Assemble_Ok_Cntr counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. Exists when any one of the RX/TX MMC counters are enabled during FPE Enabled configuration. 0b - MMC Rx Packet Assembly OK Counter Interrupt Status not detected 1b - MMC Rx Packet Assembly OK Counter Interrupt Status detected
1 PSECIS	MMC Rx Packet SMD Error Counter Interrupt Status This bit is set when the Rx_Packet_SMD_Err_Cntr counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. Exists when any one of the RX/TX MMC counters are enabled during FPE Enabled configuration. 0b - MMC Rx Packet SMD Error Counter Interrupt Status not detected 1b - MMC Rx Packet SMD Error Counter Interrupt Status detected
0 PAECIS	MMC Rx Packet Assembly Error Counter Interrupt Status This bit is set when the Rx_Packet_Assemble_Err_Cntr counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. Exists when any one of the RX/TX MMC counters are enabled during FPE Enabled configuration. 0b - MMC Rx Packet Assembly Error Counter Interrupt Status not detected 1b - MMC Rx Packet Assembly Error Counter Interrupt Status detected

76.17.158 MMC FPE Receive Interrupt Mask (MMC_FPE_Rx_Interrupt_Mask)

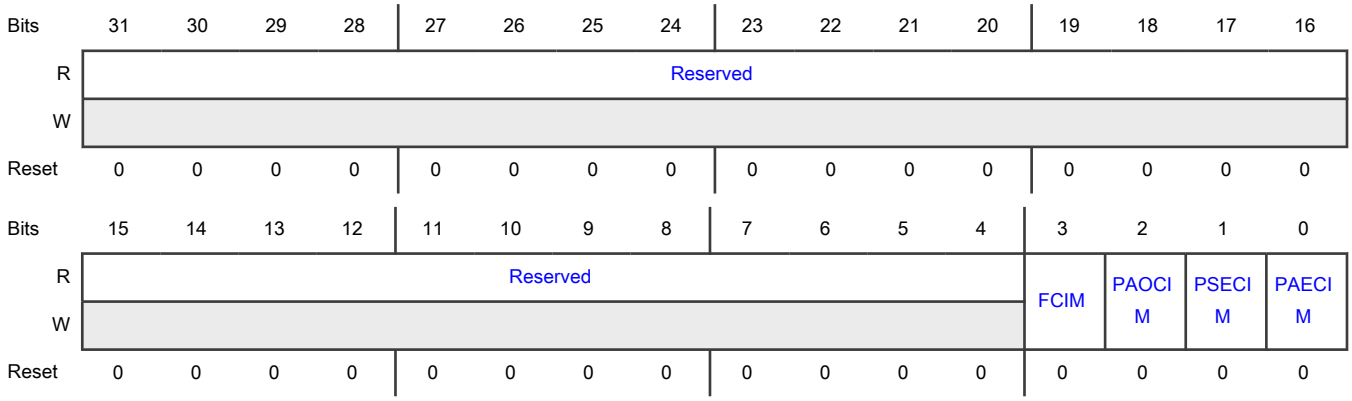
Offset

Register	Offset
MMC_FPE_Rx_Interrupt_Mask	8C4h

Function

This register maintains the masks for interrupts generated from all FPE related Receive statistics counters. The MMC Receive Interrupt Mask register maintains the masks for the interrupts generated when FPE related receive statistic counters reach half of their maximum value or the maximum values. This register is 32 bit wide.

Diagram



Fields

Field	Function
31-4 —	Reserved
3 FCIM	MMC Rx FPE Fragment Counter Interrupt Mask Setting this bit masks the interrupt when the Tx_FPE_Fragment_Cntr counter reaches half of the maximum value or the maximum value. Exists when any one of the RX/TX MMC counters are enabled during FPE Enabled configuration. 0b - MMC Rx FPE Fragment Counter Interrupt Mask is disabled 1b - MMC Rx FPE Fragment Counter Interrupt Mask is enabled
2 PAOCIM	MMC Rx Packet Assembly OK Counter Interrupt Mask Setting this bit masks the interrupt when the Rx_Packet_Assemble_Ok_Cntr counter reaches half of the maximum value or the maximum value. Exists when any one of the RX/TX MMC counters are enabled during FPE Enabled configuration. 0b - MMC Rx Packet Assembly OK Counter Interrupt Mask is disabled 1b - MMC Rx Packet Assembly OK Counter Interrupt Mask is enabled
1 PSECIM	MMC Rx Packet SMD Error Counter Interrupt Mask Setting this bit masks the interrupt when the R Rx_Packet_SMD_Err_Cntr counter reaches half of the maximum value or the maximum value. Exists when any one of the RX/TX MMC counters are enabled during FPE Enabled configuration. 0b - MMC Rx Packet SMD Error Counter Interrupt Mask is disabled 1b - MMC Rx Packet SMD Error Counter Interrupt Mask is enabled
0 PAECIM	MMC Rx Packet Assembly Error Counter Interrupt Mask Setting this bit masks the interrupt when the R Rx_Packet_Assemble_Err_Cntr counter reaches half of the maximum value or the maximum value. Exists when any one of the RX/TX MMC counters are enabled during FPE Enabled configuration.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - MMC Rx Packet Assembly Error Counter Interrupt Mask is disabled 1b - MMC Rx Packet Assembly Error Counter Interrupt Mask is enabled

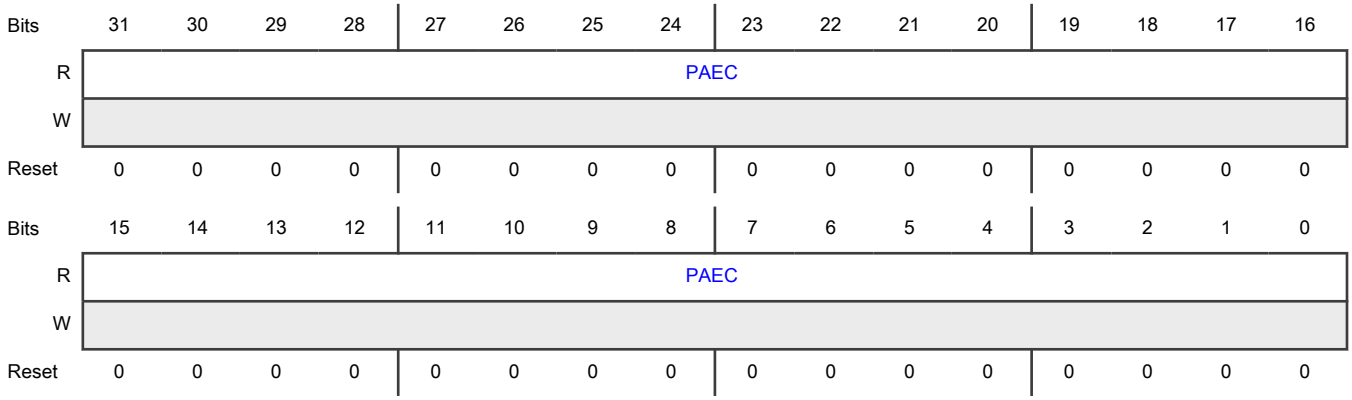
76.17.159 MMC Receive Packet Assembly Error Counter (MMC_Rx_Packet_Assembly_Err_Cntr)

Offset

Register	Offset
MMC_Rx_Packet_Assembly_Err_Cntr	8C8h

Function
This register provides the number of MAC frames with reassembly errors on the Receiver, due to mismatch in the Fragment Count value.

Diagram



Fields

Field	Function
31-0 PAEC	Rx Packet Assembly Error Counter This field indicates the number of MAC frames with reassembly errors on the Receiver, due to mismatch in the Fragment Count value. Exists when any one of the RX/TX MMC counters are enabled during FPE Enabled configuration.

76.17.160 MMC Receive Packet SMD Error Counter (MMC_Rx_Packet_SMD_Err_Cntr)

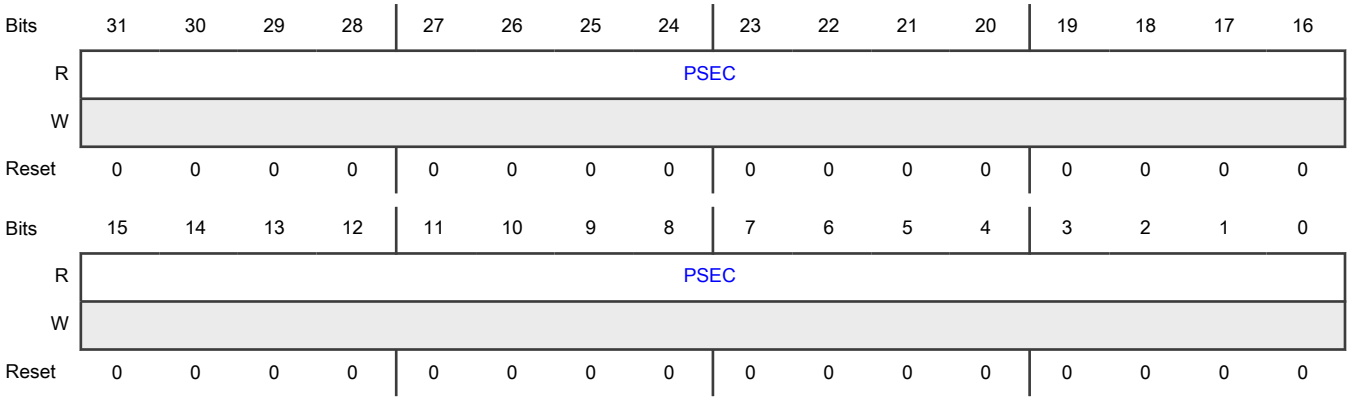
Offset

Register	Offset
MMC_Rx_Packet_SMD_Err_Cntr	8CCh

Function

This register provides the number of received MAC frames rejected due to unknown SMD value and MAC frame fragments rejected due to arriving with an SMD-C when there was no preceding preempted frame.

Diagram



Fields

Field	Function
31-0 PSEC	Rx Packet SMD Error Counter This field indicates the number of MAC frames rejected due to unknown SMD value and MAC frame fragments rejected due to arriving with an SMD-C when there was no preceding preempted frame. Exists when at least one of the RX/TX MMC counters are enabled during FPE Enabled configuration.

76.17.161 MMC Receive Packet Assembly OK Counter (MMC_Rx_Packet_Assembly_OK_Cntr)

Offset

Register	Offset
MMC_Rx_Packet_Assembly_OK_Cntr	8D0h

Function

This register provides the number of MAC frames that were successfully reassembled and delivered to MAC.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PAOC															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PAOC															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PAOC	Rx Packet Assembly OK Counter This field indicates the number of MAC frames that were successfully reassembled and delivered to MAC. Exists when at least one of the RX/TX MMC counters are enabled during FPE Enabled configuration.

76.17.162 MMC Receive FPE Fragment Counter (MMC_Rx_FPE_Fragment_Cntr)

Offset

Register	Offset
MMC_Rx_FPE_Fragment_Cntr	8D4h

Function

This register provides the number of additional mPackets received due to preemption.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	FFC															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FFC															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 FFC	Rx FPE Fragment Counter This field indicates the number of additional mPackets received due to preemption Exists when at least one of the RX/TX MMC counters are enabled during FPE Enabled configuration.

76.17.163 MAC Layer 3 Layer 4 Control 0 (MAC_L3_L4_Control0)

Offset

Register	Offset
MAC_L3_L4_Control0	900h

Function
The Layer 3 and Layer 4 Control register controls the operations of filter 0 of Layer 3 and Layer 4.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved			DMCHEN0	Reserved		DMCHN0		Reserved		L4DPIM0	L4DPM0	L4SPIM0	L4SPM0	Reserved	L4PEN0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	L3HDBM0				L3HSBM0						L3DAIM0	L3DAM0	L3SAIM0	L3SAM0	Reserved	L3PEN0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	Reserved
28 DMCHEN0	DMA Channel Select Enable When set, this bit enables the selection of the DMA channel number for the packet that is passed by this L3_L4 filter. The DMA channel is indicated by the DMCHN bits. When this bit is reset, the DMA channel is not decided by this filter. 0b - DMA Channel Select is disabled 1b - DMA Channel Select is enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
27-26 —	Reserved
25-24 DMCHN0	DMA Channel Number When DMCHEN is set high, this field selects the DMA Channel number to which the packet passed by this filter is routed. The width of this field depends on the number of the DMA channels present in your configuration.
23-22 —	Reserved
21 L4DPIM0	Layer 4 Destination Port Inverse Match Enable When this bit is set, the Layer 4 Destination Port number field is enabled for inverse matching. When this bit is reset, the Layer 4 Destination Port number field is enabled for perfect matching. This bit is valid and applicable only when the L4DPM0 bit is set high. 0b - Layer 4 Destination Port Inverse Match is disabled 1b - Layer 4 Destination Port Inverse Match is enabled
20 L4DPM0	Layer 4 Destination Port Match Enable When this bit is set, the Layer 4 Destination Port number field is enabled for matching. When this bit is reset, the MAC ignores the Layer 4 Destination Port number field for matching. 0b - Layer 4 Destination Port Match is disabled 1b - Layer 4 Destination Port Match is enabled
19 L4SPIM0	Layer 4 Source Port Inverse Match Enable When this bit is set, the Layer 4 Source Port number field is enabled for inverse matching. When this bit is reset, the Layer 4 Source Port number field is enabled for perfect matching. This bit is valid and applicable only when the L4SPM0 bit is set high. 0b - Layer 4 Source Port Inverse Match is disabled 1b - Layer 4 Source Port Inverse Match is enabled
18 L4SPM0	Layer 4 Source Port Match Enable When this bit is set, the Layer 4 Source Port number field is enabled for matching. When this bit is reset, the MAC ignores the Layer 4 Source Port number field for matching. 0b - Layer 4 Source Port Match is disabled 1b - Layer 4 Source Port Match is enabled
17 —	Reserved
16	Layer 4 Protocol Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
L4PEN0	When this bit is set, the Source and Destination Port number fields of UDP packets are used for matching. When this bit is reset, the Source and Destination Port number fields of TCP packets are used for matching. The Layer 4 matching is done only when the L4SPM0 or L4DPM0 bit is set. 0b - Layer 4 Protocol is disabled 1b - Layer 4 Protocol is enabled
15-11 L3HDBM0	Layer 3 IP DA Higher Bits Match IPv4 Packets: This field contains the number of higher bits of IP Destination Address that are matched in the IPv4 packets. The following list describes the values of this field: - 0: No bits are masked. - 1: LSb[0] is masked - 2: Two LSbs [1:0] are masked - .. - 31: All bits except MSb are masked. IPv6 Packets: Bits[12:11] of this field correspond to Bits[6:5] of L3HSBM0 which indicate the number of lower bits of IP Source or Destination Address that are masked in the IPv6 packets. The following list describes the concatenated values of the L3HDBM0[1:0] and L3HSBM0 bits: - 0: No bits are masked. - 1: LSb[0] is masked. - 2: Two LSbs [1:0] are masked - .. - 127: All bits except MSb are masked. This field is valid and applicable only when the L3DAM0 or L3SAM0 bit is set.
10-6 L3HSBM0	Layer 3 IP SA Higher Bits Match IPv4 Packets: This field contains the number of lower bits of IP Source Address that are masked for matching in the IPv4 packets. The following list describes the values of this field: - 0: No bits are masked. - 1: LSb[0] is masked - 2: Two LSbs [1:0] are masked - .. - 31: All bits except MSb are masked. IPv6 Packets: This field contains Bits[4:0] of L3HSBM0. These bits indicate the number of higher bits of IP Source or Destination Address matched in the IPv6 packets. This field is valid and applicable only when the L3DAM0 or L3SAM0 bit is set high.
5 L3DAIM0	Layer 3 IP DA Inverse Match Enable When this bit is set, the Layer 3 IP Destination Address field is enabled for inverse matching. When this bit is reset, the Layer 3 IP Destination Address field is enabled for perfect matching. This bit is valid and applicable only when the L3DAM0 bit is set high. 0b - Layer 3 IP DA Inverse Match is disabled 1b - Layer 3 IP DA Inverse Match is enabled
4 L3DAM0	Layer 3 IP DA Match Enable When this bit is set, the Layer 3 IP Destination Address field is enabled for matching. When this bit is reset, the MAC ignores the Layer 3 IP Destination Address field for matching. Note: When the L3PEN0 bit is set, you should set either this bit or the L3SAM0 bit because either IPv6 DA or SA can be checked for filtering. 0b - Layer 3 IP DA Match is disabled 1b - Layer 3 IP DA Match is enabled
3 L3SAIM0	Layer 3 IP SA Inverse Match Enable When this bit is set, the Layer 3 IP Source Address field is enabled for inverse matching. When this bit reset, the Layer 3 IP Source Address field is enabled for perfect matching. This bit is valid and applicable only when the L3SAM0 bit is set.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Layer 3 IP SA Inverse Match is disabled 1b - Layer 3 IP SA Inverse Match is enabled
2 L3SAM0	Layer 3 IP SA Match Enable When this bit is set, the Layer 3 IP Source Address field is enabled for matching. When this bit is reset, the MAC ignores the Layer 3 IP Source Address field for matching. Note: When the L3PEN0 bit is set, you should set either this bit or the L3DAM0 bit because either IPv6 SA or DA can be checked for filtering. 0b - Layer 3 IP SA Match is disabled 1b - Layer 3 IP SA Match is enabled
1 —	Reserved
0 L3PEN0	Layer 3 Protocol Enable When this bit is set, the Layer 3 IP Source or Destination Address matching is enabled for IPv6 packets. When this bit is reset, the Layer 3 IP Source or Destination Address matching is enabled for IPv4 packets. The Layer 3 matching is done only when the L3SAM0 or L3DAM0 bit is set. 0b - Layer 3 Protocol is disabled 1b - Layer 3 Protocol is enabled

76.17.164 MAC Layer 4 Address 0 (MAC_Layer4_Address0)

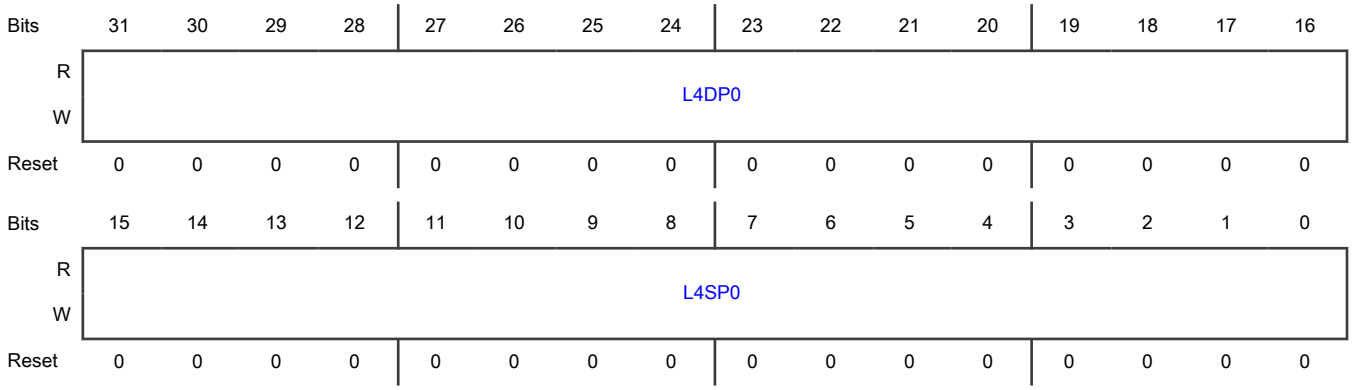
Offset

Register	Offset
MAC_Layer4_Address0	904h

Function

The MAC_Layer4_Address(#i), MAC_L3_L4_Control(#i), MAC_Layer3_Addr0_Reg(#i), MAC_Layer3_Addr1_Reg(#i), MAC_Layer3_Addr2_Reg(#i) and MAC_Layer3_Addr3_Reg(#i) registers are reserved (RO with default value) if Enable Layer 3 and Layer 4 Packet Filter option is not selected while configuring the IP. You can configure the Layer 3 and Layer 4 Address Registers to be double-synchronized by selecting the Synchronize Layer 3 and Layer 4 Address Registers to Rx Clock Domain option while configuring the IP. When you select this option, the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the Layer 3 and Layer 4 Address Registers are written. For proper synchronization updates, you should perform consecutive writes to same Layer 3 and Layer 4 Address Registers after at least four clock cycles delay of the destination clock.

Diagram



Fields

Field	Function
31-16 L4DP0	Layer 4 Destination Port Number Field When the L4PEN0 bit is reset and the L4DPM0 bit is set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with the TCP Destination Port Number field in the IPv4 or IPv6 packets. When the L4PEN0 and L4DPM0 bits are set in MAC_L3_L4_Control0 register, this field contains the value to be matched with the UDP Destination Port Number field in the IPv4 or IPv6 packets.
15-0 L4SP0	Layer 4 Source Port Number Field When the L4PEN0 bit is reset and the L4SPM0 bit is set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with the TCP Source Port Number field in the IPv4 or IPv6 packets. When the L4PEN0 and L4SPM0 bits are set in MAC_L3_L4_Control0 register, this field contains the value to be matched with the UDP Source Port Number field in the IPv4 or IPv6 packets.

76.17.165 MAC Layer 3 Address 0 Reg 0 (MAC_Layer3_Addr0_Reg0)

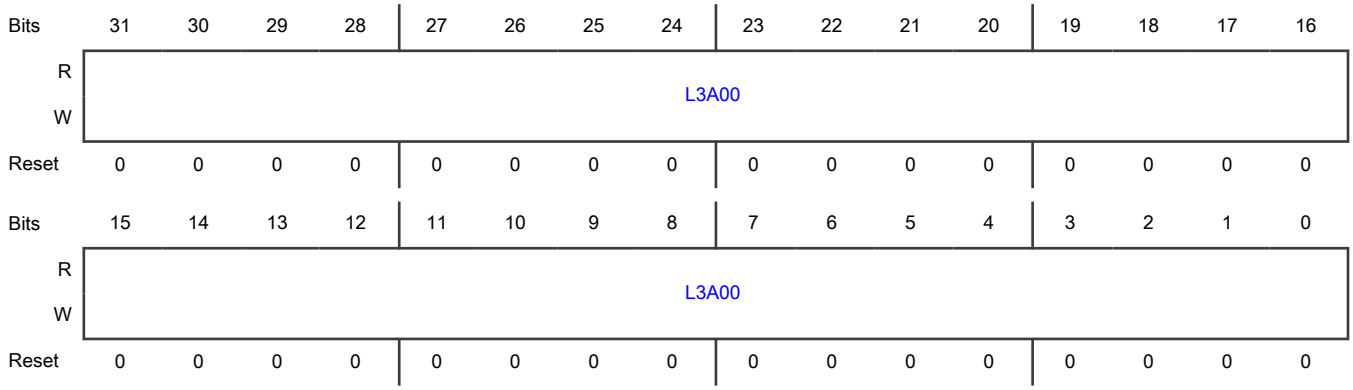
Offset

Register	Offset
MAC_Layer3_Addr0_Reg0	910h

Function

For IPv4 packets, the Layer 3 Address 0 Register 0 register contains the 32-bit IP Source Address field. For IPv6 packets, it contains Bits[31:0] of the 128-bit IP Source Address or Destination Address field.

Diagram



Fields

Field	Function
31-0 L3A00	Layer 3 Address 0 Field When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[31:0] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[31:0] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset and the L3SAM0 bit is set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with the IP Source Address field in the IPv4 packets.

76.17.166 MAC Layer 3 Address 1 Reg 0 (MAC_Layer3_Addr1_Reg0)

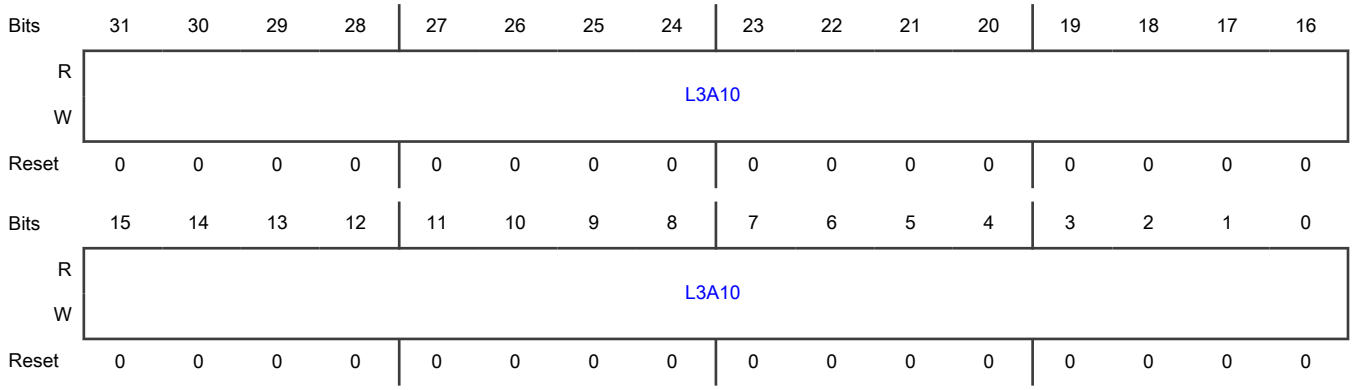
Offset

Register	Offset
MAC_Layer3_Addr1_Reg0	914h

Function

For IPv4 packets, the Layer 3 Address 1 Register 0 register contains the 32-bit IP Destination Address field. For IPv6 packets, it contains Bits[63:32] of the 128-bit IP Source Address or Destination Address field.

Diagram



Fields

Field	Function
31-0 L3A10	Layer 3 Address 1 Field When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[63:32] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[63:32] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset and the L3SAM0 bit is set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with the IP Destination Address field in the IPv4 packets.

76.17.167 MAC Layer 3 Address 2 Reg 0 (MAC_Layer3_Addr2_Reg0)

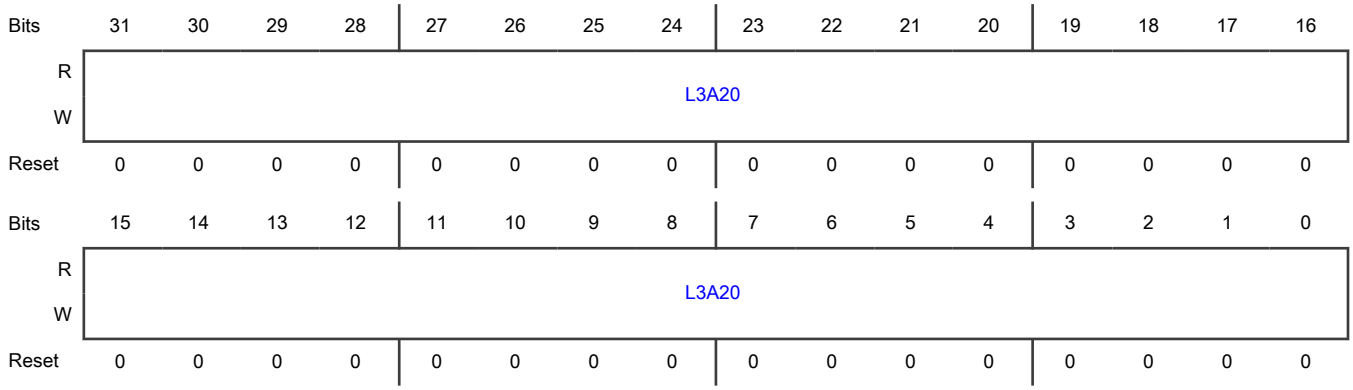
Offset

Register	Offset
MAC_Layer3_Addr2_Reg0	918h

Function

The Layer 3 Address 2 Register 0 register is reserved for IPv4 packets. For IPv6 packets, it contains Bits[95:64] of 128-bit IP Source Address or Destination Address field.

Diagram



Fields

Field	Function
31-0 L3A20	Layer 3 Address 2 Field When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[95:64] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[95:64] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset in the MAC_L3_L4_Control0 register, this field is not used.

76.17.168 MAC Layer 3 Address 3 Reg 0 (MAC_Layer3_Addr3_Reg0)

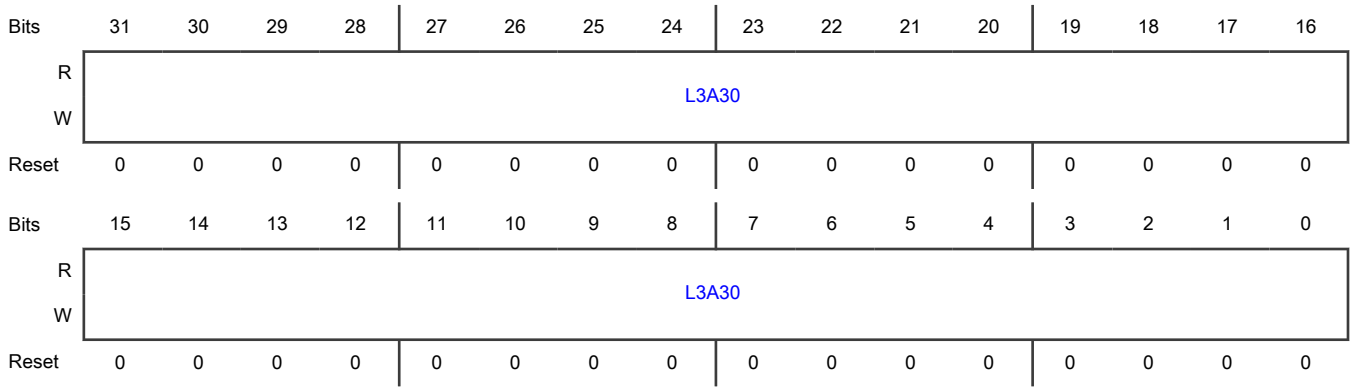
Offset

Register	Offset
MAC_Layer3_Addr3_Reg0	91Ch

Function

The Layer 3 Address 3 Register 0 register is reserved for IPv4 packets. For IPv6 packets, it contains Bits[127:96] of 128-bit IP Source Address or Destination Address field.

Diagram



Fields

Field	Function
31-0 L3A30	Layer 3 Address 3 Field When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[127:96] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[127:96] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset in the MAC_L3_L4_Control0 register, this field is not used.

76.17.169 MAC Layer 3 Layer 4 Control 1 (MAC_L3_L4_Control1)

Offset

Register	Offset
MAC_L3_L4_Control1	930h

Function
The Layer 3 and Layer 4 Control register controls the operations of filter 0 of Layer 3 and Layer 4.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved			DMCHEN1	Reserved		DMCHN1		Reserved		L4DPIM1	L4DPM1	L4SPIM1	L4SPM1	Reserved	L4PEN1
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	L3HDBM1				L3HSBM1						L3DAIM1	L3DAM1	L3SAIM1	L3SAM1	Reserved	L3PEN1
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	Reserved
28 DMCHEN1	DMA Channel Select Enable When set, this bit enables the selection of the DMA channel number for the packet that is passed by this L3_L4 filter. The DMA channel is indicated by the DMCHN bits. When this bit is reset, the DMA channel is not decided by this filter.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - DMA Channel Select is disabled 1b - DMA Channel Select is enabled
27-26 —	Reserved
25-24 DMCHN1	DMA Channel Number When DMCHEN is set high, this field selects the DMA Channel number to which the packet passed by this filter is routed. The width of this field depends on the number of the DMA channels present in your configuration.
23-22 —	Reserved
21 L4DPIM1	Layer 4 Destination Port Inverse Match Enable When this bit is set, the Layer 4 Destination Port number field is enabled for inverse matching. When this bit is reset, the Layer 4 Destination Port number field is enabled for perfect matching. This bit is valid and applicable only when the L4DPM0 bit is set high. 0b - Layer 4 Destination Port Inverse Match is disabled 1b - Layer 4 Destination Port Inverse Match is enabled
20 L4DPM1	Layer 4 Destination Port Match Enable When this bit is set, the Layer 4 Destination Port number field is enabled for matching. When this bit is reset, the MAC ignores the Layer 4 Destination Port number field for matching. 0b - Layer 4 Destination Port Match is disabled 1b - Layer 4 Destination Port Match is enabled
19 L4SPIM1	Layer 4 Source Port Inverse Match Enable When this bit is set, the Layer 4 Source Port number field is enabled for inverse matching. When this bit is reset, the Layer 4 Source Port number field is enabled for perfect matching. This bit is valid and applicable only when the L4SPM0 bit is set high. 0b - Layer 4 Source Port Inverse Match is disabled 1b - Layer 4 Source Port Inverse Match is enabled
18 L4SPM1	Layer 4 Source Port Match Enable When this bit is set, the Layer 4 Source Port number field is enabled for matching. When this bit is reset, the MAC ignores the Layer 4 Source Port number field for matching. 0b - Layer 4 Source Port Match is disabled 1b - Layer 4 Source Port Match is enabled
17	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
16 L4PEN1	Layer 4 Protocol Enable When this bit is set, the Source and Destination Port number fields of UDP packets are used for matching. When this bit is reset, the Source and Destination Port number fields of TCP packets are used for matching. The Layer 4 matching is done only when the L4SPM0 or L4DPM0 bit is set. 0b - Layer 4 Protocol is disabled 1b - Layer 4 Protocol is enabled
15-11 L3HDBM1	Layer 3 IP DA Higher Bits Match IPv4 Packets: This field contains the number of higher bits of IP Destination Address that are matched in the IPv4 packets. The following list describes the values of this field: - 0: No bits are masked. - 1: LSb[0] is masked - 2: Two LSbs [1:0] are masked - .. - 31: All bits except MSb are masked. IPv6 Packets: Bits[12:11] of this field correspond to Bits[6:5] of L3HSBM0 which indicate the number of lower bits of IP Source or Destination Address that are masked in the IPv6 packets. The following list describes the concatenated values of the L3HDBM0[1:0] and L3HSBM0 bits: - 0: No bits are masked. - 1: LSb[0] is masked. - 2: Two LSbs [1:0] are masked - .. - 127: All bits except MSb are masked. This field is valid and applicable only when the L3DAM0 or L3SAM0 bit is set.
10-6 L3HSBM1	Layer 3 IP SA Higher Bits Match IPv4 Packets: This field contains the number of lower bits of IP Source Address that are masked for matching in the IPv4 packets. The following list describes the values of this field: - 0: No bits are masked. - 1: LSb[0] is masked - 2: Two LSbs [1:0] are masked - .. - 31: All bits except MSb are masked. IPv6 Packets: This field contains Bits[4:0] of L3HSBM0. These bits indicate the number of higher bits of IP Source or Destination Address matched in the IPv6 packets. This field is valid and applicable only when the L3DAM0 or L3SAM0 bit is set high.
5 L3DAIM1	Layer 3 IP DA Inverse Match Enable When this bit is set, the Layer 3 IP Destination Address field is enabled for inverse matching. When this bit is reset, the Layer 3 IP Destination Address field is enabled for perfect matching. This bit is valid and applicable only when the L3DAM0 bit is set high. 0b - Layer 3 IP DA Inverse Match is disabled 1b - Layer 3 IP DA Inverse Match is enabled
4 L3DAM1	Layer 3 IP DA Match Enable When this bit is set, the Layer 3 IP Destination Address field is enabled for matching. When this bit is reset, the MAC ignores the Layer 3 IP Destination Address field for matching. Note: When the L3PEN0 bit is set, you should set either this bit or the L3SAM0 bit because either IPv6 DA or SA can be checked for filtering. 0b - Layer 3 IP DA Match is disabled 1b - Layer 3 IP DA Match is enabled
3 L3SAIM1	Layer 3 IP SA Inverse Match Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When this bit is set, the Layer 3 IP Source Address field is enabled for inverse matching. When this bit reset, the Layer 3 IP Source Address field is enabled for perfect matching. This bit is valid and applicable only when the L3SAM0 bit is set. 0b - Layer 3 IP SA Inverse Match is disabled 1b - Layer 3 IP SA Inverse Match is enabled
2 L3SAM1	Layer 3 IP SA Match Enable When this bit is set, the Layer 3 IP Source Address field is enabled for matching. When this bit is reset, the MAC ignores the Layer 3 IP Source Address field for matching. Note: When the L3PEN0 bit is set, you should set either this bit or the L3DAM0 bit because either IPv6 SA or DA can be checked for filtering. 0b - Layer 3 IP SA Match is disabled 1b - Layer 3 IP SA Match is enabled
1 —	Reserved
0 L3PEN1	Layer 3 Protocol Enable When this bit is set, the Layer 3 IP Source or Destination Address matching is enabled for IPv6 packets. When this bit is reset, the Layer 3 IP Source or Destination Address matching is enabled for IPv4 packets. The Layer 3 matching is done only when the L3SAM0 or L3DAM0 bit is set. 0b - Layer 3 Protocol is disabled 1b - Layer 3 Protocol is enabled

76.17.170 MAC Layer 4 Address 1 (MAC_Layer4_Address1)

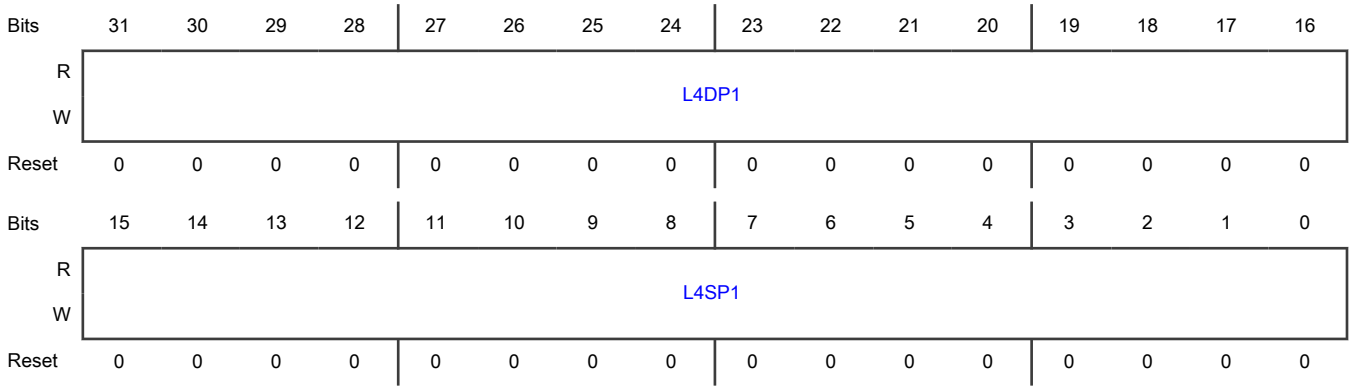
Offset

Register	Offset
MAC_Layer4_Address1	934h

Function

The MAC_Layer4_Address(#i), MAC_L3_L4_Control(#i), MAC_Layer3_Addr0_Reg(#i), MAC_Layer3_Addr1_Reg(#i), MAC_Layer3_Addr2_Reg(#i) and MAC_Layer3_Addr3_Reg(#i) registers are reserved (RO with default value) if Enable Layer 3 and Layer 4 Packet Filter option is not selected while configuring the IP. You can configure the Layer 3 and Layer 4 Address Registers to be double-synchronized by selecting the Synchronize Layer 3 and Layer 4 Address Registers to Rx Clock Domain option while configuring the IP. When you select this option, the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the Layer 3 and Layer 4 Address Registers are written. For proper synchronization updates, you should perform consecutive writes to same Layer 3 and Layer 4 Address Registers after at least four clock cycles delay of the destination clock.

Diagram



Fields

Field	Function
31-16 L4DP1	Layer 4 Destination Port Number Field When the L4PEN0 bit is reset and the L4DPM0 bit is set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with the TCP Destination Port Number field in the IPv4 or IPv6 packets. When the L4PEN0 and L4DPM0 bits are set in MAC_L3_L4_Control0 register, this field contains the value to be matched with the UDP Destination Port Number field in the IPv4 or IPv6 packets.
15-0 L4SP1	Layer 4 Source Port Number Field When the L4PEN0 bit is reset and the L4SPM0 bit is set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with the TCP Source Port Number field in the IPv4 or IPv6 packets. When the L4PEN0 and L4SPM0 bits are set in MAC_L3_L4_Control0 register, this field contains the value to be matched with the UDP Source Port Number field in the IPv4 or IPv6 packets.

76.17.171 MAC Layer 3 Address 0 Reg 1 (MAC_Layer3_Addr0_Reg1)

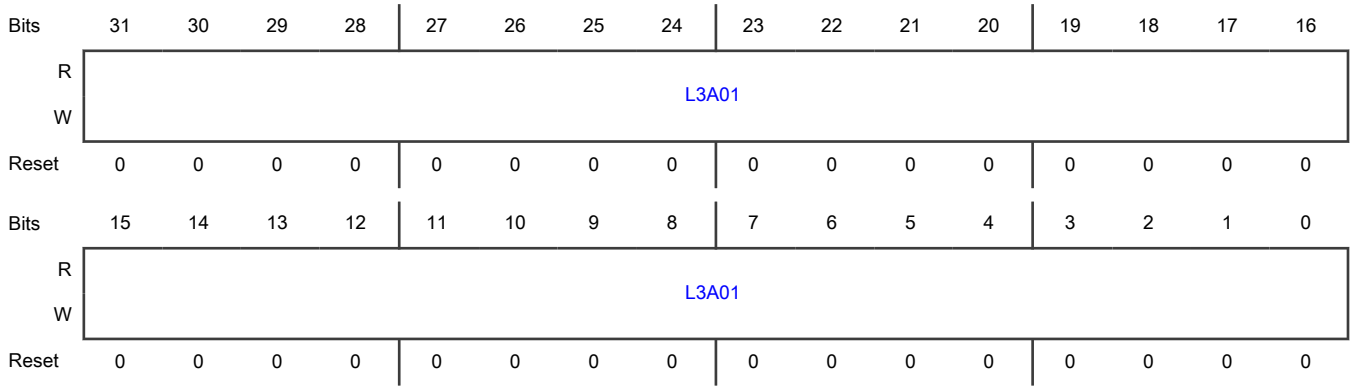
Offset

Register	Offset
MAC_Layer3_Addr0_Reg1	940h

Function

For IPv4 packets, the Layer 3 Address 0 Register 0 register contains the 32-bit IP Source Address field. For IPv6 packets, it contains Bits[31:0] of the 128-bit IP Source Address or Destination Address field.

Diagram



Fields

Field	Function
31-0 L3A01	Layer 3 Address 0 Field When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[31:0] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[31:0] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset and the L3SAM0 bit is set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with the IP Source Address field in the IPv4 packets.

76.17.172 MAC Layer 3 Address 1 Reg 1 (MAC_Layer3_Addr1_Reg1)

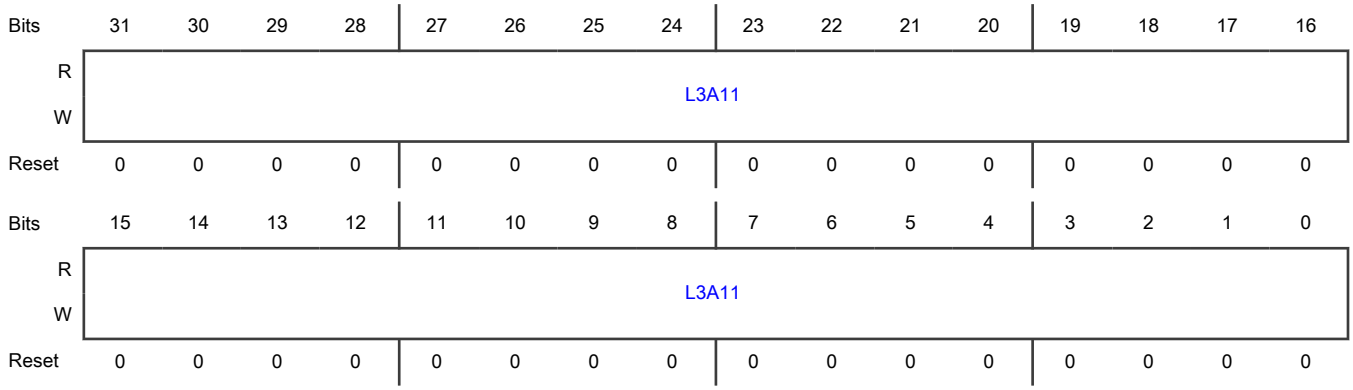
Offset

Register	Offset
MAC_Layer3_Addr1_Reg1	944h

Function

For IPv4 packets, the Layer 3 Address 1 Register 0 register contains the 32-bit IP Destination Address field. For IPv6 packets, it contains Bits[63:32] of the 128-bit IP Source Address or Destination Address field.

Diagram



Fields

Field	Function
31-0 L3A11	Layer 3 Address 1 Field When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[63:32] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[63:32] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset and the L3SAM0 bit is set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with the IP Destination Address field in the IPv4 packets.

76.17.173 MAC Layer 3 Address 2 Reg 1 (MAC_Layer3_Addr2_Reg1)

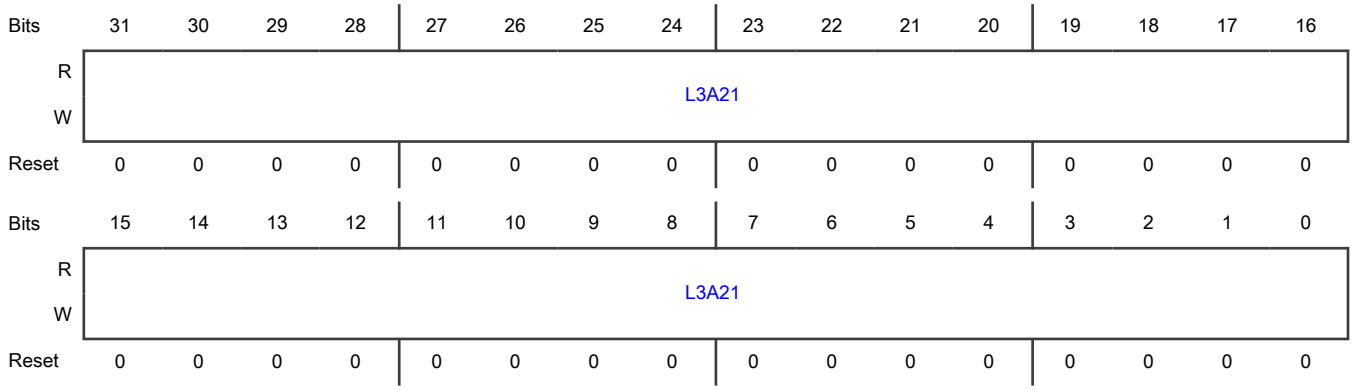
Offset

Register	Offset
MAC_Layer3_Addr2_Reg1	948h

Function

The Layer 3 Address 2 Register 0 register is reserved for IPv4 packets. For IPv6 packets, it contains Bits[95:64] of 128-bit IP Source Address or Destination Address field.

Diagram



Fields

Field	Function
31-0 L3A21	Layer 3 Address 2 Field When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[95:64] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[95:64] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset in the MAC_L3_L4_Control0 register, this field is not used.

76.17.174 MAC Layer 3 Address 3 Reg 1 (MAC_Layer3_Addr3_Reg1)

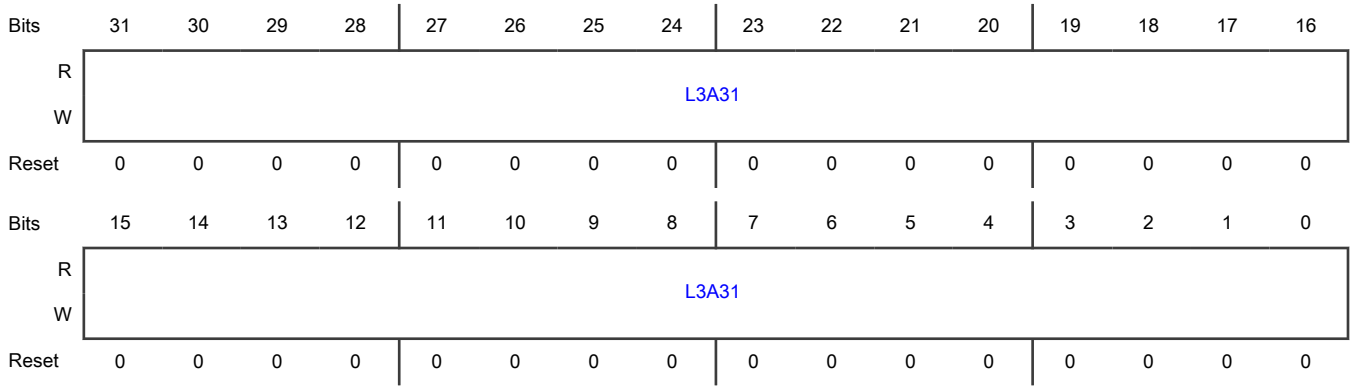
Offset

Register	Offset
MAC_Layer3_Addr3_Reg1	94Ch

Function

The Layer 3 Address 3 Register 0 register is reserved for IPv4 packets. For IPv6 packets, it contains Bits[127:96] of 128-bit IP Source Address or Destination Address field.

Diagram



Fields

Field	Function
31-0 L3A31	Layer 3 Address 3 Field When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[127:96] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[127:96] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset in the MAC_L3_L4_Control0 register, this field is not used.

76.17.175 MAC Layer 3 Layer 4 Control 2 (MAC_L3_L4_Control2)

Offset

Register	Offset
MAC_L3_L4_Control2	960h

Function
The Layer 3 and Layer 4 Control register controls the operations of filter 0 of Layer 3 and Layer 4.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved			DMCHEN2	Reserved		DMCHN2		Reserved		L4DPIM2	L4DPM2	L4SPIM2	L4SPM2	Reserved	L4PEN2
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	L3HDBM2				L3HSBM2				L3DAIM2	L3DAM2	L3SAIM2	L3SAM2	Reserved	L3PEN2		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	Reserved
28 DMCHEN2	DMA Channel Select Enable When set, this bit enables the selection of the DMA channel number for the packet that is passed by this L3_L4 filter. The DMA channel is indicated by the DMCHN bits. When this bit is reset, the DMA channel is not decided by this filter.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - DMA Channel Select is disabled 1b - DMA Channel Select is enabled
27-26 —	Reserved
25-24 DMCHN2	DMA Channel Number When DMCHEN is set high, this field selects the DMA Channel number to which the packet passed by this filter is routed. The width of this field depends on the number of the DMA channels present in your configuration.
23-22 —	Reserved
21 L4DPIM2	Layer 4 Destination Port Inverse Match Enable When this bit is set, the Layer 4 Destination Port number field is enabled for inverse matching. When this bit is reset, the Layer 4 Destination Port number field is enabled for perfect matching. This bit is valid and applicable only when the L4DPM0 bit is set high. 0b - Layer 4 Destination Port Inverse Match is disabled 1b - Layer 4 Destination Port Inverse Match is enabled
20 L4DPM2	Layer 4 Destination Port Match Enable When this bit is set, the Layer 4 Destination Port number field is enabled for matching. When this bit is reset, the MAC ignores the Layer 4 Destination Port number field for matching. 0b - Layer 4 Destination Port Match is disabled 1b - Layer 4 Destination Port Match is enabled
19 L4SPIM2	Layer 4 Source Port Inverse Match Enable When this bit is set, the Layer 4 Source Port number field is enabled for inverse matching. When this bit is reset, the Layer 4 Source Port number field is enabled for perfect matching. This bit is valid and applicable only when the L4SPM0 bit is set high. 0b - Layer 4 Source Port Inverse Match is disabled 1b - Layer 4 Source Port Inverse Match is enabled
18 L4SPM2	Layer 4 Source Port Match Enable When this bit is set, the Layer 4 Source Port number field is enabled for matching. When this bit is reset, the MAC ignores the Layer 4 Source Port number field for matching. 0b - Layer 4 Source Port Match is disabled 1b - Layer 4 Source Port Match is enabled
17	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
16 L4PEN2	Layer 4 Protocol Enable When this bit is set, the Source and Destination Port number fields of UDP packets are used for matching. When this bit is reset, the Source and Destination Port number fields of TCP packets are used for matching. The Layer 4 matching is done only when the L4SPM0 or L4DPM0 bit is set. 0b - Layer 4 Protocol is disabled 1b - Layer 4 Protocol is enabled
15-11 L3HDBM2	Layer 3 IP DA Higher Bits Match IPv4 Packets: This field contains the number of higher bits of IP Destination Address that are matched in the IPv4 packets. The following list describes the values of this field: - 0: No bits are masked. - 1: LSb[0] is masked - 2: Two LSbs [1:0] are masked - .. - 31: All bits except MSb are masked. IPv6 Packets: Bits[12:11] of this field correspond to Bits[6:5] of L3HSBM0 which indicate the number of lower bits of IP Source or Destination Address that are masked in the IPv6 packets. The following list describes the concatenated values of the L3HDBM0[1:0] and L3HSBM0 bits: - 0: No bits are masked. - 1: LSb[0] is masked. - 2: Two LSbs [1:0] are masked - .. - 127: All bits except MSb are masked. This field is valid and applicable only when the L3DAM0 or L3SAM0 bit is set.
10-6 L3HSBM2	Layer 3 IP SA Higher Bits Match IPv4 Packets: This field contains the number of lower bits of IP Source Address that are masked for matching in the IPv4 packets. The following list describes the values of this field: - 0: No bits are masked. - 1: LSb[0] is masked - 2: Two LSbs [1:0] are masked - .. - 31: All bits except MSb are masked. IPv6 Packets: This field contains Bits[4:0] of L3HSBM0. These bits indicate the number of higher bits of IP Source or Destination Address matched in the IPv6 packets. This field is valid and applicable only when the L3DAM0 or L3SAM0 bit is set high.
5 L3DAIM2	Layer 3 IP DA Inverse Match Enable When this bit is set, the Layer 3 IP Destination Address field is enabled for inverse matching. When this bit is reset, the Layer 3 IP Destination Address field is enabled for perfect matching. This bit is valid and applicable only when the L3DAM0 bit is set high. 0b - Layer 3 IP DA Inverse Match is disabled 1b - Layer 3 IP DA Inverse Match is enabled
4 L3DAM2	Layer 3 IP DA Match Enable When this bit is set, the Layer 3 IP Destination Address field is enabled for matching. When this bit is reset, the MAC ignores the Layer 3 IP Destination Address field for matching. Note: When the L3PEN0 bit is set, you should set either this bit or the L3SAM0 bit because either IPv6 DA or SA can be checked for filtering. 0b - Layer 3 IP DA Match is disabled 1b - Layer 3 IP DA Match is enabled
3 L3SAIM2	Layer 3 IP SA Inverse Match Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When this bit is set, the Layer 3 IP Source Address field is enabled for inverse matching. When this bit reset, the Layer 3 IP Source Address field is enabled for perfect matching. This bit is valid and applicable only when the L3SAM0 bit is set. 0b - Layer 3 IP SA Inverse Match is disabled 1b - Layer 3 IP SA Inverse Match is enabled
2 L3SAM2	Layer 3 IP SA Match Enable When this bit is set, the Layer 3 IP Source Address field is enabled for matching. When this bit is reset, the MAC ignores the Layer 3 IP Source Address field for matching. Note: When the L3PEN0 bit is set, you should set either this bit or the L3DAM0 bit because either IPv6 SA or DA can be checked for filtering. 0b - Layer 3 IP SA Match is disabled 1b - Layer 3 IP SA Match is enabled
1 —	Reserved
0 L3PEN2	Layer 3 Protocol Enable When this bit is set, the Layer 3 IP Source or Destination Address matching is enabled for IPv6 packets. When this bit is reset, the Layer 3 IP Source or Destination Address matching is enabled for IPv4 packets. The Layer 3 matching is done only when the L3SAM0 or L3DAM0 bit is set. 0b - Layer 3 Protocol is disabled 1b - Layer 3 Protocol is enabled

76.17.176 MAC Layer 4 Address 2 (MAC_Layer4_Address2)

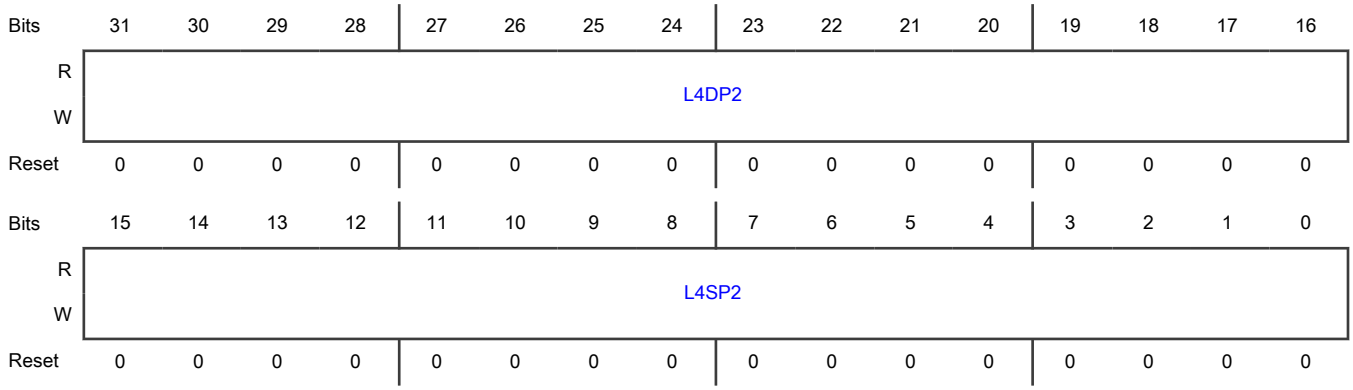
Offset

Register	Offset
MAC_Layer4_Address2	964h

Function

The MAC_Layer4_Address(#i), MAC_L3_L4_Control(#i), MAC_Layer3_Addr0_Reg(#i), MAC_Layer3_Addr1_Reg(#i), MAC_Layer3_Addr2_Reg(#i) and MAC_Layer3_Addr3_Reg(#i) registers are reserved (RO with default value) if Enable Layer 3 and Layer 4 Packet Filter option is not selected while configuring the IP. You can configure the Layer 3 and Layer 4 Address Registers to be double-synchronized by selecting the Synchronize Layer 3 and Layer 4 Address Registers to Rx Clock Domain option while configuring the IP. When you select this option, the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the Layer 3 and Layer 4 Address Registers are written. For proper synchronization updates, you should perform consecutive writes to same Layer 3 and Layer 4 Address Registers after at least four clock cycles delay of the destination clock.

Diagram



Fields

Field	Function
31-16 L4DP2	Layer 4 Destination Port Number Field When the L4PEN0 bit is reset and the L4DPM0 bit is set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with the TCP Destination Port Number field in the IPv4 or IPv6 packets. When the L4PEN0 and L4DPM0 bits are set in MAC_L3_L4_Control0 register, this field contains the value to be matched with the UDP Destination Port Number field in the IPv4 or IPv6 packets.
15-0 L4SP2	Layer 4 Source Port Number Field When the L4PEN0 bit is reset and the L4SPM0 bit is set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with the TCP Source Port Number field in the IPv4 or IPv6 packets. When the L4PEN0 and L4SPM0 bits are set in MAC_L3_L4_Control0 register, this field contains the value to be matched with the UDP Source Port Number field in the IPv4 or IPv6 packets.

76.17.177 MAC Layer 3 Address 0 Reg 2 (MAC_Layer3_Addr0_Reg2)

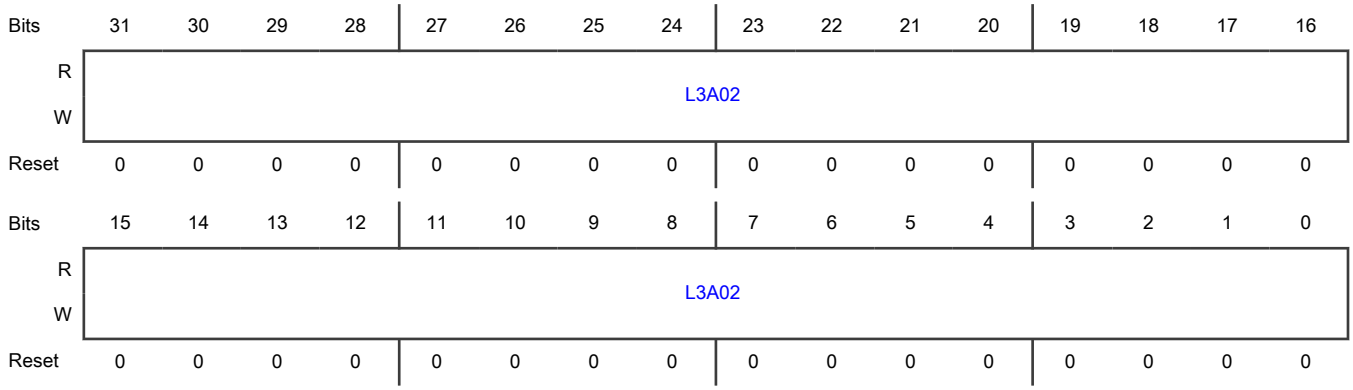
Offset

Register	Offset
MAC_Layer3_Addr0_Reg2	970h

Function

For IPv4 packets, the Layer 3 Address 0 Register 0 register contains the 32-bit IP Source Address field. For IPv6 packets, it contains Bits[31:0] of the 128-bit IP Source Address or Destination Address field.

Diagram



Fields

Field	Function
31-0 L3A02	Layer 3 Address 0 Field When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[31:0] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[31:0] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset and the L3SAM0 bit is set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with the IP Source Address field in the IPv4 packets.

76.17.178 MAC Layer 3 Address 1 Reg 2 (MAC_Layer3_Addr1_Reg2)

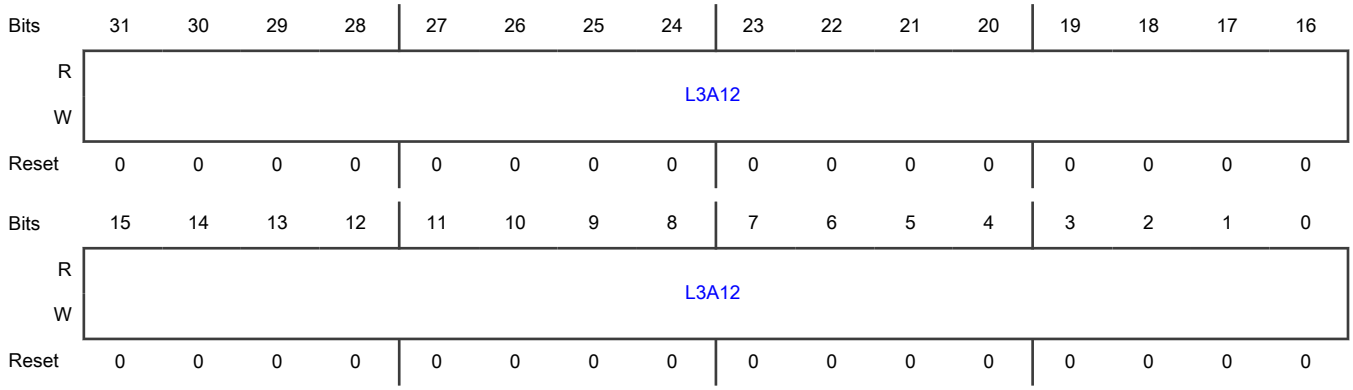
Offset

Register	Offset
MAC_Layer3_Addr1_Reg2	974h

Function

For IPv4 packets, the Layer 3 Address 1 Register 0 register contains the 32-bit IP Destination Address field. For IPv6 packets, it contains Bits[63:32] of the 128-bit IP Source Address or Destination Address field.

Diagram



Fields

Field	Function
31-0 L3A12	Layer 3 Address 1 Field When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[63:32] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[63:32] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset and the L3SAM0 bit is set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with the IP Destination Address field in the IPv4 packets.

76.17.179 MAC Layer 3 Address 2 Reg 2 (MAC_Layer3_Addr2_Reg2)

Offset

Register	Offset
MAC_Layer3_Addr2_Reg2	978h

Function

The Layer 3 Address 2 Register 0 register is reserved for IPv4 packets. For IPv6 packets, it contains Bits[95:64] of 128-bit IP Source Address or Destination Address field.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	L3A22															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	L3A22															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 L3A22	Layer 3 Address 2 Field When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[95:64] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[95:64] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset in the MAC_L3_L4_Control0 register, this field is not used.

76.17.180 MAC Layer 3 Address 3 Reg 2 (MAC_Layer3_Addr3_Reg2)

Offset

Register	Offset
MAC_Layer3_Addr3_Reg2	97Ch

Function

The Layer 3 Address 3 Register 0 register is reserved for IPv4 packets. For IPv6 packets, it contains Bits[127:96] of 128-bit IP Source Address or Destination Address field.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	L3A32															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	L3A32															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 L3A32	Layer 3 Address 3 Field When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[127:96] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[127:96] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset in the MAC_L3_L4_Control0 register, this field is not used.

76.17.181 MAC Layer 3 Layer 4 Control 3 (MAC_L3_L4_Control3)

Offset

Register	Offset
MAC_L3_L4_Control3	990h

Function
The Layer 3 and Layer 4 Control register controls the operations of filter 0 of Layer 3 and Layer 4.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved			DMCHEN3	Reserved		DMCHN3		Reserved		L4DPIM3	L4DPM3	L4SPIM3	L4SPM3	Reserved	L4PEN3
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	L3HDBM3				L3HSBM3						L3DAIM3	L3DAM3	L3SAIM3	L3SAM3	Reserved	L3PEN3
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	Reserved
28 DMCHEN3	DMA Channel Select Enable When set, this bit enables the selection of the DMA channel number for the packet that is passed by this L3_L4 filter. The DMA channel is indicated by the DMCHN bits. When this bit is reset, the DMA channel is not decided by this filter.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - DMA Channel Select is disabled 1b - DMA Channel Select is enabled
27-26 —	Reserved
25-24 DMCHN3	DMA Channel Number When DMCHEN is set high, this field selects the DMA Channel number to which the packet passed by this filter is routed. The width of this field depends on the number of the DMA channels present in your configuration.
23-22 —	Reserved
21 L4DPIM3	Layer 4 Destination Port Inverse Match Enable When this bit is set, the Layer 4 Destination Port number field is enabled for inverse matching. When this bit is reset, the Layer 4 Destination Port number field is enabled for perfect matching. This bit is valid and applicable only when the L4DPM0 bit is set high. 0b - Layer 4 Destination Port Inverse Match is disabled 1b - Layer 4 Destination Port Inverse Match is enabled
20 L4DPM3	Layer 4 Destination Port Match Enable When this bit is set, the Layer 4 Destination Port number field is enabled for matching. When this bit is reset, the MAC ignores the Layer 4 Destination Port number field for matching. 0b - Layer 4 Destination Port Match is disabled 1b - Layer 4 Destination Port Match is enabled
19 L4SPIM3	Layer 4 Source Port Inverse Match Enable When this bit is set, the Layer 4 Source Port number field is enabled for inverse matching. When this bit is reset, the Layer 4 Source Port number field is enabled for perfect matching. This bit is valid and applicable only when the L4SPM0 bit is set high. 0b - Layer 4 Source Port Inverse Match is disabled 1b - Layer 4 Source Port Inverse Match is enabled
18 L4SPM3	Layer 4 Source Port Match Enable When this bit is set, the Layer 4 Source Port number field is enabled for matching. When this bit is reset, the MAC ignores the Layer 4 Source Port number field for matching. 0b - Layer 4 Source Port Match is disabled 1b - Layer 4 Source Port Match is enabled
17	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
16 L4PEN3	Layer 4 Protocol Enable When this bit is set, the Source and Destination Port number fields of UDP packets are used for matching. When this bit is reset, the Source and Destination Port number fields of TCP packets are used for matching. The Layer 4 matching is done only when the L4SPM0 or L4DPM0 bit is set. 0b - Layer 4 Protocol is disabled 1b - Layer 4 Protocol is enabled
15-11 L3HDBM3	Layer 3 IP DA Higher Bits Match IPv4 Packets: This field contains the number of higher bits of IP Destination Address that are matched in the IPv4 packets. The following list describes the values of this field: - 0: No bits are masked. - 1: LSb[0] is masked - 2: Two LSbs [1:0] are masked - .. - 31: All bits except MSb are masked. IPv6 Packets: Bits[12:11] of this field correspond to Bits[6:5] of L3HSBM0 which indicate the number of lower bits of IP Source or Destination Address that are masked in the IPv6 packets. The following list describes the concatenated values of the L3HDBM0[1:0] and L3HSBM0 bits: - 0: No bits are masked. - 1: LSb[0] is masked. - 2: Two LSbs [1:0] are masked - .. - 127: All bits except MSb are masked. This field is valid and applicable only when the L3DAM0 or L3SAM0 bit is set.
10-6 L3HSBM3	Layer 3 IP SA Higher Bits Match IPv4 Packets: This field contains the number of lower bits of IP Source Address that are masked for matching in the IPv4 packets. The following list describes the values of this field: - 0: No bits are masked. - 1: LSb[0] is masked - 2: Two LSbs [1:0] are masked - .. - 31: All bits except MSb are masked. IPv6 Packets: This field contains Bits[4:0] of L3HSBM0. These bits indicate the number of higher bits of IP Source or Destination Address matched in the IPv6 packets. This field is valid and applicable only when the L3DAM0 or L3SAM0 bit is set high.
5 L3DAIM3	Layer 3 IP DA Inverse Match Enable When this bit is set, the Layer 3 IP Destination Address field is enabled for inverse matching. When this bit is reset, the Layer 3 IP Destination Address field is enabled for perfect matching. This bit is valid and applicable only when the L3DAM0 bit is set high. 0b - Layer 3 IP DA Inverse Match is disabled 1b - Layer 3 IP DA Inverse Match is enabled
4 L3DAM3	Layer 3 IP DA Match Enable When this bit is set, the Layer 3 IP Destination Address field is enabled for matching. When this bit is reset, the MAC ignores the Layer 3 IP Destination Address field for matching. Note: When the L3PEN0 bit is set, you should set either this bit or the L3SAM0 bit because either IPv6 DA or SA can be checked for filtering. 0b - Layer 3 IP DA Match is disabled 1b - Layer 3 IP DA Match is enabled
3 L3SAIM3	Layer 3 IP SA Inverse Match Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When this bit is set, the Layer 3 IP Source Address field is enabled for inverse matching. When this bit reset, the Layer 3 IP Source Address field is enabled for perfect matching. This bit is valid and applicable only when the L3SAM0 bit is set. 0b - Layer 3 IP SA Inverse Match is disabled 1b - Layer 3 IP SA Inverse Match is enabled
2 L3SAM3	Layer 3 IP SA Match Enable When this bit is set, the Layer 3 IP Source Address field is enabled for matching. When this bit is reset, the MAC ignores the Layer 3 IP Source Address field for matching. Note: When the L3PEN0 bit is set, you should set either this bit or the L3DAM0 bit because either IPv6 SA or DA can be checked for filtering. 0b - Layer 3 IP SA Match is disabled 1b - Layer 3 IP SA Match is enabled
1 —	Reserved
0 L3PEN3	Layer 3 Protocol Enable When this bit is set, the Layer 3 IP Source or Destination Address matching is enabled for IPv6 packets. When this bit is reset, the Layer 3 IP Source or Destination Address matching is enabled for IPv4 packets. The Layer 3 matching is done only when the L3SAM0 or L3DAM0 bit is set. 0b - Layer 3 Protocol is disabled 1b - Layer 3 Protocol is enabled

76.17.182 MAC Layer 4 Address 3 (MAC_Layer4_Address3)

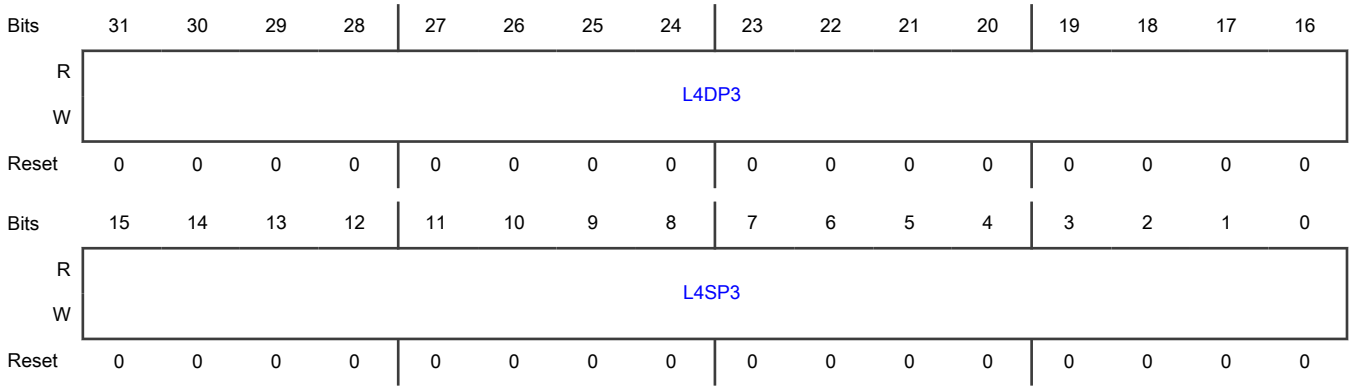
Offset

Register	Offset
MAC_Layer4_Address3	994h

Function

The MAC_Layer4_Address(#i), MAC_L3_L4_Control(#i), MAC_Layer3_Addr0_Reg(#i), MAC_Layer3_Addr1_Reg(#i), MAC_Layer3_Addr2_Reg(#i) and MAC_Layer3_Addr3_Reg(#i) registers are reserved (RO with default value) if Enable Layer 3 and Layer 4 Packet Filter option is not selected while configuring the IP. You can configure the Layer 3 and Layer 4 Address Registers to be double-synchronized by selecting the Synchronize Layer 3 and Layer 4 Address Registers to Rx Clock Domain option while configuring the IP. When you select this option, the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the Layer 3 and Layer 4 Address Registers are written. For proper synchronization updates, you should perform consecutive writes to same Layer 3 and Layer 4 Address Registers after at least four clock cycles delay of the destination clock.

Diagram



Fields

Field	Function
31-16 L4DP3	Layer 4 Destination Port Number Field When the L4PEN0 bit is reset and the L4DPM0 bit is set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with the TCP Destination Port Number field in the IPv4 or IPv6 packets. When the L4PEN0 and L4DPM0 bits are set in MAC_L3_L4_Control0 register, this field contains the value to be matched with the UDP Destination Port Number field in the IPv4 or IPv6 packets.
15-0 L4SP3	Layer 4 Source Port Number Field When the L4PEN0 bit is reset and the L4SPM0 bit is set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with the TCP Source Port Number field in the IPv4 or IPv6 packets. When the L4PEN0 and L4SPM0 bits are set in MAC_L3_L4_Control0 register, this field contains the value to be matched with the UDP Source Port Number field in the IPv4 or IPv6 packets.

76.17.183 MAC Layer 3 Address 0 Reg 3 (MAC_Layer3_Addr0_Reg3)

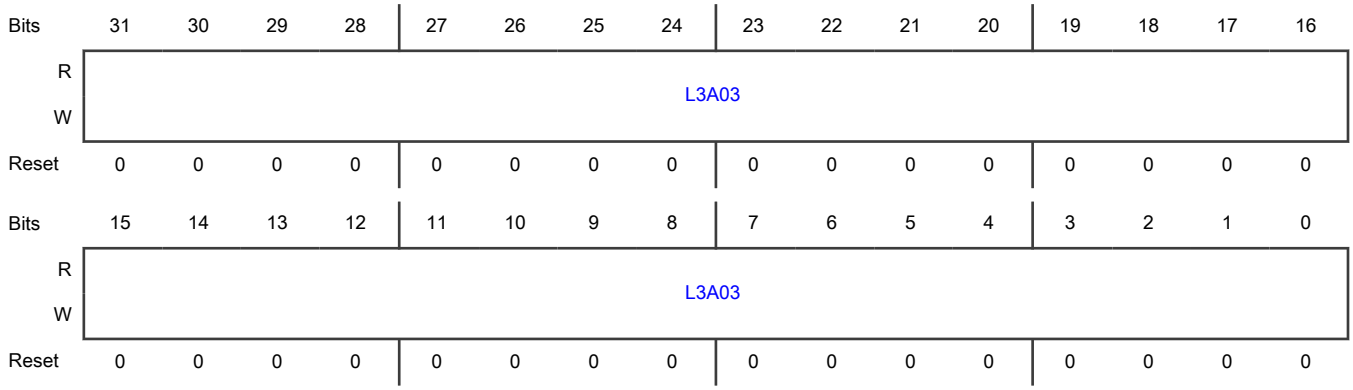
Offset

Register	Offset
MAC_Layer3_Addr0_Reg3	9A0h

Function

For IPv4 packets, the Layer 3 Address 0 Register 0 register contains the 32-bit IP Source Address field. For IPv6 packets, it contains Bits[31:0] of the 128-bit IP Source Address or Destination Address field.

Diagram



Fields

Field	Function
31-0 L3A03	Layer 3 Address 0 Field When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[31:0] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[31:0] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset and the L3SAM0 bit is set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with the IP Source Address field in the IPv4 packets.

76.17.184 MAC Layer 3 Address 1 Reg 3 (MAC_Layer3_Addr1_Reg3)

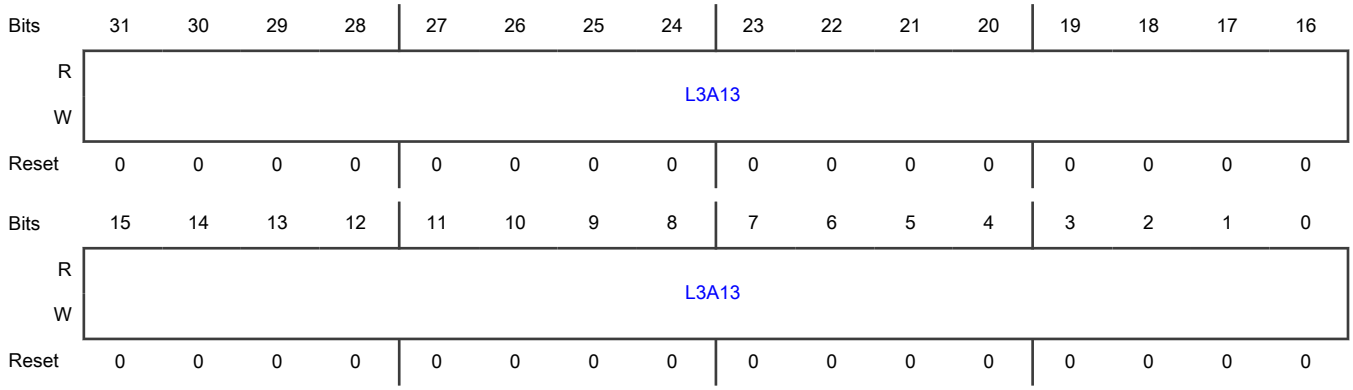
Offset

Register	Offset
MAC_Layer3_Addr1_Reg3	9A4h

Function

For IPv4 packets, the Layer 3 Address 1 Register 0 register contains the 32-bit IP Destination Address field. For IPv6 packets, it contains Bits[63:32] of the 128-bit IP Source Address or Destination Address field.

Diagram



Fields

Field	Function
31-0 L3A13	Layer 3 Address 1 Field When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[63:32] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[63:32] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset and the L3SAM0 bit is set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with the IP Destination Address field in the IPv4 packets.

76.17.185 MAC Layer 3 Address 2 Reg 3 (MAC_Layer3_Addr2_Reg3)

Offset

Register	Offset
MAC_Layer3_Addr2_Reg3	9A8h

Function

The Layer 3 Address 2 Register 0 register is reserved for IPv4 packets. For IPv6 packets, it contains Bits[95:64] of 128-bit IP Source Address or Destination Address field.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	L3A23															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	L3A23															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 L3A23	Layer 3 Address 2 Field When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[95:64] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[95:64] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset in the MAC_L3_L4_Control0 register, this field is not used.

76.17.186 MAC Layer 3 Address 3 Reg 3 (MAC_Layer3_Addr3_Reg3)

Offset

Register	Offset
MAC_Layer3_Addr3_Reg3	9ACh

Function

The Layer 3 Address 3 Register 0 register is reserved for IPv4 packets. For IPv6 packets, it contains Bits[127:96] of 128-bit IP Source Address or Destination Address field.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	L3A33															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	L3A33															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 L3A33	Layer 3 Address 3 Field When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[127:96] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[127:96] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset in the MAC_L3_L4_Control0 register, this field is not used.

76.17.187 MAC Layer 3 Layer 4 Control 4 (MAC_L3_L4_Control4)

Offset

Register	Offset
MAC_L3_L4_Control4	9C0h

Function
The Layer 3 and Layer 4 Control register controls the operations of filter 0 of Layer 3 and Layer 4.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved			DMCHEN4	Reserved		DMCHN4		Reserved		L4DPIM4	L4DPM4	L4SPIM4	L4SPM4	Reserved	L4PEN4
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	L3HDBM4				L3HSBM4				L3DAIM4	L3DAM4	L3SAIM4	L3SAM4	Reserved	L3PEN4		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	Reserved
28 DMCHEN4	DMA Channel Select Enable When set, this bit enables the selection of the DMA channel number for the packet that is passed by this L3_L4 filter. The DMA channel is indicated by the DMCHN bits. When this bit is reset, the DMA channel is not decided by this filter.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - DMA Channel Select is disabled 1b - DMA Channel Select is enabled
27-26 —	Reserved
25-24 DMCHN4	DMA Channel Number When DMCHEN is set high, this field selects the DMA Channel number to which the packet passed by this filter is routed. The width of this field depends on the number of the DMA channels present in your configuration.
23-22 —	Reserved
21 L4DPIM4	Layer 4 Destination Port Inverse Match Enable When this bit is set, the Layer 4 Destination Port number field is enabled for inverse matching. When this bit is reset, the Layer 4 Destination Port number field is enabled for perfect matching. This bit is valid and applicable only when the L4DPM0 bit is set high. 0b - Layer 4 Destination Port Inverse Match is disabled 1b - Layer 4 Destination Port Inverse Match is enabled
20 L4DPM4	Layer 4 Destination Port Match Enable When this bit is set, the Layer 4 Destination Port number field is enabled for matching. When this bit is reset, the MAC ignores the Layer 4 Destination Port number field for matching. 0b - Layer 4 Destination Port Match is disabled 1b - Layer 4 Destination Port Match is enabled
19 L4SPIM4	Layer 4 Source Port Inverse Match Enable When this bit is set, the Layer 4 Source Port number field is enabled for inverse matching. When this bit is reset, the Layer 4 Source Port number field is enabled for perfect matching. This bit is valid and applicable only when the L4SPM0 bit is set high. 0b - Layer 4 Source Port Inverse Match is disabled 1b - Layer 4 Source Port Inverse Match is enabled
18 L4SPM4	Layer 4 Source Port Match Enable When this bit is set, the Layer 4 Source Port number field is enabled for matching. When this bit is reset, the MAC ignores the Layer 4 Source Port number field for matching. 0b - Layer 4 Source Port Match is disabled 1b - Layer 4 Source Port Match is enabled
17	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
16 L4PEN4	Layer 4 Protocol Enable When this bit is set, the Source and Destination Port number fields of UDP packets are used for matching. When this bit is reset, the Source and Destination Port number fields of TCP packets are used for matching. The Layer 4 matching is done only when the L4SPM0 or L4DPM0 bit is set. 0b - Layer 4 Protocol is disabled 1b - Layer 4 Protocol is enabled
15-11 L3HDBM4	Layer 3 IP DA Higher Bits Match IPv4 Packets: This field contains the number of higher bits of IP Destination Address that are matched in the IPv4 packets. The following list describes the values of this field: - 0: No bits are masked. - 1: LSb[0] is masked - 2: Two LSbs [1:0] are masked - .. - 31: All bits except MSb are masked. IPv6 Packets: Bits[12:11] of this field correspond to Bits[6:5] of L3HSBM0 which indicate the number of lower bits of IP Source or Destination Address that are masked in the IPv6 packets. The following list describes the concatenated values of the L3HDBM0[1:0] and L3HSBM0 bits: - 0: No bits are masked. - 1: LSb[0] is masked. - 2: Two LSbs [1:0] are masked - .. - 127: All bits except MSb are masked. This field is valid and applicable only when the L3DAM0 or L3SAM0 bit is set.
10-6 L3HSBM4	Layer 3 IP SA Higher Bits Match IPv4 Packets: This field contains the number of lower bits of IP Source Address that are masked for matching in the IPv4 packets. The following list describes the values of this field: - 0: No bits are masked. - 1: LSb[0] is masked - 2: Two LSbs [1:0] are masked - .. - 31: All bits except MSb are masked. IPv6 Packets: This field contains Bits[4:0] of L3HSBM0. These bits indicate the number of higher bits of IP Source or Destination Address matched in the IPv6 packets. This field is valid and applicable only when the L3DAM0 or L3SAM0 bit is set high.
5 L3DAIM4	Layer 3 IP DA Inverse Match Enable When this bit is set, the Layer 3 IP Destination Address field is enabled for inverse matching. When this bit is reset, the Layer 3 IP Destination Address field is enabled for perfect matching. This bit is valid and applicable only when the L3DAM0 bit is set high. 0b - Layer 3 IP DA Inverse Match is disabled 1b - Layer 3 IP DA Inverse Match is enabled
4 L3DAM4	Layer 3 IP DA Match Enable When this bit is set, the Layer 3 IP Destination Address field is enabled for matching. When this bit is reset, the MAC ignores the Layer 3 IP Destination Address field for matching. Note: When the L3PEN0 bit is set, you should set either this bit or the L3SAM0 bit because either IPv6 DA or SA can be checked for filtering. 0b - Layer 3 IP DA Match is disabled 1b - Layer 3 IP DA Match is enabled
3 L3SAIM4	Layer 3 IP SA Inverse Match Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When this bit is set, the Layer 3 IP Source Address field is enabled for inverse matching. When this bit reset, the Layer 3 IP Source Address field is enabled for perfect matching. This bit is valid and applicable only when the L3SAM0 bit is set. 0b - Layer 3 IP SA Inverse Match is disabled 1b - Layer 3 IP SA Inverse Match is enabled
2 L3SAM4	Layer 3 IP SA Match Enable When this bit is set, the Layer 3 IP Source Address field is enabled for matching. When this bit is reset, the MAC ignores the Layer 3 IP Source Address field for matching. Note: When the L3PEN0 bit is set, you should set either this bit or the L3DAM0 bit because either IPv6 SA or DA can be checked for filtering. 0b - Layer 3 IP SA Match is disabled 1b - Layer 3 IP SA Match is enabled
1 —	Reserved
0 L3PEN4	Layer 3 Protocol Enable When this bit is set, the Layer 3 IP Source or Destination Address matching is enabled for IPv6 packets. When this bit is reset, the Layer 3 IP Source or Destination Address matching is enabled for IPv4 packets. The Layer 3 matching is done only when the L3SAM0 or L3DAM0 bit is set. 0b - Layer 3 Protocol is disabled 1b - Layer 3 Protocol is enabled

76.17.188 MAC Layer 4 Address 4 (MAC_Layer4_Address4)

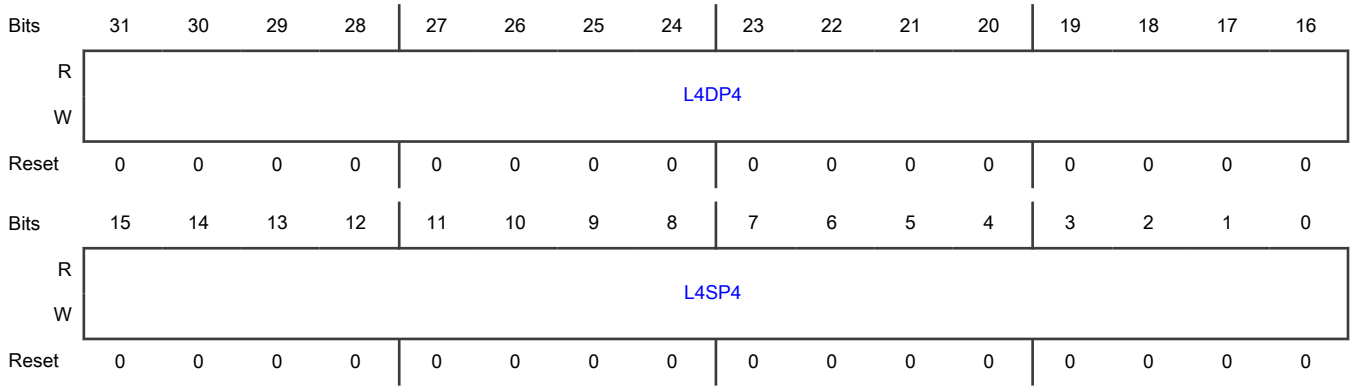
Offset

Register	Offset
MAC_Layer4_Address4	9C4h

Function

The MAC_Layer4_Address(#i), MAC_L3_L4_Control(#i), MAC_Layer3_Addr0_Reg(#i), MAC_Layer3_Addr1_Reg(#i), MAC_Layer3_Addr2_Reg(#i) and MAC_Layer3_Addr3_Reg(#i) registers are reserved (RO with default value) if Enable Layer 3 and Layer 4 Packet Filter option is not selected while configuring the IP. You can configure the Layer 3 and Layer 4 Address Registers to be double-synchronized by selecting the Synchronize Layer 3 and Layer 4 Address Registers to Rx Clock Domain option while configuring the IP. When you select this option, the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the Layer 3 and Layer 4 Address Registers are written. For proper synchronization updates, you should perform consecutive writes to same Layer 3 and Layer 4 Address Registers after at least four clock cycles delay of the destination clock.

Diagram



Fields

Field	Function
31-16 L4DP4	Layer 4 Destination Port Number Field When the L4PEN0 bit is reset and the L4DPM0 bit is set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with the TCP Destination Port Number field in the IPv4 or IPv6 packets. When the L4PEN0 and L4DPM0 bits are set in MAC_L3_L4_Control0 register, this field contains the value to be matched with the UDP Destination Port Number field in the IPv4 or IPv6 packets.
15-0 L4SP4	Layer 4 Source Port Number Field When the L4PEN0 bit is reset and the L4SPM0 bit is set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with the TCP Source Port Number field in the IPv4 or IPv6 packets. When the L4PEN0 and L4SPM0 bits are set in MAC_L3_L4_Control0 register, this field contains the value to be matched with the UDP Source Port Number field in the IPv4 or IPv6 packets.

76.17.189 MAC Layer 3 Address 0 Reg 4 (MAC_Layer3_Addr0_Reg4)

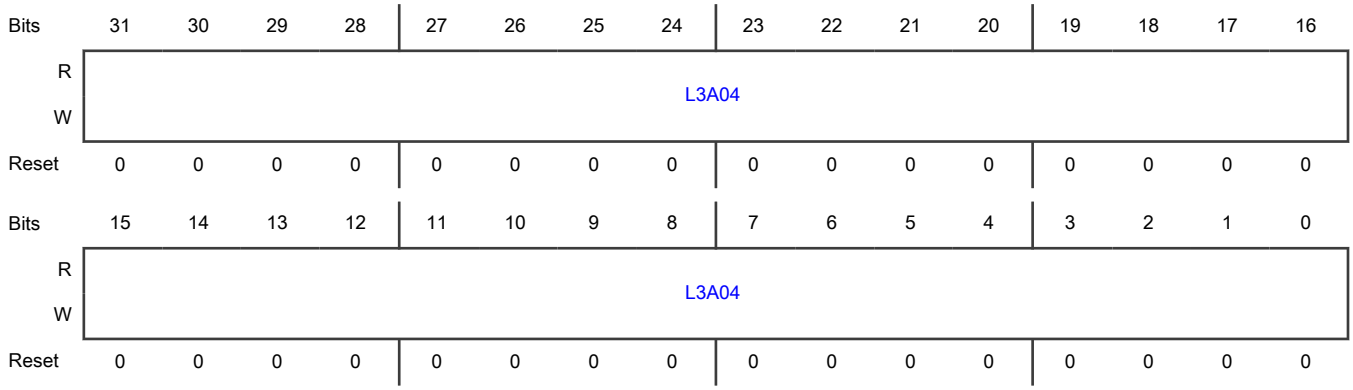
Offset

Register	Offset
MAC_Layer3_Addr0_Reg4	9D0h

Function

For IPv4 packets, the Layer 3 Address 0 Register 0 register contains the 32-bit IP Source Address field. For IPv6 packets, it contains Bits[31:0] of the 128-bit IP Source Address or Destination Address field.

Diagram



Fields

Field	Function
31-0 L3A04	Layer 3 Address 0 Field When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[31:0] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[31:0] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset and the L3SAM0 bit is set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with the IP Source Address field in the IPv4 packets.

76.17.190 MAC Layer 3 Address 1 Reg 4 (MAC_Layer3_Addr1_Reg4)

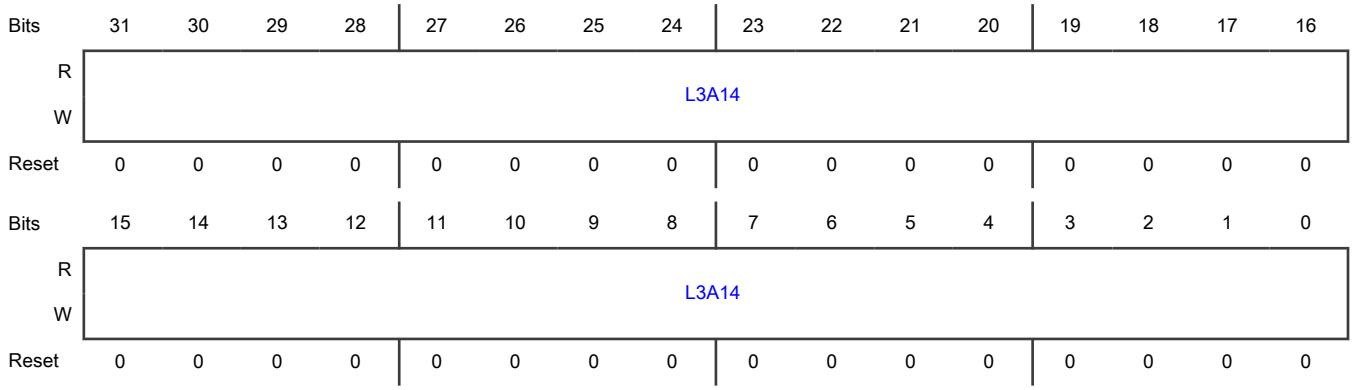
Offset

Register	Offset
MAC_Layer3_Addr1_Reg4	9D4h

Function

For IPv4 packets, the Layer 3 Address 1 Register 0 register contains the 32-bit IP Destination Address field. For IPv6 packets, it contains Bits[63:32] of the 128-bit IP Source Address or Destination Address field.

Diagram



Fields

Field	Function
31-0 L3A14	Layer 3 Address 1 Field When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[63:32] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[63:32] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset and the L3SAM0 bit is set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with the IP Destination Address field in the IPv4 packets.

76.17.191 MAC Layer 3 Address 2 Reg 4 (MAC_Layer3_Addr2_Reg4)

Offset

Register	Offset
MAC_Layer3_Addr2_Reg4	9D8h

Function

The Layer 3 Address 2 Register 0 register is reserved for IPv4 packets. For IPv6 packets, it contains Bits[95:64] of 128-bit IP Source Address or Destination Address field.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	L3A24															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	L3A24															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 L3A24	Layer 3 Address 2 Field When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[95:64] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[95:64] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset in the MAC_L3_L4_Control0 register, this field is not used.

76.17.192 MAC Layer 3 Address 3 Reg 4 (MAC_Layer3_Addr3_Reg4)

Offset

Register	Offset
MAC_Layer3_Addr3_Reg4	9DCh

Function

The Layer 3 Address 3 Register 0 register is reserved for IPv4 packets. For IPv6 packets, it contains Bits[127:96] of 128-bit IP Source Address or Destination Address field.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	L3A34															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	L3A34															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 L3A34	Layer 3 Address 3 Field When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[127:96] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[127:96] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset in the MAC_L3_L4_Control0 register, this field is not used.

76.17.193 MAC Layer 3 Layer 4 Control 5 (MAC_L3_L4_Control5)

Offset

Register	Offset
MAC_L3_L4_Control5	9F0h

Function
The Layer 3 and Layer 4 Control register controls the operations of filter 0 of Layer 3 and Layer 4.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved			DMCHEN5	Reserved		DMCHN5		Reserved		L4DPIM5	L4DPM5	L4SPIM5	L4SPM5	Reserved	L4PEN5
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	L3HDBM5				L3HSBM5				L3DAIM5	L3DAM5	L3SAIM5	L3SAM5	Reserved	L3PEN5		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	Reserved
28 DMCHEN5	DMA Channel Select Enable When set, this bit enables the selection of the DMA channel number for the packet that is passed by this L3_L4 filter. The DMA channel is indicated by the DMCHN bits. When this bit is reset, the DMA channel is not decided by this filter.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - DMA Channel Select is disabled 1b - DMA Channel Select is enabled
27-26 —	Reserved
25-24 DMCHN5	DMA Channel Number When DMCHEN is set high, this field selects the DMA Channel number to which the packet passed by this filter is routed. The width of this field depends on the number of the DMA channels present in your configuration.
23-22 —	Reserved
21 L4DPIM5	Layer 4 Destination Port Inverse Match Enable When this bit is set, the Layer 4 Destination Port number field is enabled for inverse matching. When this bit is reset, the Layer 4 Destination Port number field is enabled for perfect matching. This bit is valid and applicable only when the L4DPM0 bit is set high. 0b - Layer 4 Destination Port Inverse Match is disabled 1b - Layer 4 Destination Port Inverse Match is enabled
20 L4DPM5	Layer 4 Destination Port Match Enable When this bit is set, the Layer 4 Destination Port number field is enabled for matching. When this bit is reset, the MAC ignores the Layer 4 Destination Port number field for matching. 0b - Layer 4 Destination Port Match is disabled 1b - Layer 4 Destination Port Match is enabled
19 L4SPIM5	Layer 4 Source Port Inverse Match Enable When this bit is set, the Layer 4 Source Port number field is enabled for inverse matching. When this bit is reset, the Layer 4 Source Port number field is enabled for perfect matching. This bit is valid and applicable only when the L4SPM0 bit is set high. 0b - Layer 4 Source Port Inverse Match is disabled 1b - Layer 4 Source Port Inverse Match is enabled
18 L4SPM5	Layer 4 Source Port Match Enable When this bit is set, the Layer 4 Source Port number field is enabled for matching. When this bit is reset, the MAC ignores the Layer 4 Source Port number field for matching. 0b - Layer 4 Source Port Match is disabled 1b - Layer 4 Source Port Match is enabled
17	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
16 L4PEN5	Layer 4 Protocol Enable When this bit is set, the Source and Destination Port number fields of UDP packets are used for matching. When this bit is reset, the Source and Destination Port number fields of TCP packets are used for matching. The Layer 4 matching is done only when the L4SPM0 or L4DPM0 bit is set. 0b - Layer 4 Protocol is disabled 1b - Layer 4 Protocol is enabled
15-11 L3HDBM5	Layer 3 IP DA Higher Bits Match IPv4 Packets: This field contains the number of higher bits of IP Destination Address that are matched in the IPv4 packets. The following list describes the values of this field: - 0: No bits are masked. - 1: LSb[0] is masked - 2: Two LSbs [1:0] are masked - .. - 31: All bits except MSb are masked. IPv6 Packets: Bits[12:11] of this field correspond to Bits[6:5] of L3HSBM0 which indicate the number of lower bits of IP Source or Destination Address that are masked in the IPv6 packets. The following list describes the concatenated values of the L3HDBM0[1:0] and L3HSBM0 bits: - 0: No bits are masked. - 1: LSb[0] is masked. - 2: Two LSbs [1:0] are masked - .. - 127: All bits except MSb are masked. This field is valid and applicable only when the L3DAM0 or L3SAM0 bit is set.
10-6 L3HSBM5	Layer 3 IP SA Higher Bits Match IPv4 Packets: This field contains the number of lower bits of IP Source Address that are masked for matching in the IPv4 packets. The following list describes the values of this field: - 0: No bits are masked. - 1: LSb[0] is masked - 2: Two LSbs [1:0] are masked - .. - 31: All bits except MSb are masked. IPv6 Packets: This field contains Bits[4:0] of L3HSBM0. These bits indicate the number of higher bits of IP Source or Destination Address matched in the IPv6 packets. This field is valid and applicable only when the L3DAM0 or L3SAM0 bit is set high.
5 L3DAIM5	Layer 3 IP DA Inverse Match Enable When this bit is set, the Layer 3 IP Destination Address field is enabled for inverse matching. When this bit is reset, the Layer 3 IP Destination Address field is enabled for perfect matching. This bit is valid and applicable only when the L3DAM0 bit is set high. 0b - Layer 3 IP DA Inverse Match is disabled 1b - Layer 3 IP DA Inverse Match is enabled
4 L3DAM5	Layer 3 IP DA Match Enable When this bit is set, the Layer 3 IP Destination Address field is enabled for matching. When this bit is reset, the MAC ignores the Layer 3 IP Destination Address field for matching. Note: When the L3PEN0 bit is set, you should set either this bit or the L3SAM0 bit because either IPv6 DA or SA can be checked for filtering. 0b - Layer 3 IP DA Match is disabled 1b - Layer 3 IP DA Match is enabled
3 L3SAIM5	Layer 3 IP SA Inverse Match Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When this bit is set, the Layer 3 IP Source Address field is enabled for inverse matching. When this bit reset, the Layer 3 IP Source Address field is enabled for perfect matching. This bit is valid and applicable only when the L3SAM0 bit is set. 0b - Layer 3 IP SA Inverse Match is disabled 1b - Layer 3 IP SA Inverse Match is enabled
2 L3SAM5	Layer 3 IP SA Match Enable When this bit is set, the Layer 3 IP Source Address field is enabled for matching. When this bit is reset, the MAC ignores the Layer 3 IP Source Address field for matching. Note: When the L3PEN0 bit is set, you should set either this bit or the L3DAM0 bit because either IPv6 SA or DA can be checked for filtering. 0b - Layer 3 IP SA Match is disabled 1b - Layer 3 IP SA Match is enabled
1 —	Reserved
0 L3PEN5	Layer 3 Protocol Enable When this bit is set, the Layer 3 IP Source or Destination Address matching is enabled for IPv6 packets. When this bit is reset, the Layer 3 IP Source or Destination Address matching is enabled for IPv4 packets. The Layer 3 matching is done only when the L3SAM0 or L3DAM0 bit is set. 0b - Layer 3 Protocol is disabled 1b - Layer 3 Protocol is enabled

76.17.194 MAC Layer 4 Address 5 (MAC_Layer4_Address5)

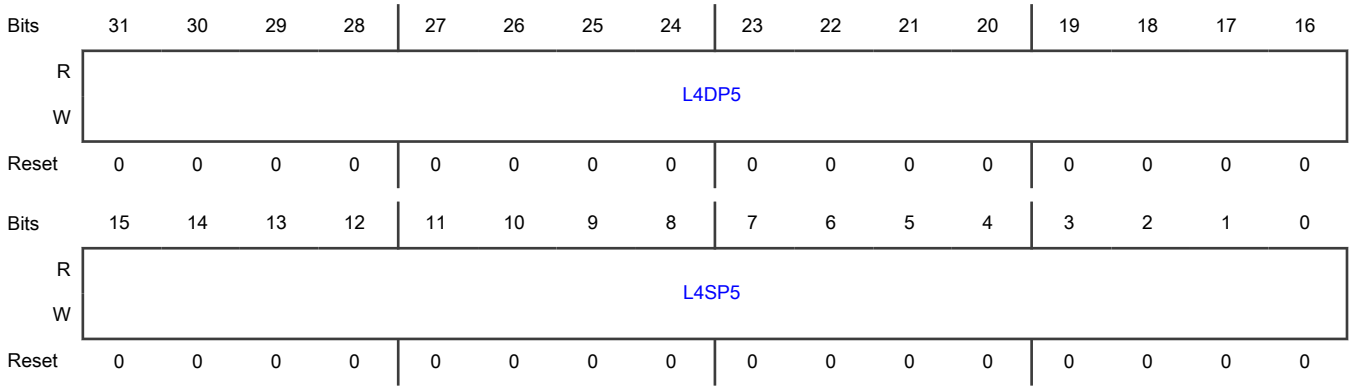
Offset

Register	Offset
MAC_Layer4_Address5	9F4h

Function

The MAC_Layer4_Address(#i), MAC_L3_L4_Control(#i), MAC_Layer3_Addr0_Reg(#i), MAC_Layer3_Addr1_Reg(#i), MAC_Layer3_Addr2_Reg(#i) and MAC_Layer3_Addr3_Reg(#i) registers are reserved (RO with default value) if Enable Layer 3 and Layer 4 Packet Filter option is not selected while configuring the IP. You can configure the Layer 3 and Layer 4 Address Registers to be double-synchronized by selecting the Synchronize Layer 3 and Layer 4 Address Registers to Rx Clock Domain option while configuring the IP. When you select this option, the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the Layer 3 and Layer 4 Address Registers are written. For proper synchronization updates, you should perform consecutive writes to same Layer 3 and Layer 4 Address Registers after at least four clock cycles delay of the destination clock.

Diagram



Fields

Field	Function
31-16 L4DP5	Layer 4 Destination Port Number Field When the L4PEN0 bit is reset and the L4DPM0 bit is set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with the TCP Destination Port Number field in the IPv4 or IPv6 packets. When the L4PEN0 and L4DPM0 bits are set in MAC_L3_L4_Control0 register, this field contains the value to be matched with the UDP Destination Port Number field in the IPv4 or IPv6 packets.
15-0 L4SP5	Layer 4 Source Port Number Field When the L4PEN0 bit is reset and the L4SPM0 bit is set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with the TCP Source Port Number field in the IPv4 or IPv6 packets. When the L4PEN0 and L4SPM0 bits are set in MAC_L3_L4_Control0 register, this field contains the value to be matched with the UDP Source Port Number field in the IPv4 or IPv6 packets.

76.17.195 MAC Layer 3 Address 0 Reg 5 (MAC_Layer3_Addr0_Reg5)

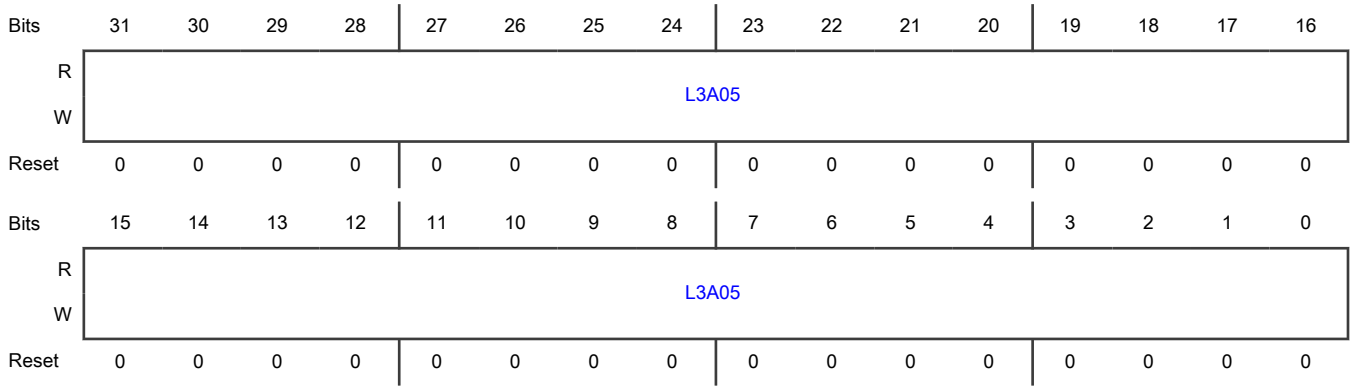
Offset

Register	Offset
MAC_Layer3_Addr0_Reg5	A00h

Function

For IPv4 packets, the Layer 3 Address 0 Register 0 register contains the 32-bit IP Source Address field. For IPv6 packets, it contains Bits[31:0] of the 128-bit IP Source Address or Destination Address field.

Diagram



Fields

Field	Function
31-0 L3A05	Layer 3 Address 0 Field When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[31:0] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[31:0] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset and the L3SAM0 bit is set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with the IP Source Address field in the IPv4 packets.

76.17.196 MAC Layer 3 Address 1 Reg 5 (MAC_Layer3_Addr1_Reg5)

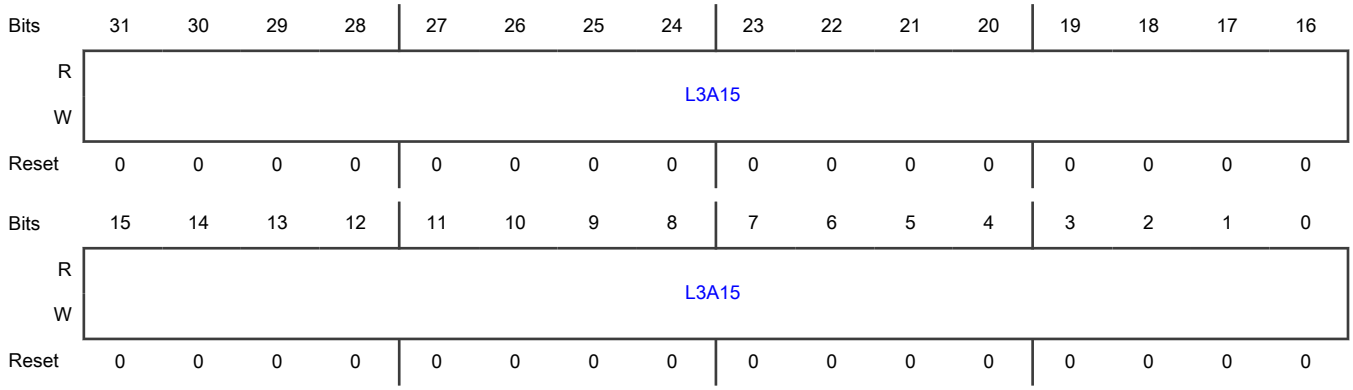
Offset

Register	Offset
MAC_Layer3_Addr1_Reg5	A04h

Function

For IPv4 packets, the Layer 3 Address 1 Register 0 register contains the 32-bit IP Destination Address field. For IPv6 packets, it contains Bits[63:32] of the 128-bit IP Source Address or Destination Address field.

Diagram



Fields

Field	Function
31-0 L3A15	Layer 3 Address 1 Field When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[63:32] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[63:32] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset and the L3SAM0 bit is set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with the IP Destination Address field in the IPv4 packets.

76.17.197 MAC Layer 3 Address 2 Reg 5 (MAC_Layer3_Addr2_Reg5)

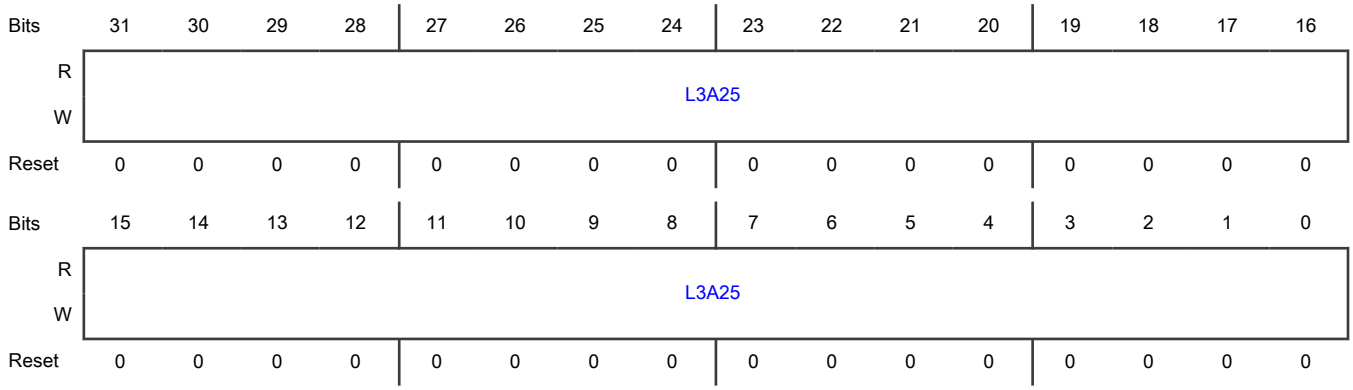
Offset

Register	Offset
MAC_Layer3_Addr2_Reg5	A08h

Function

The Layer 3 Address 2 Register 0 register is reserved for IPv4 packets. For IPv6 packets, it contains Bits[95:64] of 128-bit IP Source Address or Destination Address field.

Diagram



Fields

Field	Function
31-0 L3A25	Layer 3 Address 2 Field When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[95:64] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[95:64] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset in the MAC_L3_L4_Control0 register, this field is not used.

76.17.198 MAC Layer 3 Address 3 Reg 5 (MAC_Layer3_Addr3_Reg5)

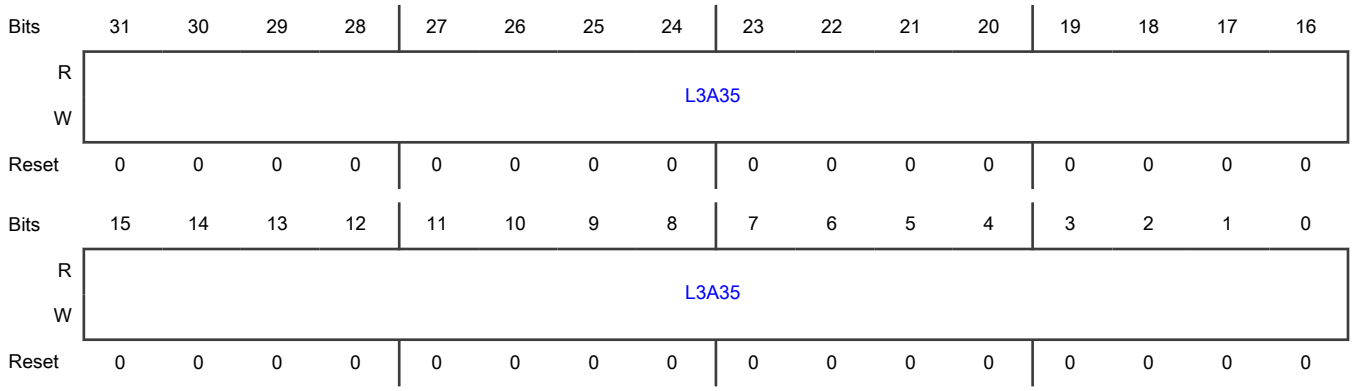
Offset

Register	Offset
MAC_Layer3_Addr3_Reg5	A0Ch

Function

The Layer 3 Address 3 Register 0 register is reserved for IPv4 packets. For IPv6 packets, it contains Bits[127:96] of 128-bit IP Source Address or Destination Address field.

Diagram



Fields

Field	Function
31-0 L3A35	Layer 3 Address 3 Field When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[127:96] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[127:96] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset in the MAC_L3_L4_Control0 register, this field is not used.

76.17.199 MAC Layer 3 Layer 4 Control 6 (MAC_L3_L4_Control6)

Offset

Register	Offset
MAC_L3_L4_Control6	A20h

Function
The Layer 3 and Layer 4 Control register controls the operations of filter 0 of Layer 3 and Layer 4.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved			DMCHEN6	Reserved		DMCHN6		Reserved		L4DPIM6	L4DPM6	L4SPIM6	L4SPM6	Reserved	L4PEN6
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	L3HDBM6				L3HSBM6				L3DAIM6	L3DAM6	L3SAIM6	L3SAM6	Reserved	L3PEN6		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	Reserved
28 DMCHEN6	DMA Channel Select Enable When set, this bit enables the selection of the DMA channel number for the packet that is passed by this L3_L4 filter. The DMA channel is indicated by the DMCHN bits. When this bit is reset, the DMA channel is not decided by this filter.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - DMA Channel Select is disabled 1b - DMA Channel Select is enabled
27-26 —	Reserved
25-24 DMCHN6	DMA Channel Number When DMCHEN is set high, this field selects the DMA Channel number to which the packet passed by this filter is routed. The width of this field depends on the number of the DMA channels present in your configuration.
23-22 —	Reserved
21 L4DPIM6	Layer 4 Destination Port Inverse Match Enable When this bit is set, the Layer 4 Destination Port number field is enabled for inverse matching. When this bit is reset, the Layer 4 Destination Port number field is enabled for perfect matching. This bit is valid and applicable only when the L4DPM0 bit is set high. 0b - Layer 4 Destination Port Inverse Match is disabled 1b - Layer 4 Destination Port Inverse Match is enabled
20 L4DPM6	Layer 4 Destination Port Match Enable When this bit is set, the Layer 4 Destination Port number field is enabled for matching. When this bit is reset, the MAC ignores the Layer 4 Destination Port number field for matching. 0b - Layer 4 Destination Port Match is disabled 1b - Layer 4 Destination Port Match is enabled
19 L4SPIM6	Layer 4 Source Port Inverse Match Enable When this bit is set, the Layer 4 Source Port number field is enabled for inverse matching. When this bit is reset, the Layer 4 Source Port number field is enabled for perfect matching. This bit is valid and applicable only when the L4SPM0 bit is set high. 0b - Layer 4 Source Port Inverse Match is disabled 1b - Layer 4 Source Port Inverse Match is enabled
18 L4SPM6	Layer 4 Source Port Match Enable When this bit is set, the Layer 4 Source Port number field is enabled for matching. When this bit is reset, the MAC ignores the Layer 4 Source Port number field for matching. 0b - Layer 4 Source Port Match is disabled 1b - Layer 4 Source Port Match is enabled
17	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
16 L4PEN6	Layer 4 Protocol Enable When this bit is set, the Source and Destination Port number fields of UDP packets are used for matching. When this bit is reset, the Source and Destination Port number fields of TCP packets are used for matching. The Layer 4 matching is done only when the L4SPM0 or L4DPM0 bit is set. 0b - Layer 4 Protocol is disabled 1b - Layer 4 Protocol is enabled
15-11 L3HDBM6	Layer 3 IP DA Higher Bits Match IPv4 Packets: This field contains the number of higher bits of IP Destination Address that are matched in the IPv4 packets. The following list describes the values of this field: - 0: No bits are masked. - 1: LSb[0] is masked - 2: Two LSbs [1:0] are masked - .. - 31: All bits except MSb are masked. IPv6 Packets: Bits[12:11] of this field correspond to Bits[6:5] of L3HSBM0 which indicate the number of lower bits of IP Source or Destination Address that are masked in the IPv6 packets. The following list describes the concatenated values of the L3HDBM0[1:0] and L3HSBM0 bits: - 0: No bits are masked. - 1: LSb[0] is masked. - 2: Two LSbs [1:0] are masked - .. - 127: All bits except MSb are masked. This field is valid and applicable only when the L3DAM0 or L3SAM0 bit is set.
10-6 L3HSBM6	Layer 3 IP SA Higher Bits Match IPv4 Packets: This field contains the number of lower bits of IP Source Address that are masked for matching in the IPv4 packets. The following list describes the values of this field: - 0: No bits are masked. - 1: LSb[0] is masked - 2: Two LSbs [1:0] are masked - .. - 31: All bits except MSb are masked. IPv6 Packets: This field contains Bits[4:0] of L3HSBM0. These bits indicate the number of higher bits of IP Source or Destination Address matched in the IPv6 packets. This field is valid and applicable only when the L3DAM0 or L3SAM0 bit is set high.
5 L3DAIM6	Layer 3 IP DA Inverse Match Enable When this bit is set, the Layer 3 IP Destination Address field is enabled for inverse matching. When this bit is reset, the Layer 3 IP Destination Address field is enabled for perfect matching. This bit is valid and applicable only when the L3DAM0 bit is set high. 0b - Layer 3 IP DA Inverse Match is disabled 1b - Layer 3 IP DA Inverse Match is enabled
4 L3DAM6	Layer 3 IP DA Match Enable When this bit is set, the Layer 3 IP Destination Address field is enabled for matching. When this bit is reset, the MAC ignores the Layer 3 IP Destination Address field for matching. Note: When the L3PEN0 bit is set, you should set either this bit or the L3SAM0 bit because either IPv6 DA or SA can be checked for filtering. 0b - Layer 3 IP DA Match is disabled 1b - Layer 3 IP DA Match is enabled
3 L3SAIM6	Layer 3 IP SA Inverse Match Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When this bit is set, the Layer 3 IP Source Address field is enabled for inverse matching. When this bit reset, the Layer 3 IP Source Address field is enabled for perfect matching. This bit is valid and applicable only when the L3SAM0 bit is set. 0b - Layer 3 IP SA Inverse Match is disabled 1b - Layer 3 IP SA Inverse Match is enabled
2 L3SAM6	Layer 3 IP SA Match Enable When this bit is set, the Layer 3 IP Source Address field is enabled for matching. When this bit is reset, the MAC ignores the Layer 3 IP Source Address field for matching. Note: When the L3PEN0 bit is set, you should set either this bit or the L3DAM0 bit because either IPv6 SA or DA can be checked for filtering. 0b - Layer 3 IP SA Match is disabled 1b - Layer 3 IP SA Match is enabled
1 —	Reserved
0 L3PEN6	Layer 3 Protocol Enable When this bit is set, the Layer 3 IP Source or Destination Address matching is enabled for IPv6 packets. When this bit is reset, the Layer 3 IP Source or Destination Address matching is enabled for IPv4 packets. The Layer 3 matching is done only when the L3SAM0 or L3DAM0 bit is set. 0b - Layer 3 Protocol is disabled 1b - Layer 3 Protocol is enabled

76.17.200 MAC Layer 4 Address 6 (MAC_Layer4_Address6)

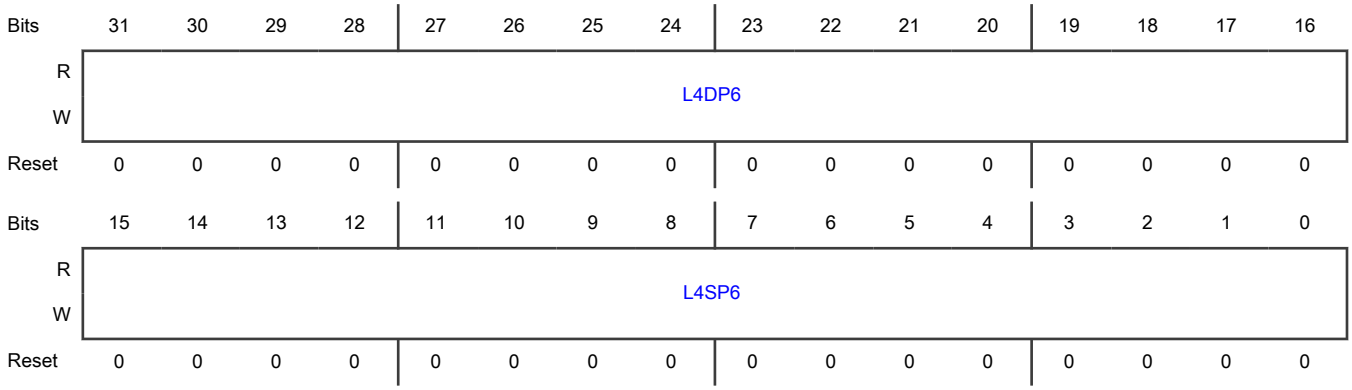
Offset

Register	Offset
MAC_Layer4_Address6	A24h

Function

The MAC_Layer4_Address(#i), MAC_L3_L4_Control(#i), MAC_Layer3_Addr0_Reg(#i), MAC_Layer3_Addr1_Reg(#i), MAC_Layer3_Addr2_Reg(#i) and MAC_Layer3_Addr3_Reg(#i) registers are reserved (RO with default value) if Enable Layer 3 and Layer 4 Packet Filter option is not selected while configuring the IP. You can configure the Layer 3 and Layer 4 Address Registers to be double-synchronized by selecting the Synchronize Layer 3 and Layer 4 Address Registers to Rx Clock Domain option while configuring the IP. When you select this option, the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the Layer 3 and Layer 4 Address Registers are written. For proper synchronization updates, you should perform consecutive writes to same Layer 3 and Layer 4 Address Registers after at least four clock cycles delay of the destination clock.

Diagram



Fields

Field	Function
31-16 L4DP6	Layer 4 Destination Port Number Field When the L4PEN0 bit is reset and the L4DPM0 bit is set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with the TCP Destination Port Number field in the IPv4 or IPv6 packets. When the L4PEN0 and L4DPM0 bits are set in MAC_L3_L4_Control0 register, this field contains the value to be matched with the UDP Destination Port Number field in the IPv4 or IPv6 packets.
15-0 L4SP6	Layer 4 Source Port Number Field When the L4PEN0 bit is reset and the L4SPM0 bit is set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with the TCP Source Port Number field in the IPv4 or IPv6 packets. When the L4PEN0 and L4SPM0 bits are set in MAC_L3_L4_Control0 register, this field contains the value to be matched with the UDP Source Port Number field in the IPv4 or IPv6 packets.

76.17.201 MAC Layer 3 Address 0 Reg 6 (MAC_Layer3_Addr0_Reg6)

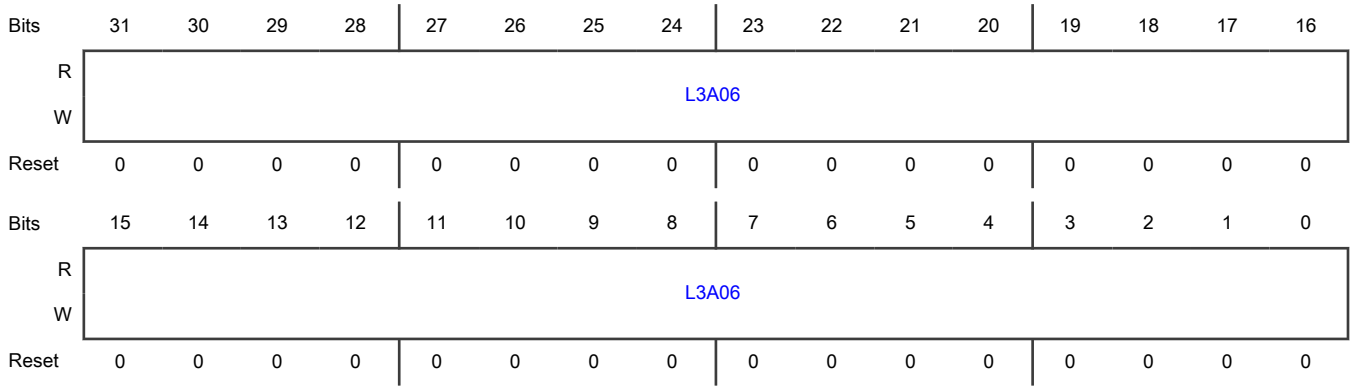
Offset

Register	Offset
MAC_Layer3_Addr0_Reg6	A30h

Function

For IPv4 packets, the Layer 3 Address 0 Register 0 register contains the 32-bit IP Source Address field. For IPv6 packets, it contains Bits[31:0] of the 128-bit IP Source Address or Destination Address field.

Diagram



Fields

Field	Function
31-0 L3A06	Layer 3 Address 0 Field When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[31:0] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[31:0] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset and the L3SAM0 bit is set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with the IP Source Address field in the IPv4 packets.

76.17.202 MAC Layer 3 Address 1 Reg 6 (MAC_Layer3_Addr1_Reg6)

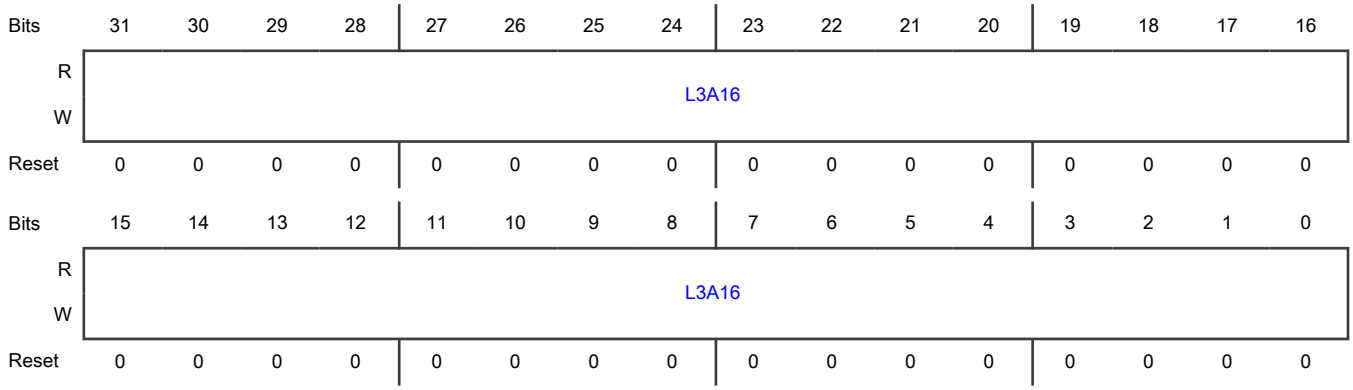
Offset

Register	Offset
MAC_Layer3_Addr1_Reg6	A34h

Function

For IPv4 packets, the Layer 3 Address 1 Register 0 register contains the 32-bit IP Destination Address field. For IPv6 packets, it contains Bits[63:32] of the 128-bit IP Source Address or Destination Address field.

Diagram



Fields

Field	Function
31-0 L3A16	Layer 3 Address 1 Field When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[63:32] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[63:32] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset and the L3SAM0 bit is set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with the IP Destination Address field in the IPv4 packets.

76.17.203 MAC Layer 3 Address 2 Reg 6 (MAC_Layer3_Addr2_Reg6)

Offset

Register	Offset
MAC_Layer3_Addr2_Reg6	A38h

Function

The Layer 3 Address 2 Register 0 register is reserved for IPv4 packets. For IPv6 packets, it contains Bits[95:64] of 128-bit IP Source Address or Destination Address field.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	L3A26															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	L3A26															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 L3A26	Layer 3 Address 2 Field When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[95:64] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[95:64] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset in the MAC_L3_L4_Control0 register, this field is not used.

76.17.204 MAC Layer 3 Address 3 Reg 6 (MAC_Layer3_Addr3_Reg6)

Offset

Register	Offset
MAC_Layer3_Addr3_Reg6	A3Ch

Function

The Layer 3 Address 3 Register 0 register is reserved for IPv4 packets. For IPv6 packets, it contains Bits[127:96] of 128-bit IP Source Address or Destination Address field.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	L3A36															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	L3A36															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 L3A36	Layer 3 Address 3 Field When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[127:96] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[127:96] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset in the MAC_L3_L4_Control0 register, this field is not used.

76.17.205 MAC Layer 3 Layer 4 Control 7 (MAC_L3_L4_Control7)

Offset

Register	Offset
MAC_L3_L4_Control7	A50h

Function
The Layer 3 and Layer 4 Control register controls the operations of filter 0 of Layer 3 and Layer 4.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved			DMCHEN7	Reserved		DMCHN7		Reserved		L4DPIM7	L4DPM7	L4SPIM7	L4SPM7	Reserved	L4PEN7
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	L3HDBM7				L3HSBM7						L3DAIM7	L3DAM7	L3SAIM7	L3SAM7	Reserved	L3PEN7
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	Reserved
28 DMCHEN7	DMA Channel Select Enable When set, this bit enables the selection of the DMA channel number for the packet that is passed by this L3_L4 filter. The DMA channel is indicated by the DMCHN bits. When this bit is reset, the DMA channel is not decided by this filter.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - DMA Channel Select is disabled 1b - DMA Channel Select is enabled
27-26 —	Reserved
25-24 DMCHN7	DMA Channel Number When DMCHEN is set high, this field selects the DMA Channel number to which the packet passed by this filter is routed. The width of this field depends on the number of the DMA channels present in your configuration.
23-22 —	Reserved
21 L4DPIM7	Layer 4 Destination Port Inverse Match Enable When this bit is set, the Layer 4 Destination Port number field is enabled for inverse matching. When this bit is reset, the Layer 4 Destination Port number field is enabled for perfect matching. This bit is valid and applicable only when the L4DPM0 bit is set high. 0b - Layer 4 Destination Port Inverse Match is disabled 1b - Layer 4 Destination Port Inverse Match is enabled
20 L4DPM7	Layer 4 Destination Port Match Enable When this bit is set, the Layer 4 Destination Port number field is enabled for matching. When this bit is reset, the MAC ignores the Layer 4 Destination Port number field for matching. 0b - Layer 4 Destination Port Match is disabled 1b - Layer 4 Destination Port Match is enabled
19 L4SPIM7	Layer 4 Source Port Inverse Match Enable When this bit is set, the Layer 4 Source Port number field is enabled for inverse matching. When this bit is reset, the Layer 4 Source Port number field is enabled for perfect matching. This bit is valid and applicable only when the L4SPM0 bit is set high. 0b - Layer 4 Source Port Inverse Match is disabled 1b - Layer 4 Source Port Inverse Match is enabled
18 L4SPM7	Layer 4 Source Port Match Enable When this bit is set, the Layer 4 Source Port number field is enabled for matching. When this bit is reset, the MAC ignores the Layer 4 Source Port number field for matching. 0b - Layer 4 Source Port Match is disabled 1b - Layer 4 Source Port Match is enabled
17	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
16 L4PEN7	Layer 4 Protocol Enable When this bit is set, the Source and Destination Port number fields of UDP packets are used for matching. When this bit is reset, the Source and Destination Port number fields of TCP packets are used for matching. The Layer 4 matching is done only when the L4SPM0 or L4DPM0 bit is set. 0b - Layer 4 Protocol is disabled 1b - Layer 4 Protocol is enabled
15-11 L3HDBM7	Layer 3 IP DA Higher Bits Match IPv4 Packets: This field contains the number of higher bits of IP Destination Address that are matched in the IPv4 packets. The following list describes the values of this field: - 0: No bits are masked. - 1: LSb[0] is masked - 2: Two LSbs [1:0] are masked - .. - 31: All bits except MSb are masked. IPv6 Packets: Bits[12:11] of this field correspond to Bits[6:5] of L3HSBM0 which indicate the number of lower bits of IP Source or Destination Address that are masked in the IPv6 packets. The following list describes the concatenated values of the L3HDBM0[1:0] and L3HSBM0 bits: - 0: No bits are masked. - 1: LSb[0] is masked. - 2: Two LSbs [1:0] are masked - .. - 127: All bits except MSb are masked. This field is valid and applicable only when the L3DAM0 or L3SAM0 bit is set.
10-6 L3HSBM7	Layer 3 IP SA Higher Bits Match IPv4 Packets: This field contains the number of lower bits of IP Source Address that are masked for matching in the IPv4 packets. The following list describes the values of this field: - 0: No bits are masked. - 1: LSb[0] is masked - 2: Two LSbs [1:0] are masked - .. - 31: All bits except MSb are masked. IPv6 Packets: This field contains Bits[4:0] of L3HSBM0. These bits indicate the number of higher bits of IP Source or Destination Address matched in the IPv6 packets. This field is valid and applicable only when the L3DAM0 or L3SAM0 bit is set high.
5 L3DAIM7	Layer 3 IP DA Inverse Match Enable When this bit is set, the Layer 3 IP Destination Address field is enabled for inverse matching. When this bit is reset, the Layer 3 IP Destination Address field is enabled for perfect matching. This bit is valid and applicable only when the L3DAM0 bit is set high. 0b - Layer 3 IP DA Inverse Match is disabled 1b - Layer 3 IP DA Inverse Match is enabled
4 L3DAM7	Layer 3 IP DA Match Enable When this bit is set, the Layer 3 IP Destination Address field is enabled for matching. When this bit is reset, the MAC ignores the Layer 3 IP Destination Address field for matching. Note: When the L3PEN0 bit is set, you should set either this bit or the L3SAM0 bit because either IPv6 DA or SA can be checked for filtering. 0b - Layer 3 IP DA Match is disabled 1b - Layer 3 IP DA Match is enabled
3 L3SAIM7	Layer 3 IP SA Inverse Match Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When this bit is set, the Layer 3 IP Source Address field is enabled for inverse matching. When this bit reset, the Layer 3 IP Source Address field is enabled for perfect matching. This bit is valid and applicable only when the L3SAM0 bit is set. 0b - Layer 3 IP SA Inverse Match is disabled 1b - Layer 3 IP SA Inverse Match is enabled
2 L3SAM7	Layer 3 IP SA Match Enable When this bit is set, the Layer 3 IP Source Address field is enabled for matching. When this bit is reset, the MAC ignores the Layer 3 IP Source Address field for matching. Note: When the L3PEN0 bit is set, you should set either this bit or the L3DAM0 bit because either IPv6 SA or DA can be checked for filtering. 0b - Layer 3 IP SA Match is disabled 1b - Layer 3 IP SA Match is enabled
1 —	Reserved
0 L3PEN7	Layer 3 Protocol Enable When this bit is set, the Layer 3 IP Source or Destination Address matching is enabled for IPv6 packets. When this bit is reset, the Layer 3 IP Source or Destination Address matching is enabled for IPv4 packets. The Layer 3 matching is done only when the L3SAM0 or L3DAM0 bit is set. 0b - Layer 3 Protocol is disabled 1b - Layer 3 Protocol is enabled

76.17.206 MAC Layer 4 Address 7 (MAC_Layer4_Address7)

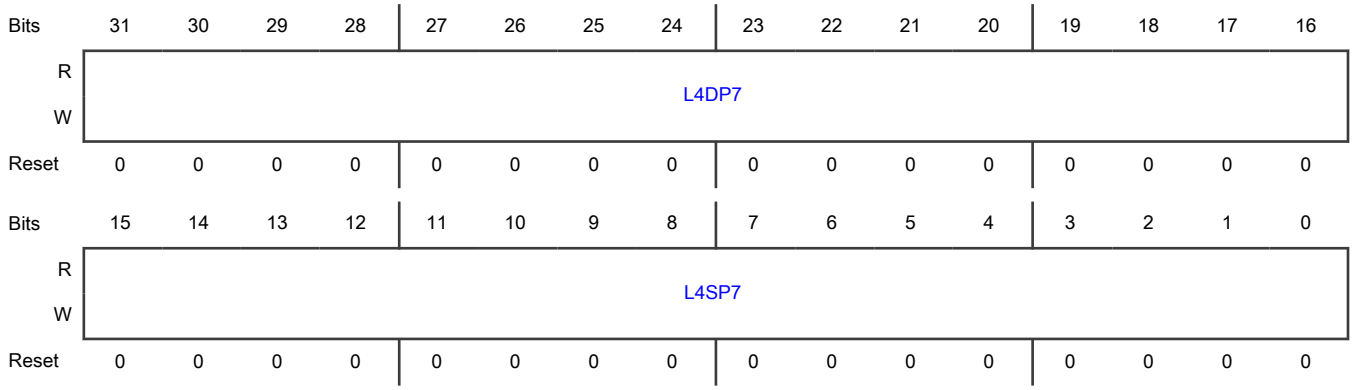
Offset

Register	Offset
MAC_Layer4_Address7	A54h

Function

The MAC_Layer4_Address(#i), MAC_L3_L4_Control(#i), MAC_Layer3_Addr0_Reg(#i), MAC_Layer3_Addr1_Reg(#i), MAC_Layer3_Addr2_Reg(#i) and MAC_Layer3_Addr3_Reg(#i) registers are reserved (RO with default value) if Enable Layer 3 and Layer 4 Packet Filter option is not selected while configuring the IP. You can configure the Layer 3 and Layer 4 Address Registers to be double-synchronized by selecting the Synchronize Layer 3 and Layer 4 Address Registers to Rx Clock Domain option while configuring the IP. When you select this option, the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the Layer 3 and Layer 4 Address Registers are written. For proper synchronization updates, you should perform consecutive writes to same Layer 3 and Layer 4 Address Registers after at least four clock cycles delay of the destination clock.

Diagram



Fields

Field	Function
31-16 L4DP7	Layer 4 Destination Port Number Field When the L4PEN0 bit is reset and the L4DPM0 bit is set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with the TCP Destination Port Number field in the IPv4 or IPv6 packets. When the L4PEN0 and L4DPM0 bits are set in MAC_L3_L4_Control0 register, this field contains the value to be matched with the UDP Destination Port Number field in the IPv4 or IPv6 packets.
15-0 L4SP7	Layer 4 Source Port Number Field When the L4PEN0 bit is reset and the L4SPM0 bit is set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with the TCP Source Port Number field in the IPv4 or IPv6 packets. When the L4PEN0 and L4SPM0 bits are set in MAC_L3_L4_Control0 register, this field contains the value to be matched with the UDP Source Port Number field in the IPv4 or IPv6 packets.

76.17.207 MAC Layer 3 Address 0 Reg 7 (MAC_Layer3_Addr0_Reg7)

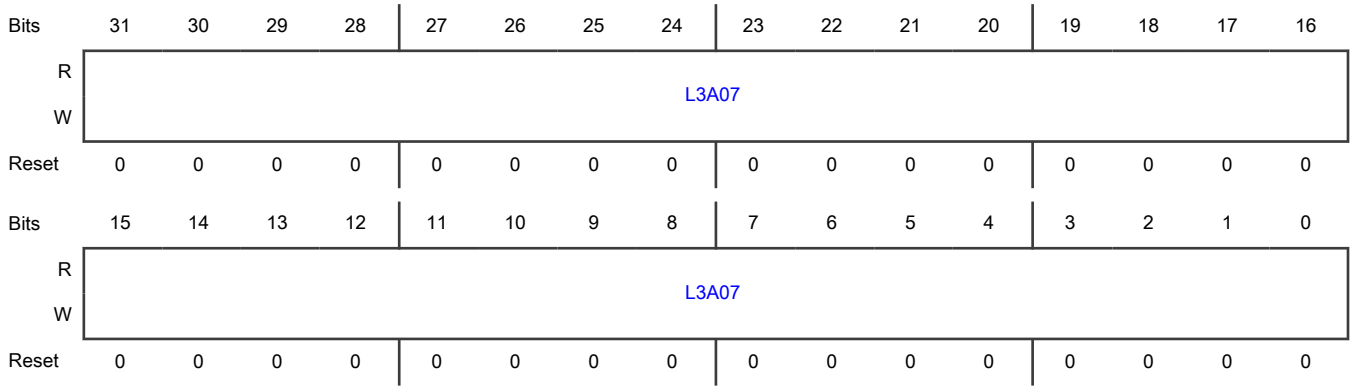
Offset

Register	Offset
MAC_Layer3_Addr0_Reg7	A60h

Function

For IPv4 packets, the Layer 3 Address 0 Register 0 register contains the 32-bit IP Source Address field. For IPv6 packets, it contains Bits[31:0] of the 128-bit IP Source Address or Destination Address field.

Diagram



Fields

Field	Function
31-0 L3A07	Layer 3 Address 0 Field When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[31:0] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[31:0] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset and the L3SAM0 bit is set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with the IP Source Address field in the IPv4 packets.

76.17.208 MAC Layer 3 Address 1 Reg 7 (MAC_Layer3_Addr1_Reg7)

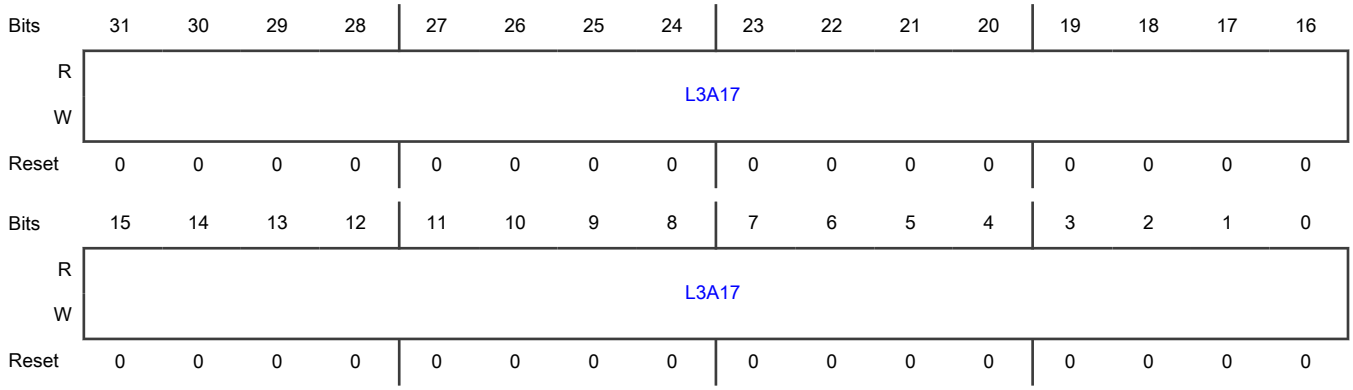
Offset

Register	Offset
MAC_Layer3_Addr1_Reg7	A64h

Function

For IPv4 packets, the Layer 3 Address 1 Register 0 register contains the 32-bit IP Destination Address field. For IPv6 packets, it contains Bits[63:32] of the 128-bit IP Source Address or Destination Address field.

Diagram



Fields

Field	Function
31-0 L3A17	Layer 3 Address 1 Field When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[63:32] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[63:32] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset and the L3SAM0 bit is set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with the IP Destination Address field in the IPv4 packets.

76.17.209 MAC Layer 3 Address 2 Reg 7 (MAC_Layer3_Addr2_Reg7)

Offset

Register	Offset
MAC_Layer3_Addr2_Reg7	A68h

Function

The Layer 3 Address 2 Register 0 register is reserved for IPv4 packets. For IPv6 packets, it contains Bits[95:64] of 128-bit IP Source Address or Destination Address field.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	L3A27															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	L3A27															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 L3A27	Layer 3 Address 2 Field When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[95:64] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[95:64] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset in the MAC_L3_L4_Control0 register, this field is not used.

76.17.210 MAC Layer 3 Address 3 Reg 7 (MAC_Layer3_Addr3_Reg7)

Offset

Register	Offset
MAC_Layer3_Addr3_Reg7	A6Ch

Function

The Layer 3 Address 3 Register 0 register is reserved for IPv4 packets. For IPv6 packets, it contains Bits[127:96] of 128-bit IP Source Address or Destination Address field.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	L3A37															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	L3A37															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 L3A37	Layer 3 Address 3 Field When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[127:96] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[127:96] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset in the MAC_L3_L4_Control0 register, this field is not used.

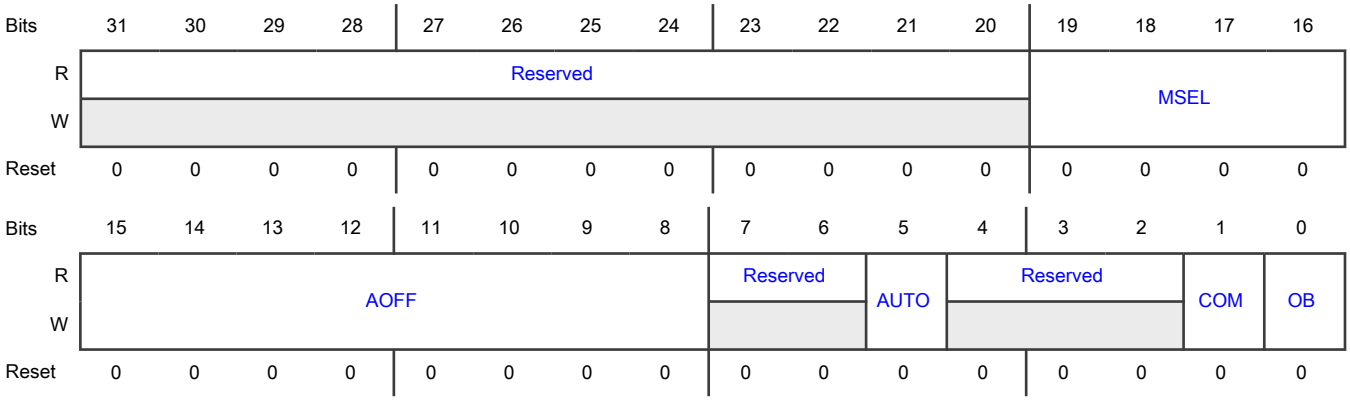
76.17.211 MAC Indirect Access Control (MAC_Indir_Access_Ctrl)

Offset

Register	Offset
MAC_Indir_Access_Ctrl	A70h

Function
This register provides the Indirect Access control and status for MAC_<MSEl>_<AOFF> registers.

Diagram



Fields

Field	Function
31-20 —	Reserved
19-16 MSEL	Mode Select This field is used in indirect access of MAC_<MSEL>_<AOFF>. This field must be set along with initiation of read/write to MAC_<MSEL>_<AOFF> and should not be changed until the OB is reset. Values: - 0000:Typ_RXQ_ (Type based RXQ mappping) - 0001-1111: Reserved for future use.

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-8 AOFF	Address Offset This field is used in indirect access of MAC_<MSEL>_<AOFF>. This field must be set along with initiation of read/write to MAC_<MSEL>_<AOFF> and should not be changed until the OB is reset. Values: - 00000000: IndReg0(Indirect register 0) - 00000001: IndReg1(Indirect register 1) - ... - 00000111: IndReg7(Indirect register 7) - 00001000-11111111: Reserved for future use.
7-6 —	Reserved
5 AUTO	Auto increment - 1: AOFF is incremented by 1. Software should ensure not to cause a wrap condition. Byte wise read/write is not supported when auto increment is enabled. - 0: AOFF is not incremented automatically. Software should program the correct Address Offset for each access.
4-2 —	Reserved
1 COM	Command type This bit indicates the register access type. - 1: Indicates a read operation. - 0: Indicates a write operation. 0b - Write operation 1b - Read operation
0 OB	Operation Busy. This bit is set along with a read or write command for initiating the indirect access to MAC_<MSEL>_<AOFF> register. This bit is reset when the read or write command to MAC_<MSEL>_<AOFF> register is complete. The next indirect register access can be initiated only after this bit is reset. During a write operation, the bit is reset only after the data has been written into MAC_<MSEL>_<AOFF> register. During a read operation, the data should be read from MAC_Indir_Access_Data register only after this bit is reset. **NOTE** This field is not testable by an automated register test.

76.17.212 MAC Indirect Access Data (MAC_Indir_Access_Data)

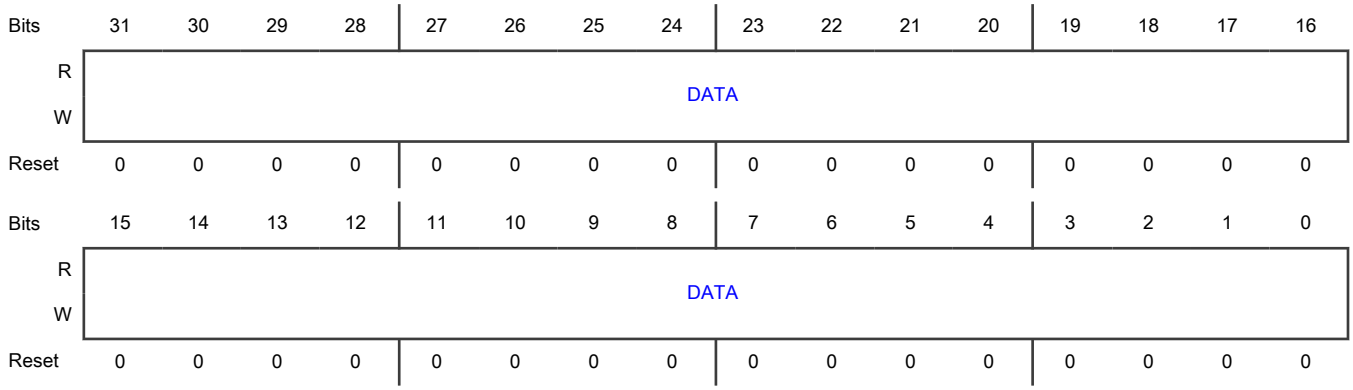
Offset

Register	Offset
MAC_Indir_Access_Data	A74h

Function

This register holds the read/write data for Indirect Access of MAC_<MSEL_<AOFF> registers.

Diagram



Fields

Field	Function
31-0 DATA	This field contains data to read/write for Indirect address access associated with MAC_Indir_Access_Ctrl register. This field contains data to read/write for Indirect address access associated with MAC_Indir_Access_Ctrl register.

76.17.213 MAC TMR Queue Regs 0 (MAC_TMRQ_Regs0)

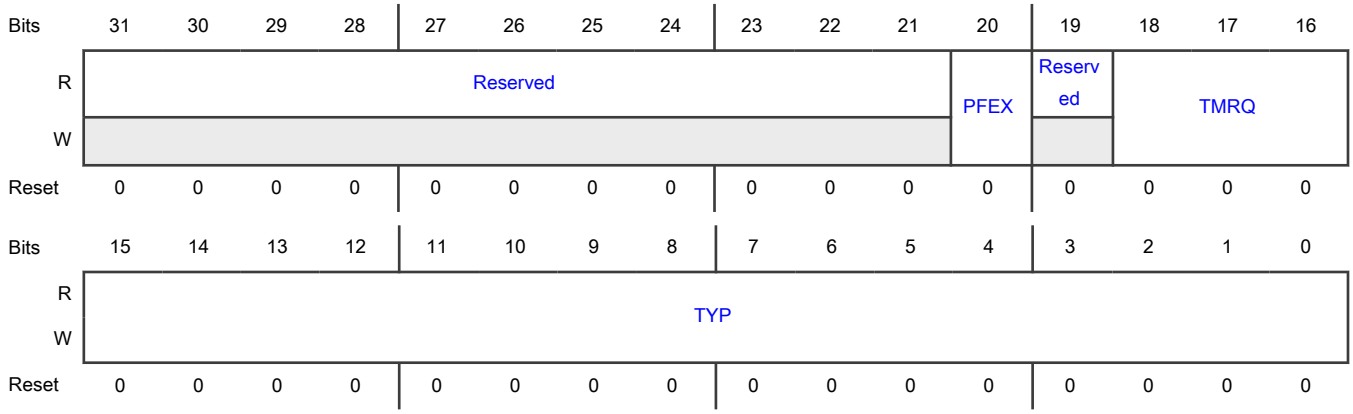
Offset

Register	Offset
MAC_TMRQ_Regs0	A74h

Function

This register contains the type, associated queue number and paket type (Pre emption/EXpress) related to Type based RXQ mapping.

Diagram



Fields

Field	Function
31-21 —	Reserved
20 PFEX	Preemption or Express Packet This bit indicates if it is a preemption packet or express packet.
19 —	Reserved
18-16 TMRQ	Type Match Rx Queue Number Indicates the receive queue number to which the packet needs to be forwarded on detecting a type match.
15-0 TYP	Type field Value Indicates the type value of packet that needs to be compared with received Ethernet packet. This field is valid when programmed to a value greater than or equal to 0x0600.

76.17.214 MAC TMR Queue Regs 1 (MAC_TMRQ_Regs1)

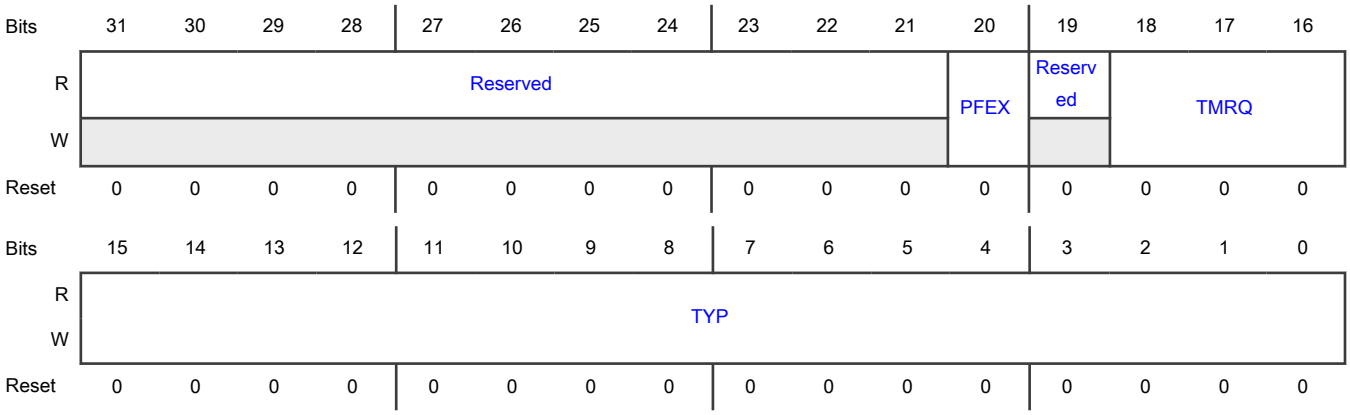
Offset

Register	Offset
MAC_TMRQ_Regs1	A74h

Function

This register contains the type, associated queue number and paket type (Pre emption/EXpress) related to Type based RXQ mapping.

Diagram



Fields

Field	Function
31-21 —	Reserved
20 PFEX	Preemption or Express Packet This bit indicates if it is a preemption packet or express packet.
19 —	Reserved
18-16 TMRQ	Type Match Rx Queue Number Indicates the receive queue number to which the packet needs to be forwarded on detecting a type match.
15-0 TYP	Type field Value Indicates the type value of packet that needs to be compared with received Ethernet packet. This field is valid when programmed to a value greater than or equal to 0x0600.

76.17.215 MAC TMR Queue Regs 2 (MAC_TMRQ_Regs2)

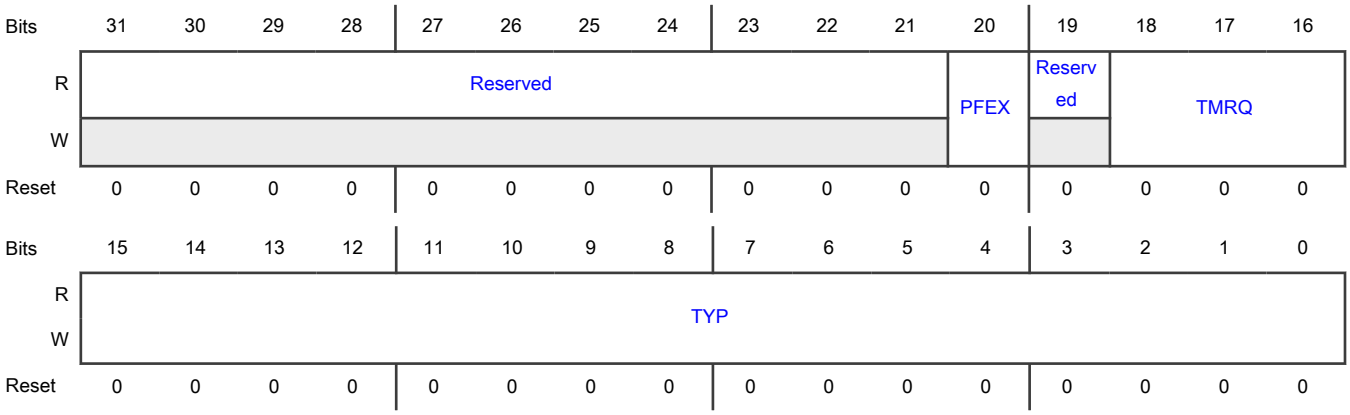
Offset

Register	Offset
MAC_TMRQ_Regs2	A74h

Function

This register contains the type, associated queue number and paket type (Pre emption/EXpress) related to Type based RXQ mapping.

Diagram



Fields

Field	Function
31-21 —	Reserved
20 PFEX	Preemption or Express Packet This bit indicates if it is a preemption packet or express packet.
19 —	Reserved
18-16 TMRQ	Type Match Rx Queue Number Indicates the receive queue number to which the packet needs to be forwarded on detecting a type match.
15-0 TYP	Type field Value Indicates the type value of packet that needs to be compared with received Ethernet packet. This field is valid when programmed to a value greater than or equal to 0x0600.

76.17.216 MAC TMR Queue Regs 3 (MAC_TMRQ_Regs3)

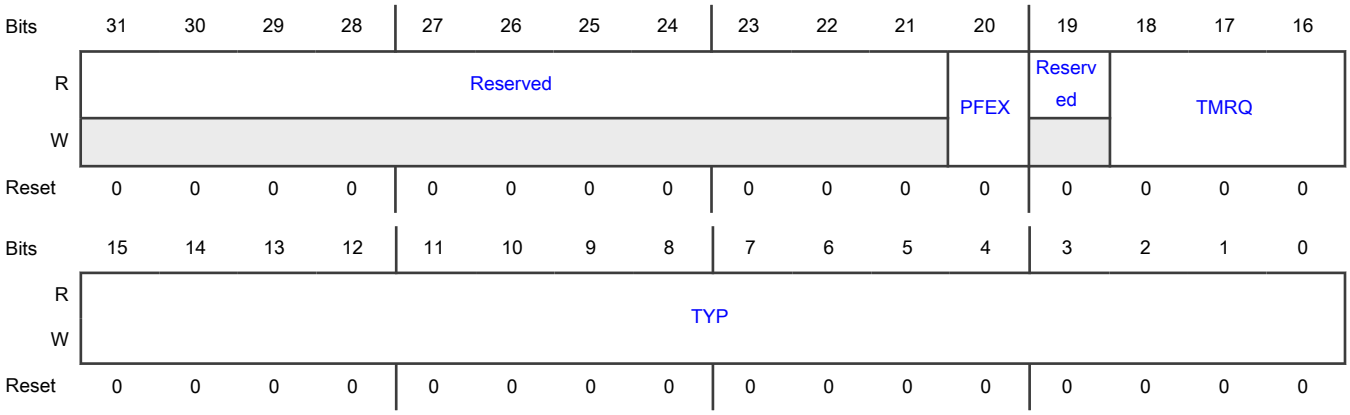
Offset

Register	Offset
MAC_TMRQ_Regs3	A74h

Function

This register contains the type, associated queue number and paket type (Pre emption/EXpress) related to Type based RXQ mapping.

Diagram



Fields

Field	Function
31-21 —	Reserved
20 PFEX	Preemption or Express Packet This bit indicates if it is a preemption packet or express packet.
19 —	Reserved
18-16 TMRQ	Type Match Rx Queue Number Indicates the receive queue number to which the packet needs to be forwarded on detecting a type match.
15-0 TYP	Type field Value Indicates the type value of packet that needs to be compared with received Ethernet packet. This field is valid when programmed to a value greater than or equal to 0x0600.

76.17.217 MAC TMR Queue Regs 4 (MAC_TMRQ_Regs4)

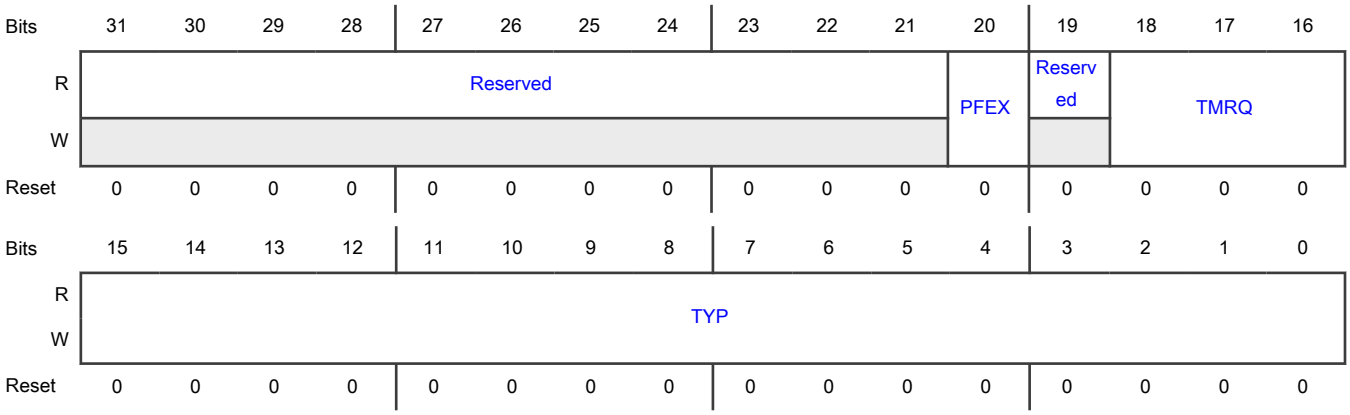
Offset

Register	Offset
MAC_TMRQ_Regs4	A74h

Function

This register contains the type, associated queue number and paket type (Pre emption/EXpress) related to Type based RXQ mapping.

Diagram



Fields

Field	Function
31-21 —	Reserved
20 PFEX	Preemption or Express Packet This bit indicates if it is a preemption packet or express packet.
19 —	Reserved
18-16 TMRQ	Type Match Rx Queue Number Indicates the receive queue number to which the packet needs to be forwarded on detecting a type match.
15-0 TYP	Type field Value Indicates the type value of packet that needs to be compared with received Ethernet packet. This field is valid when programmed to a value greater than or equal to 0x0600.

76.17.218 MAC TMR Queue Regs 5 (MAC_TMRQ_Regs5)

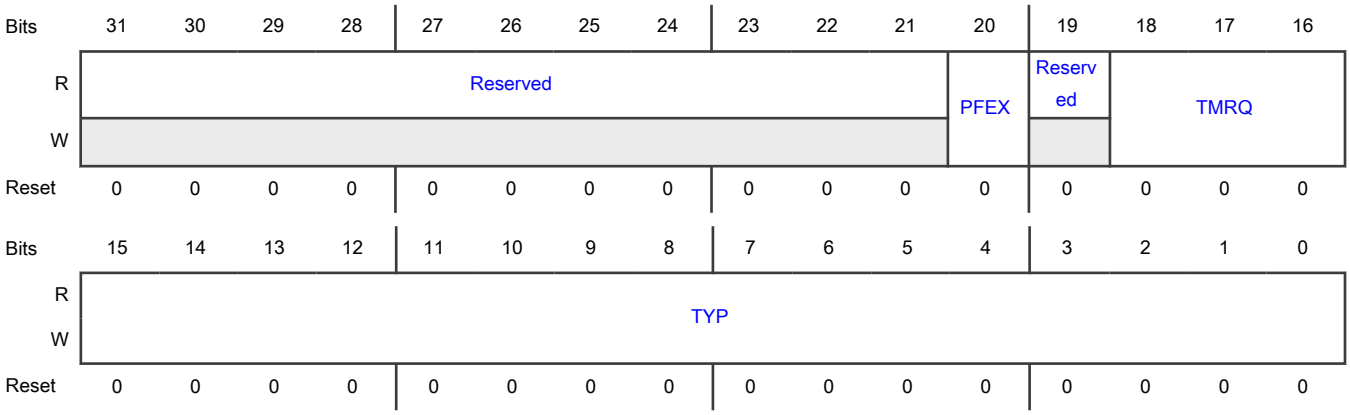
Offset

Register	Offset
MAC_TMRQ_Regs5	A74h

Function

This register contains the type, associated queue number and paket type (Pre emption/EXpress) related to Type based RXQ mapping.

Diagram



Fields

Field	Function
31-21 —	Reserved
20 PFEX	Preemption or Express Packet This bit indicates if it is a preemption packet or express packet.
19 —	Reserved
18-16 TMRQ	Type Match Rx Queue Number Indicates the receive queue number to which the packet needs to be forwarded on detecting a type match.
15-0 TYP	Type field Value Indicates the type value of packet that needs to be compared with received Ethernet packet. This field is valid when programmed to a value greater than or equal to 0x0600.

76.17.219 MAC TMR Queue Regs 6 (MAC_TMRQ_Regs6)

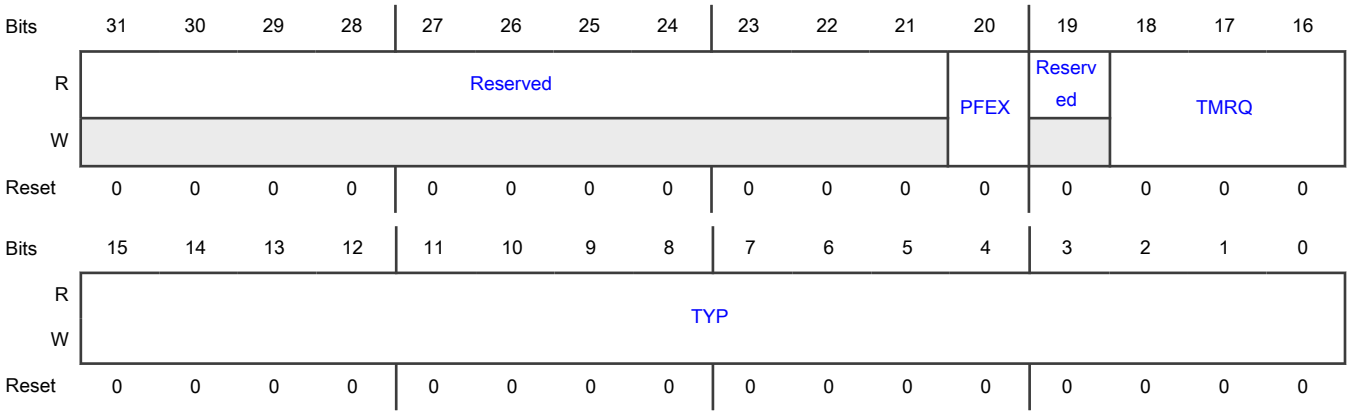
Offset

Register	Offset
MAC_TMRQ_Regs6	A74h

Function

This register contains the type, associated queue number and paket type (Pre emption/EXpress) related to Type based RXQ mapping.

Diagram



Fields

Field	Function
31-21 —	Reserved
20 PFEX	Preemption or Express Packet This bit indicates if it is a preemption packet or express packet.
19 —	Reserved
18-16 TMRQ	Type Match Rx Queue Number Indicates the receive queue number to which the packet needs to be forwarded on detecting a type match.
15-0 TYP	Type field Value Indicates the type value of packet that needs to be compared with received Ethernet packet. This field is valid when programmed to a value greater than or equal to 0x0600.

76.17.220 MAC TMR Queue Regs 7 (MAC_TMRQ_Regs7)

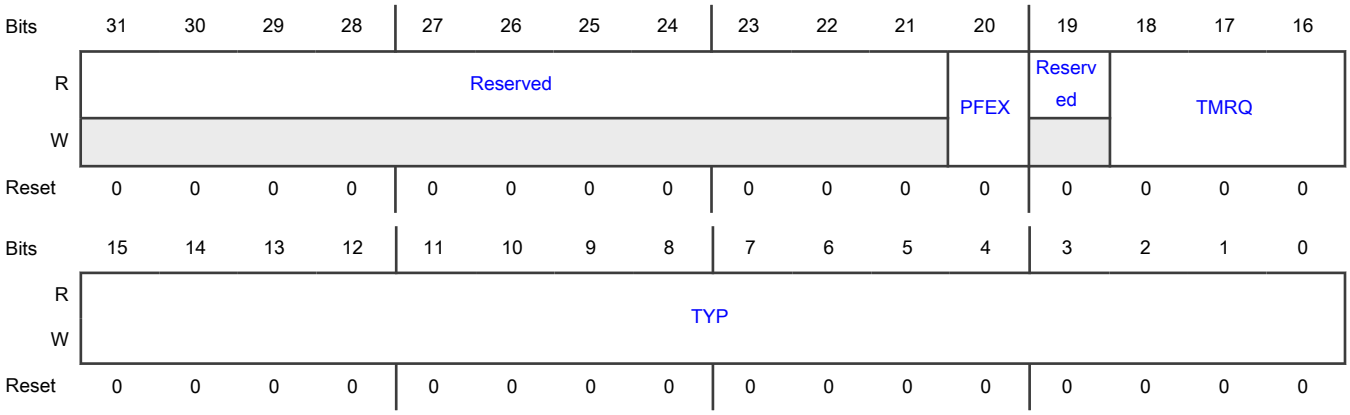
Offset

Register	Offset
MAC_TMRQ_Regs7	A74h

Function

This register contains the type, associated queue number and paket type (Pre emption/EXpress) related to Type based RXQ mapping.

Diagram



Fields

Field	Function
31-21 —	Reserved
20 PFEX	Preemption or Express Packet This bit indicates if it is a preemption packet or express packet.
19 —	Reserved
18-16 TMRQ	Type Match Rx Queue Number Indicates the receive queue number to which the packet needs to be forwarded on detecting a type match.
15-0 TYP	Type field Value Indicates the type value of packet that needs to be compared with received Ethernet packet. This field is valid when programmed to a value greater than or equal to 0x0600.

76.17.221 MAC Timestamp Control (MAC_Timestamp_Control)

Offset

Register	Offset
MAC_Timestamp_Control	B00h

Function

This register controls the operation of the System Time generator and processing of PTP packets for timestamping in the Receiver.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved			AV8021A...	Reserved			TXTSSTSM	Reserved			ESTI	Reserved	TSENMAC...	SNAPTYPSEL	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TSMSTRE...	TSEVNTE...	TSIPV4E...	TSIPV6E...	TSIPENA	TSVER2E...	TSCTRLS...	TSENALL	Reserved	PTGE	TSADDREG	Reserved	TSUPDT	TSINIT	TSCFUPDT	TSENA
W																
Reset	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	Reserved
28 AV8021ASMEN	AV 802.1AS Mode Enable When this bit is set, the MAC processes only untagged PTP over Ethernet packets for providing PTP status and capturing timestamp snapshots, that is, IEEE 802.1AS mode of operation. When PTP offload feature is enabled, for the purpose of PTP offload, the transport specific field in the PTP header is generated and checked based on the value of this bit. 0b - AV 802.1AS Mode is disabled 1b - AV 802.1AS Mode is enabled
27-25 —	Reserved
24 TXTSSTSM	Transmit Timestamp Status Mode When this bit is set, the MAC overwrites the earlier transmit timestamp status even if it is not read by the software. The MAC indicates this by setting the TXTSSMIS bit of the MAC_Tx_Timestamp_Status_Nanoseconds register. When this bit is reset, the MAC ignores the timestamp status of current packet if the timestamp status of previous packet is not read by the software. The MAC indicates this by setting the TXTSSMIS bit of the MAC_Tx_Timestamp_Status_Nanoseconds register. 0b - Transmit Timestamp Status Mode is disabled 1b - Transmit Timestamp Status Mode is enabled
23-21 —	Reserved
20 ESTI	External System Time Input When this bit is set, the MAC uses the external 64-bit reference System Time input for the following: - To take the timestamp provided as status - To insert the timestamp in transmit PTP packets when One-step Timestamp or Timestamp Offload feature is enabled. When this bit is reset, the MAC uses the internal reference System Time. 0b - External System Time Input is disabled 1b - External System Time Input is enabled
19 —	Reserved
18 TSENMACADDR	Enable MAC Address for PTP Packet Filtering When this bit is set, the DA MAC address (that matches any MAC Address register) is used to filter the PTP packets when PTP is directly sent over Ethernet. When this bit is set, received PTP packets with DA containing a special multicast or unicast address that matches the one programmed in MAC address registers are considered for processing as indicated, when PTP is directly sent over Ethernet. For normal

Table continues on the next page...

Table continued from the previous page...

Field	Function
	time stamping operation, MAC address registers 0 to 31 is considered for unicast destination address matching. For PTP offload, only MAC address register 0 is considered for unicast destination address matching. 0b - MAC Address for PTP Packet Filtering is disabled 1b - MAC Address for PTP Packet Filtering is enabled
17-16 SNAPTYPSEL	Select PTP packets for Taking Snapshots These bits, along with Bits 15 and 14, decide the set of PTP packet types for which snapshot needs to be taken. The encoding is given in Timestamp Snapshot Dependency on Register Bits Table.
15 TSMSTRENA	Enable Snapshot for Messages Relevant to Master When this bit is set, the snapshot is taken only for the messages that are relevant to the master node. Otherwise, the snapshot is taken for the messages relevant to the slave node. 0b - Snapshot for Messages Relevant to Master is disabled 1b - Snapshot for Messages Relevant to Master is enabled
14 TSEVNTENA	Enable Timestamp Snapshot for Event Messages When this bit is set, the timestamp snapshot is taken only for event messages (SYNC, Delay_Req, Pdelay_Req, or Pdelay_Resp). When this bit is reset, the snapshot is taken for all messages except Announce, Management, and Signaling. For more information about the timestamp snapshots, see Timestamp Snapshot Dependency on Register Bits Table. 0b - Timestamp Snapshot for Event Messages is disabled 1b - Timestamp Snapshot for Event Messages is enabled
13 TSIPV4ENA	Enable Processing of PTP Packets Sent over IPv4-UDP When this bit is set, the MAC receiver processes the PTP packets encapsulated in IPv4-UDP packets. When this bit is reset, the MAC ignores the PTP transported over IPv4-UDP packets. This bit is set by default. 0b - Processing of PTP Packets Sent over IPv4-UDP is disabled 1b - Processing of PTP Packets Sent over IPv4-UDP is enabled
12 TSIPV6ENA	Enable Processing of PTP Packets Sent over IPv6-UDP When this bit is set, the MAC receiver processes the PTP packets encapsulated in IPv6-UDP packets. When this bit is clear, the MAC ignores the PTP transported over IPv6-UDP packets. 0b - Processing of PTP Packets Sent over IPv6-UDP is disabled 1b - Processing of PTP Packets Sent over IPv6-UDP is enabled
11 TSIPENA	Enable Processing of PTP over Ethernet Packets When this bit is set, the MAC receiver processes the PTP packets encapsulated directly in the Ethernet packets. When this bit is reset, the MAC ignores the PTP over Ethernet packets. 0b - Processing of PTP over Ethernet Packets is disabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Processing of PTP over Ethernet Packets is enabled
10 TSVER2ENA	Enable PTP Packet Processing for Version 2 Format When this bit is set, the IEEE 1588 version 2 format is used to process the PTP packets. When this bit is reset, the IEEE 1588 version 1 format is used to process the PTP packets. The IEEE 1588 formats are described in 'PTP Processing and Control'. 0b - PTP Packet Processing for Version 2 Format is disabled 1b - PTP Packet Processing for Version 2 Format is enabled
9 TSCTRLSSR	Timestamp Digital or Binary Rollover Control When this bit is set, the Timestamp Low register rolls over after 0x3B9A_C9FF value (that is, 1 nanosecond accuracy) and increments the timestamp (High) seconds. When this bit is reset, the rollover value of sub-second register is 0x7FFF_FFFF. The sub-second increment must be programmed correctly depending on the PTP reference clock frequency and the value of this bit. 0b - Timestamp Digital or Binary Rollover Control is disabled 1b - Timestamp Digital or Binary Rollover Control is enabled
8 TSENALL	Enable Timestamp for All Packets When this bit is set, the timestamp snapshot is enabled for all packets received by the MAC. 0b - Timestamp for All Packets disabled 1b - Timestamp for All Packets enabled
7 —	Reserved
6 PTGE	Presentation Time Generation Enable When this bit is set the Presentation Time generation is enabled. 0b - Presentation Time Generation is disabled 1b - Presentation Time Generation is enabled
5 TSADDREG	Update Addend Register When this bit is set, the content of the Timestamp Addend register is updated in the PTP block for fine correction. This bit is cleared when the update is complete. This bit should be zero before it is set. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. **NOTE** This field is not testable by an automated register test. 0b - Addend Register is not updated 1b - Addend Register is updated
4	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
3 TSUPDT	Update Timestamp When this bit is set, the system time is updated (added or subtracted) with the value specified in MAC_System_Time_Seconds_Update and MAC_System_Time_Nanoseconds_Update registers. This bit should be zero before updating it. This bit is reset when the update is complete in hardware. The Timestamp Higher Word register (if enabled while configuring the IP) is not updated. When Media Clock Generation and Recovery is configured (DWC_EQOS_FLEXI_PPS_OUT_EN) and enabled MAC_Presn_Time_Updt should also be updated before setting this field. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. **NOTE** This field is not testable by an automated register test. 0b - Timestamp is not updated 1b - Timestamp is updated
2 TSINIT	Initialize Timestamp When this bit is set, the system time is initialized (overwritten) with the value specified in the MAC_System_Time_Seconds_Update and MAC_System_Time_Nanoseconds_Update registers. This bit should be zero before it is updated. This bit is reset when the initialization is complete. The Timestamp Higher Word register (if enabled while configuration the IP) can only be initialized. When Media Clock Generation and Recovery is configured (DWC_EQOS_FLEXI_PPS_OUT_EN) and enabled MAC_Presn_Time_Updt should also be updated before setting this field. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. **NOTE** This field is not testable by an automated register test. 0b - Timestamp is not initialized 1b - Timestamp is initialized
1 TSCFUPDT	Fine or Coarse Timestamp Update When this bit is set, the Fine method is used to update system timestamp. When this bit is reset, Coarse method is used to update the system timestamp. 0b - Coarse method is used to update system timestamp 1b - Fine method is used to update system timestamp
0 TSENA	Enable Timestamp When this bit is set, the timestamp is added for Transmit and Receive packets. When disabled, timestamp is not added for transmit and receive packets and the Timestamp Generator is also suspended. You need to initialize the Timestamp (system time) after enabling this mode. On the Receive side, the MAC processes the 1588 packets only if this bit is set. 0b - Timestamp is disabled 1b - Timestamp is enabled

76.17.222 MAC Sub Second Increment (MAC_Sub_Second_Increment)

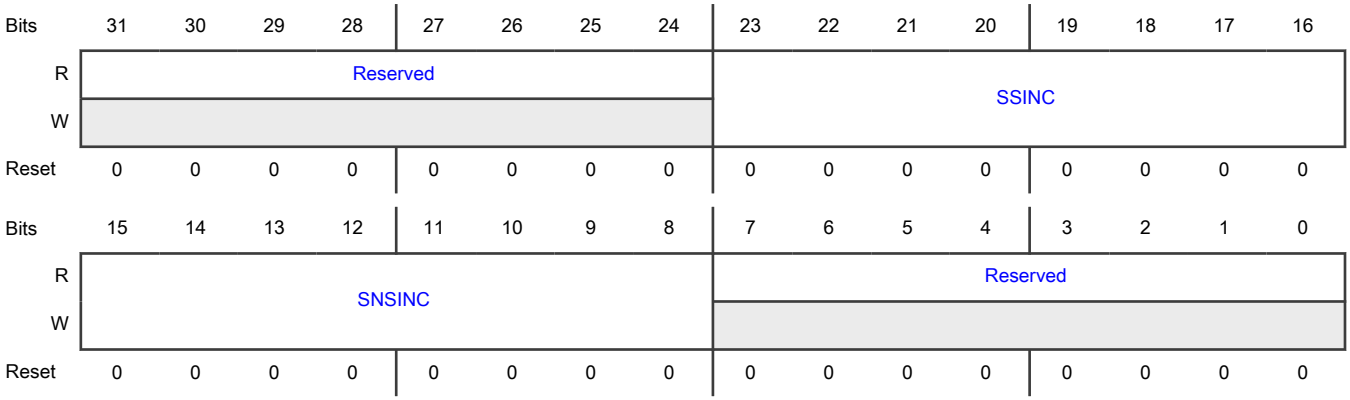
Offset

Register	Offset
MAC_Sub_Second_Increment	B04h

Function

This register specifies the value to be added to the internal system time register every cycle of CLK_PTP_REF_I clock.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-16 SSINC	Sub-second Increment Value The value programmed in this field is accumulated every clock cycle (of clk_ptp_i) with the contents of the sub-second register. For example, when the PTP clock is 50 MHz (period is 20 ns), you should program 20 (0x14) when the System Time Nanoseconds register has an accuracy of 1 ns [Bit 9 (TSCTRLSSR) is set in MAC_Timestamp_Control]. When TSCTRLSSR is clear, the Nanoseconds register has a resolution of ~0.465 ns. In this case, you should program a value of 43 (0x2B) which is derived by 20 ns/0.465.
15-8 SNSINC	Sub-nanosecond Increment Value This field contains the sub-nanosecond increment value, represented in nanoseconds multiplied by 2^8. This value is accumulated with the sub-nanoseconds field of the subsecond register. For example, when TSCTRLSSR field in the MAC_Timestamp_Control register is set. and if the required increment is 5.3ns, then SSINC should be 0x05 and SNSINC should be 0x4C.
7-0 —	Reserved

76.17.223 MAC System Time In Seconds (MAC_System_Time_Seconds)

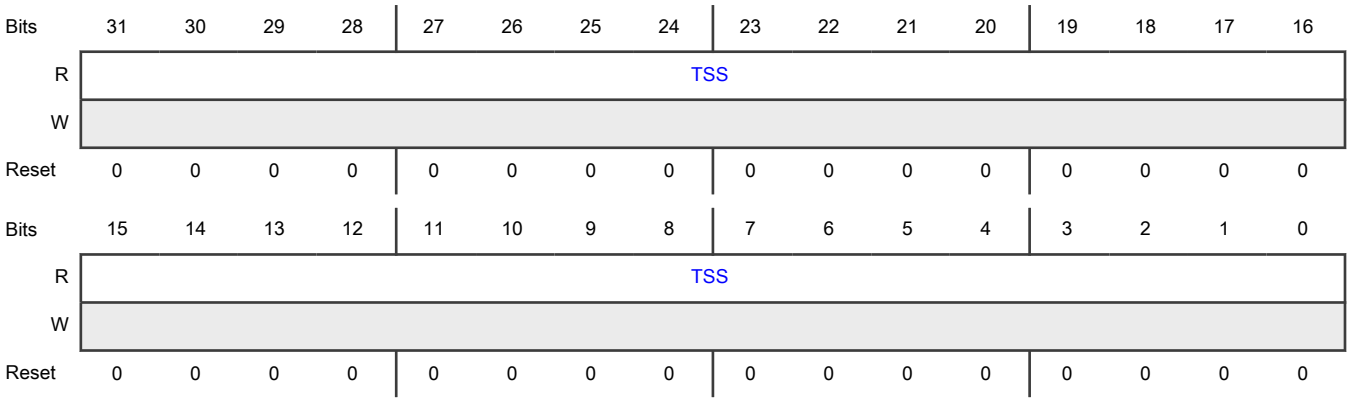
Offset

Register	Offset
MAC_System_Time_Seconds	B08h

Function

The System Time Seconds register, along with System Time Nanoseconds register, indicates the current value of the system time maintained by the MAC. Though it is updated on a continuous basis, there is some delay from the actual time because of clock domain transfer latencies (from CLK_PTP_REF_I to CSR clock).

Diagram



Fields

Field	Function
31-0 TSS	Timestamp Second The value in this field indicates the current value in seconds of the System Time maintained by the MAC. **NOTE** This field is not testable by an automated register test.

76.17.224 MAC System Time In Nanoseconds (MAC_System_Time_Nanoseconds)

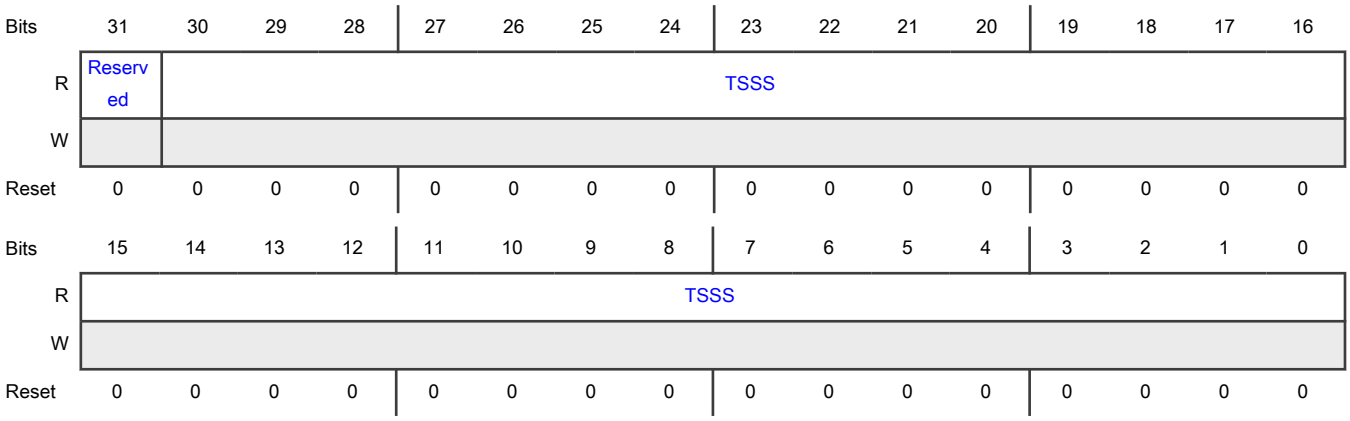
Offset

Register	Offset
MAC_System_Time_Nanoseconds	B0Ch

Function

The System Time Nanoseconds register, along with System Time Seconds register, indicates the current value of the system time maintained by the MAC.

Diagram



Fields

Field	Function
31 —	Reserved
30-0 TSSS	Timestamp Sub Seconds The value in this field has the sub-second representation of time, with an accuracy of 0.46 ns. When Bit 9 is set in MAC_Timestamp_Control, each bit represents 1 ns. The maximum value is 0x3B9A_C9FF after which it rolls-over to zero. **NOTE** This field is not testable by an automated register test.

76.17.225 MAC System Time Seconds Update (MAC_System_Time_Seconds_Update)

Offset

Register	Offset
MAC_System_Time_Seconds_Update	B10h

Function

The System Time Seconds Update register, along with the System Time Nanoseconds Update register, initializes or updates the system time maintained by the MAC. You must write both registers before setting the TSINIT or TSUPDT bits in DWC_eqos_top_map/EQOS_MAC/MAC_Timestamp_Control.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	TSS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	TSS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TSS	Timestamp Seconds The value in this field is the seconds part of the update. When ADDSUB is reset, this field must be programmed with the seconds part of the update value. When ADDSUB is set, this field must be programmed with the complement of the seconds part of the update value. For example, to subtract 2.000000001 seconds from the system time, the TSS field in the MAC_Timestamp_Seconds_Update register must be 0xFFFF_FFFE (that is, 2^32 - 2).

76.17.226 MAC System Time Nanoseconds Update (MAC_System_Time_Nanoseconds_Update)

Offset

Register	Offset
MAC_System_Time_Nanoseconds_Update	B14h

Function
MAC System Time Nanoseconds Update register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	ADDSUB	TSSS														
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	TSSS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 ADDSUB	Add or Subtract Time When this bit is set, the time value is subtracted with the contents of the update register. When this bit is reset, the time value is added with the contents of the update register. 0b - Add time 1b - Subtract time
30-0 TSSS	Timestamp Sub Seconds The value in this field is the sub-seconds part of the update. - ADDSUB is 1: This field must be programmed with the complement of the sub-seconds part of the update value as described. - ADDSUB is 0: This field must be programmed with the sub-seconds part of the update value, with an accuracy based on the TSCTRLSSR bit of the MAC_Timestamp_Control register. - TSCTRLSSR field in the MAC_Timestamp_Control register is 1: -- The programmed value must be 10^9 - <sub-second value>. -- Each bit represents 1 ns and the programmed value should not exceed 0x3B9A_C9FF. - TSCTRLSSR field in the MAC_Timestamp_Control register is 0: -- The programmed value must be 2^31 - <sub-second_value>. -- Each bit represents an accuracy of 0.46 ns. For example, to subtract 2.000000001 seconds from the system time, then the TSSS field in the MAC_Timestamp_Nanoseconds_Update register must be 0x7FFF_FFFF (that is, 2^31 - 1), when TSCTRLSSR bit in MAC_Timestamp_Control is reset and 0x3B9A_C9FF (that is, 10^9 - 1), when TSCTRLSSR bit in MAC_Timestamp_Control is set.

76.17.227 MAC Timestamp Addend (MAC_Timestamp_Addend)

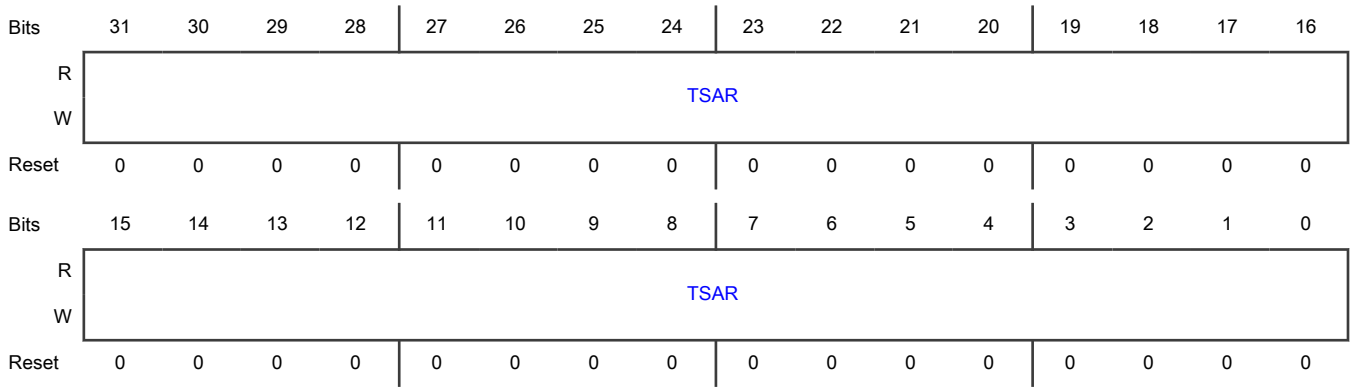
Offset

Register	Offset
MAC_Timestamp_Addend	B18h

Function

This register value is used only when the system time is configured for Fine Update mode (TSCFUPDT bit in the MAC_Timestamp_Control register). The content of this register is added to a 32-bit accumulator in every clock cycle (of CLK_PTP_REF_I) and the system time is updated when the accumulator overflows.

Diagram



Fields

Field	Function
31-0 TSAR	Timestamp Addend Register This field indicates the 32-bit time value to be added to the Accumulator register to achieve time synchronization.

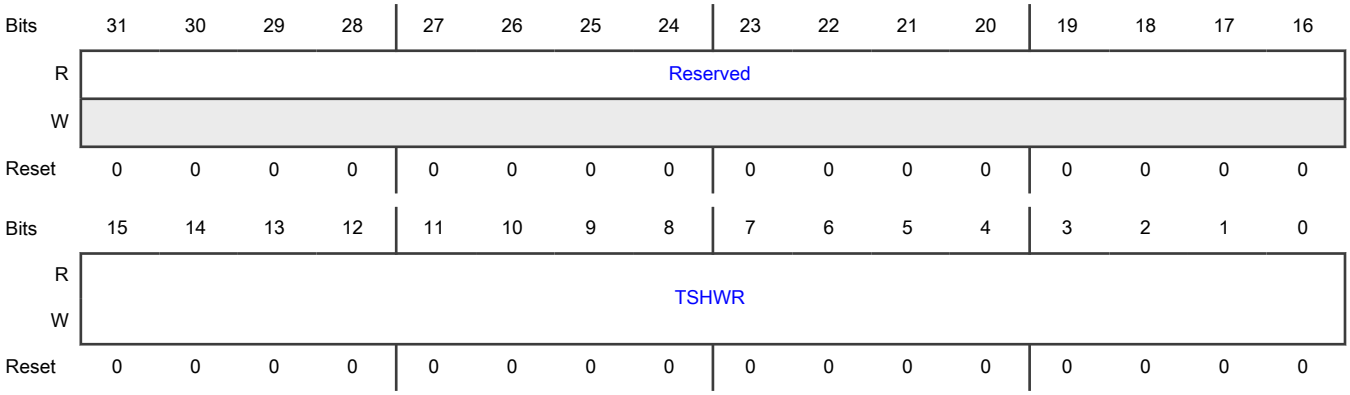
76.17.228 MAC System Time Higher Word In Seconds (MAC_System_Time_Higher_Word_Seconds)

Offset

Register	Offset
MAC_System_Time_Higher_Word_Seconds	B1Ch

Function
System Time - Higher Word Seconds register.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 TSHWR	Timestamp Higher Word Register This field contains the most-significant 16-bits of timestamp seconds value. This register is optional. You can add this register by selecting the Add IEEE 1588 Higher Word Register option. This register is directly written to initialize the value and it is incremented when there is an overflow from 32-bits of the System Time Seconds register. Access restriction applies. Updated based on the event. Setting 1 sets. Setting 0 clears.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field is not testable by an automated register test.

76.17.229 MAC Timestamp Status (MAC_Timestamp_Status)

Offset

Register	Offset
MAC_Timestamp_Status	B20h

Function
Timestamp Status register. All bits except Bits[27:25] gets cleared when the application reads this register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved		Reserved					Reserved	Reserved				Reserved			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TXTSSIS	Reserved					TSTRGTE...	TSTARGT3	TSTRGTE...	TSTARGT2	TSTRGTE...	TSTARGT1	TSTRGTE...	Reserved	TSTARGT0	TSSOVF
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29-25 —	Reserved
24 —	Reserved
23-20	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
19-16 —	Reserved
15 TXTSSIS	Tx Timestamp Status Interrupt Status In non-EQOS_CORE configurations when drop transmit status is enabled in MTL, this bit is set when the captured transmit timestamp is updated in the MAC_Tx_Timestamp_Status_Nanoseconds and MAC_Tx_Timestamp_Status_Seconds registers. When PTP offload feature is enabled, this bit is set when the captured transmit timestamp is updated in the MAC_Tx_Timestamp_Status_Nanoseconds and MAC_Tx_Timestamp_Status_Seconds registers, for PTO generated Delay Request and Pdelay request packets. This bit is cleared when the MAC_Tx_Timestamp_Status_Seconds register is read (or write to MAC_Tx_Timestamp_Status_Seconds register when RCWE bit of MAC_CSR_SW_Ctrl register is set). 0b - Tx Timestamp Status Interrupt status not detected 1b - Tx Timestamp Status Interrupt status detected
14-10 —	Reserved
9 TSTRGTERR3	Timestamp Target Time Error This bit is set when the latest target time programmed in the MAC_PPS3_Target_Time_Seconds and MAC_PPS3_Target_Time_Nanoseconds registers elapses. This bit is cleared when the application reads this bit. Access restriction applies. Clears on read (or this bit is written to 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0b - Timestamp Target Time Error status not detected 1b - Timestamp Target Time Error status detected
8 TSTARGT3	Timestamp Target Time Reached for Target Time PPS3 When this bit is set and MCGREN3 of MAC_PPS_Control register is reset, it indicates that the value of system time is greater than or equal to the value specified in the MAC_PPS3_Target_Time_Seconds and MAC_PPS3_Target_Time_Nanoseconds registers. When this bit is set and MCGREN3 of MAC_PPS_Control register is set, it indicates that mcgr_dma_req_o[3] is asserted. Access restriction applies. Clears on read (or this bit is written to 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0b - Timestamp Target Time Reached for Target Time PPS3 status not detected 1b - Timestamp Target Time Reached for Target Time PPS3 status detected
7 TSTRGTERR2	Timestamp Target Time Error This bit is set when the latest target time programmed in the MAC_PPS2_Target_Time_Seconds and MAC_PPS2_Target_Time_Nanoseconds registers elapses. This bit is cleared when the application reads this bit. Access restriction applies. Clears on read (or this bit is written to 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Timestamp Target Time Error status not detected 1b - Timestamp Target Time Error status detected
6 TSTARGT2	Timestamp Target Time Reached for Target Time PPS2 When this bit is set and MCGREN2 of MAC_PPS_Control register is reset, it indicates that the value of system time is greater than or equal to the value specified in the MAC_PPS2_Target_Time_Seconds and MAC_PPS2_Target_Time_Nanoseconds registers. When this bit is set and MCGREN2 of MAC_PPS_Control register is set, it indicates that mcgr_dma_req_o[2] is asserted. Access restriction applies. Clears on read (or this bit is written to 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0b - Timestamp Target Time Reached for Target Time PPS2 status not detected 1b - Timestamp Target Time Reached for Target Time PPS2 status detected
5 TSTRGTERR1	Timestamp Target Time Error This bit is set when the latest target time programmed in the MAC_PPS1_Target_Time_Seconds and MAC_PPS1_Target_Time_Nanoseconds registers elapses. This bit is cleared when the application reads this bit. Access restriction applies. Clears on read (or this bit is written to 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0b - Timestamp Target Time Error status not detected 1b - Timestamp Target Time Error status detected
4 TSTARGT1	Timestamp Target Time Reached for Target Time PPS1 When this bit is set and MCGREN1 of MAC_PPS_Control register is reset, it indicates that the value of system time is greater than or equal to the value specified in the MAC_PPS1_Target_Time_Seconds and MAC_PPS1_Target_Time_Nanoseconds registers. When this bit is set and MCGREN1 of MAC_PPS_Control register is set, it indicates that mcgr_dma_req_o[1] is asserted. Access restriction applies. Clears on read (or this bit is written to 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0b - Timestamp Target Time Reached for Target Time PPS1 status not detected 1b - Timestamp Target Time Reached for Target Time PPS1 status detected
3 TSTRGTERR0	Timestamp Target Time Error This bit is set when the latest target time programmed in the MAC_PPS0_Target_Time_Seconds and MAC_PPS0_Target_Time_Nanoseconds registers elapses. This bit is cleared when the application reads this bit. Access restriction applies. Clears on read (or this bit is written to 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0b - Timestamp Target Time Error status not detected 1b - Timestamp Target Time Error status detected
2 —	Reserved
1	Timestamp Target Time Reached

Table continues on the next page...

Table continued from the previous page...

Field	Function
TSTARGT0	When this bit is set and MCGREN0 of MAC_PPS_Control register is reset, it indicates that the value of system time is greater than or equal to the value specified in the MAC_PPS0_Target_Time_Seconds and MAC_PPS0_Target_Time_Nanoseconds registers. When this bit is set and MCGREN0 of MAC_PPS_Control register is set, it indicates that mcgr_dma_req_o[0] is asserted. Access restriction applies. Clears on read (or this bit is written to 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0b - Timestamp Target Time Reached status not detected 1b - Timestamp Target Time Reached status detected
0 TSSOVF	Timestamp Seconds Overflow When this bit is set, it indicates that the seconds value of the timestamp (when supporting version 2 format) has overflowed beyond 32'hFFFF_FFFF. Access restriction applies. Clears on read (or this bit is written to 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0b - Timestamp Seconds Overflow status not detected 1b - Timestamp Seconds Overflow status detected

76.17.230 MAC Transmit Timestamp Status In Nanoseconds (MAC_Tx_Timestamp_Status_Nanoseconds)

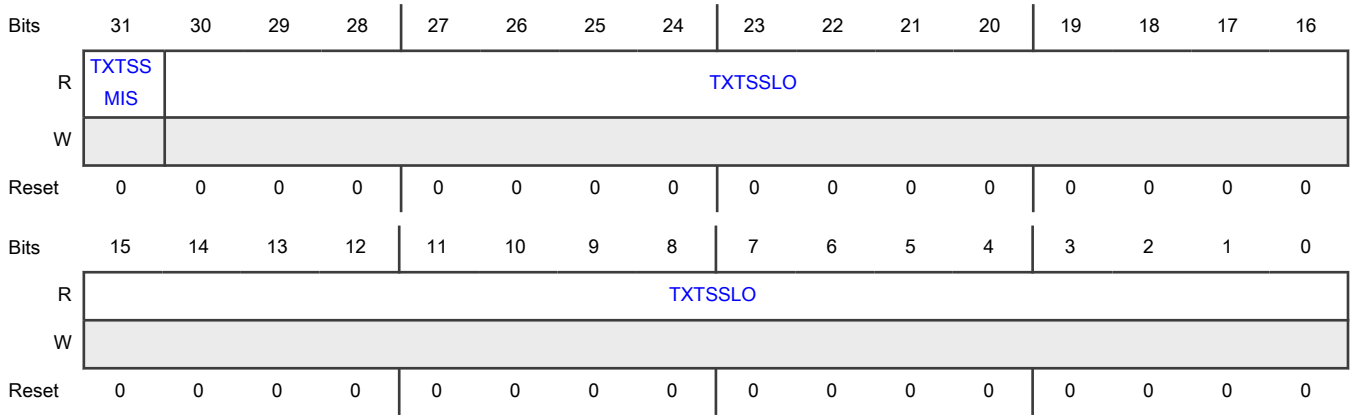
Offset

Register	Offset
MAC_Tx_Timestamp_Status_Nanoseconds	B30h

Function

This register contains the nanosecond part of timestamp captured for Transmit packets when Tx status is disabled.
The MAC_Tx_Timestamp_Status_Nanoseconds register, along with MAC_Tx_Timestamp_Status_Seconds, gives the 64-bit timestamp captured for the PTP packet successfully transmitted by the MAC. This value is considered to be read by the application when the last byte of MAC_Tx_Timestamp_Status_Nanoseconds is read. In the little-endian mode, this means when bits[31:24] are read; in big-endian mode, bits[7:0] are read. If the application does not read these registers and timestamp of another packet is captured, then either the current timestamp is lost (overwritten) or the new timestamp is lost (dropped), depending on the setting of the TXTSSTSM bit of the MAC_Timestamp_Control register. The status bit TXTSSIS bit [15] in MAC_Timestamp_Status register is set when the MAC transmitter captures the timestamp.

Diagram



Fields

Field	Function
31 TXTSSMIS	Transmit Timestamp Status Missed When this bit is set, it indicates one of the following: - The timestamp of the current packet is ignored if TXTSSTSM bit of the MAC_Timestamp_Control register is reset - The timestamp of the previous packet is overwritten with timestamp of the current packet if TXTSSTSM bit of the MAC_Timestamp_Control register is set. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - Transmit Timestamp Status Missed status not detected 1b - Transmit Timestamp Status Missed status detected
30-0 TXTSSLO	Transmit Timestamp Status Low This field contains the 31 bits of the Nanoseconds field of the Transmit packet's captured timestamp.

76.17.231 MAC Transmit Timestamp Status In Seconds (MAC_Tx_Timestamp_Status_Seconds)

Offset

Register	Offset
MAC_Tx_Timestamp_Status_Seconds	B34h

Function

The register contains the higher 32 bits of the timestamp (in seconds) captured when a PTP packet is transmitted.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TXTSSHI															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TXTSSHI															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TXTSSHI	Transmit Timestamp Status High This field contains the lower 32 bits of the Seconds field of Transmit packet's captured timestamp.

76.17.232 MAC Timestamp Ingress Asymmetry Correction (MAC_Timestamp_Ingress_Asym_Corr)

Offset

Register	Offset
MAC_Timestamp_Ingress_Asym_Corr	B50h

Function

The MAC Timestamp Ingress Asymmetry Correction register contains the Ingress Asymmetry Correction value to be used while updating correction field in PDelay_Resp PTP messages.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	OSTIAC															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	OSTIAC															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 OSTIAC	One-Step Timestamp Ingress Asymmetry Correction This field contains the ingress path asymmetry value to be added to correctionField of Pdelay_Resp PTP packet. The programmed value should be in units of nanoseconds and multiplied by 2^16. For example, 2.5 ns is represented as 0x00028000. The value can also be negative, which is represented in 2's complement form with bit 31 representing the sign bit.

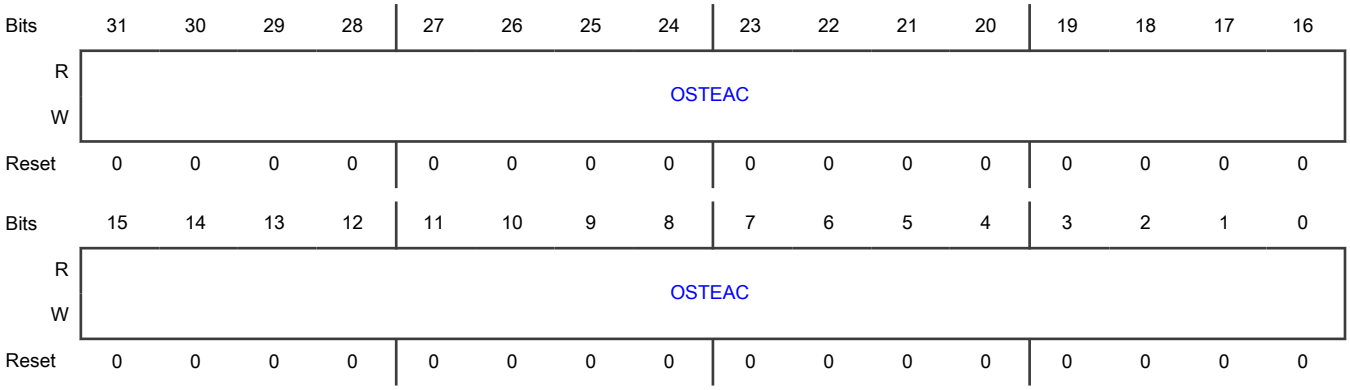
76.17.233 MAC Timestamp Egress Asymmetry Correction (MAC_Timestamp_Egress_Asym_Corr)

Offset

Register	Offset
MAC_Timestamp_Egress_Asym_Corr	B54h

Function
The MAC Timestamp Egress Asymmetry Correction register contains the Egress Asymmetry Correction value to be used while updating the correction field in PDelay_Req PTP messages.

Diagram



Fields

Field	Function
31-0 OSTEAC	One-Step Timestamp Egress Asymmetry Correction This field contains the egress path asymmetry value to be subtracted from correctionField of Pdelay_Resp PTP packet. The programmed value must be the negated value in units of nanoseconds multiplied by 2^16. For example, if the required correction is +2.5 ns, the programmed value must be 0xFFFD_8000, which is the 2's complement of 0x0002_8000(2.5 * 216). Similarly, if the required correction is -3.3 ns, the programmed value is 0x0003_4CCC (3.3 * 216).

76.17.234 MAC Timestamp Ingress Correction In Nanoseconds (MAC_Timestamp_Ingress_Corr_Nanosecond)

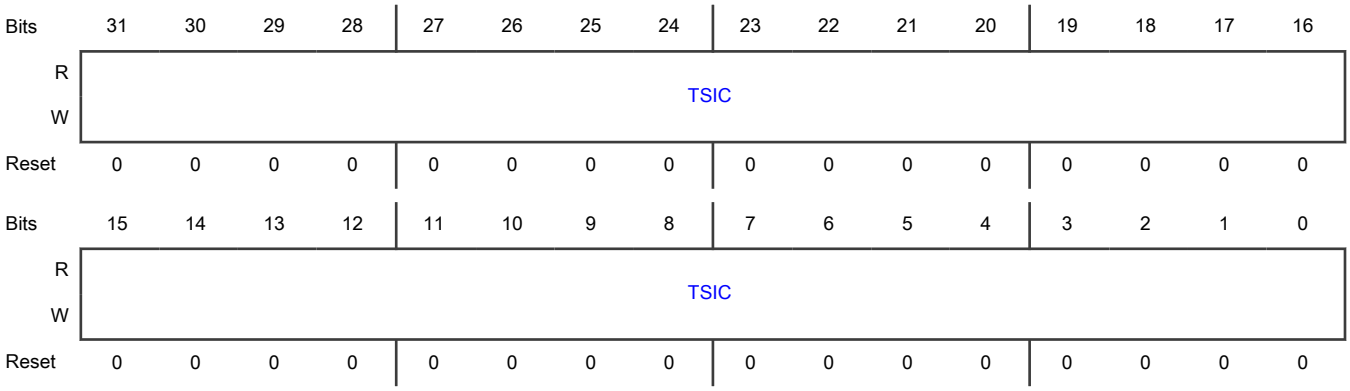
Offset

Register	Offset
MAC_Timestamp_Ingress_Corr_Nanosecond	B58h

Function

This register contains the correction value in nanoseconds to be used with the captured timestamp value in the ingress path.

Diagram



Fields

Field	Function
31-0 TSIC	Timestamp Ingress Correction This field contains the ingress path correction value as defined by the Ingress Correction expression.

76.17.235 MAC Timestamp Egress Correction In Nanoseconds (MAC_Timestamp_Egress_Corr_Nanosecond)

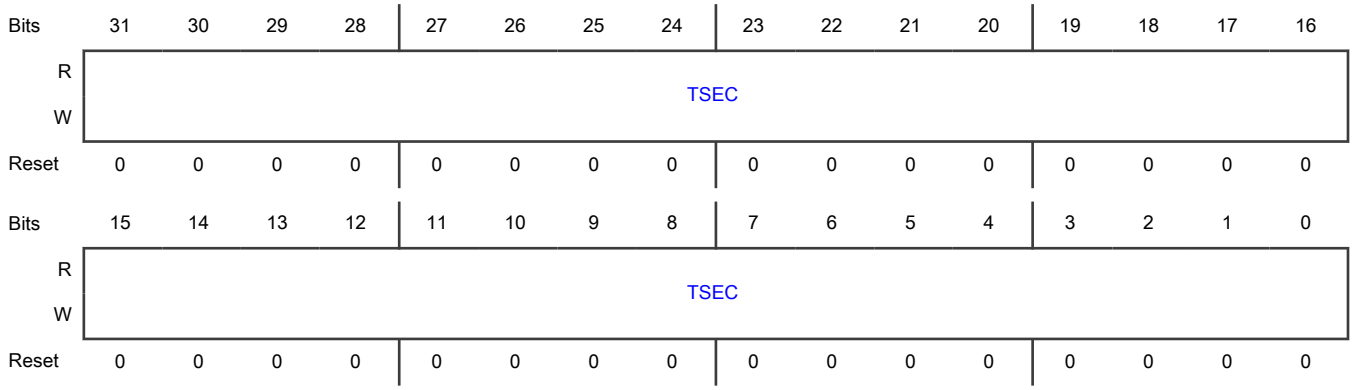
Offset

Register	Offset
MAC_Timestamp_Egress_Corr_Nanosecond	B5Ch

Function

This register contains the correction value in nanoseconds to be used with the captured timestamp value in the egress path.

Diagram



Fields

Field	Function
31-0 TSEC	Timestamp Egress Correction This field contains the nanoseconds part of the egress path correction value as defined by the Egress Correction expression.

76.17.236 MAC Timestamp Ingress Correction In Subnanoseconds (MAC_Timestamp_Ingress_Corr_Subnanosec)

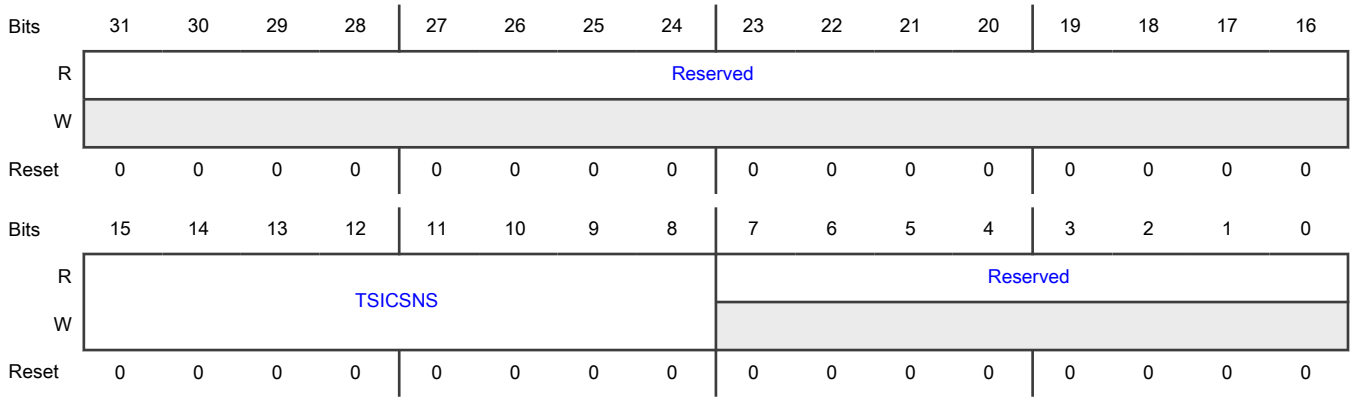
Offset

Register	Offset
MAC_Timestamp_Ingress_Corr_Subnanosec	B60h

Function

This register contains the sub-nanosecond part of the correction value to be used with the captured timestamp value, for ingress direction.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-8 TSICSNS	Timestamp Ingress Correction, sub-nanoseconds This field contains the sub-nanoseconds part of the ingress path correction value as defined by the "Ingress Correction" expression.
7-0 —	Reserved

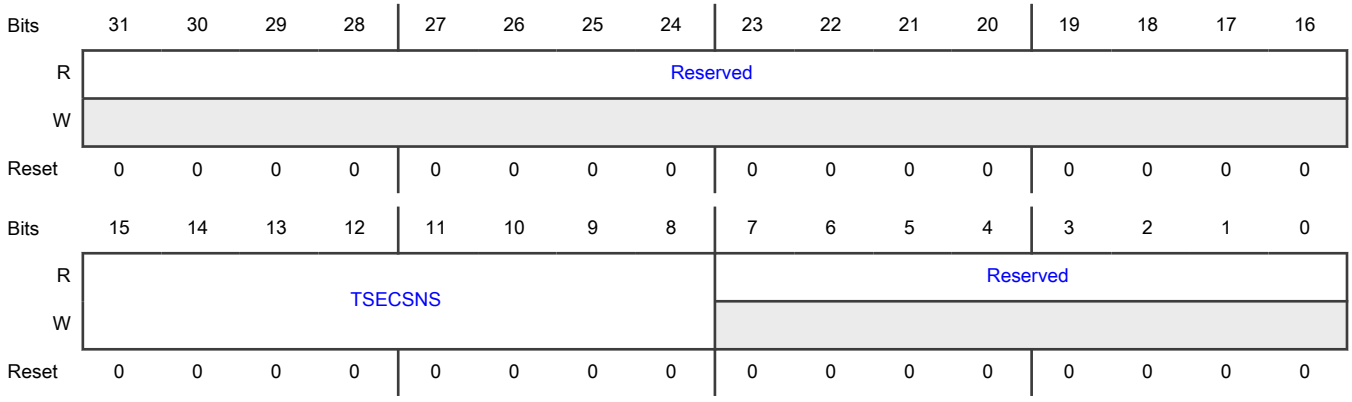
76.17.237 MAC Timestamp Egress Correction In Subnanoseconds (MAC_Timestamp_Egress_Corr_Subnanosec)

Offset

Register	Offset
MAC_Timestamp_Egress_Corr_Subnanosec	B64h

Function
This register contains the sub-nanosecond part of the correction value to be used with the captured timestamp value, for egress direction.

Diagram



Fields

Field	Function
31-16	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
15-8 TSECSNS	Timestamp Egress Correction, sub-nanoseconds This field contains the sub-nanoseconds part of the egress path correction value as defined by the "Egress Correction" expression.
7-0 —	Reserved

76.17.238 MAC Timestamp Ingress Latency (MAC_Timestamp_Ingress_Latency)

Offset

Register	Offset
MAC_Timestamp_Ingress_Latency	B68h

Function
This register holds the Ingress MAC latency.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved				ITLNS											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ITLSNS								Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved
27-16	Ingress Timestamp Latency, in sub-nanoseconds

Table continues on the next page...

Table continued from the previous page...

Field	Function
ITLNS	This register holds the average latency in sub-nanoseconds between the input ports (PHY_RXD_I) of MAC and the actual point (GMII/MII) where the ingress timestamp is taken. Ingress correction value is computed as described in the section, "Ingress Correction".
15-8 ITLSNS	Ingress Timestamp Latency, in nanoseconds This register holds the average latency in nanoseconds between the input ports (PHY_RXD_I) of MAC and the actual point (GMII/MII) where the ingress timestamp is taken. Ingress correction value is computed as described in the section, "Ingress Correction".
7-0 —	Reserved

76.17.239 MAC Timestamp Egress Latency (MAC_Timestamp_Egress_Latency)

Offset

Register	Offset
MAC_Timestamp_Egress_Latency	B6Ch

Function
This register holds the Egress MAC latency.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved				ETLNS											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ETLSNS								Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
27-16 ETLNS	Egress Timestamp Latency, in sub-nanoseconds This register holds the average latency in nanoseconds between the actual point (GMII/MII) where the egress timestamp is taken and the output ports (PHY_TXD_O) of the MAC. Ingress correction value is computed as described in the section, "Ingress Correction".
15-8 ETLSNS	Egress Timestamp Latency, in sub-nanoseconds This register holds the average latency in sub-nanoseconds between the actual point (GMII/MII) where the egress timestamp is taken and the output ports (PHY_TXD_O) of the MAC. Ingress correction value is computed as described in the section "Ingress Correction".
7-0 —	Reserved

76.17.240 MAC PPS Control (MAC_PPS_Control)

Offset

Register	Offset
MAC_PPS_Control	B70h

Function

PPS Control register. Bits[30:24] of this register are valid only when four Flexible PPS outputs are selected. Bits[22:16] are valid only when three or more Flexible PPS outputs are selected. Bits[14:8] are valid only when two or more Flexible PPS outputs are selected. Bits[6:4] are valid only when Flexible PPS feature is selected.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MCGREN3	TRGTMODSEL3		TIMESEL	PPSCMD3				MCGREN2	TRGTMODSEL2		Reserved	PPSCMD2			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MCGREN1	TRGTMODSEL1		Reserved	PPSCMD1				MCGREN0	TRGTMODSEL0		PPSEN0	PPSCTRL_PPSCMD			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 MCGREN3	MCGR Mode Enable for PPS3 Output This field enables the 3rd PPS instance to operate in PPS or MCGR mode. - 1: Operates in MCGR mode - 0: Operates in PPS mode
30-29 TRGTMODSEL3	Target Time Register Mode for PPS3 Output This field indicates the Target Time registers (MAC_PPS3_Target_Time_Seconds and MAC_PPS3_Target_Time_Nanoseconds) mode for PPS3 output signal. 00b - Target Time registers are programmed only for generating the interrupt event. The Flexible PPS function must not be enabled in this mode, otherwise spurious transitions may be observed on the corresponding PTP_PPS_O output port 01b - Enables MCGR Interrupt whose status bit is indicated by TSTARGT3 (MAC_Timestamp_Status[8]) 10b - Target Time registers are programmed for generating the interrupt event and starting or stopping the PPS0 output signal generation 11b - Target Time registers are programmed only for starting or stopping the PPS0 output signal generation. No interrupt is asserted
28 TIMESEL	Time Select When this bit is set, 64 bit PTP time is used to capture time at MCGR trigger[0] input When this bit is reset, presentation time is used to capture time at trigger input, maintaining backward compatibility
27-24 PPSCMD3	Flexible PPS3 Output Control This field controls the flexible PPS3 output (ptp_pps_o[3]) signal. This field is similar to the PPSCMD0[2:0] field. - When MCGREN3 is 1, PPSCMD3 indicated by the 4 bits [27:24] are taken as Presentation Time Control bits for media clock generation and recovery for comparator instance 3. This field is similar to the PPSCMD0 Presentation Time Control bits. - When MCGREN3 is 0, only 3 bits from [26:24] are used as PPSCMD3 and the 4th bit must be set as 0. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect.
23 MCGREN2	MCGR Mode Enable for PPS2 Output This field enables the 2nd PPS instance to operate in PPS or MCGR mode. - 1: Operates in MCGR mode - 0: Operates in PPS mode 0b - 2nd PPS instance is disabled to operate in PPS or MCGR mode 1b - 2nd PPS instance is enabled to operate in PPS or MCGR mode
22-21 TRGTMODSEL2	Target Time Register Mode for PPS2 Output This field indicates the Target Time registers (MAC_PPS2_Target_Time_Seconds and MAC_PPS2_Target_Time_Nanoseconds) mode for PPS2 output signal. 00b - Target Time registers are programmed only for generating the interrupt event. The Flexible PPS function must not be enabled in this mode, otherwise spurious transitions may be observed on the corresponding ptp_pps_o output port 01b - Enables MCGR Interrupt whose status bit is indicated by TSTARGT2 (MAC_Timestamp_Status[6])

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10b - Target Time registers are programmed for generating the interrupt event and starting or stopping the PPS0 output signal generation 11b - Target Time registers are programmed only for starting or stopping the PPS0 output signal generation. No interrupt is asserted
20 —	Reserved
19-16 PPSCMD2	Flexible PPS2 Output Control This field controls the flexible PPS2 output (ptp_pps_o[2]) signal. This field is similar to the PPSCMD0 field. - When MCGREN2 is 1, PPSCMD2 indicated by the 4 bits, [19:16] are taken as Presentation Time Control bits for media clock generation and recovery for comparator instance 2. This field is similar to the PPSCMD0 Presentation Time Control bits. - When MCGREN2 is 0, only 3 bits, [18:16] are used as PPSCMD2 and the 4th bit must be set as 0. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect.
15 MCGREN1	MCGR Mode Enable for PPS1 Output This field enables the 1st PPS instance to operate in PPS or MCGR mode. - 1: Operates in MCGR mode - 0: Operates in PPS mode 0b - 1st PPS instance is disabled to operate in PPS or MCGR mode 1b - 1st PPS instance is enabled to operate in PPS or MCGR mode
14-13 TRGTMODSEL1	Target Time Register Mode for PPS1 Output This field indicates the Target Time registers (MAC_PPS1_Target_Time_Seconds and MAC_PPS1_Target_Time_Nanoseconds) mode for PPS1 output signal. 00b - Target Time registers are programmed only for generating the interrupt event. The Flexible PPS function must not be enabled in this mode, otherwise spurious transitions may be observed on the corresponding PTP_PPS_O output port 01b - Enables MCGR Interrupt whose status bit is indicated by TSTARGT1 (MAC_Timestamp_Status[4]) 10b - Target Time registers are programmed for generating the interrupt event and starting or stopping the PPS0 output signal generation 11b - Target Time registers are programmed only for starting or stopping the PPS0 output signal generation. No interrupt is asserted
12 —	Reserved
11-8 PPSCMD1	Flexible PPS1 Output Control This field controls the flexible PPS1 output (ptp_pps_o[1]) signal. This field is similar to the PPSCMD0 field. - When MCGREN1 is 1, PPSCMD1 indicated by the 4 bits [11:8] are taken as Presentation Time Control bits for media clock generation and recovery for comparator instance 1. This field is similar to the PPSCMD0 Presentation Time Control bits. - When MCGREN1 is 0, only 3 bits, [10:8] are used as

Table continues on the next page...

Table continued from the previous page...

Field	Function
	PPSCMD1 and the 4th bit must be set as 0. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect.
7 MCGREN0	MCGR Mode Enable for PPS0 Output This field enables the 0th PPS instance to operate in PPS or MCGR mode. When set it operates in MCGR mode and on reset it operates in PPS mode. 0b - 0th PPS instance is enabled to operate in PPS mode 1b - 0th PPS instance is enabled to operate in MCGR mode
6-5 TRGTMODSEL0	Target Time Register Mode for PPS0 Output This field indicates the Target Time registers (MAC_PPS0_Target_Time_Seconds and MAC_PPS0_Target_Time_Nanoseconds) mode for PPS0 output signal. 00b - Target Time registers are programmed only for generating the interrupt event. The Flexible PPS function must not be enabled in this mode, otherwise spurious transitions may be observed on the corresponding PTP_PPS_O output port 01b - Enables MCGR Interrupt whose status bit is indicated by TSTARGT0 (MAC_Timestamp_Status[1]) 10b - Target Time registers are programmed for generating the interrupt event and starting or stopping the PPS0 output signal generation 11b - Target Time registers are programmed only for starting or stopping the PPS0 output signal generation. No interrupt is asserted
4 PPSEN0	Flexible PPS Output Mode Enable When this bit is - 1: Bits[3:0] function as PPSCMD. - 0: Bits[3:0] function as PPSCTRL (Fixed PPS mode). 0b - Flexible PPS Output Mode is disabled 1b - Flexible PPS Output Mode is enabled
3-0 PPSCTRL_PPSCMD	PPS Output Frequency Control This field controls the frequency of the PPS0 output (ptp_pps_o[0]) signal. The default value of PPSCTRL is 0000, and the PPS output is 1 pulse (of width clk_ptp_i) every second. For other values of PPSCTRL, the PPS output becomes a generated clock of following frequencies: - 0001: The binary rollover is 2 Hz, and the digital rollover is 1 Hz. - 0010: The binary rollover is 4 Hz, and the digital rollover is 2 Hz. - 0011: The binary rollover is 8 Hz, and the digital rollover is 4 Hz. - 0100: The binary rollover is 16 Hz, and the digital rollover is 8 Hz. - .. - 1111: The binary rollover is 32.768 KHz and the digital rollover is 16.384 KHz. Note: In the binary rollover mode, the PPS output (ptp_pps_o) has a duty cycle of 50 percent with these frequencies. In the digital rollover mode, the PPS output frequency is an average number. The actual clock is of different frequency that gets synchronized every second. For example: - When PPSCTRL = 0001, the PPS (1 Hz) has a low period of 537 ms and a high period of 463 ms - When PPSCTRL = 0010, the PPS (2 Hz) is a sequence of One clock of 50 percent duty cycle and 537 ms period Second clock of 463 ms period (268 ms low and 195 ms high) - When PPSCTRL = 0011, the PPS (4 Hz) is a sequence of Three clocks of 50 percent duty cycle and 268 ms period Fourth clock of 195 ms period (134 ms low and 61 ms high) This behavior is because of the non-linear toggling of bits in the digital rollover mode in the MAC_System_Time_Nanoseconds

Table continues on the next page...

Table continued from the previous page...

Field	Function
	register. or Flexible PPS Output (ptp_pps_o[0]) Control Programming these bits with a non-zero value instructs the MAC to initiate an event. When the command is transferred or synchronized to the PTP clock domain, these bits get cleared automatically. The software should ensure that these bits are programmed only when they are 'all-zero'. The following list describes the values of PPSCMD0: - 0000: No Command - 0001: START Single Pulse This command generates single pulse rising at the start point defined in MAC_PPS0_Target_Time_Seconds and MAC_PPS0_Target_Time_Nanoseconds register and of a duration defined in the PPS0 Width Register. - 0010: START Pulse Train This command generates the train of pulses rising at the start point defined in the Target Time Registers and of a duration defined in the PPS0 Width Register and repeated at interval defined in the PPS Interval Register. By default, the PPS pulse train is free-running unless stopped by the 'Stop Pulse train at time' or 'Stop Pulse Train immediately' commands. - 0011: Cancel START This command cancels the START Single Pulse and START Pulse Train commands if the system time has not crossed the programmed start time. - 0100: STOP Pulse train at time This command stops the train of pulses initiated by the START Pulse Train command (PPSCMD = 0010) after the time programmed in the Target Time registers elapses. - 0101: STOP Pulse Train immediately This command immediately stops the train of pulses initiated by the START Pulse Train command (PPSCMD = 0010). - 110: Cancel STOP Pulse train This command cancels the STOP pulse train at time command if the programmed stop time has not elapsed. The PPS pulse train becomes free-running on the successful execution of this command. - 0111-1111: Reserved or Presentation Time Control When MCGREN0 is 1, these bits are treated as Presentation time control bits. The following list describes the values of PPSCMD0: - 0000: MCGR operation is not carried out. If set to this value in the mid of clock recovery or generation, all the processing inputs are flushed - 0001: Capture the Presentation time at the rising edge of MCG_PST_TRIG_I[0] into the MAC_PPS0_Target_Time_Seconds register - 0010: Capture the Presentation time at the falling edge of MCG_PST_TRIG_I[0] into the MAC_PPS0_Target_Time_Seconds register - 0011: Capture the Presentation time at both edges of MCG_PST_TRIG_I[0] into the MAC_PPS0_Target_Time_Seconds register - 0100-1000: Reserved - 1001: Toggle output on compare - 1010: Pulse output low on compare for one PTP-clock cycle - 1011: Pulse output high on compare for one PTP-clock cycle - 1100-1111: Reserved

76.17.241 MAC PPS0 Target Time In Seconds (MAC_PPS0_Target_Time_Seconds)

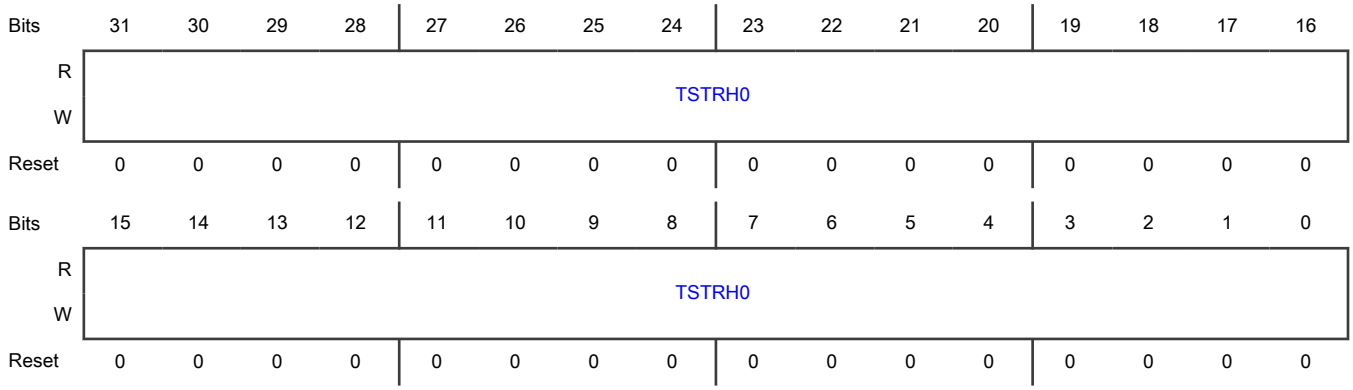
Offset

Register	Offset
MAC_PPS0_Target_Time_Seconds	B80h

Function

The PPS Target Time Seconds register, along with PPS Target Time Nanoseconds register, is used to schedule an interrupt event [Bit 1 of MAC_Timestamp_Status] when the system time exceeds the value programmed in these registers.

Diagram



Fields

Field	Function
31-0 TSTRH0	PPS Target Time Seconds Register This field stores the time in seconds. When the timestamp value matches or exceeds both Target Timestamp registers, the MAC starts or stops the PPS signal output and generates an interrupt (if enabled) based on Target Time mode selected for the corresponding PPS output in the MAC_PPS_Control register. If DWC_EQOS_FLEXI_PPS_OUT_EN is enabled in the configuration and PTGE field of MAC_Timestamp_Control Register is set with Presentation time control set in recovery mode, then these bits indicate the TPT being programmed by the application and in generation mode it indicates the CPT generated at the sampled trigger.

76.17.242 MAC PPS0 Target Time In Nanoseconds (MAC_PPS0_Target_Time_Nanoseconds)

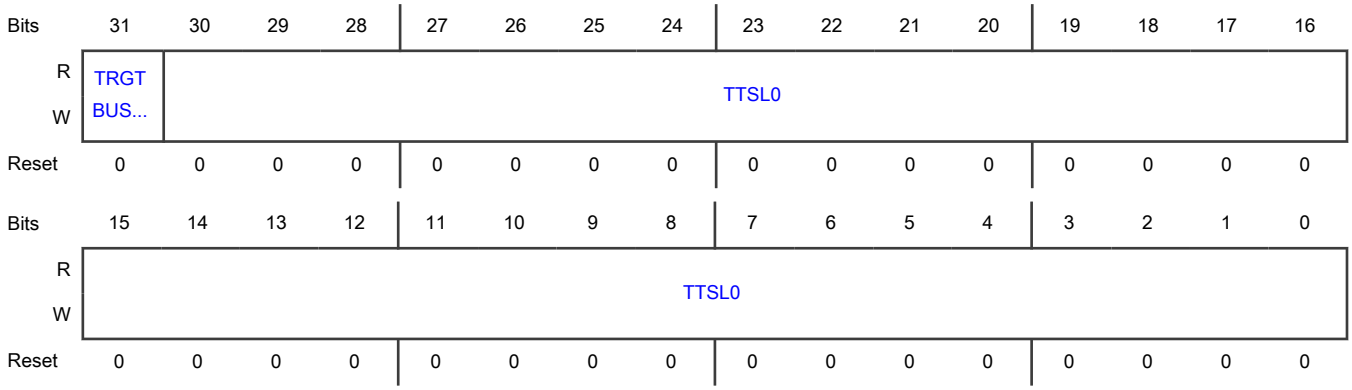
Offset

Register	Offset
MAC_PPS0_Target_Time_Nanoseconds	B84h

Function

PPS0 Target Time Nanoseconds register.

Diagram



Fields

Field	Function
31 TRGTBUSY0	PPS Target Time Register Busy The MAC sets this bit when the PPSCMD0 field in the MAC_PPS_Control register is programmed to 010 or 011. Programming the PPSCMD0 field to 010 or 011 instructs the MAC to synchronize the Target Time Registers to the PTP clock domain. The MAC clears this bit after synchronizing the Target Time Registers to the PTP clock domain The application must not update the Target Time Registers when this bit is read as 1. Otherwise, the synchronization of the previous programmed time gets corrupted. **NOTE** This field is not testable by an automated register test. 0b - PPS Target Time Register Busy status is not detected 1b - PPS Target Time Register Busy is detected
30-0 TTSL0	Target Time Low for PPS Register This register stores the time in (signed) nanoseconds. When the value of the timestamp matches the value in both Target Timestamp registers, the MAC starts or stops the PPS signal output and generates an interrupt (if enabled) based on the TRGTMODSEL0 field (Bits [6:5]) in MAC_PPS_Control. When the TSCTRLSSR bit is reset in the MAC_Timestamp_Control register, this value should be (time in ns / 0.465). The actual start or stop time of the PPS signal output might have an error margin up to one unit of sub-second increment value. When the TSCTRLSSR bit is set in the MAC_Timestamp_Control register, this value should not exceed 0x3B9A_C9FF. The actual start or stop time of the PPS signal output might have an error margin up to one unit of sub-second increment value. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect.

76.17.243 MAC PPS0 Interval (MAC_PPS0_Interval)

Offset

Register	Offset
MAC_PPS0_Interval	B88h

Function

The PPS0 Interval register contains the number of units of sub-second increment value between the rising edges of PPS0 signal output (PTP_PPS_O[0]).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PPSINT0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PPSINT0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PPSINT0	PPS Output Signal Interval These bits store the interval between the rising edges of PPS0 signal output. The interval is stored in terms of number of units of sub-second increment value. You need to program one value less than the required interval. For example, if the PTP reference clock is 50 MHz (period of 20 ns), and desired interval between the rising edges of PPS0 signal output is 100 ns (that is, 5 units of sub-second increment value), you should program value 4 (5-1) in this register.

76.17.244 MAC PPS0 Width (MAC_PPS0_Width)

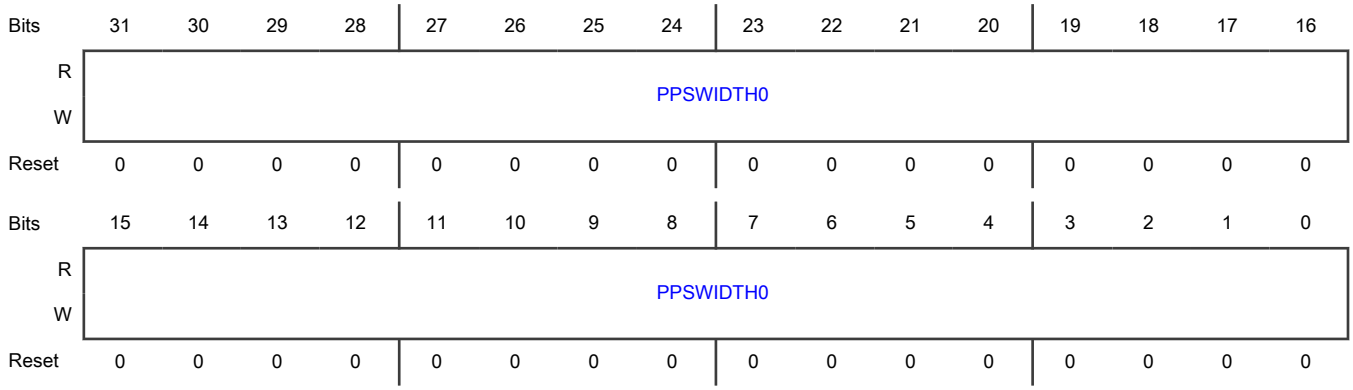
Offset

Register	Offset
MAC_PPS0_Width	B8Ch

Function

The PPS0 Width register contains the number of units of sub-second increment value between the rising and corresponding falling edges of PPS0 signal output (PTP_PPS_O[0]).

Diagram



Fields

Field	Function
31-0 PPSWIDTH0	PPS Output Signal Width These bits store the width between the rising edge and corresponding falling edge of PPS0 signal output. The width is stored in terms of number of units of sub-second increment value. You need to program one value less than the required interval. For example, if PTP reference clock is 50 MHz (period of 20 ns), and width between the rising and corresponding falling edges of PPS0 signal output is 80 ns (that is, four units of sub-second increment value), you should program value 3 (4-1) in this register. Note: The value programmed in this register must be lesser than the value programmed in MAC_PPS0_Interval.

76.17.245 MAC PPS1 Target Time In Seconds (MAC_PPS1_Target_Time_Seconds)

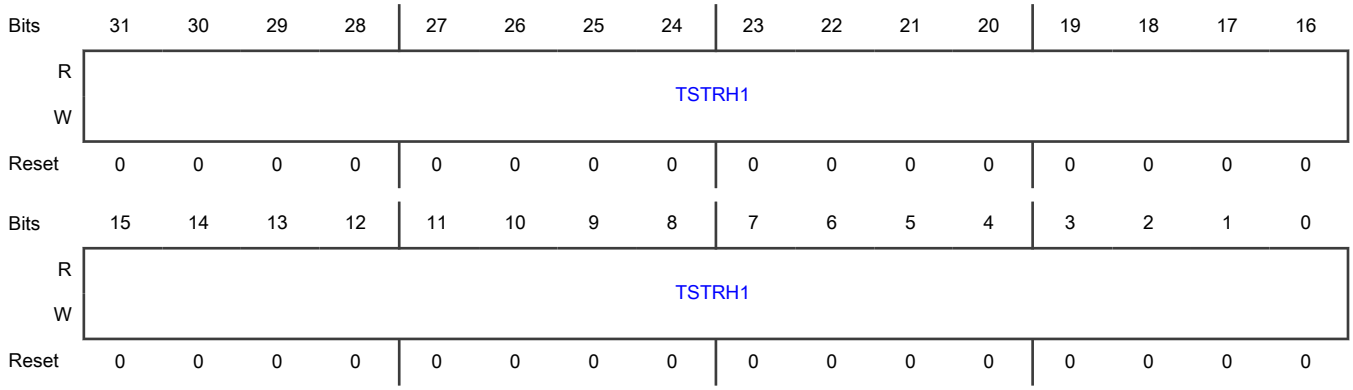
Offset

Register	Offset
MAC_PPS1_Target_Time_Seconds	B90h

Function

The PPS Target Time Seconds register, along with PPS Target Time Nanoseconds register, is used to schedule an interrupt event [Bit 1 of MAC_Timestamp_Status] when the system time exceeds the value programmed in these registers.

Diagram



Fields

Field	Function
31-0 TSTRH1	PPS Target Time Seconds Register This field stores the time in seconds. When the timestamp value matches or exceeds both Target Timestamp registers, the MAC starts or stops the PPS signal output and generates an interrupt (if enabled) based on Target Time mode selected for the corresponding PPS output in the MAC_PPS_Control register. If DWC_EQOS_FLEXI_PPS_OUT_EN is enabled in the configuration and PTGE field of MAC_Timestamp_Control Register is set with Presentation time control set in recovery mode, then these bits indicate the TPT being programmed by the application and in generation mode it indicates the CPT generated at the sampled trigger.

76.17.246 MAC PPS1 Target Time In Nanoseconds (MAC_PPS1_Target_Time_Nanoseconds)

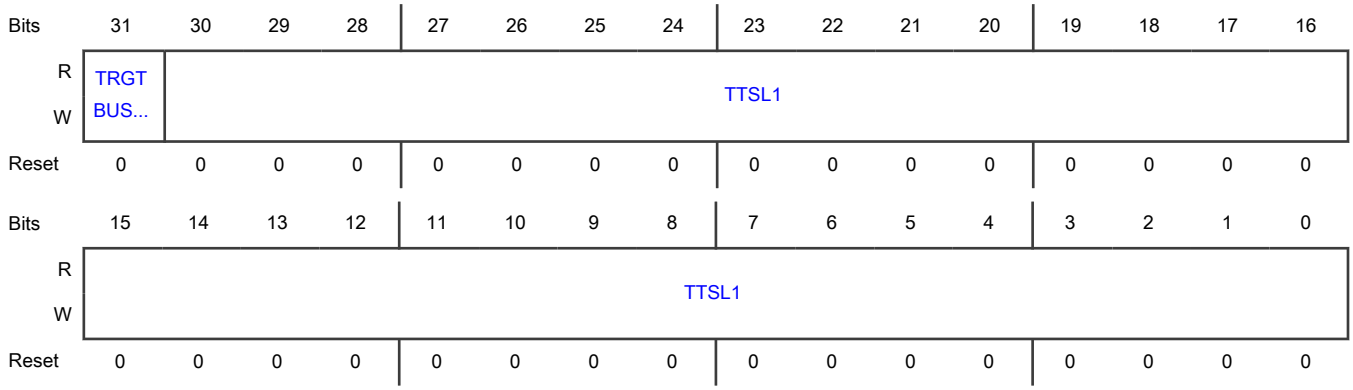
Offset

Register	Offset
MAC_PPS1_Target_Time_Nanoseconds	B94h

Function

PPS0 Target Time Nanoseconds register.

Diagram



Fields

Field	Function
31 TRGTBUSY1	PPS Target Time Register Busy The MAC sets this bit when the PPSCMD0 field in the MAC_PPS_Control register is programmed to 010 or 011. Programming the PPSCMD0 field to 010 or 011 instructs the MAC to synchronize the Target Time Registers to the PTP clock domain. The MAC clears this bit after synchronizing the Target Time Registers to the PTP clock domain The application must not update the Target Time Registers when this bit is read as 1. Otherwise, the synchronization of the previous programmed time gets corrupted. **NOTE** This field is not testable by an automated register test. 0b - PPS Target Time Register Busy status is not detected 1b - PPS Target Time Register Busy is detected
30-0 TTSL1	Target Time Low for PPS Register This register stores the time in (signed) nanoseconds. When the value of the timestamp matches the value in both Target Timestamp registers, the MAC starts or stops the PPS signal output and generates an interrupt (if enabled) based on the TRGTMODSEL0 field (Bits [6:5]) in MAC_PPS_Control. When the TSCTRLSSR bit is reset in the MAC_Timestamp_Control register, this value should be (time in ns / 0.465). The actual start or stop time of the PPS signal output might have an error margin up to one unit of sub-second increment value. When the TSCTRLSSR bit is set in the MAC_Timestamp_Control register, this value should not exceed 0x3B9A_C9FF. The actual start or stop time of the PPS signal output might have an error margin up to one unit of sub-second increment value. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect.

76.17.247 MAC PPS1 Interval (MAC_PPS1_Interval)

Offset

Register	Offset
MAC_PPS1_Interval	B98h

Function

The PPS0 Interval register contains the number of units of sub-second increment value between the rising edges of PPS0 signal output (PTP_PPS_O[0]).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PPSINT1															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PPSINT1															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PPSINT1	PPS Output Signal Interval These bits store the interval between the rising edges of PPS0 signal output. The interval is stored in terms of number of units of sub-second increment value. You need to program one value less than the required interval. For example, if the PTP reference clock is 50 MHz (period of 20 ns), and desired interval between the rising edges of PPS0 signal output is 100 ns (that is, 5 units of sub-second increment value), you should program value 4 (5-1) in this register.

76.17.248 MAC PPS1 Width (MAC_PPS1_Width)

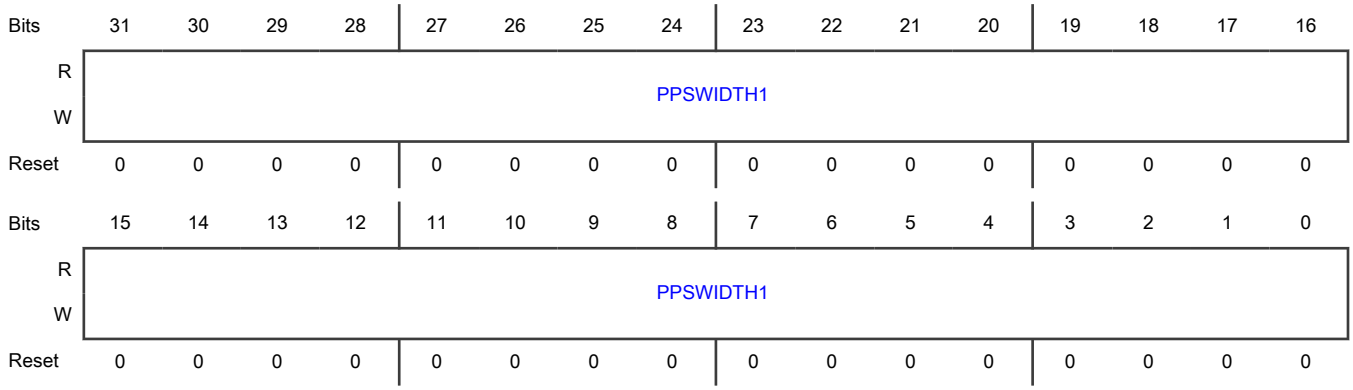
Offset

Register	Offset
MAC_PPS1_Width	B9Ch

Function

The PPS0 Width register contains the number of units of sub-second increment value between the rising and corresponding falling edges of PPS0 signal output (PTP_PPS_O[0]).

Diagram



Fields

Field	Function
31-0 PPSWIDTH1	PPS Output Signal Width These bits store the width between the rising edge and corresponding falling edge of PPS0 signal output. The width is stored in terms of number of units of sub-second increment value. You need to program one value less than the required interval. For example, if PTP reference clock is 50 MHz (period of 20 ns), and width between the rising and corresponding falling edges of PPS0 signal output is 80 ns (that is, four units of sub-second increment value), you should program value 3 (4-1) in this register. Note: The value programmed in this register must be lesser than the value programmed in MAC_PPS0_Interval.

76.17.249 MAC PPS2 Target Time In Seconds (MAC_PPS2_Target_Time_Seconds)

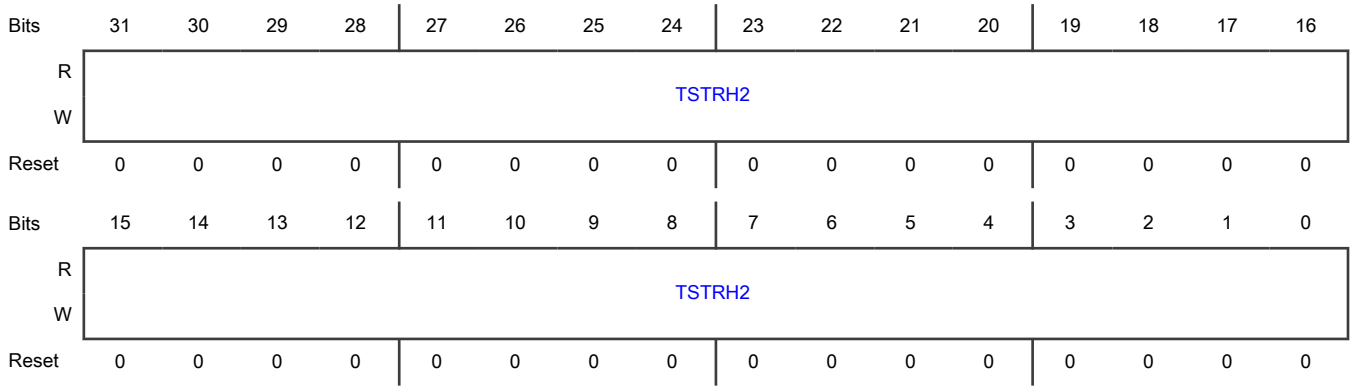
Offset

Register	Offset
MAC_PPS2_Target_Time_Seconds	BA0h

Function

The PPS Target Time Seconds register, along with PPS Target Time Nanoseconds register, is used to schedule an interrupt event [Bit 1 of MAC_Timestamp_Status] when the system time exceeds the value programmed in these registers.

Diagram



Fields

Field	Function
31-0 TSTRH2	PPS Target Time Seconds Register This field stores the time in seconds. When the timestamp value matches or exceeds both Target Timestamp registers, the MAC starts or stops the PPS signal output and generates an interrupt (if enabled) based on Target Time mode selected for the corresponding PPS output in the MAC_PPS_Control register. If DWC_EQOS_FLEXI_PPS_OUT_EN is enabled in the configuration and PTGE field of MAC_Timestamp_Control Register is set with Presentation time control set in recovery mode, then these bits indicate the TPT being programmed by the application and in generation mode it indicates the CPT generated at the sampled trigger.

76.17.250 MAC PPS2 Target Time In Nanoseconds (MAC_PPS2_Target_Time_Nanoseconds)

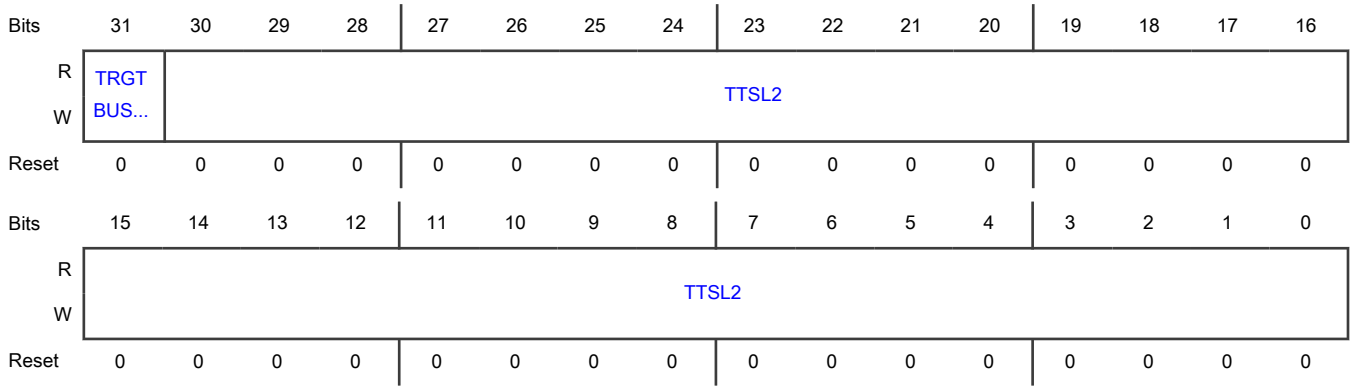
Offset

Register	Offset
MAC_PPS2_Target_Time_Nanoseconds	BA4h

Function

PPS0 Target Time Nanoseconds register.

Diagram



Fields

Field	Function
31 TRGTBUSY2	PPS Target Time Register Busy The MAC sets this bit when the PPSCMD0 field in the MAC_PPS_Control register is programmed to 010 or 011. Programming the PPSCMD0 field to 010 or 011 instructs the MAC to synchronize the Target Time Registers to the PTP clock domain. The MAC clears this bit after synchronizing the Target Time Registers to the PTP clock domain The application must not update the Target Time Registers when this bit is read as 1. Otherwise, the synchronization of the previous programmed time gets corrupted. **NOTE** This field is not testable by an automated register test. 0b - PPS Target Time Register Busy status is not detected 1b - PPS Target Time Register Busy is detected
30-0 TTSL2	Target Time Low for PPS Register This register stores the time in (signed) nanoseconds. When the value of the timestamp matches the value in both Target Timestamp registers, the MAC starts or stops the PPS signal output and generates an interrupt (if enabled) based on the TRGTMODSEL0 field (Bits [6:5]) in MAC_PPS_Control. When the TSCTRLSSR bit is reset in the MAC_Timestamp_Control register, this value should be (time in ns / 0.465). The actual start or stop time of the PPS signal output might have an error margin up to one unit of sub-second increment value. When the TSCTRLSSR bit is set in the MAC_Timestamp_Control register, this value should not exceed 0x3B9A_C9FF. The actual start or stop time of the PPS signal output might have an error margin up to one unit of sub-second increment value. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect.

76.17.251 MAC PPS2 Interval (MAC_PPS2_Interval)

Offset

Register	Offset
MAC_PPS2_Interval	BA8h

Function

The PPS0 Interval register contains the number of units of sub-second increment value between the rising edges of PPS0 signal output (PTP_PPS_O[0]).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PPSINT2															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PPSINT2															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PPSINT2	PPS Output Signal Interval These bits store the interval between the rising edges of PPS0 signal output. The interval is stored in terms of number of units of sub-second increment value. You need to program one value less than the required interval. For example, if the PTP reference clock is 50 MHz (period of 20 ns), and desired interval between the rising edges of PPS0 signal output is 100 ns (that is, 5 units of sub-second increment value), you should program value 4 (5-1) in this register.

76.17.252 MAC PPS2 Width (MAC_PPS2_Width)

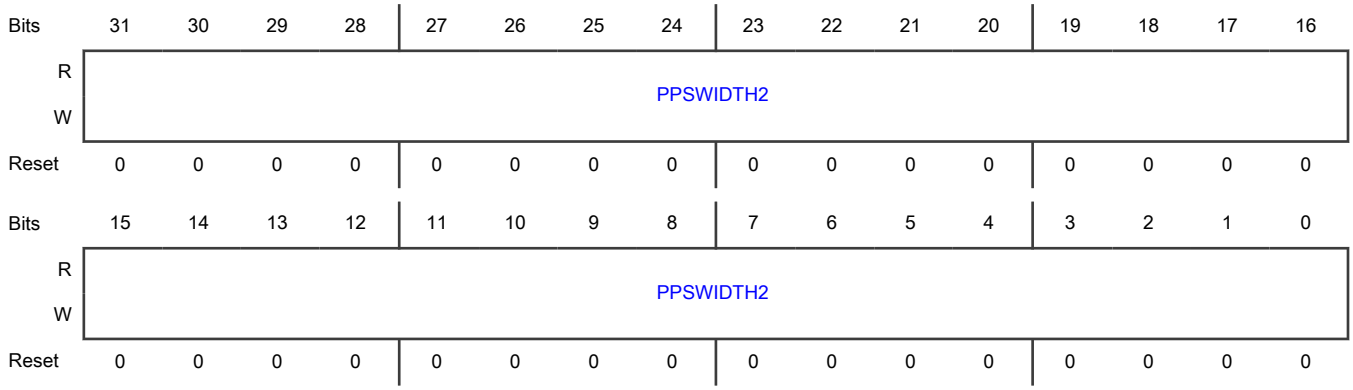
Offset

Register	Offset
MAC_PPS2_Width	BACh

Function

The PPS0 Width register contains the number of units of sub-second increment value between the rising and corresponding falling edges of PPS0 signal output (PTP_PPS_O[0]).

Diagram



Fields

Field	Function
31-0 PPSWIDTH2	PPS Output Signal Width These bits store the width between the rising edge and corresponding falling edge of PPS0 signal output. The width is stored in terms of number of units of sub-second increment value. You need to program one value less than the required interval. For example, if PTP reference clock is 50 MHz (period of 20 ns), and width between the rising and corresponding falling edges of PPS0 signal output is 80 ns (that is, four units of sub-second increment value), you should program value 3 (4-1) in this register. Note: The value programmed in this register must be lesser than the value programmed in MAC_PPS0_Interval.

76.17.253 MAC PPS3 Target Time In Seconds (MAC_PPS3_Target_Time_Seconds)

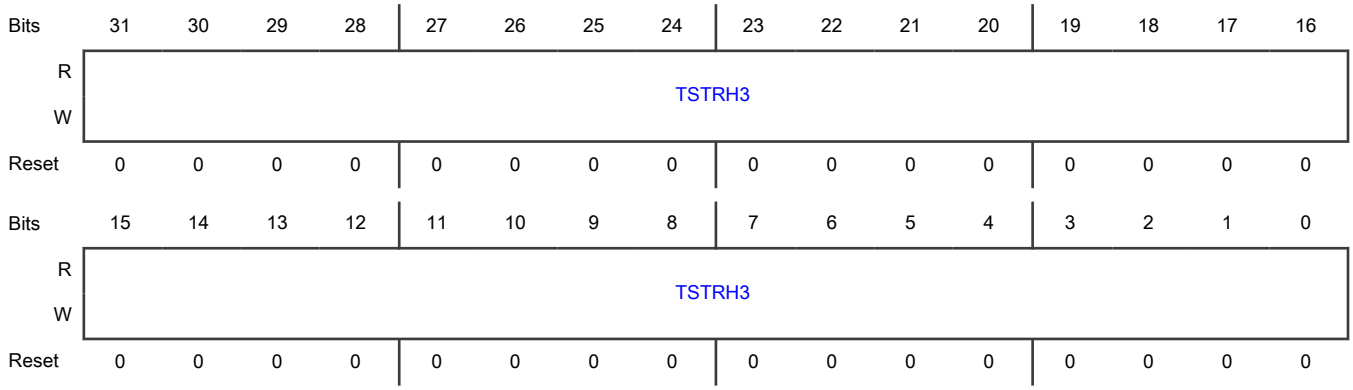
Offset

Register	Offset
MAC_PPS3_Target_Time_Seconds	BB0h

Function

The PPS Target Time Seconds register, along with PPS Target Time Nanoseconds register, is used to schedule an interrupt event [Bit 1 of MAC_Timestamp_Status] when the system time exceeds the value programmed in these registers.

Diagram



Fields

Field	Function
31-0 TSTRH3	PPS Target Time Seconds Register This field stores the time in seconds. When the timestamp value matches or exceeds both Target Timestamp registers, the MAC starts or stops the PPS signal output and generates an interrupt (if enabled) based on Target Time mode selected for the corresponding PPS output in the MAC_PPS_Control register. If DWC_EQOS_FLEXI_PPS_OUT_EN is enabled in the configuration and PTGE field of MAC_Timestamp_Control Register is set with Presentation time control set in recovery mode, then these bits indicate the TPT being programmed by the application and in generation mode it indicates the CPT generated at the sampled trigger.

76.17.254 MAC PPS3 Target Time In Nanoseconds (MAC_PPS3_Target_Time_Nanoseconds)

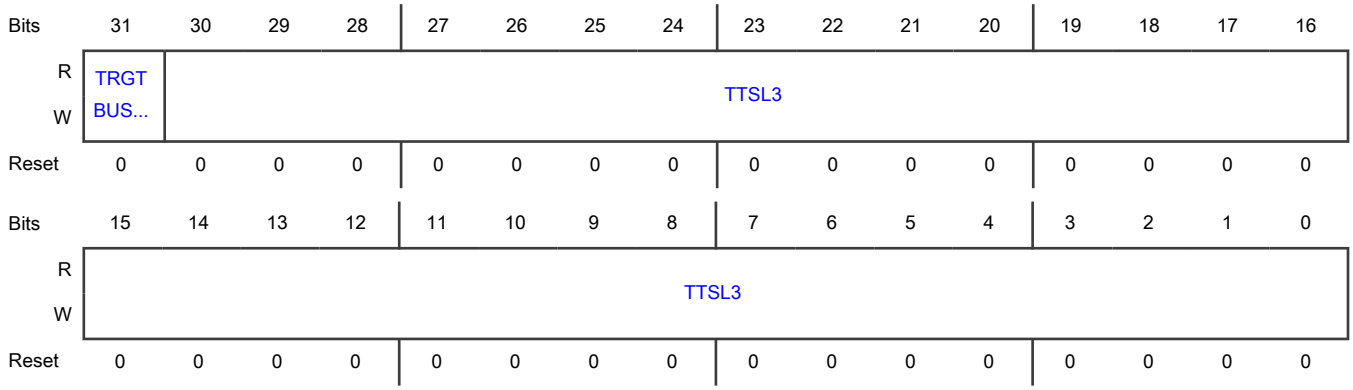
Offset

Register	Offset
MAC_PPS3_Target_Time_Nanoseconds	BB4h

Function

PPS0 Target Time Nanoseconds register.

Diagram



Fields

Field	Function
31 TRGTBUSY3	PPS Target Time Register Busy The MAC sets this bit when the PPSCMD0 field in the MAC_PPS_Control register is programmed to 010 or 011. Programming the PPSCMD0 field to 010 or 011 instructs the MAC to synchronize the Target Time Registers to the PTP clock domain. The MAC clears this bit after synchronizing the Target Time Registers to the PTP clock domain The application must not update the Target Time Registers when this bit is read as 1. Otherwise, the synchronization of the previous programmed time gets corrupted. **NOTE** This field is not testable by an automated register test. 0b - PPS Target Time Register Busy status is not detected 1b - PPS Target Time Register Busy is detected
30-0 TTSL3	Target Time Low for PPS Register This register stores the time in (signed) nanoseconds. When the value of the timestamp matches the value in both Target Timestamp registers, the MAC starts or stops the PPS signal output and generates an interrupt (if enabled) based on the TRGTMODSEL0 field (Bits [6:5]) in MAC_PPS_Control. When the TSCTRLSSR bit is reset in the MAC_Timestamp_Control register, this value should be (time in ns / 0.465). The actual start or stop time of the PPS signal output might have an error margin up to one unit of sub-second increment value. When the TSCTRLSSR bit is set in the MAC_Timestamp_Control register, this value should not exceed 0x3B9A_C9FF. The actual start or stop time of the PPS signal output might have an error margin up to one unit of sub-second increment value. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect.

76.17.255 MAC PPS3 Interval (MAC_PPS3_Interval)

Offset

Register	Offset
MAC_PPS3_Interval	BB8h

Function

The PPS0 Interval register contains the number of units of sub-second increment value between the rising edges of PPS0 signal output (PTP_PPS_O[0]).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PPSINT3															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PPSINT3															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PPSINT3	PPS Output Signal Interval These bits store the interval between the rising edges of PPS0 signal output. The interval is stored in terms of number of units of sub-second increment value. You need to program one value less than the required interval. For example, if the PTP reference clock is 50 MHz (period of 20 ns), and desired interval between the rising edges of PPS0 signal output is 100 ns (that is, 5 units of sub-second increment value), you should program value 4 (5-1) in this register.

76.17.256 MAC PPS3 Width (MAC_PPS3_Width)

Offset

Register	Offset
MAC_PPS3_Width	BBCh

Function

The PPS0 Width register contains the number of units of sub-second increment value between the rising and corresponding falling edges of PPS0 signal output (PTP_PPS_O[0]).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PPSWIDTH3															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PPSWIDTH3															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PPSWIDTH3	PPS Output Signal Width These bits store the width between the rising edge and corresponding falling edge of PPS0 signal output. The width is stored in terms of number of units of sub-second increment value. You need to program one value less than the required interval. For example, if PTP reference clock is 50 MHz (period of 20 ns), and width between the rising and corresponding falling edges of PPS0 signal output is 80 ns (that is, four units of sub-second increment value), you should program value 3 (4-1) in this register. Note: The value programmed in this register must be lesser than the value programmed in MAC_PPS0_Interval.

76.17.257 MTL Operation Mode (MTL_Operation_Mode)

Offset

Register	Offset
MTL_Operation_Mode	C00h

Function

The Operation Mode register establishes the Transmit and Receive operating modes and commands.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FRPE	RXPED	Reserved		CNTCLR	CNTPRST	Reserved		SCHALG		Reserved		RAA	DTXSTS	Reserved	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15 FRPE	Flexible Rx parser Enable - 1: Enables Programmable Rx Parser - 0: Disables Programmable Rx Parser When the Rx parser is disabled when parsing is in progress, the Rx parser is disabled only after the current packet parsing complete. When the Rx parser is enabled, the parser gets activated for the next packet. 0b - Flexible Rx parser is disabled 1b - Flexible Rx parser is enabled
14 RXPED	RxParser Software Error/Incomplete Parsing Packet Drop Enable when this bit is set to 0, packets encountering software programming errors (NPE/NVE/frame offset overflow errors) or incomplete parsing are forwarded to application with the corresponding RxParser status. When this bit is set to 1, backward compatibility is maintained where all the specified packets are dropped (when RA is not set) 0b - Flexible Rx parser, packet drop in case software error is disabled 1b - Flexible Rx parser, packet drop in case software error is enabled
13-10 —	Reserved
9 CNTCLR	Counters Reset When this bit is set, all counters are reset. This bit is cleared automatically after 1 clock cycle. If this bit is set along with CNT_PRESET bit, CNT_PRESET has precedence. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. **NOTE** This field is not testable by an automated register test. 0b - Counters are not reset 1b - All counters are reset
8 CNTPRST	Counters Preset When this bit is set, - MTL_TxQ[0-7]_Underflow register is initialized/preset to 12'h7F0. - Missed Packet and Overflow Packet counters in MTL_RxQ[0-7]_Missed_Packet_Overflow_Cnt register is initialized/ preset to 12'h7F0. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. **NOTE** This field is not testable by an automated register test. 0b - Counters Preset is disabled 1b - Counters Preset is enabled
7	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
6-5 SCHALG	Tx Scheduling Algorithm This field indicates the algorithm for Tx scheduling: 00b - WRR algorithm 01b - WFQ algorithm when DCB feature is selected.Otherwise, Reserved 10b - DWRR algorithm when DCB feature is selected.Otherwise, Reserved 11b - Strict priority algorithm
4-3 —	Reserved
2 RAA	Receive Arbitration Algorithm This field is used to select the arbitration algorithm for the Rx side. - 0: Strict priority (SP). Queue 0 has the lowest priority and the last queue has the highest priority. - 1: Weighted Strict Priority (WSP) 0b - Strict priority (SP) 1b - Weighted Strict Priority (WSP)
1 DTXSTS	Drop Transmit Status - 1: Tx packet status received from the MAC is dropped in the MTL - 0: Tx packet status received from the MAC is forwarded to the application 0b - Drop Transmit Status is disabled 1b - Drop Transmit Status is enabled
0 —	Reserved

76.17.258 MTL Debus Control (MTL_DBG_CTL)

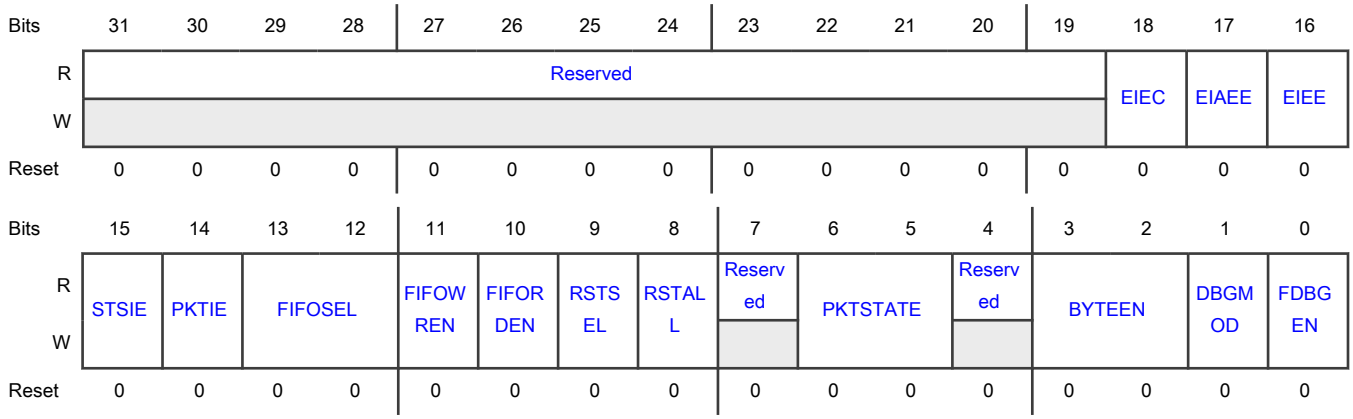
Offset

Register	Offset
MTL_DBG_CTL	C08h

Function

The FIFO Debug Access Control and Status register controls the operation mode of FIFO debug access.

Diagram



Fields

Field	Function
31-19 —	Reserved
18 EIEC	ECC Inject Error Control for Tx, Rx, TSO and DCACHE memories When EIEE or EIAEE bit of this register is set, following are the errors inserted based on the value encoded in this field. 0b - Insert 1 bit error 1b - insert 2 bit errors
17 EIAEE	ECC Inject Address Error for Tx, Rx, TSO and DCACHE memories - 1: Enables the ECC address error injection feature - 0: Disables the ECC address error injection feature 0b - Disables the ECC address error injection 1b - Enables the ECC address error injection
16 EIEE	ECC Inject Error Enable for Tx, Rx, TSO and DCACHE memories - 1: Enables the ECC error injection feature - 0: Disables the ECC error injection feature 0b - ECC Inject Error for Tx, Rx, TSO and DCACHE memories is disabled 1b - ECC Inject Error for Tx, Rx, TSO and DCACHE memories is enabled
15 STSIE	Transmit Status Available Interrupt Status Enable When this bit is set, an interrupt is generated when Transmit status is available in slave mode. 0b - Transmit Packet Available Interrupt Status is disabled 1b - Transmit Packet Available Interrupt Status is enabled
14 PKTIE	Receive Packet Available Interrupt Status Enable When this bit is set, an interrupt is generated when EOP of received packet is written to the Rx FIFO.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Receive Packet Available Interrupt Status is disabled 1b - Receive Packet Available Interrupt Status is enabled
13-12 FIFOSEL	FIFO Selected for Access This field indicates the FIFO selected for debug access. 00b - Tx FIFO 01b - Tx Status FIFO (only read access when SLVMOD is set)/ DC Memory FIFO read-write access when DBGMOD is set 10b - TSO FIFO (cannot be accessed when SLVMOD is set) 11b - Rx FIFO
11 FIFOWREN	FIFO Write Enable When this bit is set, it enables the Write operation on selected FIFO when FIFO Debug Access is enabled. This bit must not be written to 1 when FIFO Debug Access is not enabled, that is FDBGEN bit is 0. Access restriction applies. Self-cleared. Setting 0 clears. Setting 1 sets. **NOTE** This field is not testable by an automated register test. 0b - FIFO Write is disabled 1b - FIFO Write is enabled
10 FIFORDEN	FIFO Read Enable When this bit is set, it enables the Read operation on selected FIFO when FIFO Debug Access is enabled. This bit must not be written to 1 when FIFO Debug Access is not enabled, that is FDBGEN bit is 0. Access restriction applies. Self-cleared. Setting 0 clears. Setting 1 sets. **NOTE** This field is not testable by an automated register test. 0b - FIFO Read is disabled 1b - FIFO Read is enabled
9 RSTSEL	Reset Pointers of Selected FIFO When this bit is set, the pointers of the currently-selected FIFO are reset when FIFO Debug Access is enabled. This bit must not be written to 1 when FIFO Debug Access is not enabled, that is FDBGEN bit is 0. Access restriction applies. Self-cleared. Setting 0 clears. Setting 1 sets. **NOTE** This field is not testable by an automated register test. 0b - Reset Pointers of Selected FIFO is disabled 1b - Reset Pointers of Selected FIFO is enabled
8	Reset All Pointers

Table continues on the next page...

Table continued from the previous page...

Field	Function
RSTALL	When this bit is set, the pointers of all FIFOs are reset when FIFO Debug Access is enabled. This bit must not be written to 1 when FIFO Debug Access is not enabled, that is FDBGEN bit is 0. Access restriction applies. Self-cleared. Setting 0 clears. Setting 1 sets. **NOTE** This field is not testable by an automated register test. 0b - Reset All Pointers is disabled 1b - Reset All Pointers is enabled
7 —	Reserved
6-5 PKTSTATE	Encoded Packet State This field is used to write the control information to the Tx FIFO or Rx FIFO. - Tx FIFO: -- 00: Packet Data -- 01: Control Word -- 10: SOP Data -- 11: EOP Data - Rx FIFO: -- 00: Packet Data -- 01: Normal Status -- 10: Last Status -- 11: EOP 00b - Packet Data 01b - Control Word/Normal Status 10b - SOP Data/Last Status 11b - EOP Data/EOP
4 —	Reserved
3-2 BYTEEN	Byte Enables This field indicates the number of data bytes valid in the data register during Write operation. This is valid only when PKTSTATE is 2'b10 (EOP) and Tx FIFO or Rx FIFO is selected. 00b - Byte 0 valid 01b - Byte 0 and Byte 1 are valid 10b - Byte 0, Byte 1, and Byte 2 are valid 11b - All four bytes are valid
1 DBGMOD	Debug Mode Access to FIFO - 1: Indicates that the current access to the FIFO is read, write, and debug access. In this mode, the following access types are allowed: -- Read and Write access to Tx FIFO, TSO FIFO, DCACHE FIFO and Rx FIFO -- Read access is allowed to Tx Status FIFO. - 0: Indicates that the current access to the FIFO is slave access bypassing the DMA. In this mode, the following access are allowed: -- Write access to the Tx FIFO -- Read access to the Rx FIFO and Tx Status FIFO 0b - Debug Mode Access to FIFO is disabled 1b - Debug Mode Access to FIFO is enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 FDBGEN	FIFO Debug Access Enable - 1: Indicates that the debug mode access to the FIFO is enabled. - 0: Indicates that the FIFO can be accessed only through a master interface. 0b - FIFO Debug Access is disabled 1b - FIFO Debug Access is enabled

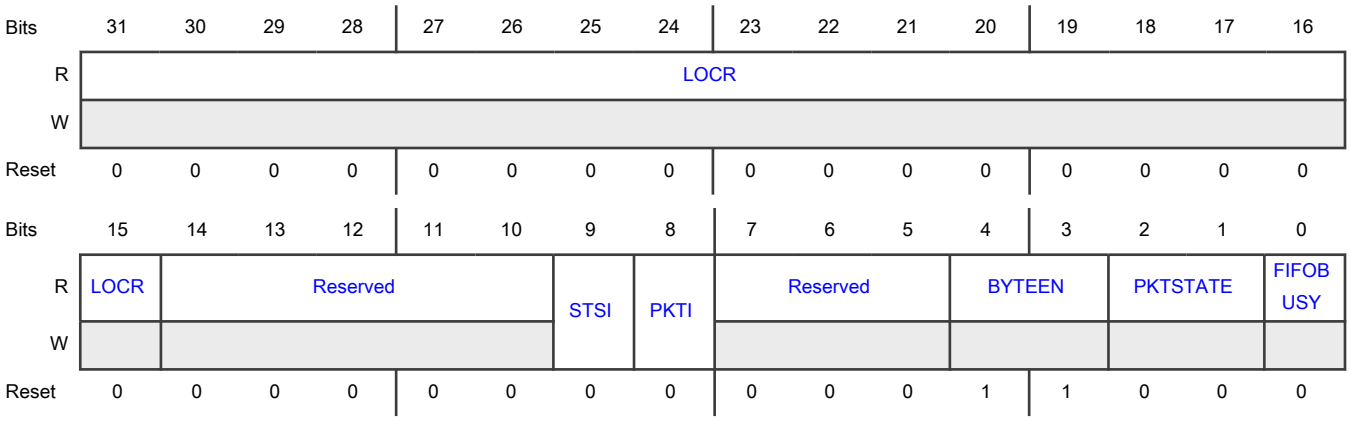
76.17.259 MTL Debus Status (MTL_DBG_STS)

Offset

Register	Offset
MTL_DBG_STS	C0Ch

Function
The FIFO Debug Status register contains the status of FIFO debug access.

Diagram



Fields

Field	Function
31-15 LOCR	Remaining Locations in the FIFO - Slave Access Mode: This field indicates the space available in the selected FIFO. - Debug Access Mode: This field contains the Write or Read pointer value of the selected FIFO during Write or Read operation, respectively. - Reset: In single Tx Queue configurations, (DWC_EQOS_TXFIFO_SIZE/(DWC_EQOS_DATAWIDTH/8)), Otherwise 0000H
14-10	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
9 STSI	Transmit Status Available Interrupt Status When set, this bit indicates that the Slave mode Tx packet is transmitted, and the status is available in Tx Status FIFO. This bit is reset when 1 is written to this bit. **NOTE** This field is not testable by an automated register test. 0b - Transmit Status Available Interrupt Status not detected 1b - Transmit Status Available Interrupt Status detected
8 PKTI	Receive Packet Available Interrupt Status When set, this bit indicates that MAC layer has written the EOP of received packet to the Rx FIFO. This bit is reset when 1 is written to this bit. **NOTE** This field is not testable by an automated register test. 0b - Receive Packet Available Interrupt Status not detected 1b - Receive Packet Available Interrupt Status detected
7-5 —	Reserved
4-3 BYTEEN	Byte Enables This field indicates the number of data bytes valid in the data register during Read operation. This is valid only when PKTSTATE is 2'b10 (EOP) and Tx FIFO or Rx FIFO is selected. 00b - Byte 0 valid 01b - Byte 0 and Byte 1 are valid 10b - Byte 0, Byte 1, and Byte 2 are valid 11b - All four bytes are valid
2-1 PKTSTATE	Encoded Packet State This field is used to get the control or status information of the selected FIFO. - Tx FIFO: -- 00: Packet Data -- 01: Control Word -- 10: SOP Data -- 11: EOP Data - Rx FIFO: -- 00: Packet Data -- 01: Normal Status -- 10: Last Status -- 11: EOP This field is applicable only for Tx FIFO and Rx FIFO during Read operation. 00b - Packet Data 01b - Control Word/Normal Status 10b - SOP Data/Last Status 11b - EOP Data/EOP

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 FIFOBUSY	FIFO Busy When set, this bit indicates that a FIFO operation is in progress in the MAC and content of the following fields is not valid: - All other fields of this register - All fields of the MTL_FIFO_Debug_Data register 0b - FIFO Busy not detected 1b - FIFO Busy detected

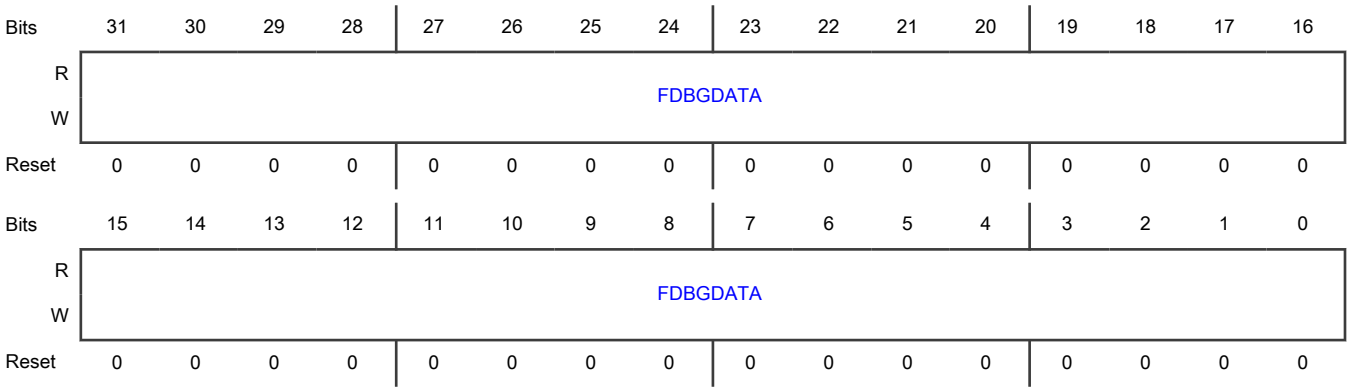
76.17.260 MTL FIFO Debug Data (MTL_FIFO_Debug_Data)

Offset

Register	Offset
MTL_FIFO_Debug_Data	C10h

Function
The FIFO Debug Data register contains the data to be written to or read from the FIFOs.

Diagram



Fields

Field	Function
31-0 FDBGDATA	FIFO Debug Data During debug or slave access write operation, this field contains the data to be written to the Tx FIFO, Rx FIFO, or TSO FIFO. During debug or slave access read operation, this field contains the data read from the Tx FIFO, Rx FIFO, TSO FIFO, or Tx Status FIFO. **NOTE** This field is not testable by an automated register test.

76.17.261 MTL Interrupt Status (MTL_Interrupt_Status)

Offset

Register	Offset
MTL_Interrupt_Status	C20h

Function

The software driver (application) reads this register during interrupt service routine or polling to determine the interrupt status of MTL queues and the MAC.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								MTLPIS	Reserved				ESTIS	DBGIS	Reserved
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								Reserved	Reserved	Reserved	Reserved	Reserved	Q2IS	Q1IS	Q0IS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23 MTLPIS	MTL Rx Parser Interrupt Status This bit indicates that there is an interrupt from Rx Parser Block. To reset this bit, the application must read the MTL_ Rxp_Interrupt_Status register to get the exact cause of the interrupt and clear its source. 0b - MTL Rx Parser Interrupt status not detected 1b - MTL Rx Parser Interrupt status detected
22-19 —	Reserved
18 ESTIS	EST (TAS- 802.1Qbv) Interrupt Status This bit indicates an interrupt event during the operation of 802.1Qbv. To reset this bit, the application must clear the error/event that has caused the Interrupt.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - EST (TAS- 802.1Qbv) Interrupt status not detected 1b - EST (TAS- 802.1Qbv) Interrupt status detected
17 DBGIS	Debug Interrupt status This bit indicates an interrupt event during the slave access. To reset this bit, the application must read the FIFO Debug Access Status register to get the exact cause of the interrupt and clear its source. 0b - Debug Interrupt status not detected 1b - Debug Interrupt status detected
16 —	Reserved
15-8 —	Reserved
7 —	Reserved
6 —	Reserved
5 —	Reserved
4 —	Reserved
3 —	Reserved
2 Q2IS	Queue 2 Interrupt status This bit indicates that there is an interrupt from Queue 2. To reset this bit, the application must read the MTL_Q2_Interrupt_Control_Status register to get the exact cause of the interrupt and clear its source. 0b - Queue 2 Interrupt status not detected 1b - Queue 2 Interrupt status detected
1 Q1IS	Queue 1 Interrupt status This bit indicates that there is an interrupt from Queue 1. To reset this bit, the application must read the MTL_Q1_Interrupt_Control_Status register to get the exact cause of the interrupt and clear its source. 0b - Queue 1 Interrupt status not detected 1b - Queue 1 Interrupt status detected

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 Q0IS	Queue 0 Interrupt status This bit indicates that there is an interrupt from Queue 0. To reset this bit, the application must read Queue 0 Interrupt Control and Status register to get the exact cause of the interrupt and clear its source. 0b - Queue 0 Interrupt status not detected 1b - Queue 0 Interrupt status detected

76.17.262 MTL Receive Queue DMA Map 0 (MTL_RxQ_DMA_Map0)

Offset

Register	Offset
MTL_RxQ_DMA_Map0	C30h

Function

The Receive Queue and DMA Channel Mapping 0 register is reserved in EQOS-CORE and EQOS-MTL configurations.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved			Reserved	Reserved		Reserved		Reserved			Q2DDMACH	Reserved		Q2MDMACH	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved			Q1DDMACH	Reserved		Q1MDMACH		Reserved			Q0DDMACH	Reserved		Q0MDMACH	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	Reserved
28 —	Reserved
27-26	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
25-24 —	Reserved
23-21 —	Reserved
20 Q2DDMACH	Queue 2 Enabled for DA-based DMA Channel Selection When set, this bit indicates that the packets received in Queue 2 are routed to a particular DMA channel as decided in the MAC Receiver based on the DMA channel number programmed in the L3-L4 filter registers, or the Ethernet DA address. When reset, this bit indicates that the packets received in Queue 2 are routed to the DMA Channel programmed in the Q2MDMACH field (Bits[18:16]). 0b - Queue 2 disabled for DA-based DMA Channel Selection 1b - Queue 2 enabled for DA-based DMA Channel Selection
19-18 —	Reserved
17-16 Q2MDMACH	Queue 2 Mapped to DMA Channel This field controls the routing of the received packet in Queue 2 to the DMA channel: - 000: DMA Channel 0 - 001: DMA Channel 1 - 010: DMA Channel 2 - 011: DMA Channel 3 - 100: DMA Channel 4 - 101: DMA Channel 5 - 110: DMA Channel 6 - 111: DMA Channel 7 This field is valid when the Q2DDMACH field is reset.
15-13 —	Reserved
12 Q1DDMACH	Queue 1 Enabled for DA-based DMA Channel Selection When set, this bit indicates that the packets received in Queue 1 are routed to a particular DMA channel as decided in the MAC Receiver based on the DMA channel number programmed in the L3-L4 filter registers, or the Ethernet DA address. When reset, this bit indicates that the packets received in Queue 1 are routed to the DMA Channel programmed in the Q1MDMACH field (Bits[10:8]). 0b - Queue 1 disabled for DA-based DMA Channel Selection 1b - Queue 1 enabled for DA-based DMA Channel Selection
11-10 —	Reserved
9-8 Q1MDMACH	Queue 1 Mapped to DMA Channel This field controls the routing of the received packet in Queue 1 to the DMA channel: - 000: DMA Channel 0 - 001: DMA Channel 1 - 010: DMA Channel 2 - 011: DMA Channel 3 - 100: DMA Channel 4 - 101: DMA Channel 5 - 110: DMA Channel 6 - 111: DMA Channel 7 This field is valid when the

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Q1DDMACH field is reset. The width of this field depends on the number of RX DMA channels and not all the values might be valid in some configurations. For example, if the number of RX DMA channels selected is 2, only 000 and 001 are valid, the other bits are reserved.
7-5 —	Reserved
4 Q0DDMACH	Queue 0 Enabled for DA-based DMA Channel Selection When set, this bit indicates that the packets received in Queue 0 are routed to a particular DMA channel as decided in the MAC Receiver based on the DMA channel number programmed in the L3-L4 filter registers, or the Ethernet DA address. When reset, this bit indicates that the packets received in Queue 0 are routed to the DMA Channel programmed in the Q0MDMACH field. 0b - Queue 0 disabled for DA-based DMA Channel Selection 1b - Queue 0 enabled for DA-based DMA Channel Selection
3-2 —	Reserved
1-0 Q0MDMACH	Queue 0 Mapped to DMA Channel This field controls the routing of the packet received in Queue 0 to the DMA channel: - 000: DMA Channel 0 - 001: DMA Channel 1 - 010: DMA Channel 2 - 011: DMA Channel 3 - 100: DMA Channel 4 - 101: DMA Channel 5 - 110: DMA Channel 6 - 111: DMA Channel 7 This field is valid when the Q0DDMACH field is reset. The width of this field depends on the number of RX DMA channels and not all the values might be valid in some configurations. For example, if the number of RX DMA channels selected is 2, only 000 and 001 are valid, the other bits are reserved.

76.17.263 MTL TBS Control (MTL_TBS_CTRL)

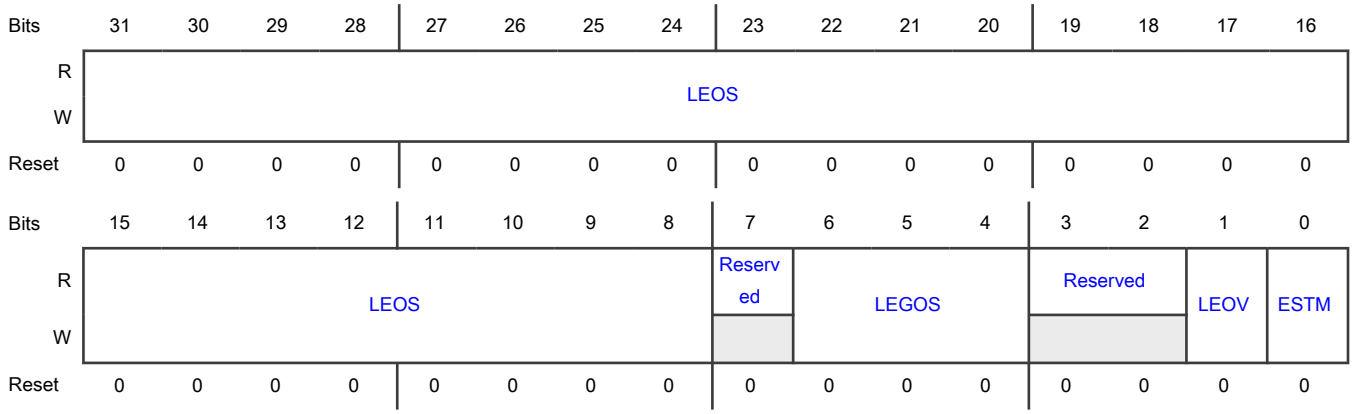
Offset

Register	Offset
MTL_TBS_CTRL	C40h

Function

This register controls the operation of Time Based Scheduling.

Diagram



Fields

Field	Function
31-8 LEOS	Launch Expiry Offset The value in units of 256 nanoseconds that has to be added to the Launch time to compute the Launch Expiry time. Value valid only when LEOV is set. Max value: 999,999,999 ns, additionally should be smaller than CTR-1 value when ESTM mode is set since this value is a modulo CTR value.
7 —	Reserved
6-4 LEGOS	Launch Expiry GSN Offset The number GSN slots that has to be added to the Launch GSN to compute the Launch Expiry time. Value valid only when LEOV is set.
3-2 —	Reserved
1 LEOV	Launch Expiry Offset Valid When set indicates the LEOS field is valid. When not set, indicates the Launch Expiry Offset is not valid and the MTL must not check for Launch expiry time. 0b - LEOS field is invalid 1b - LEOS field is valid
0 ESTM	EST offset Mode When this bit is set, the Launch Time value used in Time Based Scheduling is interpreted as an EST offset value and is added to the Base Time Register (BTR) of the current list. When reset, the Launch Time value is used as an absolute value that should be compared with the System time [39:8]. 0b - EST offset Mode is disabled 1b - EST offset Mode is enabled

76.17.264 MTL EST Control (MTL_EST_Control)

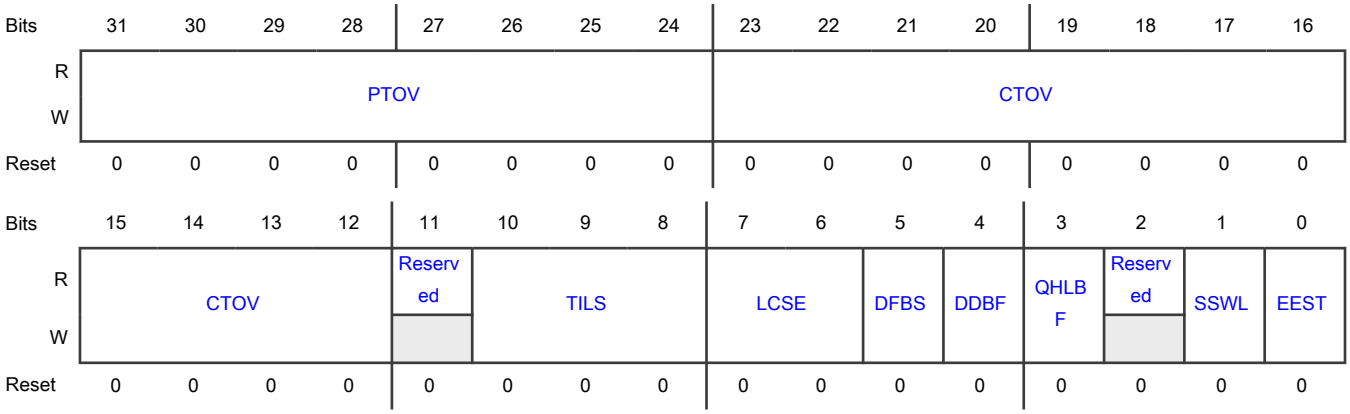
Offset

Register	Offset
MTL_EST_Control	C50h

Function

This register controls the operation of Enhancements to Scheduled Transmission (IEEE802.1Qbv).

Diagram



Fields

Field	Function
31-24 PTOV	PTP Time Offset Value The value of PTP Clock period multiplied by 6 in nanoseconds. This value is needed to avoid transmission overruns at the beginning of the installation of a new GCL.
23-12 CTOV	Current Time Offset Value Provides a 12 bit time offset value in nano second that is added to the current time to compensate for all the implementation pipeline delays such as the CDC sync delay, buffering delays, data path delays and so on. This offset helps to ensure that the impact of gate controls is visible on the line exactly at the pre-determined schedule (or as close to the schedule as possible).
11 —	Reserved
10-8 TILS	Time Interval Left Shift Amount This field provides the left shift amount for the programmed Time Interval values used in the Gate Control Lists. - 000: No left shift needed (equal to x1ns) - 001: Left shift TI by 1 bit (equal to x2ns) - 010: Left shift TI by 2 bits (equal to x4ns) - . - . - 100: Left shift TI by 7 bits (equal to x128ns) Based on the configuration one or more bits of this field should be treated as Reserved/Read-Only.
7-6	Loop Count to report Scheduling Error

Table continues on the next page...

Table continued from the previous page...

Field	Function
LCSE	Programmable number of GCL list iterations before reporting an HLBS error defined in EST_Status register. 00b - 4 iterations 01b - 8 iterations 10b - 16 iterations 11b - 32 iterations
5 DFBS	Drop Frames causing Scheduling Error When set frames reported to cause HOL Blocking due to not getting scheduled (HLBS field of EST_Status register) after 4,8,16,32 (based on LCSE field of this register) GCL iterations are dropped. 0b - Do not Drop Frames causing Scheduling Error 1b - Drop Frames causing Scheduling Error
4 DDBF	Do not Drop frames during Frame Size Error When set, frames are not be dropped during Head-of-Line blocking due to Frame Size Error (HLBF field of EST_Status register). 0b - Drop frames during Frame Size Error 1b - Do not Drop frames during Frame Size Error
3 QHLBF	Quick Assertion of HLBF Error When set, Time Window for Head-of-Line blocking due to Frame Size Error is 1 to 2 loop count of GCL list.On reset,Time Window for Head-of-Line blocking due to Frame Size Error is 2 to 3 loop counts of GCL list. 0b - Disable Quick assertion of HLBF error 1b - Quick Assertion of HLBF Error
2 —	Reserved
1 SSWL	Switch to S/W owned list When set indicates that the software has programmed that list that it currently owns (SWOL) and the hardware should switch to the new list based on the new BTR. Hardware clears this bit when the switch to the SWOL happens to indicate the completion of the switch or when an BTR error (BTRE in Status register) is set. When BTRE is set this bit is cleared but SWOL is not updated as the switch was not successful. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. 0b - Switch to S/W owned list is disabled 1b - Switch to S/W owned list is enabled
0 EEST	Enable EST When reset, the gate control list processing is halted and all gates are assumed to be in Open state. Should be set for the hardware to start processing the gate control lists. During the toggle from 0 to 1,

Table continues on the next page...

Table continued from the previous page...

Field	Function
	the gate control list processing starts only after the SSWL bit it set. When DWC_EQOS_ASP_ECC is selected during the configuration, if any uncorrectable error is detected in the EST memory the hardware resets this bit and disables the EST function. 0b - EST is disabled 1b - EST is enabled

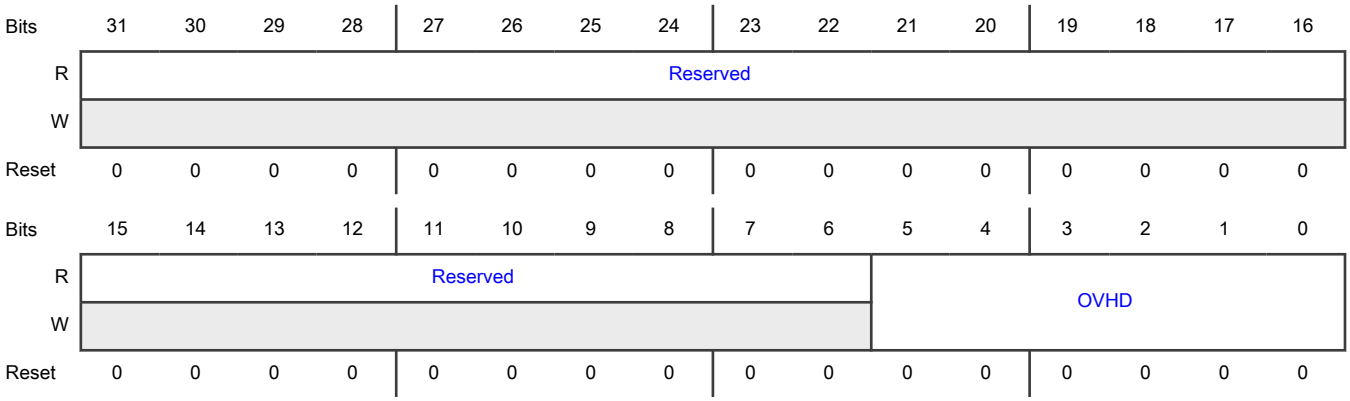
76.17.265 MTL EST Extended Control (MTL_EST_Ext_Control)

Offset

Register	Offset
MTL_EST_Ext_Control	C54h

Function
This register indicates the number of Overhead bytes for EST related scheduling.

Diagram



Fields

Field	Function
31-6 —	Reserved
5-0 OVHD	Overhead Bytes Value This field indicates the fixed overhead for every packet to account for EST Scheduler Delay, IPG or EIPG, and Preamble bytes.

76.17.266 MTL EST Status (MTL_EST_Status)

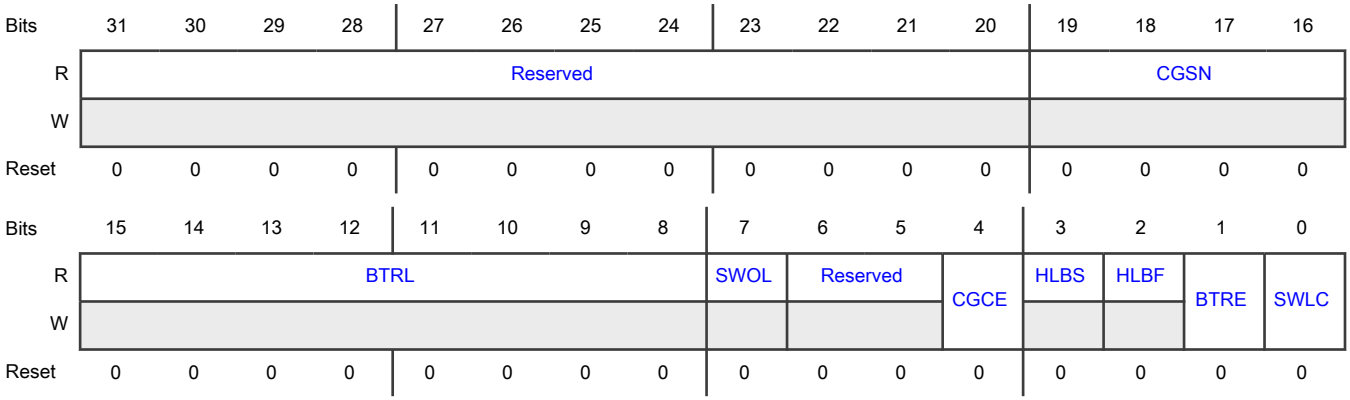
Offset

Register	Offset
MTL_EST_Status	C58h

Function

This register provides Status related to Enhancements to Scheduled Transmission (IEEE802.1Qbv).

Diagram



Fields

Field	Function
31-20 —	Reserved
19-16 CGSN	Current GCL Slot Number Indicates the slot number of the GCL list. Slot number is a modulo 16 count of the GCL List loops executed so far. Even if a new GCL list is installed, the count is incremental.
15-8 BTRL	BTR Error Loop Count Provides the minimum count (N) for which the equation Current Time =< New BTR + (N * New Cycle Time) becomes true. N = "11111111" indicates the iterations exceeded the value of 128 and the hardware was not able to update New BTR to be equal to or greater than Current Time. Software intervention is needed to update the New BTR. Value cleared when BTRE field of this register is cleared.
7 SWOL	S/W owned list When '0' indicates Gate control list number "0" is owned by software and when "1" indicates the Gate Control list "1" is owned by the software. Any reads/writes by the software (using indirect access through GCL_Control) is directed to the list indicated by this value by default. The inverse of this value is treated as HWOL. R/W operations performed by hardware are directed to the list pointed by HWOL by default. 0b - Gate control list number "0" is owned by software

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Gate control list number "1" is owned by software
6-5 —	Reserved
4 CGCE	Constant Gate Control Error This error occurs when the list length (LLR) is 1 and the Cycle Time (CTR) is less than or equal to the programmed Time Interval (TI) value after the optional Left Shifting. This implies, Gates are either always Closed or always Open based on the Gate Control values; the same effect can be achieved by other simpler (non TSN) programming mechanisms. Since the implementation does not support such a programming an error is reported. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. 0b - Constant Gate Control Error not detected 1b - Constant Gate Control Error detected
3 HLBS	Head-Of-Line Blocking due to Scheduling Set when the frame is not able to win arbitration and get scheduled even after 4 iterations of the GCL. Indicates to software a potential programming error. The one hot encoded values of the Queue Numbers that are not able to make progress are indicated in the MTL_EST_Sch_Error register. Bit cleared when MTL_EST_Sch_Error register is all zeros. 0b - Head-Of-Line Blocking due to Scheduling not detected 1b - Head-Of-Line Blocking due to Scheduling detected
2 HLBF	Head-Of-Line Blocking due to Frame Size Set when HOL Blocking is noticed on one or more Queues as a result of none of the Time Intervals of gate open in the GCL being greater than or equal to the duration needed for frame size (or frame fragment size when preemption is enabled) transmission. The one hot encoded Queue numbers that are experiencing HLBF are indicated in the MTL_EST_Frm_Size_Error register. Additionally, the first Queue number that experienced HLBF along with the frame size is captured in MTL_EST_Frm_Size_Capture register. Bit cleared when MTL_EST_Frame_Size_ Error register is all zeros. 0b - Head-Of-Line Blocking due to Frame Size not detected 1b - Head-Of-Line Blocking due to Frame Size detected
1 BTRE	BTR Error When "1" indicates a programming error in the BTR of SWOL where the programmed value is less than current time. If the BTRL = "11111111", SWOL is not updated and software should reprogram the BTR to a value greater than current time and then set SSWL to reinitiate the switch to SWOL. Else if the value of BTRL < "11111111", SWOL is updated and this field indicates the number of iterations (of + CycleTime) taken by hardware to update the BTR to a value greater than Current Time. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. 0b - BTR Error not detected 1b - BTR Error detected

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 SWLC	Switch to S/W owned list Complete When "1" indicates the hardware has successfully switched to the SWOL, and the SWOL bit has been updated to that effect. Cleared when the SSWL of EST_Control register transitions from 0 to 1, or on a software write. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. 0b - Switch to S/W owned list Complete not detected 1b - Switch to S/W owned list Complete detected

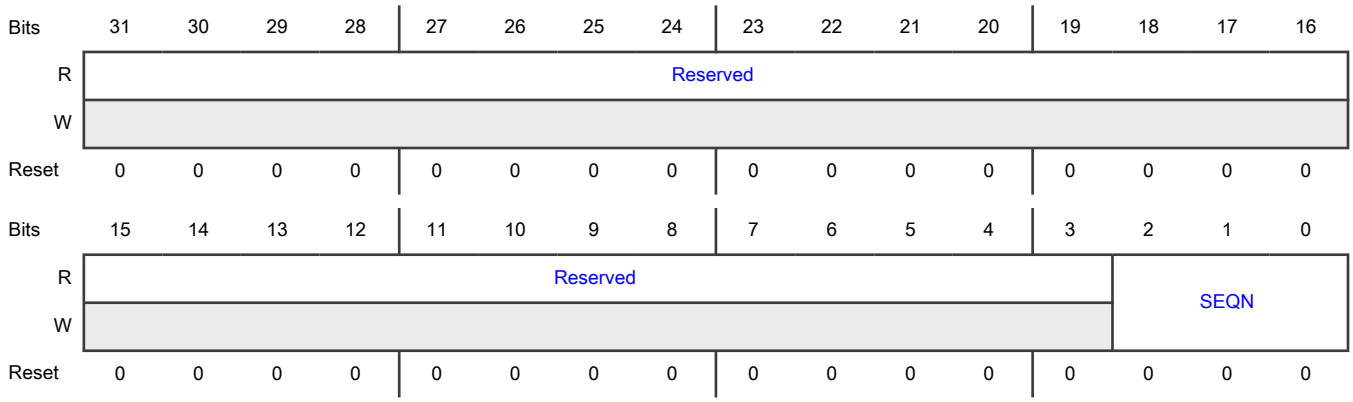
76.17.267 MTL EST Scheduling Error (MTL_EST_Sch_Error)

Offset

Register	Offset
MTL_EST_Sch_Error	C60h

Function
This register provides the One Hot encoded Queue Numbers that are having the Scheduling related error (timeout).

Diagram



Fields

Field	Function
31-3 —	Reserved
2-0 SEQN	Schedule Error Queue Number The One Hot Encoded Queue Numbers that have experienced error/timeout described in HLBS field of status register. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect.

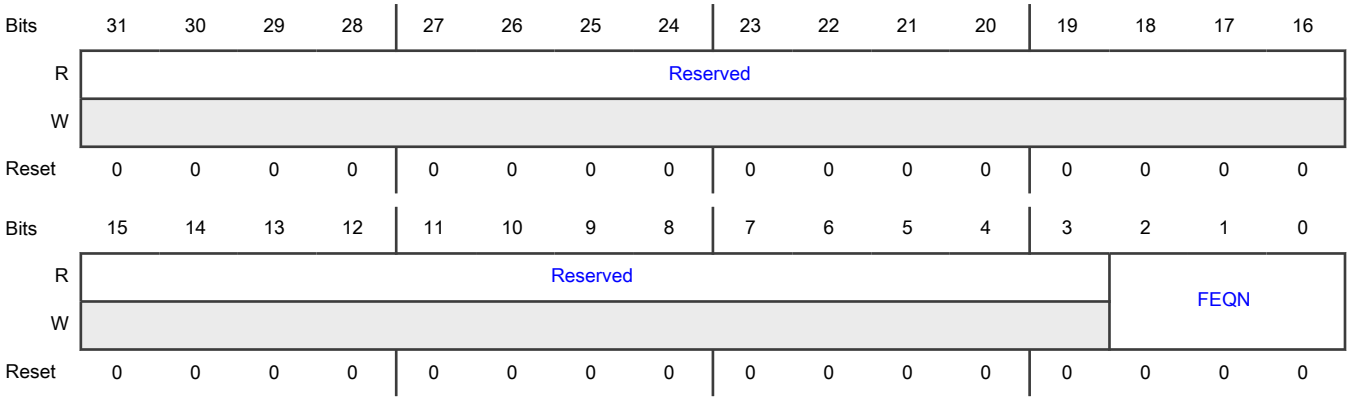
76.17.268 MTL EST Frame Size Error (MTL_EST_Frm_Size_Error)

Offset

Register	Offset
MTL_EST_Frm_Size_Error	C64h

Function
This register provides the One Hot encoded Queue Numbers that are having the Frame Size related error.

Diagram



Fields

Field	Function
31-3 —	Reserved
2-0 FEQN	Frame Size Error Queue Number The One Hot Encoded Queue Numbers that have experienced error described in HLBF field of status register. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect.

76.17.269 MTL EST Frame Size Capture (MTL_EST_Frm_Size_Capture)

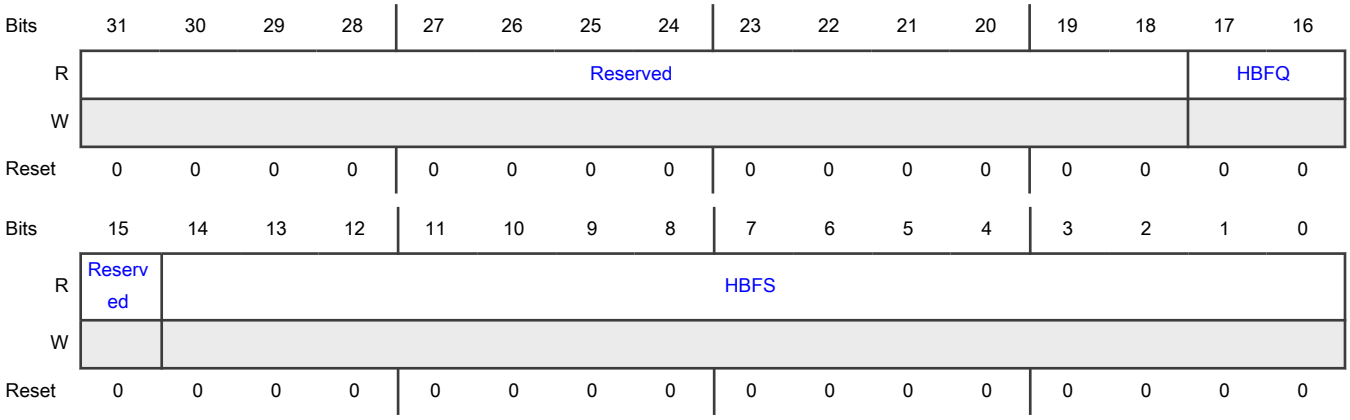
Offset

Register	Offset
MTL_EST_Frm_Size_Capture	C68h

Function

This register captures the Frame Size and Queue Number of the first occurrence of the Frame Size related error. Up on clearing it captures the data of immediate next occurrence of a similar error.

Diagram



Fields

Field	Function
31-18 —	Reserved
17-16 HBFQ	Queue Number of HLBF Captures the binary value of the of the first Queue (number) experiencing HLBF error (see HLBF field of status register). Value once written is not altered by any subsequent queue errors of similar nature. Once cleared the queue number of the next occurring HLBF error is captured. Width is based on the number of Tx Queues configured; remaining bits are Read-Only. Cleared when MTL_EST_Frm_Size_Error register is all zeros.
15 —	Reserved
14-0 HBFS	Frame Size of HLBF Captures the Frame Size of the dropped frame related to queue number indicated in HBFQ field of this register. Contents of this register should be considered invalid, if this field is zero. Cleared when MTL_EST_Frm_Size_Error register is all zeros.

76.17.270 MTL EST Interrupt Enable (MTL_EST_Intr_Enable)

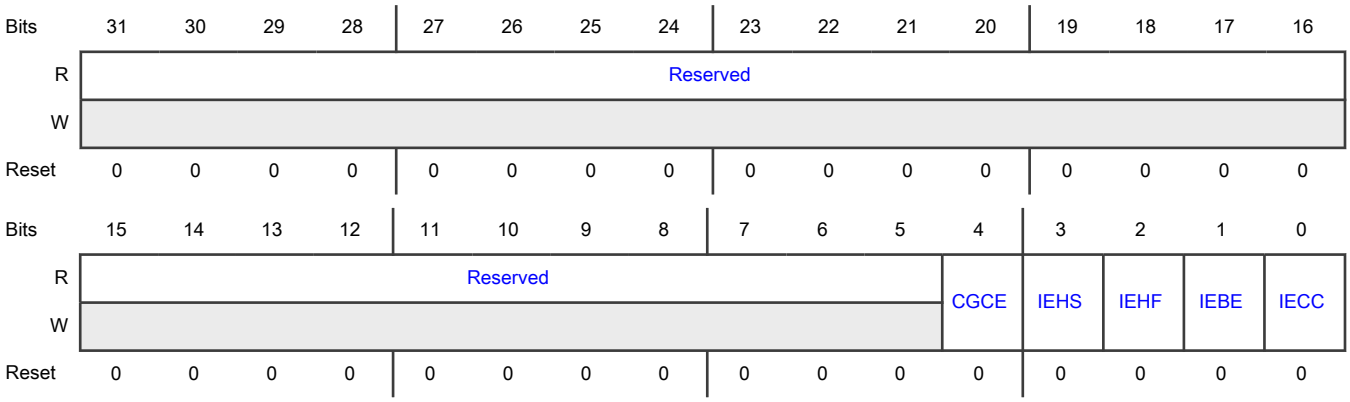
Offset

Register	Offset
MTL_EST_Intr_Enable	C70h

Function

This register implements the Interrupt Enable bits for the various events that generate an interrupt. Bit positions have a 1 to 1 correlation with the status bit positions in MTL_ETS_Status register.

Diagram



Fields

Field	Function
31-5 —	Reserved
4 CGCE	Interrupt Enable for CGCE When set, generates interrupt when the Constant Gate Control Error occurs and is indicated in the status. When reset this event does not generate an interrupt 0b - Interrupt for CGCE is disabled 1b - Interrupt for CGCE is enabled
3 IEHS	Interrupt Enable for HLBS When set, generates interrupt when the Head-of-Line Blocking due to Scheduling issue and is indicated in the status. When reset this event does not generate an interrupt. 0b - Interrupt for HLBS is disabled 1b - Interrupt for HLBS is enabled
2 IEHF	Interrupt Enable for HLBF When set, generates interrupt when the Head-of-Line Blocking due to Frame Size error occurs and is indicated in the status. When reset this event does not generate an interrupt. 0b - Interrupt for HLBF is disabled 1b - Interrupt for HLBF is enabled
1 IEBE	Interrupt Enable for BTR Error When set, generates interrupt when the BTR Error occurs and is indicated in the status. When reset this event does not generate an interrupt.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Interrupt for BTR Error is disabled 1b - Interrupt for BTR Error is enabled
0 IECC	Interrupt Enable for Switch List When set, generates interrupt when the configuration change is successful and the hardware has switched to the new list. When reset this event does not generate an interrupt. 0b - Interrupt for Switch List is disabled 1b - Interrupt for Switch List is enabled

76.17.271 MTL EST GCL Control (MTL_EST_GCL_Control)

Offset

Register	Offset
MTL_EST_GCL_Control	C80h

Function

This register provides the control information for reading/writing to the Gate Control lists.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								ESTEIEC	ESTEIAEE	ESTEIEE	Reserved				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ADDR								Reserved		DBGB	DBGM	Reserved	GCRR	R1W0	SRWO
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23	ECC Inject Error Control for EST Memory

Table continues on the next page...

Table continued from the previous page...

Field	Function
ESTEIEC	When ESTEIEE or ESTEIAEE bit of this register is set, following are the errors inserted based on the value encoded in this field. This field is valid only if DWC_EQOS_ASP_ECC feature is selected during the configuration, else it is reserved. 0b - Insert 1 bit error 1b - Insert 2 bit errors
22 ESTEIAEE	EST ECC Inject Address Error Enable When set along with EEST bit of MTL_EST_Control register, enables the ECC address error injection feature. When reset, disables the ECC address error injection feature. 0b - EST ECC Inject Address Error is disabled 1b - EST ECC Inject Address Error is enabled
21 ESTEIEE	EST ECC Inject Error Enable When set along with EEST bit of MTL_EST_Control register, enables the ECC error injection feature. When reset, disables the ECC error injection feature. 0b - EST ECC Inject Error is disabled 1b - EST ECC Inject Error is enabled
20-16 —	Reserved
15-8 ADDR	Gate Control List Address: (GCLA when GCRR is "0"). Provides the address (row number) of the Gate Control List at which the R/W operation has to be performed. By default the Gate Control List pointed by SWOL of MTL_EST_Status is selected for R/W, however if the DBGM bit of this register is set, a debug mode access is given to R/W from DBGB. The width of this field is dependent on DWC_EQOS_EST_DEP; unused bits should be treated as read only. Gate Control List Related Registers Address: (GCRA when GCRR is "1"). By default the GCL related register set pointed by SWOL of MTL_EST_Status is selected for R/W, however if the DBGM bit of this register is set, a debug mode access is given to R/W from DBGB. Lower 3 bits are only used in this mode, higher order bits are treated as dont cares. - 000: BTR Low (31:0) - 001: BTR High (63:31) - 010: CTR Low (31:0) - 011: CTR High (39:32) - 100: TER (31:0) - 101: LLR (n:0) (where n is (log{DWC_EQOS_EST_DEP} / log2)) - Others: Reserved
7-6 —	Reserved
5 DBGB	Debug Mode Bank Select When set to "0" indicates R/W in debug mode should be directed to Bank 0 (GCL0 and corresponding Time related registers). When set to "1" indicates R/W in debug mode should be directed to Bank 1 (GCL1 and corresponding Time related registers). This value is used when DBGM is set and overrides by value of SWOL which is normally used. 0b - R/W in debug mode should be directed to Bank 0 1b - R/W in debug mode should be directed to Bank 1

Table continues on the next page...

Table continued from the previous page...

Field	Function
4 DBGM	Debug Mode When set to "1" indicates R/W in debug mode where the memory bank (for GCL and Time related registers) is explicitly provided by DBGB value, when set to "0" SWOL bit is used to determine which bank to use. 0b - Debug Mode is disabled 1b - Debug Mode is enabled
3 —	Reserved
2 GCRR	Gate Control Related Registers When set to "1" indicates the R/W access is for the GCL related registers (BTR, CTR, TER, LLR) whose address is provided by GCRA. When "0" indicates R/W should be directed to GCL from the address provided by GCLA. 0b - Gate Control Related Registers are disabled 1b - Gate Control Related Registers are enabled
1 R1W0	Read '1', Write '0': - 1: Read operation - 0: Write operation. 0b - Write Operation 1b - Read Operation
0 SRWO	Start Read/Write Op - 1: Indicates the start/progress of a Read/Write operation. - Reset by hardware: Indicates the R/W operation has completed or an error has occurred (when bit 20 is set) - Reads: Data can be read from MTL_EST_GCL_Data register after this bit is reset - Writes: MTL_EST_GCL_Data should be programmed with write data before setting SRWO. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. 0b - Start Read/Write Op disabled 1b - Start Read/Write Op enabled

76.17.272 MTL EST GCL Data (MTL_EST_GCL_Data)

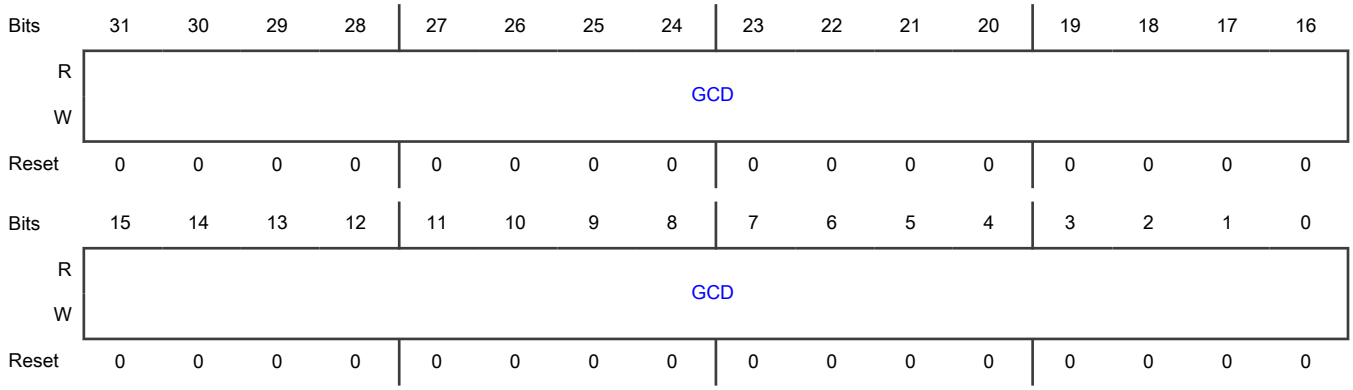
Offset

Register	Offset
MTL_EST_GCL_Data	C84h

Function

This register holds the read data or write data in case of reads and writes respectively.

Diagram



Fields

Field	Function
31-0 GCD	Gate Control Data The data corresponding to the address selected in the GCL_Control register. Used for both Read and Write operations.

76.17.273 MTL FPE Control Status (MTL_FPE_CTRL_STS)

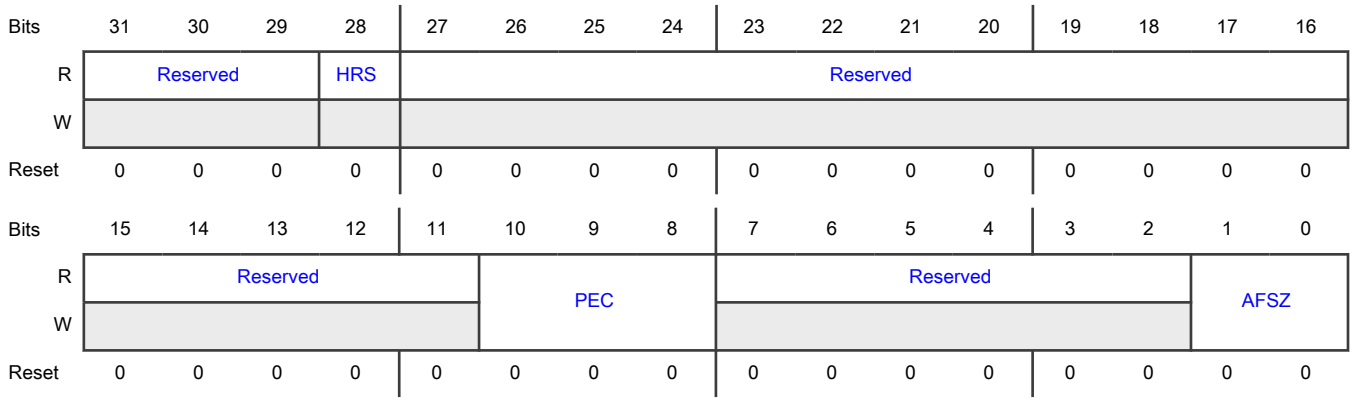
Offset

Register	Offset
MTL_FPE_CTRL_STS	C90h

Function

This register controls the operation of, and provides status for Frame Preemption (IEEE802.1Qbu/802.3br).

Diagram



Fields

Field	Function
31-29 —	Reserved
28 HRS	Hold/Release Status - 1: Indicates a Set-and-Hold-MAC operation was last executed and the pMAC is in Hold State. - 0: Indicates a Set-and-Release-MAC operation was last executed and the pMAC is in Release State. 0b - Indicates a Set-and-Release-MAC operation was last executed and the pMAC is in Release State 1b - Indicates a Set-and-Hold-MAC operation was last executed and the pMAC is in Hold State
27-16 —	Reserved
15-11 —	Reserved
10-8 PEC	Preemption Classification When set indicates the corresponding Queue must be classified as preemptable, when '0' Queue is classified as express. When both EST (Qbv) and Preemption are enabled, Queue-0 is always assumed to be preemptable. When EST (Qbv) is enabled Queues categorized as preemptable here are always assumed to be in "Open" state in the Gate Control List.
7-2 —	Reserved
1-0 AFSZ	Additional Fragment Size used to indicate, in units of 64 bytes, the minimum number of bytes over 64 bytes required in non-final fragments of preempted frames. The minimum non-final fragment size is (AFSZ +1) * 64 bytes

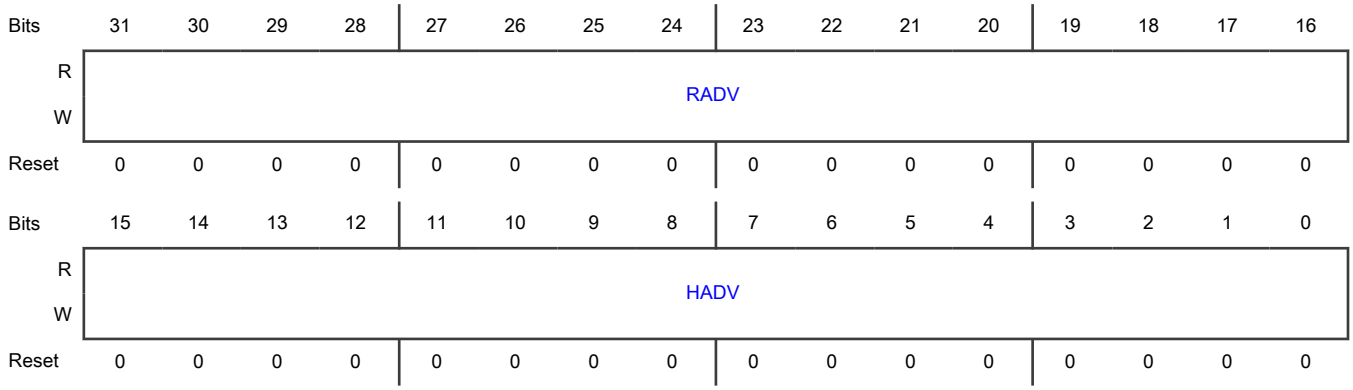
76.17.274 MTL FPE Advance (MTL_FPE_Advance)

Offset

Register	Offset
MTL_FPE_Advance	C94h

Function
This register holds the Hold and Release Advance time.

Diagram



Fields

Field	Function
31-16 RADV	Release Advance The maximum time in nanoseconds that can elapse between issuing a RELEASE to the MAC and the MAC being ready to resume transmission of preemptable frames, in the absence of there being any express frames available for transmission.
15-0 HADV	Hold Advance The maximum time in nanoseconds that can elapse between issuing a HOLD to the MAC and the MAC ceasing to transmit any preemptable frame that is in the process of transmission or any preemptable frames that are queued for transmission.

76.17.275 MTL Rx Parser Control Status (MTL_RXP_Control_Status)

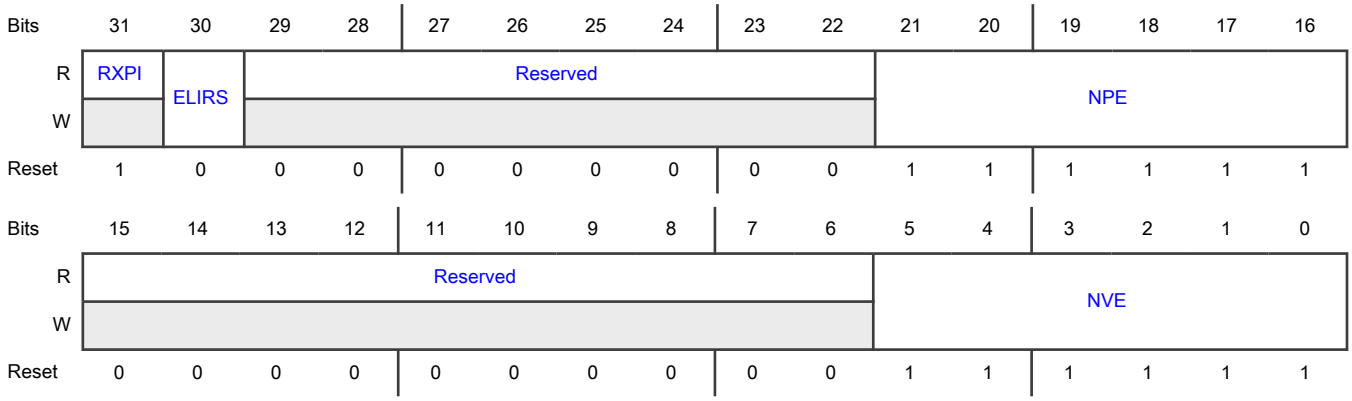
Offset

Register	Offset
MTL_RXP_Control_Status	CA0h

Function

The MTL_RXP_Control_Status register establishes the operating mode of Rx Parser and provides some status.

Diagram



Fields

Field	Function
31 RXPI	RX Parser in Idle state This status bit is set to 1 when the Rx parser is in Idle State and waiting for a new packet for processing. This bit is used as a handshake with software when parser gets disables. After disabling, when bit is set then software can update the Rx parser instruction table. 0b - RX Parser not in Idle state 1b - RX Parser in Idle state
30 ELIRS	Enable Last Instruction in RX Status When this bit is set, RDES2[31:16] indicates the index of the last instruction excecuted in Rx Parse, when set to 0 indicates MAC filter status.
29-22 —	Reserved
21-16 NPE	Number of parsable entries in the Instruction table This control indicates the number of parsable entries in the Instruction Memory. This is used in Rx parser logic to detect programming Error. In case number of parsed entries for a packet is more than this entry then NPEOVIS bit in the MTL_RXP_Interrupt_Control_Status register is set.
15-6 —	Reserved
5-0 NVE	Number of valid entry address/index in the Instruction table This control indicates the number of valid entries address/index in the Instruction Memory (i.e. when NVE field in register=31, the maximum valid entry address is NVE+1 i.e. addresses/indices=0 to 32, or 33 entries). This is used in Rx parser logic to detect any programming Error. In case while parsing Table address (memory address) found to be more than this maximum valid entry address then NVEOVIS bit in the MTL_RXP_Interrupt_Control_Status register is set. Note: The minimum value of this should be 2.

76.17.276 MTL Rx Parser Interrupt Control Status (MTL_RXP_Interrupt_Control_Status)

Offset

Register	Offset
MTL_RXP_Interrupt_Control_Status	CA4h

Function

The MTL_RXP_Interrupt_Control_Status registers provides enable control for the interrupts and provides interrupt status.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved												PDRFIE	FOOVIE	NPEOVIE	NVEOVIE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												PDRFIS	FOOVIS	NPEOVIS	NVEOVIS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-20 —	Reserved
19 PDRFIE	Packet Drop due to RF Interrupt Enable When this bit is set, the PDRFIS interrupt is enabled. When this bit is reset, the PDRFIS interrupt is disabled. 0b - Packet Drop due to RF Interrupt is disabled 1b - Packet Drop due to RF Interrupt is enabled
18 FOOVIE	Frame Offset Overflow Interrupt Enable When this bit is set, the FOOVIS interrupt is enabled. When this bit is reset, the FOOVIS interrupt is disabled. 0b - Frame Offset Overflow Interrupt is disabled 1b - Frame Offset Overflow Interrupt is enabled
17 NPEOVIE	Number of Parsable Entries Overflow Interrupt Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When this bit is set, the NPEOVIS interrupt is enabled. When this bit is reset, the NPEOVIS interrupt is disabled. 0b - Number of Parsable Entries Overflow Interrupt is disabled 1b - Number of Parsable Entries Overflow Interrupt is enabled
16 NVEOVIE	Number of Valid Entries Overflow Interrupt Enable When this bit is set, the NVEOVIS interrupt is enabled. When this bit is reset, the NVEOVIS interrupt is disabled. 0b - Number of Valid Entries Overflow Interrupt is disabled 1b - Number of Valid Entries Overflow Interrupt is enabled
15-4 —	Reserved
3 PDRFIS	Packet Dropped due to RF Interrupt Status If the Rx Parser result says to drop the packet by setting RF=1 in the instruction memory, then this bit is set to 1. This bit is cleared when the application writes 1 to this bit. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **NOTE** This field is not testable by an automated register test. 0b - Packet Dropped due to RF Interrupt Status not detected 1b - Packet Dropped due to RF Interrupt Status detected
2 FOOVIS	Frame Offset Overflow Interrupt Status While parsing if the Instruction table entry's 'Frame Offset' found to be more than EOF offset, then then this bit is set. This bit is cleared when the application writes 1 to this bit. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **NOTE** This field is not testable by an automated register test. 0b - Frame Offset Overflow Interrupt Status not detected 1b - Frame Offset Overflow Interrupt Status detected
1 NPEOVIS	Number of Parsable Entries Overflow Interrupt Status While parsing a packet if the number of parsed entries found to be more than NPE[] (Number of Parseable Entries in MTL_RXP_Control register),then this bit is set to 1. This bit is cleared when the application writes 1 to this bit. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **NOTE** This field is not testable by an automated register test.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Number of Parsable Entries Overflow Interrupt Status not detected 1b - Number of Parsable Entries Overflow Interrupt Status detected
0 NVEOVIS	Number of Valid Entry Address/Index Overflow Interrupt Status While parsing if the Instruction address found to be more than NVE (Number of Valid Entry Address/ index in MTL_RXP_Control register), then this bit is set to 1. For example, when NVE field in register=31, the maximum valid entry address/index is NVE+1 i.e. 32 (addresses/indices=0 to 32, or 33 entries), so NVEOVIS is set when currently processed entry indicates next address is 33 or more i.e. 34th or later entries. This bit is cleared when the application writes 1 to this bit. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **NOTE** This field is not testable by an automated register test. 0b - Number of Valid Entries Overflow Interrupt Status not detected 1b - Number of Valid Entries Overflow Interrupt Status detected

76.17.277 MTL Rx Parser Drop Count (MTL_RXP_Drop_Cnt)

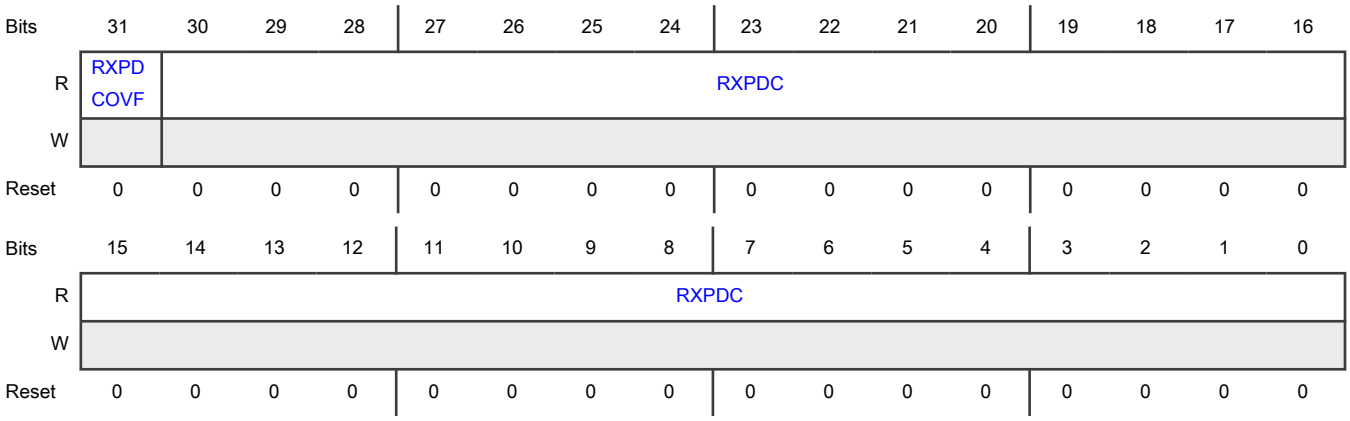
Offset

Register	Offset
MTL_RXP_Drop_Cnt	CA8h

Function

The MTL_RXP_Drop_Cnt register provides the drop count of Rx Parser initiated drops.

Diagram



Fields

Field	Function
31 RXPDCOVF	Rx Parser Drop Counter Overflow Bit When set, this bit indicates that the MTL_RXP_Drop_cnt (RXPDC) Counter field crossed the maximum limit. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - Rx Parser Drop count overflow not occurred 1b - Rx Parser Drop count overflow occurred
30-0 RXPDC	Rx Parser Drop count This 31-bit counter is implemented when a Rx Parser Drops a packet due to RF =1. The counter is cleared when the register is read.

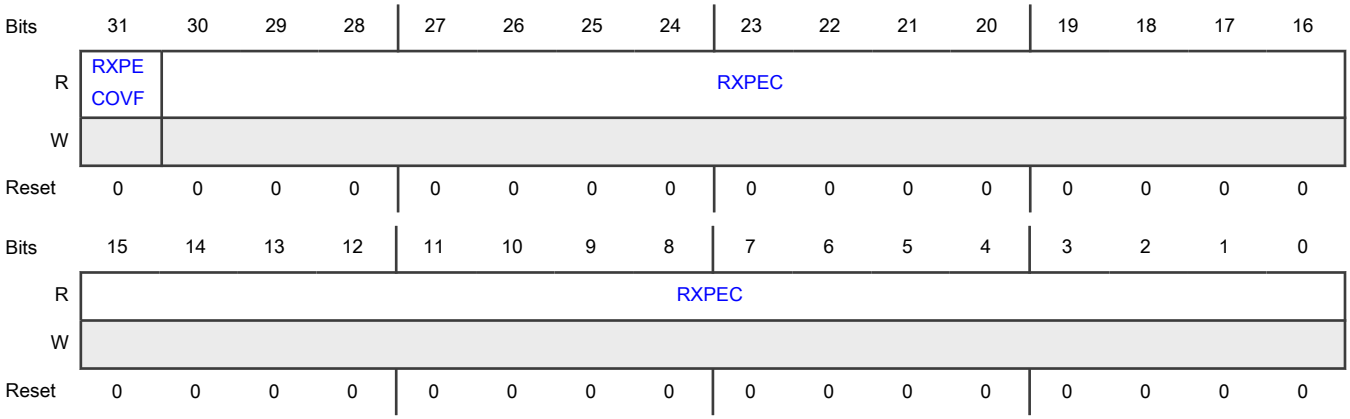
76.17.278 MTL Rx Parser Error Count (MTL_RXP_Error_Cnt)

Offset

Register	Offset
MTL_RXP_Error_Cnt	CACh

Function
The MTL_RXP_Error_Cnt register provides the Rx Parser related error occurrence count.

Diagram



Fields

Field	Function
31 RXPECOVF	Rx Parser Error Counter Overflow Bit When set, this bit indicates that the MTL_RXP_Error_cnt (RXPEC) Counter field crossed the maximum limit. Access restriction applies. Clears on read. Self-set to 1 on internal event.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Rx Parser Error count overflow not occurred 1b - Rx Parser Error count overflow occurred
30-0 RXPEC	Rx Parser Error count This 31-bit counter is implemented whenr a Rx Parser encounters following Error scenarios - Entry address >= NVE[] - Number Parsed Entries >= NPE[] - Entry address > EOF data entry address The counter is cleared when the register is read.

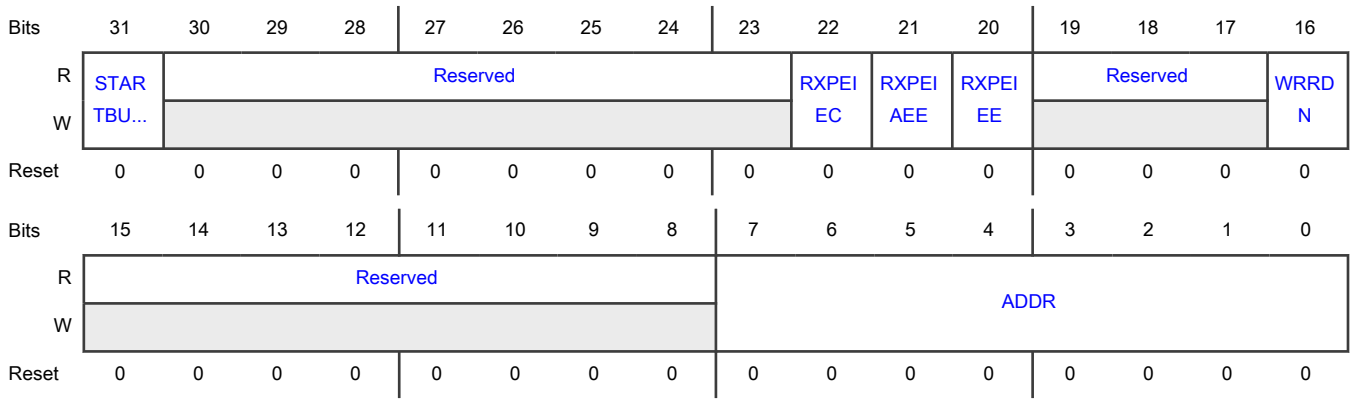
76.17.279 MTL Rx Parser Indirect Access Control Status (MTL_RXP_Indirect_Acc_Control_Status)

Offset

Register	Offset
MTL_RXP_Indirect_Acc_Control_Status	CB0h

Function
The MTL_RXP_Indirect_Acc_Control_Status register provides the Indirect Access control and status for Rx Parser memory.

Diagram



Fields

Field	Function
31 STARTBUSY	FRP Instruction Table Access Busy - Set to 1 by the software, indicates to start the Read/Write operation from/to the Rx Parser Memory. Software should read this bit as 0 before issuing read or write request to access the Parser Memory Instructions. - Set to 1 by the IP, indicates that the IP is busy until it is cleared by the IP. Software should not issue read or write operation.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field is not testable by an automated register test. 0b - hardware not busy 1b - hardware is busy (Read/Write operation from/to the Rx Parser Memory)
30-23 —	Reserved
22 RXPEIEC	ECC Inject Error Control for Rx Parser Memory When RXPEIEE or RXPEIAEE bit of this register is set, following are the errors inserted based on the value encoded in this field: - 1: Insert 1 bit error. - 0: Insert 2 bit errors. **NOTE** This field is not testable by an automated register test. 0b - Insert 1 bit error 1b - Insert 2 bit errors
21 RXPEIAEE	ECC Inject Address Error Enable for Rx Parser Memory - 1: Enables the ECC address error injection feature. - 0: Disables the ECC address error injection feature. **NOTE** This field is not testable by an automated register test. 0b - ECC Inject Address Error for Rx Parser Memory is disabled 1b - ECC Inject Address Error for Rx Parser Memory is enabled
20 RXPEIEE	ECC Inject Error Enable for Rx Parser Memory - 1: Enables the ECC error injection feature. - 0: Disables the ECC error injection feature. **NOTE** This field is not testable by an automated register test. 0b - ECC Inject Error for Rx Parser Memory is disabled 1b - ECC Inject Error for Rx Parser Memory is enabled
19-17 —	Reserved
16 WRRDN	Read Write Control - 1: Indicates the write operation to the Rx Parser Memory. - 0: Indicates the read operation to the Rx Parser Memory.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Read operation to the Rx Parser Memory 1b - Write operation to the Rx Parser Memory
15-8 —	Reserved
7-0 ADDR	FRP Instruction Table Offset Address This field indicates the ADDR of the 32-bit entry in Rx parser instruction table. Each entry has 128-bit (4x32-bit words). The X is based on the configurable DWC_EQOS_FRP_ENTRIES. If DWC_EQOS_FRP_ENTRIES is - 256: X = 9 - 128: X = 8 - 64: X = 7

76.17.280 MTL Rx Parser Indirect Access Data (MTL_RXP_Indirect_Acc_Data)

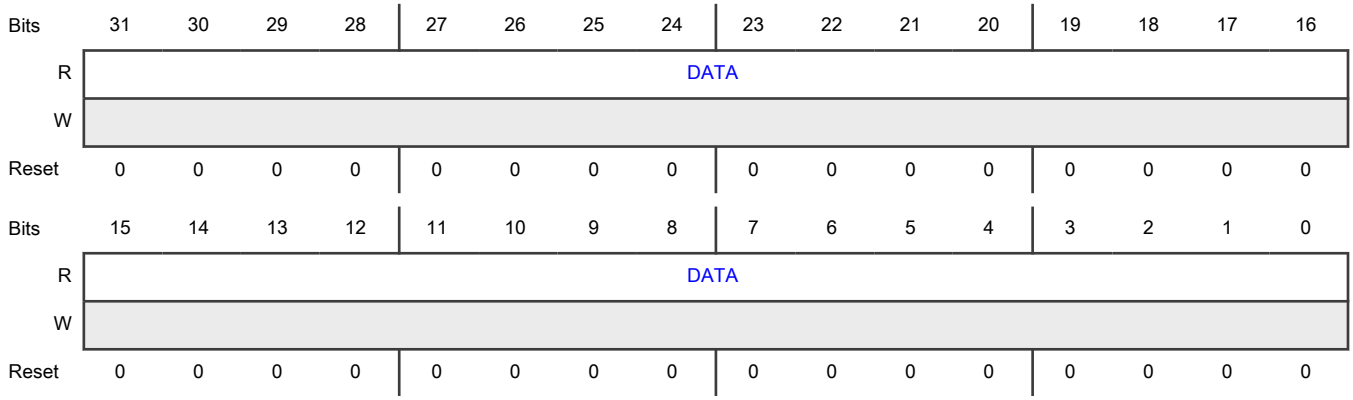
Offset

Register	Offset
MTL_RXP_Indirect_Acc_Data	CB4h

Function

The MTL_RXP_Indirect_Acc_Data registers holds the data associated to Indirect Access to Rx Parser memory.

Diagram



Fields

Field	Function
31-0 DATA	FRP Instruction Table Write/Read Data Software should write this register before issuing any write command. The hardware provides the read data from the Rx Parser Memory for read operation when STARTBUSY =0 after read command.

76.17.281 MTL Rx Parser Bypass Count (MTL_RXP_Bypass_Cnt)

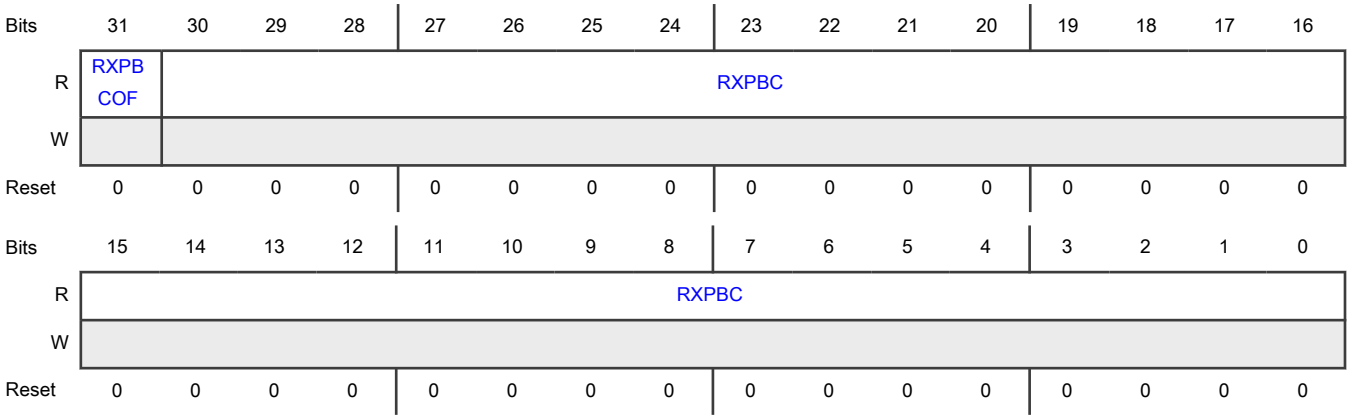
Offset

Register	Offset
MTL_RXP_Bypass_Cnt	CB8h

Function

The MTL_RXP_Bypass_Cnt register provides the bypass count of Rx Parser.

Diagram



Fields

Field	Function
31 RXPBCOF	Rx Parser bypass Counter Overflow Bit When set, this bit indicates that the MTL_RXP_Bypass_cnt (RXPBC) Counter field crossed the maximum limit. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - Rx Parser Bypass count overflow not occurred 1b - Rx Parser Bypass count overflow occurred
30-0 RXPBC	Rx Parser Bypass count This 31-bit counter is implemented when a Rx Parser bypass a packet due to AF=1 and RF =1. The counter is cleared when the register is read.

76.17.282 MTL ECC Control (MTL_ECC_Control)

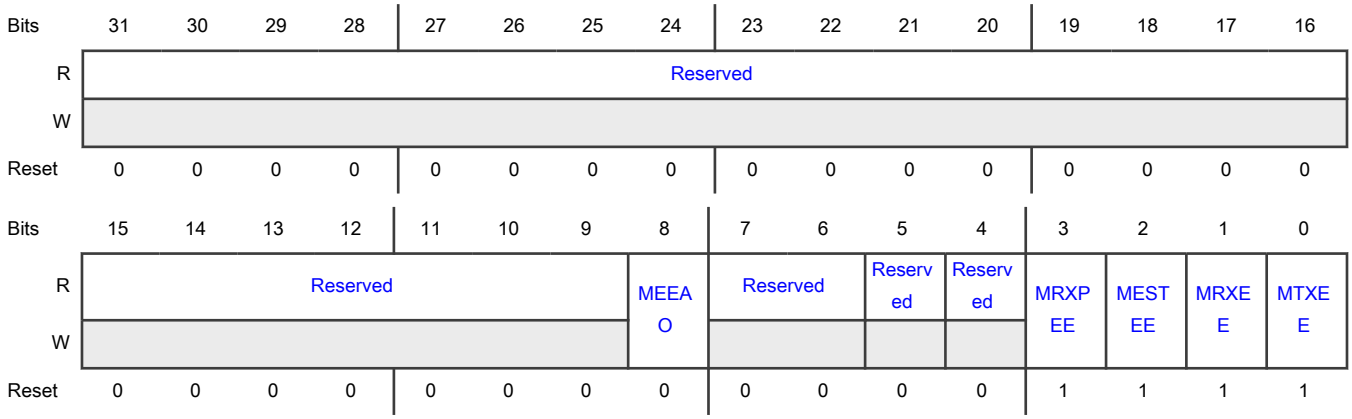
Offset

Register	Offset
MTL_ECC_Control	CC0h

Function

The MTL_ECC_Control register establishes the operating mode of ECC related to MTL memories.

Diagram



Fields

Field	Function
31-9 —	Reserved
8 MEEAO	MTL ECC Error Address Status Over-ride - 1: The EUEAS/ECEAS field of MTL_ECC_Err_Addr_Status register holds the last valid address where the error is detected. - 0: The EUEAS/ECEAS field of MTL_ECC_Err_Addr_Status register holds the first address where the error is detected. 0b - MTL ECC Error Address Status Over-ride is disabled 1b - MTL ECC Error Address Status Over-ride is enabled
7-6 —	Reserved
5 —	Reserved
4 —	Reserved
3 MRXPEE	MTL Rx Parser ECC Enable - 1: Enables the ECC feature for Rx Parser memory - 0: Disables the ECC feature for Rx Parser memory 0b - MTL Rx Parser ECC is disabled 1b - MTL Rx Parser ECC is enabled
2	MTL EST ECC Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
MESTEE	- 1: Enables the ECC feature for EST memory - 0: Disables the ECC feature for EST memory 0b - MTL EST ECC is disabled 1b - MTL EST ECC is enabled
1 MRXEE	MTL Rx FIFO ECC Enable - 1: Enables the ECC feature for MTL Rx FIFO memory. - 0: Disables the ECC feature for MTL Rx FIFO memory. 0b - MTL Rx FIFO ECC is disabled 1b - MTL Rx FIFO ECC is enabled
0 MTXEE	MTL Tx FIFO ECC Enable - 1: Enables the ECC feature for MTL Tx FIFO memory - 0: Disables the ECC feature for MTL Tx FIFO memory 0b - MTL Tx FIFO ECC is disabled 1b - MTL Tx FIFO ECC is enabled

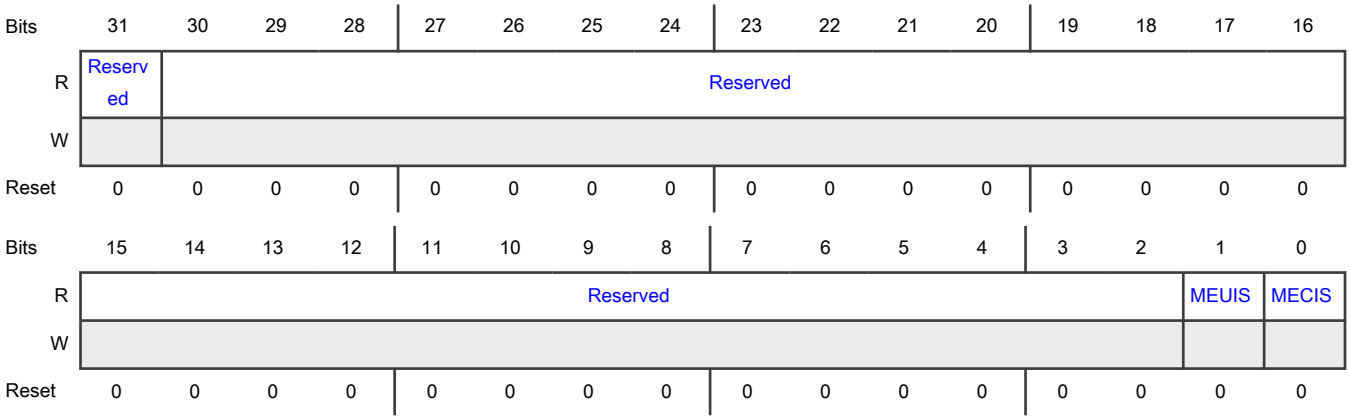
76.17.283 MTL Safety Interrupt Status (MTL_Safety_Interrupt_Status)

Offset

Register	Offset
MTL_Safety_Interrupt_Status	CC4h

Function
The MTL_Safety_Interrupt_Status registers provides Safety interrupt status.

Diagram



Fields

Field	Function
31 —	Reserved 0b - MAC Safety Uncorrectable Interrupt Status not detected 1b - MAC Safety Uncorrectable Interrupt Status detected
30-2 —	Reserved
1 MEUIS	MTL ECC Uncorrectable error Interrupt Status This bit indicates that an uncorrectable error interrupt event in the MTL ECC safety feature. To get the exact cause of the interrupt the application should read the MTL_ECC_Interrupt_Status register. 0b - MTL ECC Uncorrectable error Interrupt Status not detected 1b - MTL ECC Uncorrectable error Interrupt Status detected
0 MECIS	MTL ECC Correctable error Interrupt Status This bit indicates that a correctable error interrupt event in the MTL ECC safety feature. To get the exact cause of the interrupt the application should read the MTL_ECC_Interrupt_Status register. 0b - MTL ECC Correctable error Interrupt Status not detected 1b - MTL ECC Correctable error Interrupt Status detected

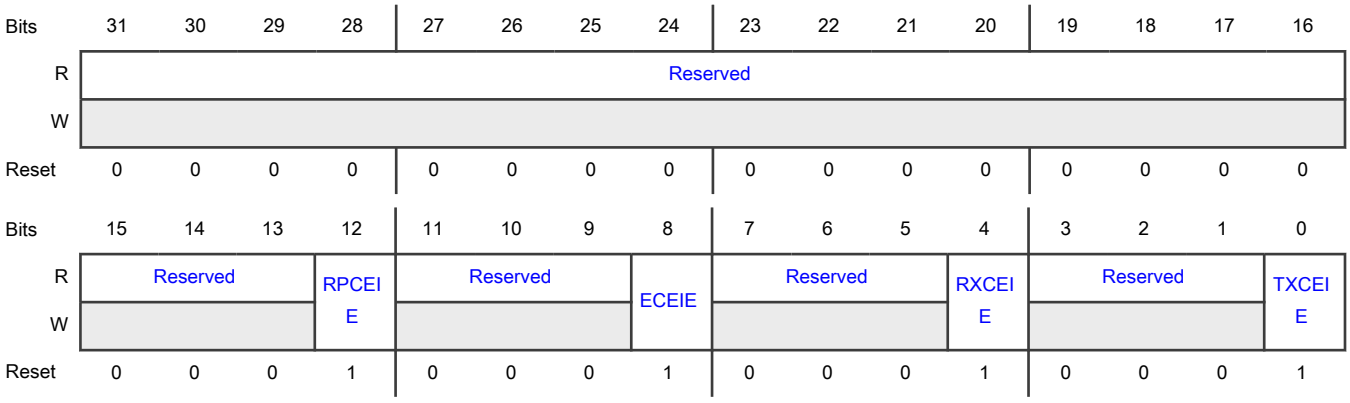
76.17.284 MTL ECC Interrupt Enable (MTL_ECC_Interrupt_Enable)

Offset

Register	Offset
MTL_ECC_Interrupt_Enable	CC8h

Function
The MTL_ECC_Interrupt_Enable register provides enable bits for the ECC interrupts.

Diagram



Fields

Field	Function
31-13 —	Reserved
12 RPCEIE	Rx Parser memory Correctable Error Interrupt Enable When set, generates an interrupt when an uncorrectable error is detected at the Rx Parser memory interface. It is indicated in RPCES status bit of MTL_ECC_Interrupt_Status register. When reset this event does not generates an interrupt. 0b - Rx Parser memory Correctable Error Interrupt is disabled 1b - Rx Parser memory Correctable Error Interrupt is enabled
11-9 —	Reserved
8 ECEIE	EST memory Correctable Error Interrupt Enable When set, generates an interrupt when a correctable error is detected at the MTL EST memory interface. It is indicated in the ECES bit of MTL_ECC_Interrupt_Status register. When reset this event does not generates an interrupt. 0b - EST memory Correctable Error Interrupt is disabled 1b - EST memory Correctable Error Interrupt is enabled
7-5 —	Reserved
4 RXCEIE	Rx memory Correctable Error Interrupt Enable When set, generates an interrupt when a correctable error is detected at the MTL Rx memory interface. It is indicated in the RXCES bit of MTL_ECC_Interrupt_Status register. When reset this event does not generates an interrupt. 0b - Rx memory Correctable Error Interrupt is disabled 1b - Rx memory Correctable Error Interrupt is enabled
3-1 —	Reserved
0 TXCEIE	Tx memory Correctable Error Interrupt Enable When set, generates an interrupt when a correctable error is detected at the MTL Tx memory interface. It is indicated in the TXCES bit of MTL_ECC_Interrupt_Status register. When reset this event does not generates an interrupt. 0b - Tx memory Correctable Error Interrupt is disabled 1b - Tx memory Correctable Error Interrupt is enabled

76.17.285 MTL ECC Interrupt Status (MTL_ECC_Interrupt_Status)

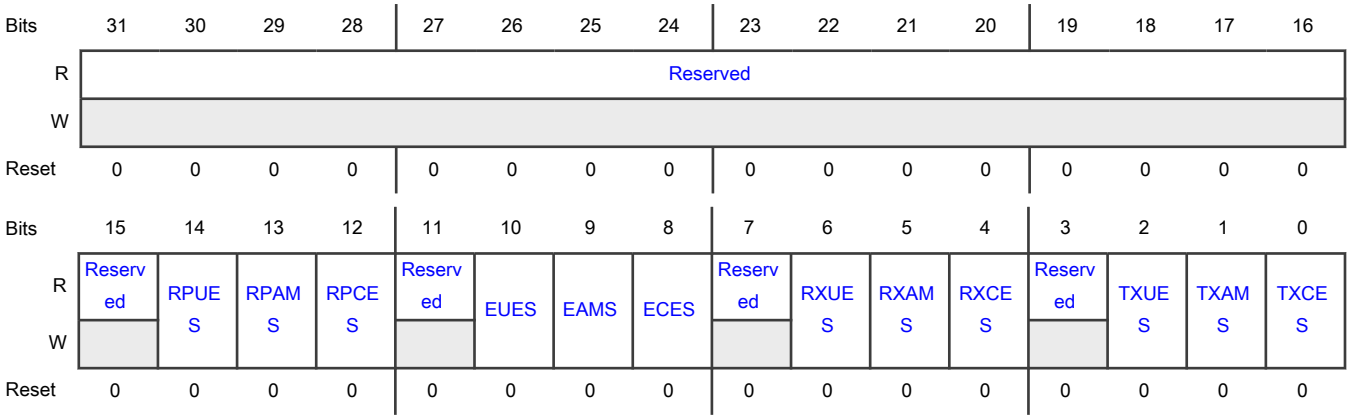
Offset

Register	Offset
MTL_ECC_Interrupt_Status	CCCh

Function

The MTL_ECC_Interrupt_Status register provides MTL ECC Interrupt Status.

Diagram



Fields

Field	Function
31-15 —	Reserved
14 RPUES	Rx Parser memory Uncorrectable Error Status When set, indicates that an uncorrectable error is detected at Rx Parser memory interface. **NOTE** This field is not testable by an automated register test. 0b - Rx Parser memory Uncorrectable Error Status not detected 1b - Rx Parser memory Uncorrectable Error Status detected
13 RPAMS	MTL Rx Parser memory Address Mismatch Status This bit when set indicates that address mismatch is found for address bus of Rx Parser memory. **NOTE** This field is not testable by an automated register test.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - MTL Rx Parser memory Address Mismatch Status not detected 1b - MTL Rx Parser memory Address Mismatch Status detected
12 RPCES	MTL Rx Parser memory Correctable Error Status This bit when set indicates that correctable error is detected at RX Parser memory interface. **NOTE** This field is not testable by an automated register test. 0b - MTL Rx Parser memory Correctable Error Status not detected 1b - MTL Rx Parser memory Correctable Error Status detected
11 —	Reserved
10 EUES	MTL EST memory Uncorrectable Error Status When set, indicates that an uncorrectable error is detected at MTL EST memory interface. **NOTE** This field is not testable by an automated register test. 0b - MTL EST memory Uncorrectable Error Status not detected 1b - MTL EST memory Uncorrectable Error Status detected
9 EAMS	MTL EST memory Address Mismatch Status This bit when set indicates that address mismatch is found for address bus of MTL EST memory. **NOTE** This field is not testable by an automated register test. 0b - MTL EST memory Address Mismatch Status not detected 1b - MTL EST memory Address Mismatch Status detected
8 ECES	MTL EST memory Correctable Error Status This bit when set indicates that correctable error is detected at the MTL EST memory. **NOTE** This field is not testable by an automated register test. 0b - MTL EST memory Correctable Error Status not detected 1b - MTL EST memory Correctable Error Status detected
7 —	Reserved
6	MTL Rx memory Uncorrectable Error Status

Table continues on the next page...

Table continued from the previous page...

Field	Function
RXUES	When set, indicates that an uncorrectable error is detected at the MTL Rx memory interface. **NOTE** This field is not testable by an automated register test. 0b - MTL Rx memory Uncorrectable Error Status not detected 1b - MTL Rx memory Uncorrectable Error Status detected
5 RXAMS	MTL Rx memory Address Mismatch Status This bit when set indicates that address mismatch is found for address bus of the MTL Rx memory. **NOTE** This field is not testable by an automated register test. 0b - MTL Rx memory Address Mismatch Status not detected 1b - MTL Rx memory Address Mismatch Status detected
4 RXCES	MTL Rx memory Correctable Error Status This bit when set indicates that correctable error is detected at the MTL Rx memory. **NOTE** This field is not testable by an automated register test. 0b - MTL Rx memory correctable Error Status not detected 1b - MTL Rx memory correctable Error Status detected
3 —	Reserved
2 TXUES	MTL Tx memory Uncorrectable Error Status When set, indicates that an uncorrectable error is detected at the MTL TX memory interface. **NOTE** This field is not testable by an automated register test. 0b - MTL Tx memory Uncorrectable Error Status not detected 1b - MTL Tx memory Uncorrectable Error Status detected
1 TXAMS	MTL Tx memory Address Mismatch Status This bit when set indicates that address mismatch is found for address bus of the MTL Tx memory. **NOTE** This field is not testable by an automated register test. 0b - MTL Tx memory Address Mismatch Status not detected 1b - MTL Tx memory Address Mismatch Status detected

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 TXCES	MTL Tx memory Correctable Error Status This bit when set indicates that a correctable error is detected at the MTL Tx memory. **NOTE** This field is not testable by an automated register test. 0b - MTL Tx memory Correctable Error Status not detected 1b - MTL Tx memory Correctable Error Status detected

76.17.286 MTL ECC Error Status (MTL_ECC_Err_Sts_Rctl)

Offset

Register	Offset
MTL_ECC_Err_Sts_Rctl	CD0h

Function
The MTL_ECC_Err_Sts_Rctl register establishes the control for ECC Error status capture.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved										CUES	CCES	EMS			EESRE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-6 —	Reserved
5 CUES	Clear Uncorrectable Error Status

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When this bit is set along with EESRE bit of this register, based on the EMS field of this register, the respective memory's uncorrectable error address and uncorrectable error count values are cleared upon reading. Hardware resets this bit when all the error status values are cleared. **NOTE** This field is not testable by an automated register test. 0b - Clear Uncorrectable Error Status not detected 1b - Clear Uncorrectable Error Status detected
4 CCES	Clear Correctable Error Status When this bit is set along with EESRE bit of this register, based on the EMS field of this register, the respective memory's correctable error address and correctable error count values are cleared upon reading. Hardware resets this bit when all the error status values are cleared. **NOTE** This field is not testable by an automated register test. 0b - Clear Correctable Error Status not detected 1b - Clear Correctable Error Status detected
3-1 EMS	MTL ECC Memory Selection When EESRE bit of this register is set, this field indicates which memory's error status value to be read. The memory selection encoding is as described. 000b - MTL Tx memory 001b - MTL Rx memory 010b - MTL EST memory 011b - MTL Rx Parser memory 100b - DMA TSO memory 101b - DMA DCACHE memory
0 EESRE	MTL ECC Error Status Read Enable When this bit is set, based on the EMS field of this register, the respective memory's error status values are captured as described: - The correctable and uncorrectable error count values are captured into MTL_ECC_Err_Cnt_Status register - The address location's of correctable and uncorrectable errors are captured into MTL_ECC_Err_Addr_Status register. Hardware resets this bit when all the status values are captured into the MTL_ECC_Err_Cnt_Status and MTL_ECC_Err_Addr_Status registers. **NOTE** This field is not testable by an automated register test. 0b - MTL ECC Error Status Read is disabled 1b - MTL ECC Error Status Read is enabled

76.17.287 MTL ECC Error Address Status (MTL_ECC_Err_Addr_Status)

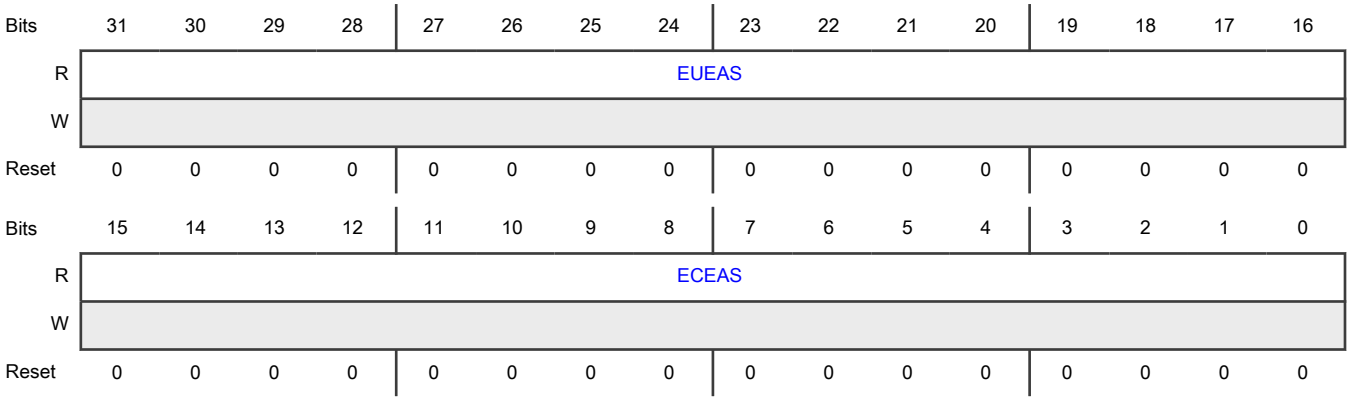
Offset

Register	Offset
MTL_ECC_Err_Addr_Status	CD4h

Function

The MTL_ECC_Err_Addr_Status register provides the memory addresses for the correctable and uncorrectable errors.

Diagram



Fields

Field	Function
31-16 EUEAS	MTL ECC Uncorrectable Error Address Status Based on the EMS field of MTL_ECC_Err_Sts_Rctl register, this field holds the respective memory's address locations for which an uncorrectable error or address mismatch is detected. - MEEAO field of MTL_ECC_Control register is 1: This field holds the last valid address of memory for which either an uncorrectable error or an address mismatch is detected. - MEEAO field of MTL_ECC_Control register is 0: This field holds the first address of the memory for which either an uncorrectable error or address mismatch is detected.
15-0 ECEAS	MTL ECC Correctable Error Address Status Based on the EMS field of MTL_ECC_Err_Sts_Rctl register, this field holds the respective memory's address locations for which a correctable error is detected. - MEEAO field of MTL_ECC_Control register is 1: This field holds the last valid address of memory for which correctable error or address mismatch is detected. - MEEAO field of MTL_ECC_Control register is 0: This field holds the first address of the memory for which correctable error is detected.

76.17.288 MTL ECC Error Control Status (MTL_ECC_Err_Cntr_Status)

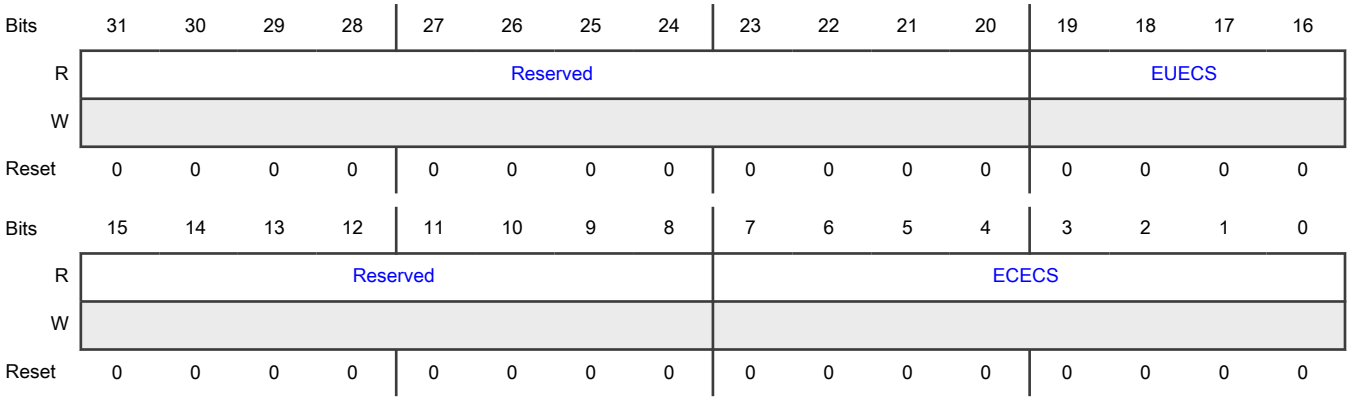
Offset

Register	Offset
MTL_ECC_Err_Cntr_Status	CD8h

Function

The MTL_ECC_Err_Cntr_Status register provides ECC Error count for Correctable and uncorrectable errors.

Diagram



Fields

Field	Function
31-20 —	Reserved
19-16 EUECS	MTL ECC Uncorrectable Error Counter Status Based on the EMS field of MTL_ECC_Err_Cntr_Rctl register, this field holds the respective memory's uncorrectable error count value.
15-8 —	Reserved
7-0 ECECS	MTL ECC Correctable Error Counter Status Based on the EMS field of MTL_ECC_Err_Cntr_Rctl register, this field holds the respective memory's correctable error count value.

76.17.289 MTL DPP Control (MTL_DPP_Control)

Offset

Register	Offset
MTL_DPP_Control	CE0h

Function

The MTL_DPP_Control establishes the operating mode of Data Parity protection and error injection.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															Reserved
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	IPEMRWC	IPEMTFC	IPEMTBU	Reserved	Reserved	Reserved	IPERID	IPEMTS	IPEMTF	IPETRD	Reserved	Reserved	Reserved	Reserved	OPE	EDPP
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Fields

Field	Function
31-17 —	Reserved
16 —	Reserved
15 IPEMRWC	Insert Parity Error in MTL RWC Data Parity Checker When set to 1, parity/data bit of first valid input parity/data of the MTL RWC data parity checker (or at PC12 as shown in received data path parity protection diagram) is flipped. Based on the EIM/BLEI field of MTL_DPP_ECC_EIC register, software can corrupt any one bit of parity/data. Hardware clears this bit once respective parity bit is flipped. 0b - Insert Parity error in MTL RWC data parity checker is disabled 1b - Insert Parity error in MTL RWC data parity
14 IPEMTFC	Insert Parity error in MAC TFC data parity checker When set to 1, parity/data bit of first valid input parity/data of the MAC TFC data parity checker (or at PC11 as shown in Transmit Data path parity protection diagram) is flipped. Based on the EIM/BLEI field

Table continues on the next page...

Table continued from the previous page...

Field	Function
	of MTL_DPP_ECC_EIC register, software can corrupt any one bit of parity/data. Hardware clears this bit once respective parity bit is flipped. **NOTE** This field is not testable by an automated register test. 0b - Insert Parity error in MAC TFC data parity checker is disabled 1b - Insert Parity error in MAC TFC data parity checker is enabled
13 IPEMTBU	Insert Parity error in MTL RWC data parity checker When set to 1, parity/data bit of first valid input parity/data of the MTL RWC data parity checker (or at PC12 as shown in Recieve data path parity protection diagram) is flipped. Based on the EIM/BLEI field of MTL_DPP_ECC_EIC register, software can corrupt any one bit of parity/data. Hardware clears this bit once respective parity bit is flipped. **NOTE** This field is not testable by an automated register test. 0b - Insert Parity error in MAC TBU data parity checker is disabled 1b - Insert Parity error in MAC TBU data parity checker is enabled
12 —	Reserved
11 —	Reserved
10 —	Reserved
9 IPERID	Insert Parity Error in RX Interface Data parity checker When set to 1, parity/data bit of first valid input parity/data of the RX Interface Data parity checker is (or at PC6 as shown in Receive data path parity protection diagram) flipped. Based on the EIM/BLEI field of MTL_DPP_ECC_EIC register, software can corrupt any one bit of parity/data. Following are the input data bus on which parity bits are generated based on configuration selected In AHB Config, hwdata_o In AXI config, wdata_m_o In DMA Config, mdc_wdata_o In MTL Config, ari_data_o Hardware clears this bit once respective parity bit is flipped. **NOTE** This field is not testable by an automated register test. 0b - Insert Parity Error in Rx Interface Data parity checker is disabled 1b - Insert Parity Error in Rx Interface Data parity checker is enabled
8 IPEMTS	Insert Parity Error in MTL Tx Status FIFO parity checker

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When set to 1, parity/data bit of first valid input parity/data of the MTL Tx Status FIFO parity checker (or at PC5 as shown in Transmit data path parity protection diagram) is flipped. Based on the EIM/BLEI field of MTL_DPP_ECC_EIC register, software can corrupt any one bit of parity/data. Hardware clears this bit once respective parity bit is flipped. **NOTE** This field is not testable by an automated register test. 0b - Insert Parity Error in MTL TX Status FIFO parity checker is disabled 1b - Insert Parity Error in MTL TX Status FIFO parity checker is enabled
7 IPEMTF	Insert Parity Error in MTL Tx FIFO write data parity checker When set to 1, parity/data bit of first valid input parity/data of the MTL Tx FIFO write data parity checker (or at PC4 as shown in Transmit data path parity protection diagram) is flipped. Based on the EIM/BLEI field of MTL_DPP_ECC_EIC register, software can corrupt any one bit of parity/data. Hardware clears this bit once respective parity bit is flipped. **NOTE** This field is not testable by an automated register test. 0b - Insert Parity Error in MTL Tx FIFO write data parity checker is disabled 1b - Insert Parity Error in MTL Tx FIFO write data parity checker is enabled
6 IPETRD	Insert Parity Error in DMA Tx/Rx Descriptor parity checker When set to 1, parity/data bit of first valid input parity/data of the DMA Tx/Rx Descriptor parity checker (or at PC3 as shown in Transmit data path parity protection diagram) is flipped. Based on the EIM/BLEI field of MTL_DPP_ECC_EIC register, software can corrupt any one bit of parity/data. Hardware clears this bit once respective parity bit is flipped. **NOTE** This field is not testable by an automated register test. 0b - Insert Parity Error in DMA Tx/Rx Descriptor parity checker is disabled 1b - Insert Parity Error in DMA Tx/Rx Descriptor parity checker is enabled
5 —	Reserved
4 —	Reserved
3 —	Reserved
2 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 OPE	Odd Parity Enable When set to 1, enables odd parity protection on all the external interfaces and when set to 0, enables even parity protection on all the external interfaces. 0b - Odd Parity is disabled 1b - Odd Parity is enabled
0 EDPP	Enable Data path Parity Protection When set to 1, enables the parity protection for EQOS datapath by generating and checking the parity on EQOS datapath. When set to 0, disables the parity protection for EQOS datapath. 0b - Data path Parity Protection is disabled 1b - Data path Parity Protection is enabled

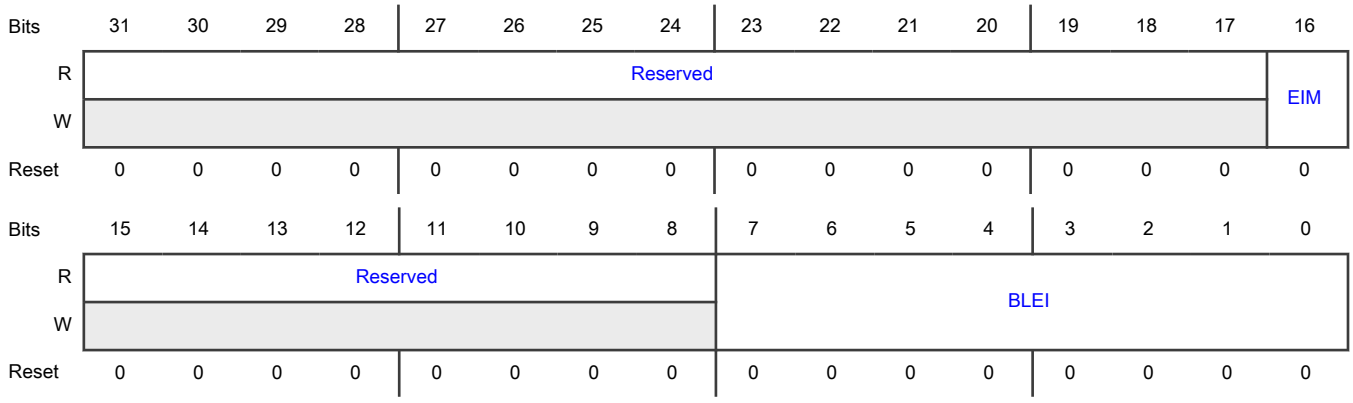
76.17.290 MTL DPP ECC Error Injection Channel (MTL_DPP_ECC_EIC)

Offset

Register	Offset
MTL_DPP_ECC_EIC	CE4h

Function
The MTL_DPP_ECC_EIC establishes the operating mode of ECC/DPP error injection.

Diagram



Fields

Field	Function
31-17	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
16 EIM	Error Injection Mode When it set to 0, indicate error injection on data; When it set to 1, indicate error injection on ECC/Parity bits(need to check the address error injection mode is disabled).
15-8 —	Reserved
7-0 BLEI	Bit Location of error injection This field indicates the bit location of DPP/ECC error injection, determination of error in Parity/ECC bits or Data (being protected) depends on the Error Injection Mode(EIM field). Depending on the interface being used for error injection not all bits of this field are valid. Example, for error injection on a 64 bit data interface this field should be programmed to a value between 63 and 0. In case of 2-bit error injection bit 0 is always included in error injection and this field should represent the second bit selection for error injection.If the second bit is programed at bit 0 when 2-bit error injection is enabled, DUT only insert 1-bit error at bit 0.

76.17.291 MTL Tx Queue 0 Operation Mode (MTL_TxQ0_Operation_Mode)

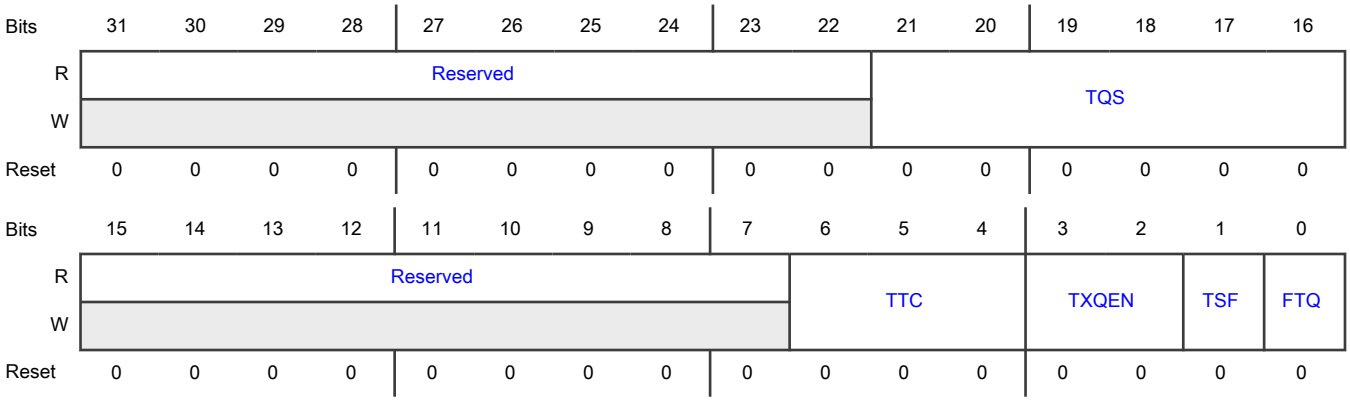
Offset

Register	Offset
MTL_TxQ0_Operation_Mode	D00h

Function

The Queue 0 Transmit Operation Mode register establishes the Transmit queue operating modes and commands.

Diagram



Fields

Field	Function
31-22 —	Reserved
21-16 TQS	Transmit Queue Size This field indicates the size of the allocated Transmit queues in blocks of 256 bytes. The TQS field is read-write only if the number of Tx Queues more than one, the reset value is 0x0 and indicates size of 256 bytes. This means that value of 0x0 = 256 bytes, 0x1 = 512 bytes and so on. So, program TQS [5:0] = 6'b001111 to allocate queue size of 4096 (4K) bytes. In general, the size of the Queue = (TQS+1)*256 bytes. When the number of Tx Queues is one, the field is read-only and the configured TX FIFO size in blocks of 256 bytes is reflected in the reset value. The width of this field depends on the Tx memory size selected in your configuration. For example, if the memory size is 2048, the width of this field is 3 bits: LOG2(2048/256) = LOG2(8) = 3 bits
15-7 —	Reserved
6-4 TTC	Transmit Threshold Control These bits control the threshold level of the MTL Tx Queue. The transmission starts when the packet size within the MTL Tx Queue is larger than the threshold. In addition, full packets with length less than the threshold are also transmitted. These bits are used only when the TSF bit is reset. 000b - 32 001b - 64 010b - 96 011b - 128 100b - 192 101b - 256 110b - 384 111b - 512
3-2 TXQEN	Transmit Queue Enable This field is used to enable/disable the transmit queue 0. - 2'b00: Not enabled - 2'b01: Reserved - 2'b10: Enabled - 2'b11: Reserved This field is Read Only in Single Queue configurations and Read Write in Multiple Queue configurations. Note: In multiple Tx queues configuration, all the queues are disabled by default. Enable the Tx queue by programming this field. 00b - Not enabled 01b - Enable in AV mode (Reserved in non-AV) 10b - Enabled 11b - Reserved
1 TSF	Transmit Store and Forward

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When this bit is set, the transmission starts when a full packet resides in the MTL Tx queue. When this bit is set, the TTC values specified in Bits[6:4] of this register are ignored. This bit should be changed only when the transmission is stopped. 0b - Transmit Store and Forward is disabled 1b - Transmit Store and Forward is enabled
0 FTQ	Flush Transmit Queue When this bit is set, the Tx queue controller logic is reset to its default values. Therefore, all the data in the Tx queue is lost or flushed. This bit is internally reset when the flushing operation is complete. Until this bit is reset, you should not write to the MTL_TxQ1_Operation_Mode register. The data which is already accepted by the MAC transmitter is not flushed. It is scheduled for transmission and results in underflow and runt packet transmission. Note: The flush operation is complete only when the Tx queue is empty and the application has accepted the pending Tx Status of all transmitted packets. To complete this flush operation, the PHY Tx clock (CLK_TX_I) should be active. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. 0b - Flush Transmit Queue is disabled 1b - Flush Transmit Queue is enabled

76.17.292 MTL Tx Queue 0 Underflow (MTL_TxQ0_Underflow)

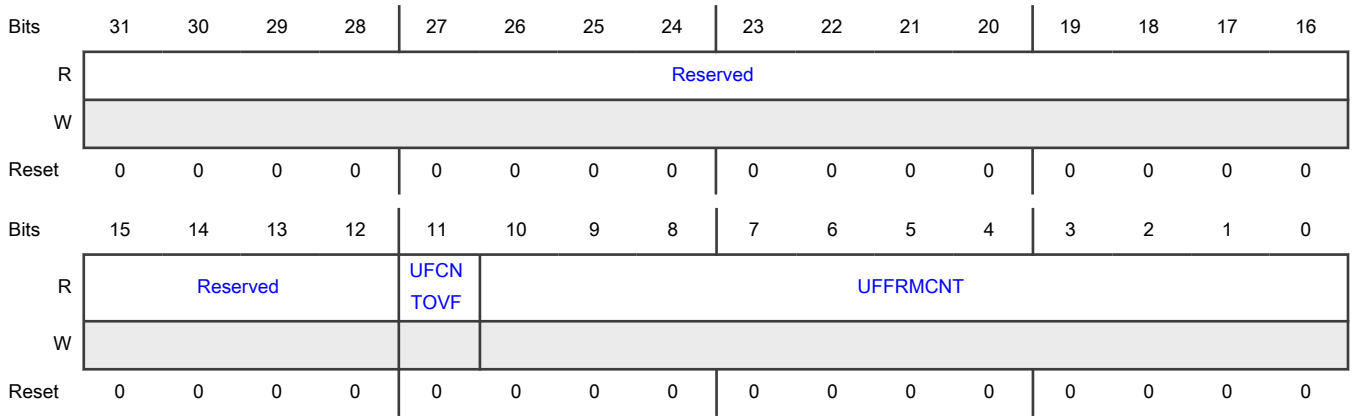
Offset

Register	Offset
MTL_TxQ0_Underflow	D04h

Function

The Queue 0 Underflow Counter register contains the counter for packets aborted because of Transmit queue underflow and packets missed because of Receive queue packet flush

Diagram



Fields

Field	Function
31-12 —	Reserved
11 UFCNTOVF	Overflow Bit for Underflow Packet Counter This bit is set every time the Tx queue Underflow Packet Counter field overflows, that is, it has crossed the maximum count. In such a scenario, the overflow packet counter is reset to all-zeros and this bit indicates that the rollover happened. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - Overflow not detected for Underflow Packet Counter 1b - Overflow detected for Underflow Packet Counter
10-0 UFFRMCNT	Underflow Packet Counter This field indicates the number of packets aborted by the controller because of Tx Queue Underflow. This counter is incremented each time the MAC aborts outgoing packet because of underflow. The counter is cleared when this register is read with mci_be_i[0] at 1'b1. Access restriction applies. Clears on read. Self-set to 1 on internal event.

76.17.293 MTL Tx Queue 0 Debug (MTL_TxQ0_Debug)

Offset

Register	Offset
MTL_TxQ0_Debug	D08h

Function
The Queue 0 Transmit Debug register gives the debug status of various blocks related to the Transmit queue.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved									STXSTSF			Reserved	PTXQ		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved										TXSTSFS...	TXQSTS	TWCSTS	TRCSTS		TXQPAUS...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-23 —	Reserved
22-20 STXSTSF	Number of Status Words in Tx Status FIFO of Queue This field indicates the current number of status in the Tx Status FIFO of this queue. When the DTXSTS bit of MTL_Operation_Mode register is set to 1, this field does not reflect the number of status words in Tx Status FIFO.
19 —	Reserved
18-16 PTXQ	Number of Packets in the Transmit Queue This field indicates the current number of packets in the Tx Queue. When the DTXSTS bit of MTL_Operation_Mode register is set to 1, this field does not reflect the number of packets in the Transmit queue.
15-6 —	Reserved
5 TXSTSFSTS	MTL Tx Status FIFO Full Status When high, this bit indicates that the MTL Tx Status FIFO is full. Therefore, the MTL cannot accept any more packets for transmission. 0b - MTL Tx Status FIFO Full status is not detected 1b - MTL Tx Status FIFO Full status is detected
4 TXQSTS	MTL Tx Queue Not Empty Status When this bit is high, it indicates that the MTL Tx Queue is not empty and some data is left for transmission. 0b - MTL Tx Queue Not Empty status is not detected 1b - MTL Tx Queue Not Empty status is detected
3 TWCSTS	MTL Tx Queue Write Controller Status When high, this bit indicates that the MTL Tx Queue Write Controller is active, and it is transferring the data to the Tx Queue. 0b - MTL Tx Queue Write Controller status is not detected 1b - MTL Tx Queue Write Controller status is detected
2-1 TRCSTS	MTL Tx Queue Read Controller Status This field indicates the state of the Tx Queue Read Controller: 00b - Idle state 01b - Read state (transferring data to the MAC transmitter)

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10b - Waiting for pending Tx Status from the MAC transmitter 11b - Flushing the Tx queue because of the Packet Abort request from the MAC
0 TXQPAUSED	Transmit Queue in Pause When this bit is high and the Rx flow control is enabled, it indicates that the Tx Queue is in the Pause condition (in the full-duplex only mode) because of the following: - Reception of the PFC packet for the priorities assigned to the Tx Queue when PFC is enabled - Reception of 802.3x Pause packet when PFC is disabled 0b - Transmit Queue in Pause status is not detected 1b - Transmit Queue in Pause status is detected

76.17.294 MTL Tx Queue 0 ETS Staus (MTL_TxQ0_ETS_Status)

Offset

Register	Offset
MTL_TxQ0_ETS_Status	D14h

Function
The Queue 0 ETS Status register provides the average traffic transmitted in Queue 0.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								ABS							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ABS															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
23-0 ABS	Average Bits per Slot This field contains the average transmitted bits per slot. When the DCB operation is enabled for Queue 0, this field is computed over every 10 million bit times slot (4 ms in 2500 Mbps; 10 ms in 1000 Mbps; 100 ms in 100 Mbps). The maximum value is 0x989680.

76.17.295 MTL Tx Queue 0 Quantum Weight (MTL_TxQ0_Quantum_Weight)

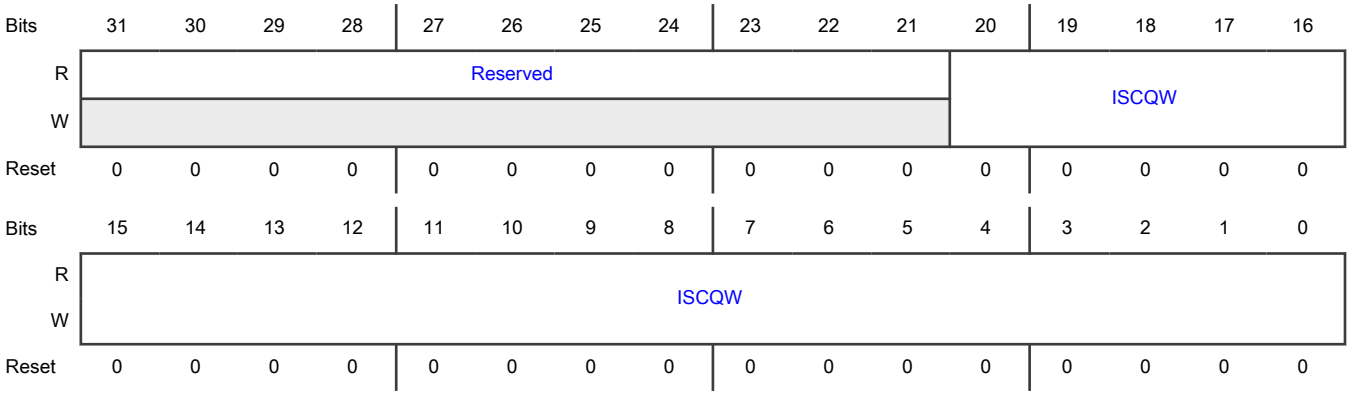
Offset

Register	Offset
MTL_TxQ0_Quantum_Weight	D18h

Function

The Queue 0 Quantum or Weights register contains the quantum value for Deficit Weighted Round Robin (DWRR), weights for the Weighted Round Robin (WRR), and Weighted Fair Queuing (WFQ) for Queue 0.

Diagram



Fields

Field	Function
31-21 —	Reserved
20-0 ISCQW	Quantum or Weights When the DCB operation is enabled with DWRR algorithm for Queue 0 traffic, this field contains the quantum value in bytes to be added to credit during every queue scanning cycle. The maximum value is 0x1312D0 bytes. When DCB operation is enabled with WFQ algorithm for Queue 0 traffic, this field contains the weight for this queue. The maximum value is 0x3FFF where weight of 0 indicates 100%

Table continues on the next page...

Table continued from the previous page...

Field	Function
	bandwidth. Bits[20:14] must be written to zero. The higher the programmed weights lesser the bandwidth allocated for the particular Transmit Queue. This is because the weights are used to compute the packet finish time (weights * packet_size). Lesser the finish time, higher the probability of the packet getting scheduled first and using more bandwidth. When DCB operation or generic queuing operation is enabled with WRR algorithm for Queue 0 traffic, this field contains the weight for this queue. The maximum value is 0x64. Bits [20:7] must be written to zero.

76.17.296 MTL Queue 0 Interrupt Control Status (MTL_Q0_Interrupt_Control_Status)

Offset

Register	Offset
MTL_Q0_Interrupt_Control_Status	D2Ch

Function
This register contains the interrupt enable and status bits for the queue 0 interrupts.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved							RXOIE	Reserved							RXOVFIS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved						ABPSIE	TXUIE	Reserved						ABPSIS	TXUNFIS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-25 —	Reserved
24 RXOIE	Receive Queue Overflow Interrupt Enable When this bit is set, the Receive Queue Overflow interrupt is enabled. When this bit is reset, the Receive Queue Overflow interrupt is disabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Receive Queue Overflow Interrupt is disabled 1b - Receive Queue Overflow Interrupt is enabled
23-17 —	Reserved
16 RXOVFIS	Receive Queue Overflow Interrupt Status This bit indicates that the Receive Queue had an overflow while receiving the packet. If a partial packet is transferred to the application, the overflow status is set in RDES3[21]. This bit is cleared when the application writes 1 to this bit. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **NOTE** This field is not testable by an automated register test. 0b - Receive Queue Overflow Interrupt Status not detected 1b - Receive Queue Overflow Interrupt Status detected
15-10 —	Reserved
9 ABPSIE	Average Bits Per Slot Interrupt Enable When this bit is set, the MAC asserts the sbd_intr_o or mci_intr_o interrupt when the average bits per slot status is updated. When this bit is cleared, the interrupt is not asserted for such an event. 0b - Average Bits Per Slot Interrupt is disabled 1b - Average Bits Per Slot Interrupt is enabled
8 TXUIE	Transmit Queue Underflow Interrupt Enable When this bit is set, the Transmit Queue Underflow interrupt is enabled. When this bit is reset, the Transmit Queue Underflow interrupt is disabled. 0b - Transmit Queue Underflow Interrupt Status is disabled 1b - Transmit Queue Underflow Interrupt Status is enabled
7-2 —	Reserved
1 ABPSIS	Average Bits Per Slot Interrupt Status When set, this bit indicates that the MAC has updated the ABS value. This bit is cleared when the application writes 1 to this bit. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **NOTE** This field is not testable by an automated register test.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Average Bits Per Slot Interrupt Status not detected 1b - Average Bits Per Slot Interrupt Status detected
0 TXUNFIS	Transmit Queue Underflow Interrupt Status This bit indicates that the Transmit Queue had an underflow while transmitting the packet. Transmission is suspended and an Underflow Error TDES3[2] is set. This bit is cleared when the application writes 1 to this bit. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **NOTE** This field is not testable by an automated register test. 0b - Transmit Queue Underflow Interrupt Status not detected 1b - Transmit Queue Underflow Interrupt Status detected

76.17.297 MTL Rx Queue 0 Operation Mode (MTL_RxQ0_Operation_Mode)

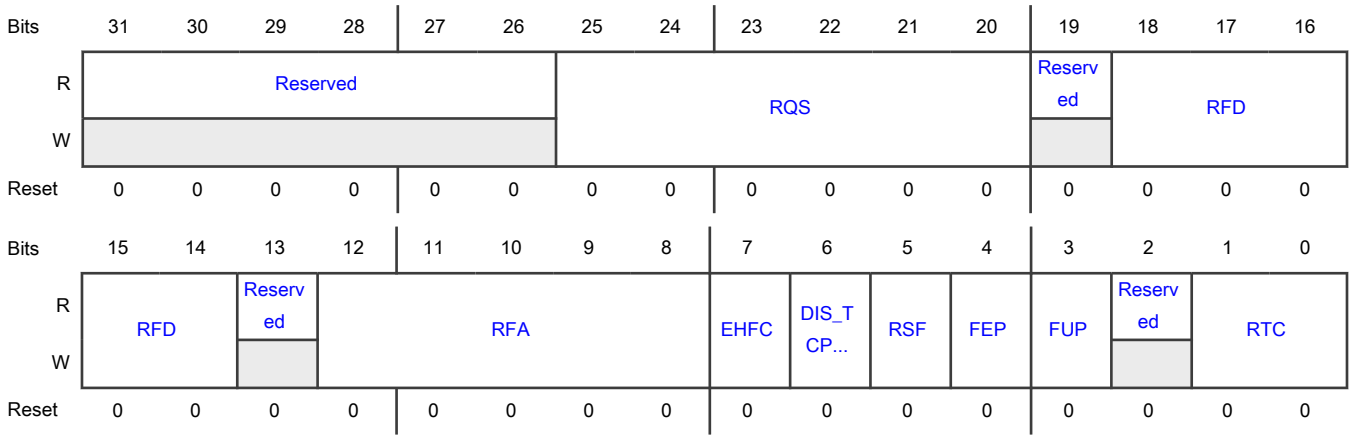
Offset

Register	Offset
MTL_RxQ0_Operation_Mode	D30h

Function

The Queue 0 Receive Operation Mode register establishes the Receive queue operating modes and command. The RFA and RFD fields are not backward compatible with the RFA and RFD fields of 4.00a release

Diagram



Fields

Field	Function
31-26 —	Reserved
25-20 RQS	Receive Queue Size This field indicates the size of the allocated Receive queues in blocks of 256 bytes. The RQS field is read-write only if the number of Rx Queues more than one, the reset value is 0x0 and indicates size of 256 bytes. This means that value of 0x0 = 256 bytes, 0x1 = 512 bytes and so on. So, program RQS [5:0] = 6'b001111 to allocate queue size of 4096 (4K) bytes. In general, the size of the Queue = (RQS+1)*256 bytes. When the number of Rx Queues is one, the field is read-only and the configured RX FIFO size in blocks of 256 bytes is reflected in the reset value. The width of this field depends on the Rx memory size selected in your configuration. For example, if the memory size is 2048, the width of this field is 3 bits: LOG2(2048/256) = LOG2(8) = 3 bits
19 —	Reserved
18-14 RFD	Threshold for Deactivating Flow Control (in half-duplex and full-duplex modes) These bits control the threshold (fill-level of Rx queue) at which the flow control is de-asserted after activation: - 0: Full minus 1 KB, that is, FULL 1 KB - 1: Full minus 1.5 KB, that is, FULL 1.5 KB - 2: Full minus 2 KB, that is, FULL 2 KB - 3: Full minus 2.5 KB, that is, FULL 2.5 KB - ... - 30: Full minus 16 KB, that is, FULL 16 KB - 31: Full minus 16.5 KB, that is, FULL 16.5 KB The de-assertion is effective only after flow control is asserted. Note: The value must be programmed in such a way to make sure that the threshold is a positive number. When the EHFC is set high, these values are applicable only when the Rx queue size determined by the RQS field of this register, is equal to or greater than 4 KB. For a given queue size, the values ranges between 0 and the encoding for FULL minus (QSIZE - 0.5 KB) and all other values are illegal. Here the term FULL and QSIZE refers to the queue size determined by the RQS field of this register. The width of this field depends on RX FIFO size selected during the configuration. Remaining bits are reserved and read only.
13 —	Reserved
12-8 RFA	Threshold for Activating Flow Control (in half-duplex and full-duplex These bits control the threshold (fill-level of Rx queue) at which the flow control is activated: For more information on encoding for this field, see RFD.
7 EHFC	Enable Hardware Flow Control When this bit is set, the flow control signal operation, based on the fill-level of Rx queue, is enabled. When reset, the flow control operation is disabled. 0b - Hardware Flow Control is disabled 1b - Hardware Flow Control is enabled
6 DIS_TCP_EF	Disable Dropping of TCP/IP Checksum Error Packets When this bit is set, the MAC does not drop the packets which only have the errors detected by the Receive Checksum Offload engine. Such packets have errors only in the encapsulated payload. There

Table continues on the next page...

Table continued from the previous page...

Field	Function
	are no errors (including FCS error) in the Ethernet packet received by the MAC. When this bit is reset, all error packets are dropped if the FEP bit is reset. 0b - Dropping of TCP/IP Checksum Error Packets is enabled 1b - Dropping of TCP/IP Checksum Error Packets is disabled
5 RSF	Receive Queue Store and Forward When this bit is set, the DWC_ether_qos reads a packet from the Rx queue only after the complete packet has been written to it, ignoring the RTC field of this register. When this bit is reset, the Rx queue operates in the Threshold (cut-through) mode, subject to the threshold specified by the RTC field of this register. 0b - Receive Queue Store and Forward is disabled 1b - Receive Queue Store and Forward is enabled
4 FEP	Forward Error Packets When this bit is reset, the Rx queue drops packets with error status (CRC error, GMII_ER, watchdog timeout, or overflow). However, if the start byte (write) pointer of a packet is already transferred to the read controller side (in Threshold mode), the packet is not dropped. When this bit is set, all packets except the runt error packets are forwarded to the application or DMA. If the RSF bit is set and the Rx queue overflows when a partial packet is written, the packet is dropped irrespective of the setting of this bit. However, if the RSF bit is reset and the Rx queue overflows when a partial packet is written, a partial packet might be forwarded to the application or DMA. 0b - Forward Error Packets is disabled 1b - Forward Error Packets is enabled
3 FUP	Forward Undersized Good Packets When this bit is set, the Rx queue forwards the undersized good packets (packets with no error and length less than 64 bytes), including pad-bytes and CRC. When this bit is reset, the Rx queue drops all packets of less than 64 bytes, unless a packet is already transferred because of the lower value of Rx Threshold, for example, RTC = 01. 0b - Forward Undersized Good Packets is disabled 1b - Forward Undersized Good Packets is enabled
2 —	Reserved
1-0 RTC	Receive Queue Threshold Control These bits control the threshold level of the MTL Rx queue (in bytes): The received packet is transferred to the application or DMA when the packet size within the MTL Rx queue is larger than the threshold. In addition, full packets with length less than the threshold are automatically transferred. This field is valid only when the RSF bit is zero. This field is ignored when the RSF bit is set to 1. 00b - 64

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01b - 32 10b - 96 11b - 128

76.17.298 MTL Rx Queue 0 Missed Packet Overflow Count (MTL_RxQ0_Missed_Packet_Overflow_Cnt)

Offset

Register	Offset
MTL_RxQ0_Missed_Packet_Overflow_Cnt	D34h

Function

The Queue 0 Missed Packet and Overflow Counter register contains the counter for packets missed because of Receive queue packet flush and packets discarded because of Receive queue overflow.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved				MISCNTO...	MISPKTCNT										
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved				OVFCNTO...	OVFPKTCNT										
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved
27 MISCNTOVF	Missed Packet Counter Overflow Bit When set, this bit indicates that the Rx Queue Missed Packet Counter crossed the maximum limit. Access restriction applies. Clears on read. Self-set to 1 on internal event.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Missed Packet Counter overflow not detected 1b - Missed Packet Counter overflow detected
26-16 MISPKTCNT	Missed Packet Counter This field indicates the number of packets missed by the DWC_ether_qos because the application asserted ari_pkt_flush_i[] for this queue. This counter is reset when this register is read with mci_be_i[0] at 1b1. This counter is incremented by 1 when the DMA discards the packet because of buffer unavailability. Access restriction applies. Clears on read. Self-set to 1 on internal event.
15-12 —	Reserved
11 OVFCNTOVF	Overflow Counter Overflow Bit When set, this bit indicates that the Rx Queue Overflow Packet Counter field crossed the maximum limit. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - Overflow Counter overflow not detected 1b - Overflow Counter overflow detected
10-0 OVFPKTCNT	Overflow Packet Counter This field indicates the number of packets discarded by the DWC_ether_qos because of Receive queue overflow. This counter is incremented each time the DWC_ether_qos discards an incoming packet because of overflow. This counter is reset when this register is read with mci_be_i[0] at 1'b1. Access restriction applies. Clears on read. Self-set to 1 on internal event.

76.17.299 MTL Rx Queue 0 Debug (MTL_RxQ0_Debug)

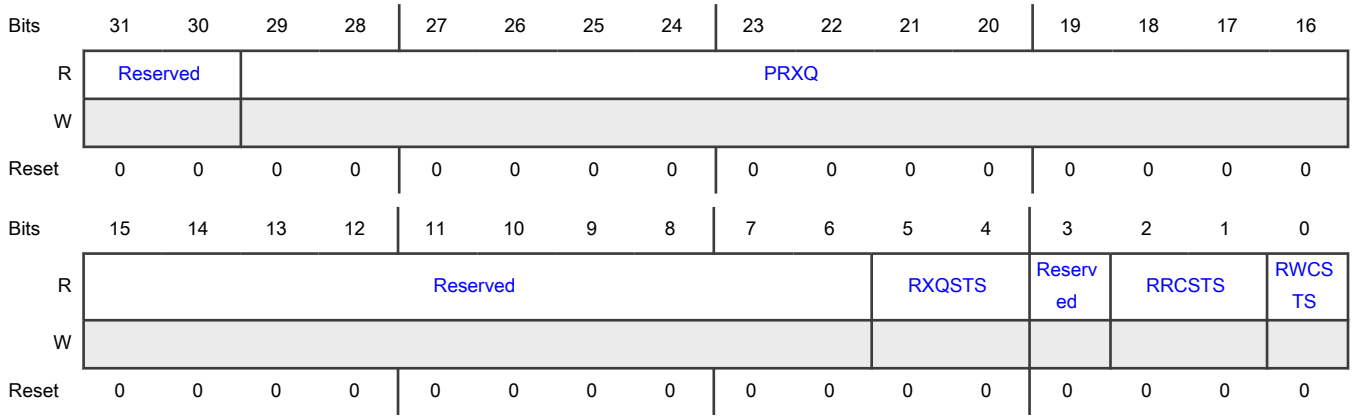
Offset

Register	Offset
MTL_RxQ0_Debug	D38h

Function

The Queue 0 Receive Debug register gives the debug status of various blocks related to the Receive queue.

Diagram



Fields

Field	Function
31-30 —	Reserved
29-16 PRXQ	Number of Packets in Receive Queue This field indicates the current number of packets in the Rx Queue. The theoretical maximum value for this field is 256KB/16B = 16K Packets, that is, Max_Queue_Size/Min_Packet_Size.
15-6 —	Reserved
5-4 RXQSTS	MTL Rx Queue Fill-Level Status This field gives the status of the fill-level of the Rx Queue: 00b - Rx Queue empty 01b - Rx Queue fill-level below flow-control deactivate threshold 10b - Rx Queue fill-level above flow-control activate threshold 11b - Rx Queue full
3 —	Reserved
2-1 RRCSTS	MTL Rx Queue Read Controller State This field gives the state of the Rx queue Read controller: 00b - Idle state 01b - Reading packet data 10b - Reading packet status (or timestamp) 11b - Flushing the packet data and status

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 RWCSTS	MTL Rx Queue Write Controller Active Status When high, this bit indicates that the MTL Rx queue Write controller is active, and it is transferring a received packet to the Rx Queue. 0b - MTL Rx Queue Write Controller Active Status not detected 1b - MTL Rx Queue Write Controller Active Status detected

76.17.300 MTL Tx Queue 0 Control (MTL_RxQ0_Control)

Offset

Register	Offset
MTL_RxQ0_Control	D3Ch

Function
The Queue Receive Control register controls the receive arbitration and passing of received packets to the application.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												RXQ_FRM...	RXQ_WEGT		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-4 —	Reserved
3 RXQ_FRM_ARBIT	Receive Queue Packet Arbitration When this bit is set, the DWC_ether_qos drives the packet data to the ARI interface such that the entire packet data of currently-selected queue is transmitted before switching to other queue. When this bit is reset, the DWC_ether_qos drives the packet data to the ARI interface such that the following amount of data of currently-selected queue is transmitted before switching to other queue: - PBL amount of data

Table continues on the next page...

Table continued from the previous page...

Field	Function
	(indicated by ari_qN_pbl_i[]) or - Complete data of a packet The status and the timestamp are not a part of the PBL data. Therefore, the DWC_ether_qos drives the complete status (including timestamp status) during first PBL request for the packet (in store-and-forward mode) or the last PBL request for the packet (in Threshold mode). 0b - Receive Queue Packet Arbitration is disabled 1b - Receive Queue Packet Arbitration is enabled
2-0 RXQ_WEGT	Receive Queue Weight This field indicates the weight assigned to the Rx Queue 0. This field needs to be programmed with one value less than the required weight, that is, reset value of 0 indicates weight of 1, value of 1 indicates weight of 2, and so on. The weight is used as the number of continuous PBL or packets requests (depending on the RXQ_FRM_ARBIT) allocated to the queue in one arbitration cycle. Note: The change in value of RXQ_WEGT takes effect only after the completion of current service round or when there is change from RAA=SP to RAA=WSP algorithm. This approach is taken so that there is smooth transition. For the RXQ_WEGT value to take effect at the start, the MTL_RxQ(#i)_Control registers must be programmed before the MTL_Operation_Mode register.

76.17.301 MTL Tx Queue 1 Operation Mode (MTL_TxQ1_Operation_Mode)

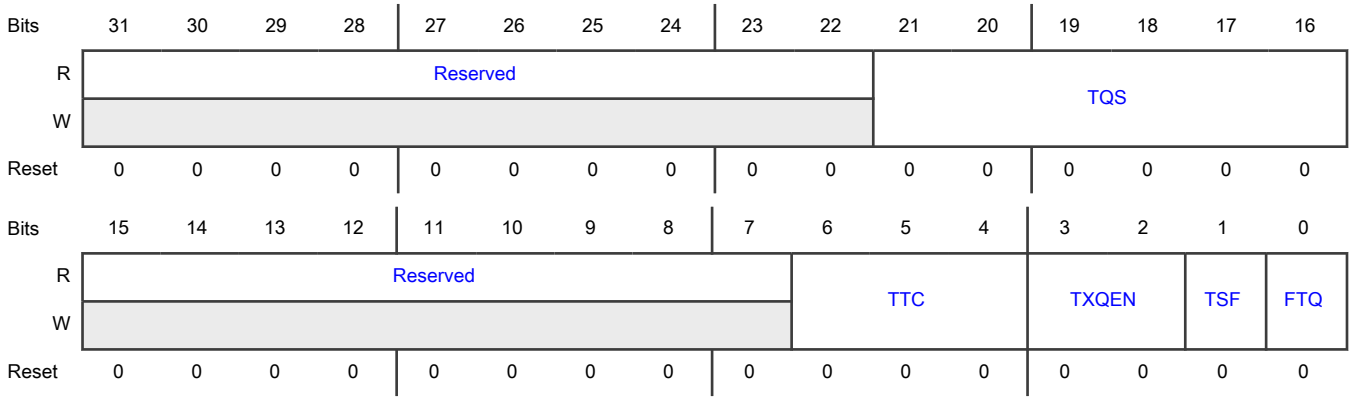
Offset

Register	Offset
MTL_TxQ1_Operation_Mode	D40h

Function

The Queue 1 Transmit Operation Mode register establishes the Transmit queue operating modes and commands.

Diagram



Fields

Field	Function
31-22 —	Reserved
21-16 TQS	Transmit Queue Size This field indicates the size of the allocated Transmit queues in blocks of 256 bytes. The TQS field is read-write only if the number of Tx Queues more than one, the reset value is 0x0 and indicates size of 256 bytes. This means that value of 0x0 = 256 bytes, 0x1 = 512 bytes and so on. So, program TQS [5:0] = 6'b001111 to allocate queue size of 4096 (4K) bytes. In general, the size of the Queue = (TQS+1)*256 bytes. When the number of Tx Queues is one, the field is read-only and the configured TX FIFO size in blocks of 256 bytes is reflected in the reset value. The width of this field depends on the Tx memory size selected in your configuration. For example, if the memory size is 2048, the width of this field is 3 bits: LOG2(2048/256) = LOG2(8) = 3 bits
15-7 —	Reserved
6-4 TTC	Transmit Threshold Control These bits control the threshold level of the MTL Tx Queue. The transmission starts when the packet size within the MTL Tx Queue is larger than the threshold. In addition, full packets with length less than the threshold are also transmitted. These bits are used only when the TSF bit is reset. 000b - 32 001b - 64 010b - 96 011b - 128 100b - 192 101b - 256 110b - 384 111b - 512
3-2 TXQEN	Transmit Queue Enable This field is used to enable/disable the transmit queue 0. - 2'b00: Not enabled - 2'b01: Enable in AV mode - 2'b10: Enabled - 2'b11: Reserved Note: In multiple Tx queues configuration, all the queues are disabled by default. Enable the Tx queue by programming this field. 00b - Not enabled 01b - Enable in AV mode (Reserved in non-AV) 10b - Enabled 11b - Reserved
1 TSF	Transmit Store and Forward

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When this bit is set, the transmission starts when a full packet resides in the MTL Tx queue. When this bit is set, the TTC values specified in Bits[6:4] of this register are ignored. This bit should be changed only when the transmission is stopped. 0b - Transmit Store and Forward is disabled 1b - Transmit Store and Forward is enabled
0 FTQ	Flush Transmit Queue When this bit is set, the Tx queue controller logic is reset to its default values. Therefore, all the data in the Tx queue is lost or flushed. This bit is internally reset when the flushing operation is complete. Until this bit is reset, you should not write to the MTL_TxQ1_Operation_Mode register. The data which is already accepted by the MAC transmitter is not flushed. It is scheduled for transmission and results in underflow and runt packet transmission. Note: The flush operation is complete only when the Tx queue is empty and the application has accepted the pending Tx Status of all transmitted packets. To complete this flush operation, the PHY Tx clock (CLK_TX_I) should be active. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. 0b - Flush Transmit Queue is disabled 1b - Flush Transmit Queue is enabled

76.17.302 MTL Tx Queue 1 Underflow (MTL_TxQ1_Underflow)

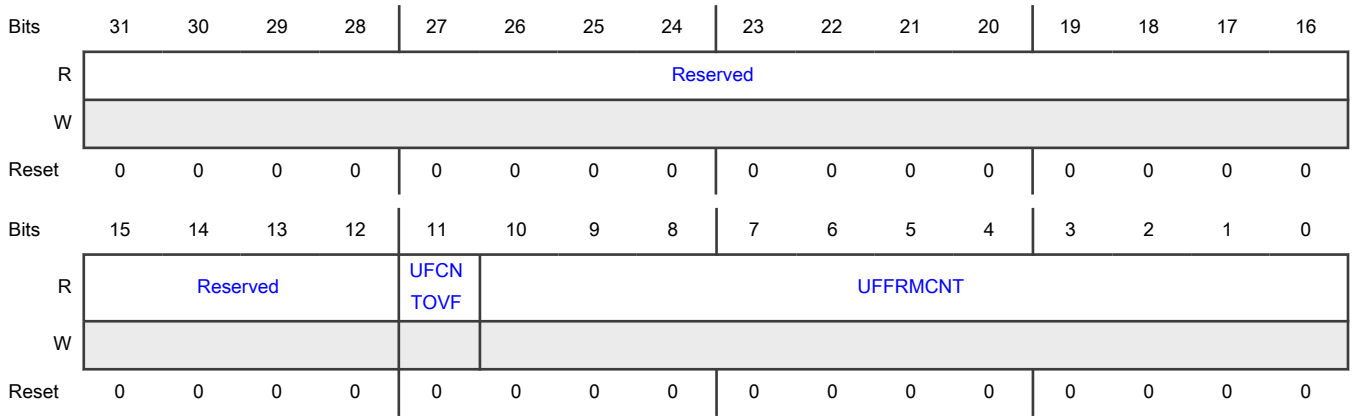
Offset

Register	Offset
MTL_TxQ1_Underflow	D44h

Function

The Queue 1 Underflow Counter register contains the counter for packets aborted because of Transmit queue underflow and packets missed because of Receive queue packet flush

Diagram



Fields

Field	Function
31-12 —	Reserved
11 UFCNTOVF	Overflow Bit for Underflow Packet Counter This bit is set every time the Tx queue Underflow Packet Counter field overflows, that is, it has crossed the maximum count. In such a scenario, the overflow packet counter is reset to all-zeros and this bit indicates that the rollover happened. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - Overflow not detected for Underflow Packet Counter 1b - Overflow detected for Underflow Packet Counter
10-0 UFFRMCNT	Underflow Packet Counter This field indicates the number of packets aborted by the controller because of Tx Queue Underflow. This counter is incremented each time the MAC aborts outgoing packet because of underflow. The counter is cleared when this register is read with mci_be_i[0] at 1'b1. Access restriction applies. Clears on read. Self-set to 1 on internal event.

76.17.303 MTL Tx Queue 1 Debug (MTL_TxQ1_Debug)

Offset

Register	Offset
MTL_TxQ1_Debug	D48h

Function
The Queue 1 Transmit Debug register gives the debug status of various blocks related to the Transmit queue.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved										STXSTSF		Reserved	PTXQ		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved										TXSTSFS...	TXQSTS	TWCSTS	TRCSTS		TXQPAUS...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-23 —	Reserved
22-20 STXSTSF	Number of Status Words in Tx Status FIFO of Queue This field indicates the current number of status in the Tx Status FIFO of this queue. When the DTXSTS bit of MTL_Operation_Mode register is set to 1, this field does not reflect the number of status words in Tx Status FIFO.
19 —	Reserved
18-16 PTXQ	Number of Packets in the Transmit Queue This field indicates the current number of packets in the Tx Queue. When the DTXSTS bit of MTL_Operation_Mode register is set to 1, this field does not reflect the number of packets in the Transmit queue.
15-6 —	Reserved
5 TXSTSFSTS	MTL Tx Status FIFO Full Status When high, this bit indicates that the MTL Tx Status FIFO is full. Therefore, the MTL cannot accept any more packets for transmission. 0b - MTL Tx Status FIFO Full status is not detected 1b - MTL Tx Status FIFO Full status is detected
4 TXQSTS	MTL Tx Queue Not Empty Status When this bit is high, it indicates that the MTL Tx Queue is not empty and some data is left for transmission. 0b - MTL Tx Queue Not Empty status is not detected 1b - MTL Tx Queue Not Empty status is detected
3 TWCSTS	MTL Tx Queue Write Controller Status When high, this bit indicates that the MTL Tx Queue Write Controller is active, and it is transferring the data to the Tx Queue. 0b - MTL Tx Queue Write Controller status is not detected 1b - MTL Tx Queue Write Controller status is detected
2-1 TRCSTS	MTL Tx Queue Read Controller Status This field indicates the state of the Tx Queue Read Controller: 00b - Idle state 01b - Read state (transferring data to the MAC transmitter)

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10b - Waiting for pending Tx Status from the MAC transmitter 11b - Flushing the Tx queue because of the Packet Abort request from the MAC
0 TXQPAUSED	Transmit Queue in Pause When this bit is high and the Rx flow control is enabled, it indicates that the Tx Queue is in the Pause condition (in the full-duplex only mode) because of the following: - Reception of the PFC packet for the priorities assigned to the Tx Queue when PFC is enabled - Reception of 802.3x Pause packet when PFC is disabled 0b - Transmit Queue in Pause status is not detected 1b - Transmit Queue in Pause status is detected

76.17.304 MTL Tx Queue 1 ETS Control (MTL_TxQ1_ETS_Control)

Offset

Register	Offset
MTL_TxQ1_ETS_Control	D50h

Function

The Queue ETS Control register controls the enhanced transmission selection operation.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								SLC				CC	AVALG	Reserved	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-7 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
6-4 SLC	Slot Count If the credit-based shaper algorithm is enabled, the software can program the number of slots (of duration programmed in DMA_CH(#i)_Slot_Interval register) over which the average transmitted bits per slot, provided in the MTL_TxQ(#i)_ETS_Status register, need to be computed for Queue. The encoding is as follows: 000b - 1 slot 001b - 2 slots 010b - 4 slots 011b - 8 slots 100b - 16 slots 101b - Reserved
3 CC	Credit Control When this bit is set, the accumulated credit parameter in the credit-based shaper algorithm logic is not reset to zero when there is positive credit and no packet to transmit in Channel 1. The credit accumulates even when there is no packet waiting in Channel 1 and another channel is transmitting. When this bit is reset, the accumulated credit parameter in the credit-based shaper algorithm logic is set to zero when there is positive credit and no packet to transmit in Channel 1. When there is no packet waiting in Channel 1 and other channel is transmitting, no credit is accumulated. 0b - Credit Control is disabled 1b - Credit Control is enabled
2 AVALG	AV Algorithm When Queue 1 is programmed for AV, this field configures the scheduling algorithm for this queue: This bit when set, indicates credit based shaper algorithm (CBS) is selected for Queue 1 traffic. When reset, strict priority is selected. 0b - CBS Algorithm is disabled 1b - CBS Algorithm is enabled
1-0 —	Reserved

76.17.305 MTL Tx Queue 1 ETS_Status (MTL_TxQ1_ETS_Status)

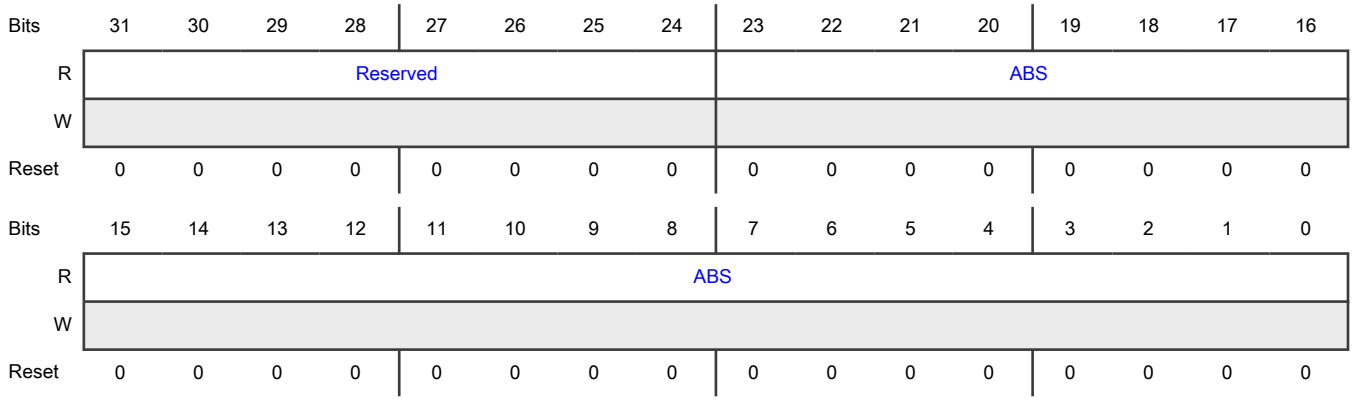
Offset

Register	Offset
MTL_TxQ1_ETS_Status	D54h

Function

The Queue 1 ETS Status register provides the average traffic transmitted in Queue 1.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-0 ABS	Average Bits per Slot This field contains the average transmitted bits per slot. If AV operation is enabled, this field is computed over number of slots, programmed in the SLC field of MTL_TxQ(#i)_ETS_CONTROL register. The maximum value of this field is 0x6_4000 in 100 Mbps, 0x3E_8000 in 1000 Mbps and 9C_4000 in 2500 Mbps mode respectively. When the DCB operation is enabled for Queue, this field is computed over every 10 million bit times slot (4 ms in 2500 Mbps; 10 ms in 1000 Mbps; 100 ms in 100 Mbps). The maximum value is 0x989680.

76.17.306 MTL Tx Queue 1 Quantum Weight (MTL_TxQ1_Quantum_Weight)

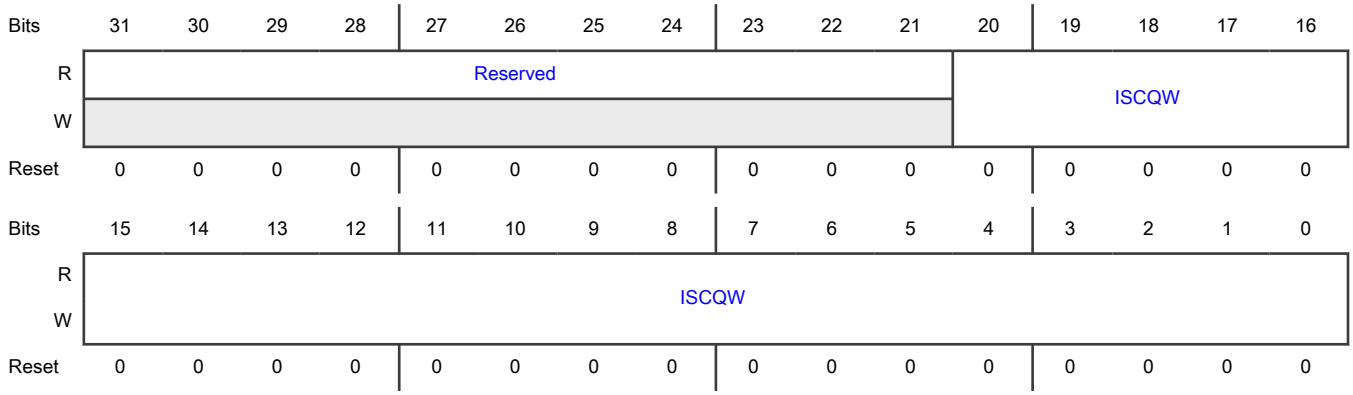
Offset

Register	Offset
MTL_TxQ1_Quantum_Weight	D58h

Function

The Queue 1 idleSlopeCredit, Quantum or Weights register provides the average traffic transmitted in Queue 1.

Diagram



Fields

Field	Function
31-21 —	Reserved
20-0 ISCQW	idleSlopeCredit, Quantum or Weights - idleSlopeCredit When AV feature is enabled, this field contains the idleSlopeCredit value required for the credit-based shaper algorithm for Queue 1. This is the rate of change of credit in bits per cycle (40 ns for 100 Mbps; 8 ns for 1000 Mbps; 3.2 ns for 2500 Mbps) when the credit is increasing. The software should program this field with computed credit in bits per cycle scaled by 1,024. The maximum value is portTransmitRate, that is, 0x2000 in 1000/2500 Mbps mode and 0x1000 in 100 Mbps mode. Bits[20:14] must be written to zero. - Quantum When the DCB operation is enabled with DWRR algorithm for Queue 1 traffic, this field contains the quantum value in bytes to be added to credit during every queue scanning cycle. The maximum value is 0x1312D0 bytes. - Weights When DCB operation is enabled with WFQ algorithm for Queue 1 traffic, this field contains the weight for this queue. The maximum value is 0x3FFF where weight of 0 indicates 100% bandwidth. Bits[20:14] must be written to zero. When DCB operation or generic queuing operation is enabled with WRR algorithm for Queue 1 traffic, this field contains the weight for this queue. The maximum value is 0x64. Bits [20:7] must be written to zero. - Note 1: In multiple Queue configuration this field in respective per queue register must be programmed to some non-zero value when multiple queues are enabled or single queue other than Q0 is enabled. This field need not be programmed when only Q0 is enabled. In general, when WRR algorithm is selected a non-zero value must be programmed on both Receive and Transmit. In Receive, the register is MTL_Operation_Mode register. - Note 2: For WFQ algorithm, higher the programmed weights lesser the bandwidth allocated for that Transmit Queue. The finish time is not a function of particular packet alone but it is as per the formula: (previous_finish_time of particular Transmit Queue + (weights*packet_size)) - Note 3: The weights programmed do not correspond to the number of packets but the fraction of bandwidth or time allocated for particular queue w.r.t. total BW or time.

76.17.307 MTL Tx Queue 1 Sendslope Credit (MTL_TxQ1_SendSlopeCredit)

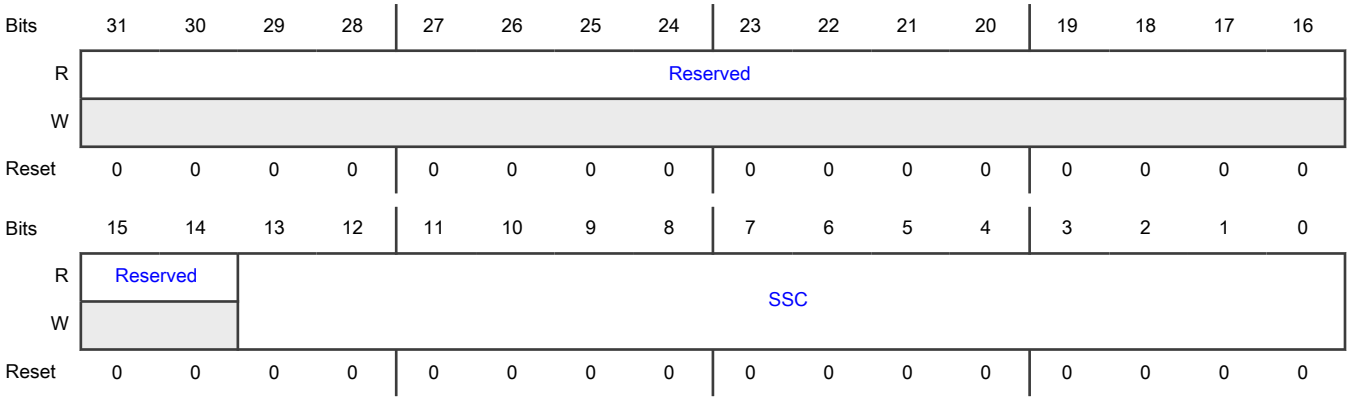
Offset

Register	Offset
MTL_TxQ1_SendSlopeCredit	D5Ch

Function

The sendSlopeCredit register contains the sendSlope credit value required for the credit-based shaper algorithm for the Queue.

Diagram



Fields

Field	Function
31-14 —	Reserved
13-0 SSC	sendSlopeCredit Value When AV operation is enabled, this field contains the sendSlopeCredit value required for credit-based shaper algorithm for Queue 1. This is the rate of change of credit in bits per cycle (40 ns, 8 ns and 3.2 ns for 100 Mbps, 1000 Mbps and 2500 Mbps respectively) when the credit is decreasing. The software should program this field with computed credit in bits per cycle scaled by 1,024. The maximum value is portTransmitRate, that is, 0x2000 in 1000/2500 Mbps mode and 0x1000 in 100 Mbps mode. This field should be programmed with absolute sendSlopeCredit value. The credit-based shaper logic subtracts it from the accumulated credit when Channel 1 is selected for transmission.

76.17.308 MTL Tx Queue 1 Hi Credit (MTL_TxQ1_HiCredit)

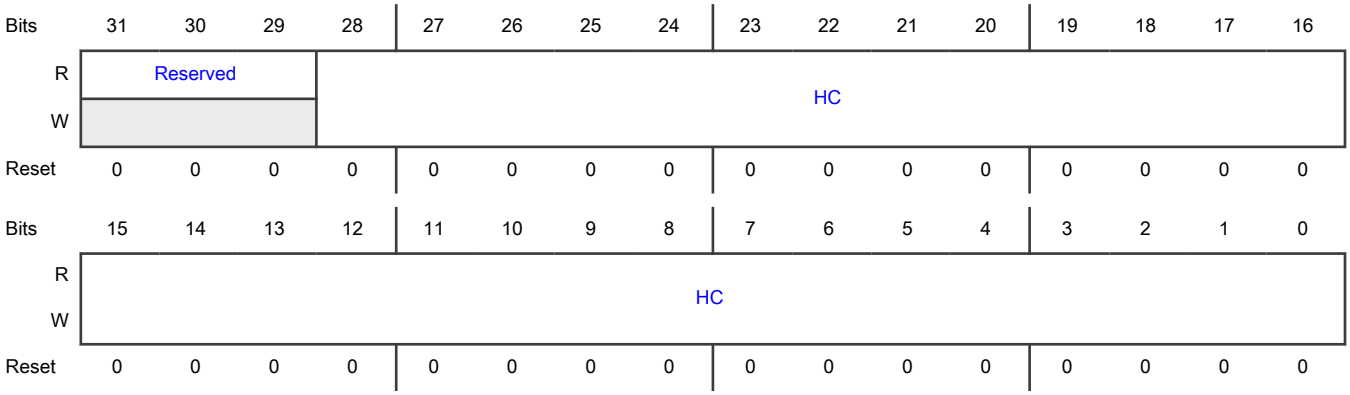
Offset

Register	Offset
MTL_TxQ1_HiCredit	D60h

Function

The hiCredit register contains the hiCredit value required for the credit-based shaper algorithm for the Queue.

Diagram



Fields

Field	Function
31-29 —	Reserved
28-0 HC	hiCredit Value When the AV feature is enabled, this field contains the hiCredit value required for the credit-based shaper algorithm. This is the maximum value that can be accumulated in the credit parameter. This is specified in bits scaled by 1,024. The maximum value is maxInterferenceSize, that is, best-effort maximum packet size (16,384 bytes or 131,072 bits). The value to be specified is 131,072 * 1,024 = 134,217,728 or 0x0800_0000.

76.17.309 MTL Tx Queue 1 Lo Credit (MTL_TxQ1_LoCredit)

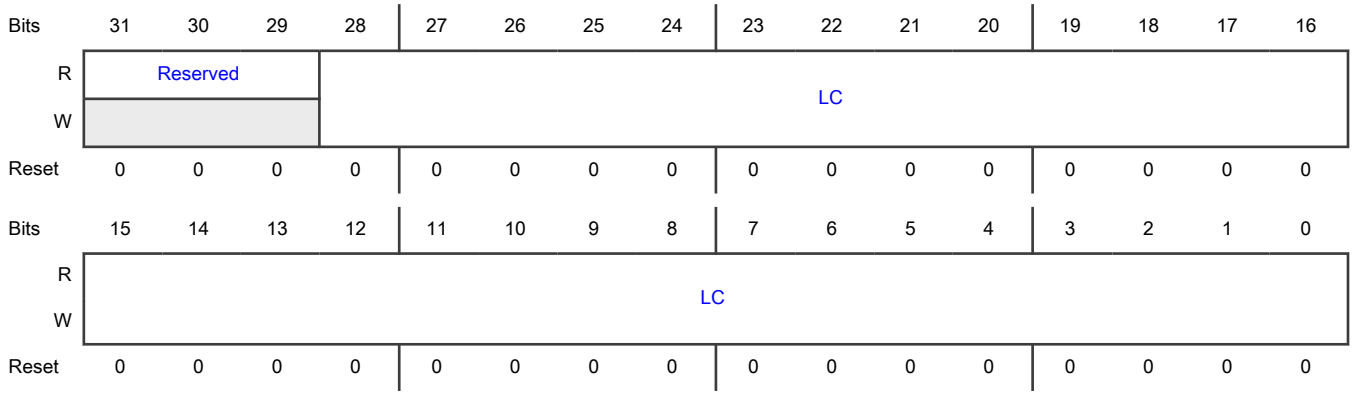
Offset

Register	Offset
MTL_TxQ1_LoCredit	D64h

Function

The loCredit register contains the loCredit value required for the credit-based shaper algorithm for the Queue.

Diagram



Fields

Field	Function
31-29 —	Reserved
28-0 LC	loCredit Value When AV operation is enabled, this field contains the loCredit value required for the credit-based shaper algorithm. This is the minimum value that can be accumulated in the credit parameter. This is specified in bits scaled by 1,024. The maximum value to be programmed is corresponds to twice the maxFrameSize transmitted from this queue. If the maxFrameSize is 8192 bytes, then (8192*2) * 8 * 1024 = 134,217,728 or 0x0800_0000. Because it is a negative value, the programmed value is 2's complement of the value, that is, 0x1800_0000.

76.17.310 MTL Queue 1 Interrupt Control Status (MTL_Q1_Interrupt_Control_Status)

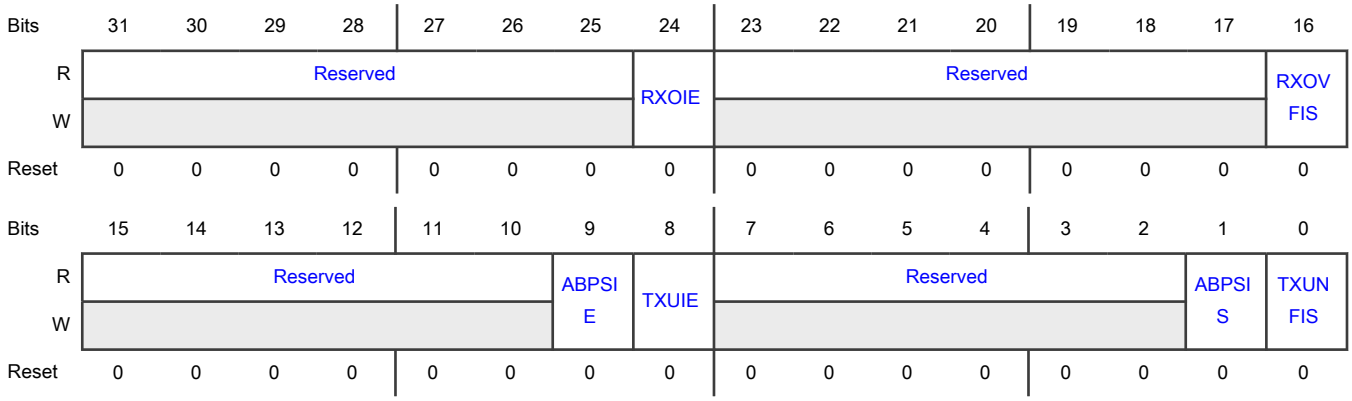
Offset

Register	Offset
MTL_Q1_Interrupt_Control_Status	D6Ch

Function

This register contains the interrupt enable and status bits for the queue 1 interrupts.

Diagram



Fields

Field	Function
31-25 —	Reserved
24 RXOIE	Receive Queue Overflow Interrupt Enable When this bit is set, the Receive Queue Overflow interrupt is enabled. When this bit is reset, the Receive Queue Overflow interrupt is disabled. 0b - Receive Queue Overflow Interrupt is disabled 1b - Receive Queue Overflow Interrupt is enabled
23-17 —	Reserved
16 RXOVFIS	Receive Queue Overflow Interrupt Status This bit indicates that the Receive Queue had an overflow while receiving the packet. If a partial packet is transferred to the application, the overflow status is set in RDES3[21]. This bit is cleared when the application writes 1 to this bit. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **NOTE** This field is not testable by an automated register test. 0b - Receive Queue Overflow Interrupt Status not detected 1b - Receive Queue Overflow Interrupt Status detected
15-10 —	Reserved
9 ABPSIE	Average Bits Per Slot Interrupt Enable When this bit is set, the MAC asserts the sbd_intr_o or mci_intr_o interrupt when the average bits per slot status is updated. When this bit is cleared, the interrupt is not asserted for such an event.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Average Bits Per Slot Interrupt is disabled 1b - Average Bits Per Slot Interrupt is enabled
8 TXUIE	Transmit Queue Underflow Interrupt Enable When this bit is set, the Transmit Queue Underflow interrupt is enabled. When this bit is reset, the Transmit Queue Underflow interrupt is disabled. 0b - Transmit Queue Underflow Interrupt Status is disabled 1b - Transmit Queue Underflow Interrupt Status is enabled
7-2 —	Reserved
1 ABPSIS	Average Bits Per Slot Interrupt Status When set, this bit indicates that the MAC has updated the ABS value. This bit is cleared when the application writes 1 to this bit. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **NOTE** This field is not testable by an automated register test. 0b - Average Bits Per Slot Interrupt Status not detected 1b - Average Bits Per Slot Interrupt Status detected
0 TXUNFIS	Transmit Queue Underflow Interrupt Status This bit indicates that the Transmit Queue had an underflow while transmitting the packet. Transmission is suspended and an Underflow Error TDES3[2] is set. This bit is cleared when the application writes 1 to this bit. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **NOTE** This field is not testable by an automated register test. 0b - Transmit Queue Underflow Interrupt Status not detected 1b - Transmit Queue Underflow Interrupt Status detected

76.17.311 MTL Rx Queue 1 Operation Mode (MTL_RxQ1_Operation_Mode)

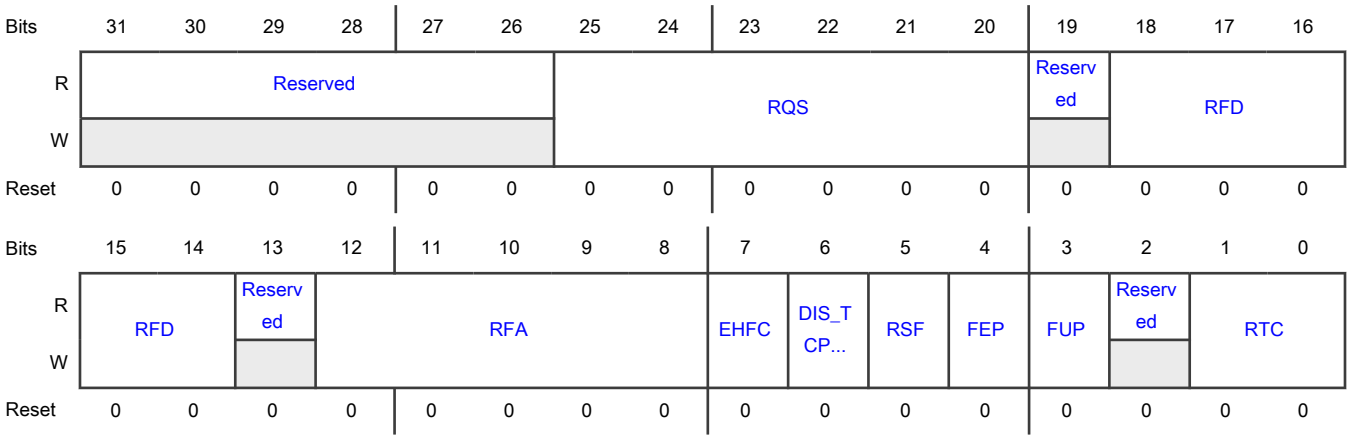
Offset

Register	Offset
MTL_RxQ1_Operation_Mode	D70h

Function

The Queue 1 Receive Operation Mode register establishes the Receive queue operating modes and command. The RFA and RFD fields are not backward compatible with the RFA and RFD fields of 4.00a release

Diagram



Fields

Field	Function
31-26 —	Reserved
25-20 RQS	Receive Queue Size This field indicates the size of the allocated Receive queues in blocks of 256 bytes. The RQS field is read-write only if the number of Rx Queues more than one, the reset value is 0x0 and indicates size of 256 bytes. This means that value of 0x0 = 256 bytes, 0x1 = 512 bytes and so on. So, program RQS [5:0] = 6'b001111 to allocate queue size of 4096 (4K) bytes. In general, the size of the Queue = (RQS+1)*256 bytes. When the number of Rx Queues is one, the field is read-only and the configured RX FIFO size in blocks of 256 bytes is reflected in the reset value. The width of this field depends on the Rx memory size selected in your configuration. For example, if the memory size is 2048, the width of this field is 3 bits: LOG2(2048/256) = LOG2(8) = 3 bits
19 —	Reserved
18-14 RFD	Threshold for Deactivating Flow Control (in half-duplex and full-duplex modes) These bits control the threshold (fill-level of Rx queue) at which the flow control is de-asserted after activation: - 0: Full minus 1 KB, that is, FULL 1 KB - 1: Full minus 1.5 KB, that is, FULL 1.5 KB - 2: Full minus 2 KB, that is, FULL 2 KB - 3: Full minus 2.5 KB, that is, FULL 2.5 KB - ... - 30: Full minus 16 KB, that is, FULL 16 KB - 31: Full minus 16.5 KB, that is, FULL 16.5 KB The de-assertion is effective only after flow control is asserted. Note: The value must be programmed in such a way to make sure that the threshold is a positive number. When the EHFC is set high, these values are applicable only when the Rx queue size determined by the RQS field of this register, is equal to or greater than 4 KB. For a given queue size, the values ranges between 0 and the encoding for FULL minus (QSIZE - 0.5 KB) and all other values are illegal. Here the term FULL and QSIZE refers to the queue size determined by the RQS

Table continues on the next page...

Table continued from the previous page...

Field	Function
	field of this register. The width of this field depends on RX FIFO size selected during the configuration. Remaining bits are reserved and read only.
13 —	Reserved
12-8 RFA	Threshold for Activating Flow Control (in half-duplex and full-duplex These bits control the threshold (fill-level of Rx queue) at which the flow control is activated: For more information on encoding for this field, see RFD.
7 EHFC	Enable Hardware Flow Control When this bit is set, the flow control signal operation, based on the fill-level of Rx queue, is enabled. When reset, the flow control operation is disabled. 0b - Hardware Flow Control is disabled 1b - Hardware Flow Control is enabled
6 DIS_TCP_EF	Disable Dropping of TCP/IP Checksum Error Packets When this bit is set, the MAC does not drop the packets which only have the errors detected by the Receive Checksum Offload engine. Such packets have errors only in the encapsulated payload. There are no errors (including FCS error) in the Ethernet packet received by the MAC. When this bit is reset, all error packets are dropped if the FEP bit is reset. 0b - Dropping of TCP/IP Checksum Error Packets is enabled 1b - Dropping of TCP/IP Checksum Error Packets is disabled
5 RSF	Receive Queue Store and Forward When this bit is set, the DWC_ether_qos reads a packet from the Rx queue only after the complete packet has been written to it, ignoring the RTC field of this register. When this bit is reset, the Rx queue operates in the Threshold (cut-through) mode, subject to the threshold specified by the RTC field of this register. 0b - Receive Queue Store and Forward is disabled 1b - Receive Queue Store and Forward is enabled
4 FEP	Forward Error Packets When this bit is reset, the Rx queue drops packets with error status (CRC error, GMII_ER, watchdog timeout, or overflow). However, if the start byte (write) pointer of a packet is already transferred to the read controller side (in Threshold mode), the packet is not dropped. When this bit is set, all packets except the runt error packets are forwarded to the application or DMA. If the RSF bit is set and the Rx queue overflows when a partial packet is written, the packet is dropped irrespective of the setting of this bit. However, if the RSF bit is reset and the Rx queue overflows when a partial packet is written, a partial packet might be forwarded to the application or DMA. 0b - Forward Error Packets is disabled 1b - Forward Error Packets is enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
3 FUP	Forward Undersized Good Packets When this bit is set, the Rx queue forwards the undersized good packets (packets with no error and length less than 64 bytes), including pad-bytes and CRC. When this bit is reset, the Rx queue drops all packets of less than 64 bytes, unless a packet is already transferred because of the lower value of Rx Threshold, for example, RTC = 01. 0b - Forward Undersized Good Packets is disabled 1b - Forward Undersized Good Packets is enabled
2 —	Reserved
1-0 RTC	Receive Queue Threshold Control These bits control the threshold level of the MTL Rx queue (in bytes): The received packet is transferred to the application or DMA when the packet size within the MTL Rx queue is larger than the threshold. In addition, full packets with length less than the threshold are automatically transferred. This field is valid only when the RSF bit is zero. This field is ignored when the RSF bit is set to 1. 00b - 64 01b - 32 10b - 96 11b - 128

76.17.312 MTL Rx Queue 1 Missed Packet Overflow Counter (MTL_RxQ1_Missed_Packet_Overflow_Cnt)

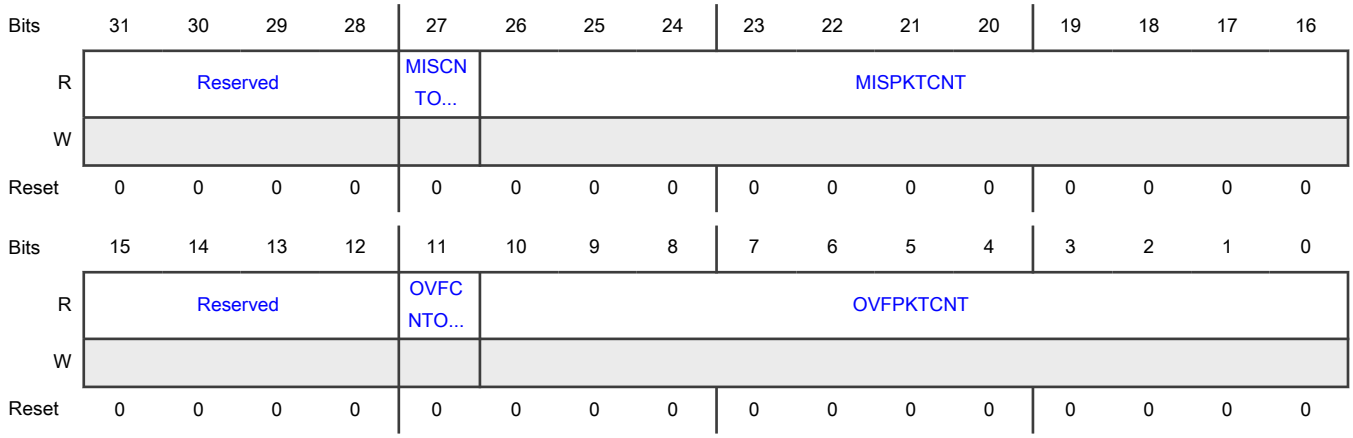
Offset

Register	Offset
MTL_RxQ1_Missed_Packet_Overflow_Cnt	D74h

Function

The Queue 1 Missed Packet and Overflow Counter register contains the counter for packets missed because of Receive queue packet flush and packets discarded because of Receive queue overflow.

Diagram



Fields

Field	Function
31-28 —	Reserved
27 MISCNTOVF	Missed Packet Counter Overflow Bit When set, this bit indicates that the Rx Queue Missed Packet Counter crossed the maximum limit. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - Missed Packet Counter overflow not detected 1b - Missed Packet Counter overflow detected
26-16 MISPKTCNT	Missed Packet Counter This field indicates the number of packets missed by the DWC_ether_qos because the application asserted ari_pkt_flush_i[] for this queue. This counter is reset when this register is read with mci_be_i[0] at 1b1. This counter is incremented by 1 when the DMA discards the packet because of buffer unavailability. Access restriction applies. Clears on read. Self-set to 1 on internal event.
15-12 —	Reserved
11 OVFCNTOVF	Overflow Counter Overflow Bit When set, this bit indicates that the Rx Queue Overflow Packet Counter field crossed the maximum limit. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - Overflow Counter overflow not detected 1b - Overflow Counter overflow detected
10-0 OVFPKTCNT	Overflow Packet Counter This field indicates the number of packets discarded by the DWC_ether_qos because of Receive queue overflow. This counter is incremented each time the DWC_ether_qos discards an incoming packet because of overflow. This counter is reset when this register is read with mci_be_i[0] at 1'b1. Access restriction applies. Clears on read. Self-set to 1 on internal event.

76.17.313 MTL Rx Queue 1 Debug (MTL_RxQ1_Debug)

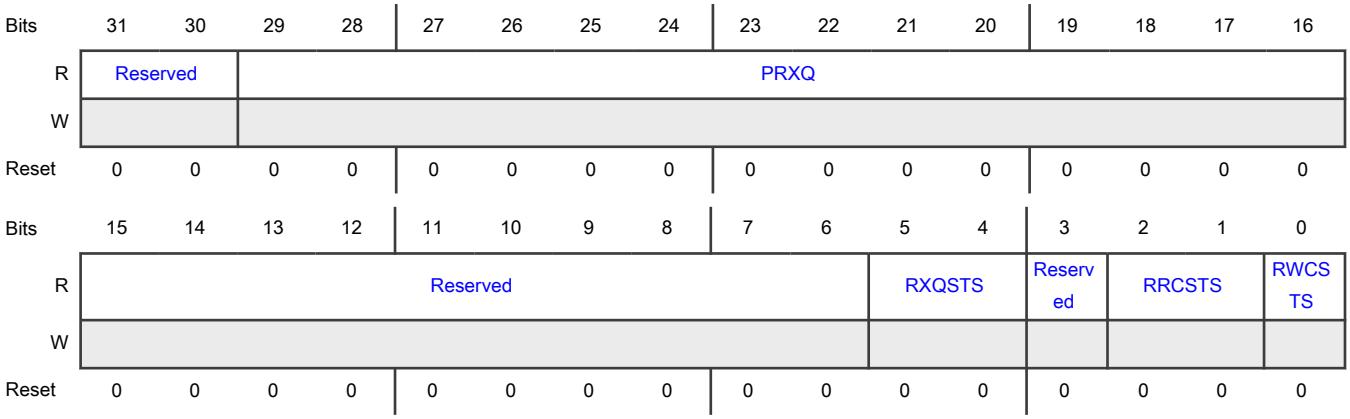
Offset

Register	Offset
MTL_RxQ1_Debug	D78h

Function

The Queue 1 Receive Debug register gives the debug status of various blocks related to the Receive queue.

Diagram



Fields

Field	Function
31-30 —	Reserved
29-16 PRXQ	Number of Packets in Receive Queue This field indicates the current number of packets in the Rx Queue. The theoretical maximum value for this field is 256KB/16B = 16K Packets, that is, Max_Queue_Size/Min_Packet_Size.
15-6 —	Reserved
5-4 RXQSTS	MTL Rx Queue Fill-Level Status This field gives the status of the fill-level of the Rx Queue: 00b - Rx Queue empty 01b - Rx Queue fill-level below flow-control deactivate threshold 10b - Rx Queue fill-level above flow-control activate threshold 11b - Rx Queue full
3	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
2-1 RRCSTS	MTL Rx Queue Read Controller State This field gives the state of the Rx queue Read controller: 00b - Idle state 01b - Reading packet data 10b - Reading packet status (or timestamp) 11b - Flushing the packet data and status
0 RWCSTS	MTL Rx Queue Write Controller Active Status When high, this bit indicates that the MTL Rx queue Write controller is active, and it is transferring a received packet to the Rx Queue. 0b - MTL Rx Queue Write Controller Active Status not detected 1b - MTL Rx Queue Write Controller Active Status detected

76.17.314 MTL Rx Queue 1 Control (MTL_RxQ1_Control)

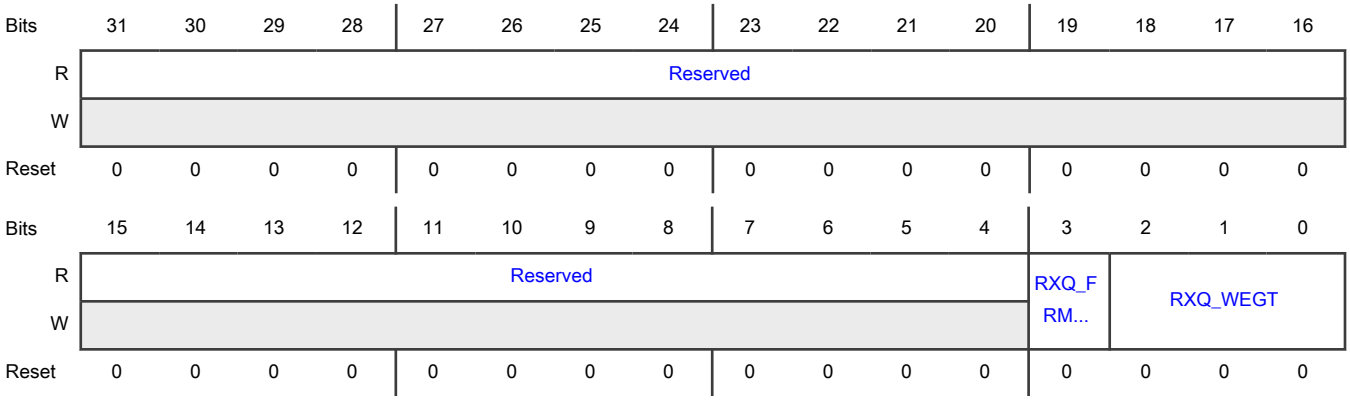
Offset

Register	Offset
MTL_RxQ1_Control	D7Ch

Function

The Queue Receive Control register controls the receive arbitration and passing of received packets to the application.

Diagram



Fields

Field	Function
31-4 —	Reserved
3 RXQ_FRM_ARBIT	Receive Queue Packet Arbitration When this bit is set, the DWC_ether_qos drives the packet data to the ARI interface such that the entire packet data of currently-selected queue is transmitted before switching to other queue. When this bit is reset, the DWC_ether_qos drives the packet data to the ARI interface such that the following amount of data of currently-selected queue is transmitted before switching to other queue: - PBL amount of data (indicated by ari_qN_pbl_i[]) or - Complete data of a packet The status and the timestamp are not a part of the PBL data. Therefore, the DWC_ether_qos drives the complete status (including timestamp status) during first PBL request for the packet (in store-and-forward mode) or the last PBL request for the packet (in Threshold mode). 0b - Receive Queue Packet Arbitration is disabled 1b - Receive Queue Packet Arbitration is enabled
2-0 RXQ_WEGT	Receive Queue Weight This field indicates the weight assigned to the Rx Queue 0. This field needs to be programmed with one value less than the required weight, that is, reset value of 0 indicates weight of 1, value of 1 indicates weight of 2, and so on. The weight is used as the number of continuous PBL or packets requests (depending on the RXQ_FRM_ARBIT) allocated to the queue in one arbitration cycle. Note: The change in value of RXQ_WEGT takes effect only after the completion of current service round or when there is change from RAA=SP to RAA=WSP algorithm. This approach is taken so that there is smooth transition. For the RXQ_WEGT value to take effect at the start, the MTL_RxQ(#i)_Control registers must be programmed before the MTL_Operation_Mode register.

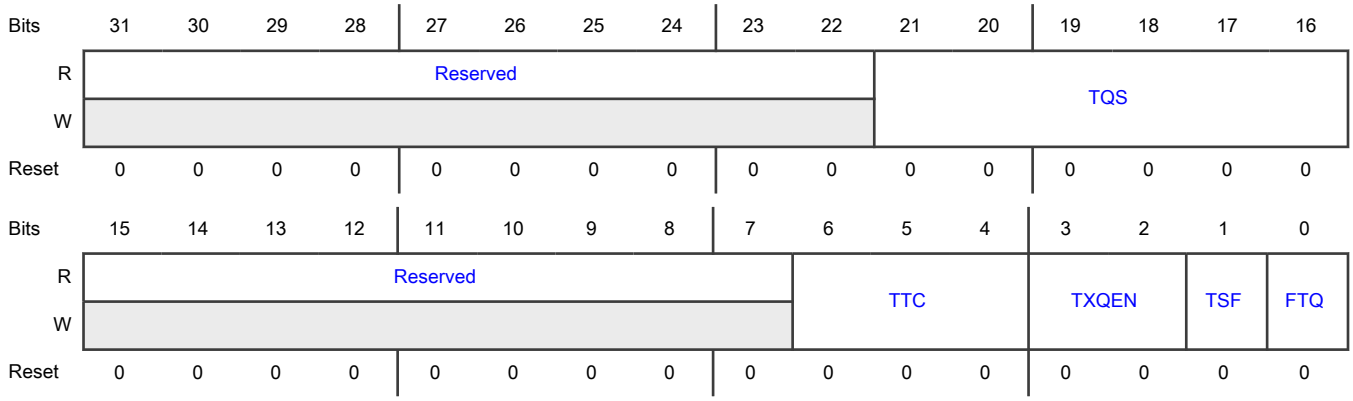
76.17.315 MTL Tx Queue 2 Operation Mode (MTL_TxQ2_Operation_Mode)

Offset

Register	Offset
MTL_TxQ2_Operation_Mode	D80h

Function
The Queue 2 Transmit Operation Mode register establishes the Transmit queue operating modes and commands.

Diagram



Fields

Field	Function
31-22 —	Reserved
21-16 TQS	Transmit Queue Size This field indicates the size of the allocated Transmit queues in blocks of 256 bytes. The TQS field is read-write only if the number of Tx Queues more than one, the reset value is 0x0 and indicates size of 256 bytes. This means that value of 0x0 = 256 bytes, 0x1 = 512 bytes and so on. So, program TQS [5:0] = 6'b001111 to allocate queue size of 4096 (4K) bytes. In general, the size of the Queue = (TQS+1)*256 bytes. When the number of Tx Queues is one, the field is read-only and the configured TX FIFO size in blocks of 256 bytes is reflected in the reset value. The width of this field depends on the Tx memory size selected in your configuration. For example, if the memory size is 2048, the width of this field is 3 bits: LOG2(2048/256) = LOG2(8) = 3 bits
15-7 —	Reserved
6-4 TTC	Transmit Threshold Control These bits control the threshold level of the MTL Tx Queue. The transmission starts when the packet size within the MTL Tx Queue is larger than the threshold. In addition, full packets with length less than the threshold are also transmitted. These bits are used only when the TSF bit is reset. 000b - 32 001b - 64 010b - 96 011b - 128 100b - 192 101b - 256 110b - 384 111b - 512

Table continues on the next page...

Table continued from the previous page...

Field	Function
3-2 TXQEN	Transmit Queue Enable This field is used to enable/disable the transmit queue 0. - 2'b00: Not enabled - 2'b01: Enable in AV mode - 2'b10: Enabled - 2'b11: Reserved Note: In multiple Tx queues configuration, all the queues are disabled by default. Enable the Tx queue by programming this field. 00b - Not enabled 01b - Enable in AV mode (Reserved in non-AV) 10b - Enabled 11b - Reserved
1 TSF	Transmit Store and Forward When this bit is set, the transmission starts when a full packet resides in the MTL Tx queue. When this bit is set, the TTC values specified in Bits[6:4] of this register are ignored. This bit should be changed only when the transmission is stopped. 0b - Transmit Store and Forward is disabled 1b - Transmit Store and Forward is enabled
0 FTQ	Flush Transmit Queue When this bit is set, the Tx queue controller logic is reset to its default values. Therefore, all the data in the Tx queue is lost or flushed. This bit is internally reset when the flushing operation is complete. Until this bit is reset, you should not write to the MTL_TxQ1_Operation_Mode register. The data which is already accepted by the MAC transmitter is not flushed. It is scheduled for transmission and results in underflow and runt packet transmission. Note: The flush operation is complete only when the Tx queue is empty and the application has accepted the pending Tx Status of all transmitted packets. To complete this flush operation, the PHY Tx clock (CLK_TX_I) should be active. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. 0b - Flush Transmit Queue is disabled 1b - Flush Transmit Queue is enabled

76.17.316 MTL Tx Queue 2 Underflow (MTL_TxQ2_Underflow)

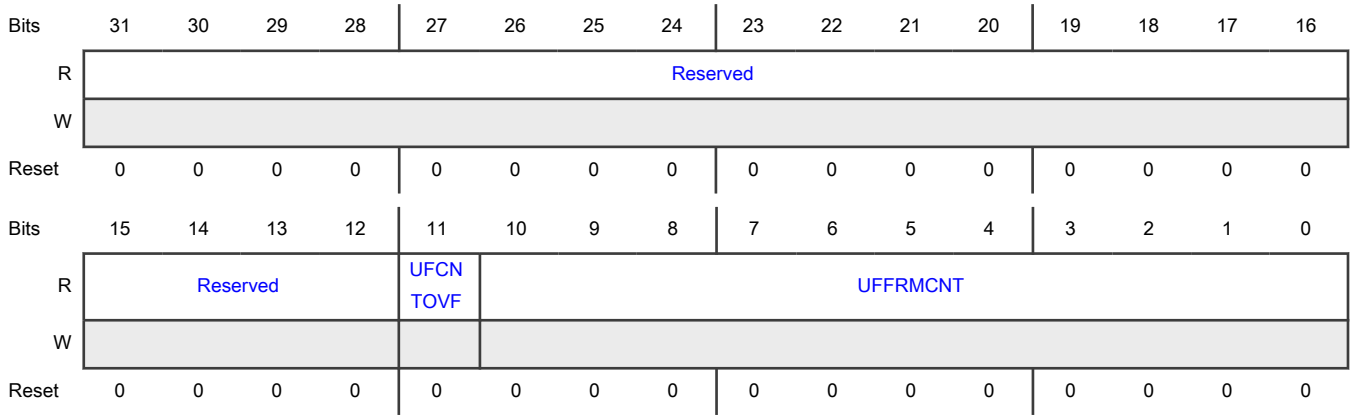
Offset

Register	Offset
MTL_TxQ2_Underflow	D84h

Function

The Queue 2 Underflow Counter register contains the counter for packets aborted because of Transmit queue underflow and packets missed because of Receive queue packet flush

Diagram



Fields

Field	Function
31-12 —	Reserved
11 UFCNTOVF	Overflow Bit for Underflow Packet Counter This bit is set every time the Tx queue Underflow Packet Counter field overflows, that is, it has crossed the maximum count. In such a scenario, the overflow packet counter is reset to all-zeros and this bit indicates that the rollover happened. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - Overflow not detected for Underflow Packet Counter 1b - Overflow detected for Underflow Packet Counter
10-0 UFFRMCNT	Underflow Packet Counter This field indicates the number of packets aborted by the controller because of Tx Queue Underflow. This counter is incremented each time the MAC aborts outgoing packet because of underflow. The counter is cleared when this register is read with mci_be_i[0] at 1'b1. Access restriction applies. Clears on read. Self-set to 1 on internal event.

76.17.317 MTL Tx Queue 2 Debug (MTL_TxQ2_Debug)

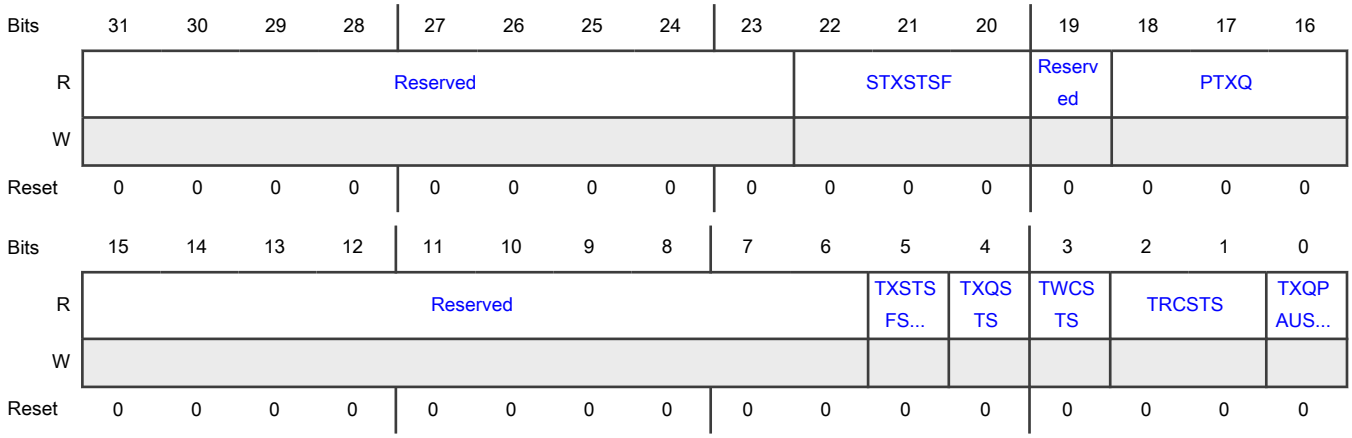
Offset

Register	Offset
MTL_TxQ2_Debug	D88h

Function

The Queue 2 Transmit Debug register gives the debug status of various blocks related to the Transmit queue.

Diagram



Fields

Field	Function
31-23 —	Reserved
22-20 STXSTSF	Number of Status Words in Tx Status FIFO of Queue This field indicates the current number of status in the Tx Status FIFO of this queue. When the DTXSTS bit of MTL_Operation_Mode register is set to 1, this field does not reflect the number of status words in Tx Status FIFO.
19 —	Reserved
18-16 PTXQ	Number of Packets in the Transmit Queue This field indicates the current number of packets in the Tx Queue. When the DTXSTS bit of MTL_Operation_Mode register is set to 1, this field does not reflect the number of packets in the Transmit queue.
15-6 —	Reserved
5 TXSTSFSTS	MTL Tx Status FIFO Full Status When high, this bit indicates that the MTL Tx Status FIFO is full. Therefore, the MTL cannot accept any more packets for transmission. 0b - MTL Tx Status FIFO Full status is not detected 1b - MTL Tx Status FIFO Full status is detected
4 TXQSTS	MTL Tx Queue Not Empty Status When this bit is high, it indicates that the MTL Tx Queue is not empty and some data is left for transmission.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - MTL Tx Queue Not Empty status is not detected 1b - MTL Tx Queue Not Empty status is detected
3 TWCSTS	MTL Tx Queue Write Controller Status When high, this bit indicates that the MTL Tx Queue Write Controller is active, and it is transferring the data to the Tx Queue. 0b - MTL Tx Queue Write Controller status is not detected 1b - MTL Tx Queue Write Controller status is detected
2-1 TRCSTS	MTL Tx Queue Read Controller Status This field indicates the state of the Tx Queue Read Controller: 00b - Idle state 01b - Read state (transferring data to the MAC transmitter) 10b - Waiting for pending Tx Status from the MAC transmitter 11b - Flushing the Tx queue because of the Packet Abort request from the MAC
0 TXQPAUSED	Transmit Queue in Pause When this bit is high and the Rx flow control is enabled, it indicates that the Tx Queue is in the Pause condition (in the full-duplex only mode) because of the following: - Reception of the PFC packet for the priorities assigned to the Tx Queue when PFC is enabled - Reception of 802.3x Pause packet when PFC is disabled 0b - Transmit Queue in Pause status is not detected 1b - Transmit Queue in Pause status is detected

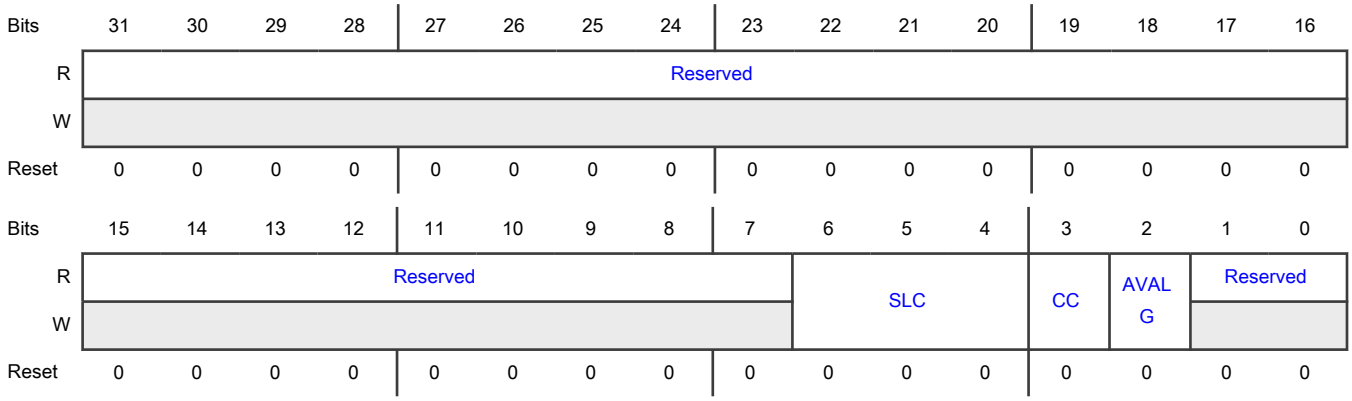
76.17.318 MTL Tx Queue 2 ETS Control (MTL_TxQ2_ETS_Control)

Offset

Register	Offset
MTL_TxQ2_ETS_Control	D90h

Function
The Queue ETS Control register controls the enhanced transmission selection operation.

Diagram



Fields

Field	Function
31-7 —	Reserved
6-4 SLC	Slot Count If the credit-based shaper algorithm is enabled, the software can program the number of slots (of duration programmed in DMA_CH(#i)_Slot_Interval register) over which the average transmitted bits per slot, provided in the MTL_TxQ(#i)_ETS_Status register, need to be computed for Queue. The encoding is as follows: 000b - 1 slot 001b - 2 slots 010b - 4 slots 011b - 8 slots 100b - 16 slots 101b - Reserved
3 CC	Credit Control When this bit is set, the accumulated credit parameter in the credit-based shaper algorithm logic is not reset to zero when there is positive credit and no packet to transmit in Channel 1. The credit accumulates even when there is no packet waiting in Channel 1 and another channel is transmitting. When this bit is reset, the accumulated credit parameter in the credit-based shaper algorithm logic is set to zero when there is positive credit and no packet to transmit in Channel 1. When there is no packet waiting in Channel 1 and other channel is transmitting, no credit is accumulated. 0b - Credit Control is disabled 1b - Credit Control is enabled
2 AVALG	AV Algorithm

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When Queue 1 is programmed for AV, this field configures the scheduling algorithm for this queue: This bit when set, indicates credit based shaper algorithm (CBS) is selected for Queue 1 traffic. When reset, strict priority is selected. 0b - CBS Algorithm is disabled 1b - CBS Algorithm is enabled
1-0 —	Reserved

76.17.319 MTL Tx Queue 2 ETS Status (MTL_TxQ2_ETS_Status)

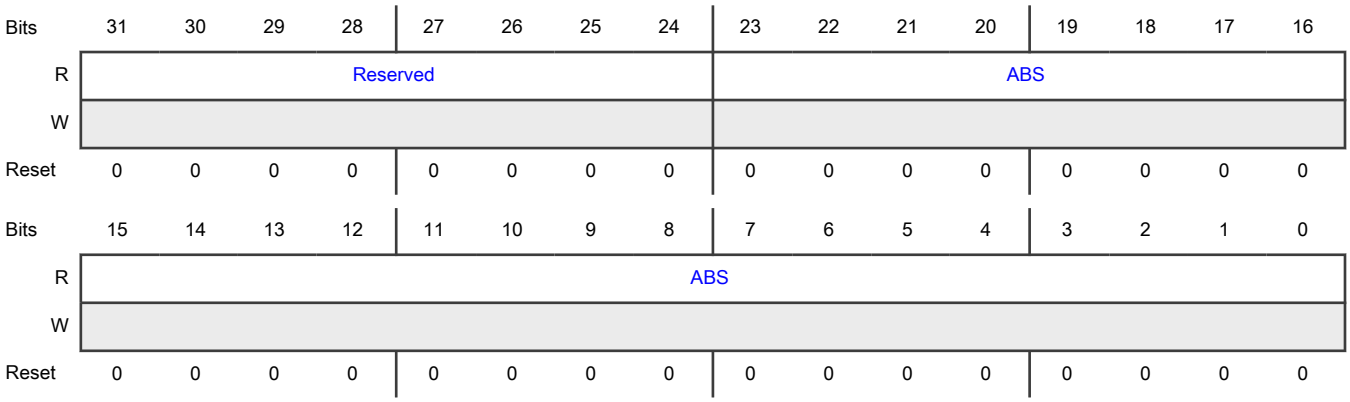
Offset

Register	Offset
MTL_TxQ2_ETS_Status	D94h

Function

The Queue 2 ETS Status register provides the average traffic transmitted in Queue 2.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-0 ABS	Average Bits per Slot

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field contains the average transmitted bits per slot. If AV operation is enabled, this field is computed over number of slots, programmed in the SLC field of MTL_TxQ(#i)_ETS_CONTROL register. The maximum value of this field is 0x6_4000 in 100 Mbps, 0x3E_8000 in 1000 Mbps and 9C_4000 in 2500 Mbps mode respectively. When the DCB operation is enabled for Queue, this field is computed over every 10 million bit times slot (4 ms in 2500 Mbps; 10 ms in 1000 Mbps; 100 ms in 100 Mbps). The maximum value is 0x989680.

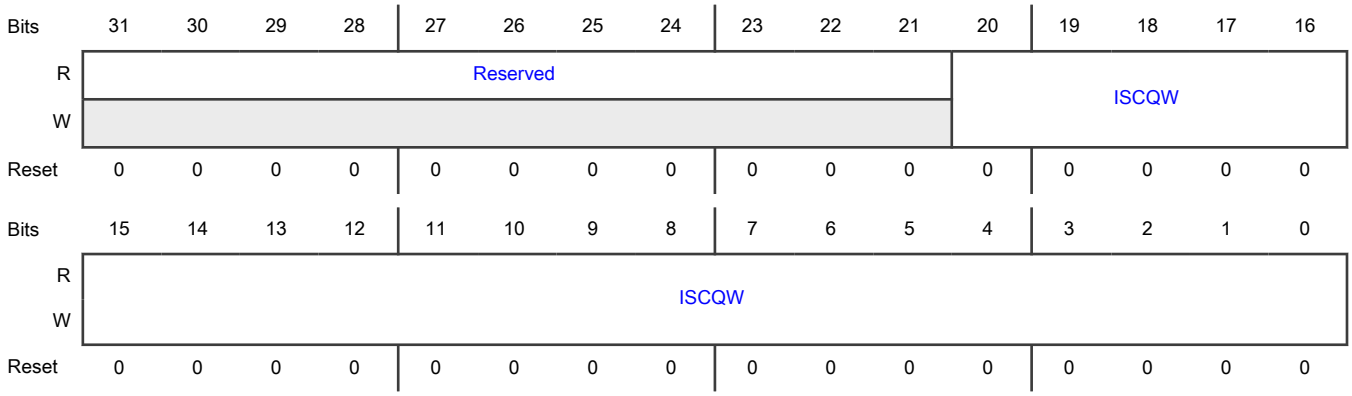
76.17.320 MTL Tx Queue 2 Quantum Weight (MTL_TxQ2_Quantum_Weight)

Offset

Register	Offset
MTL_TxQ2_Quantum_Weight	D98h

Function
The Queue 2 idleSlopeCredit, Quantum or Weights register provides the average traffic transmitted in Queue 2.

Diagram



Fields

Field	Function
31-21 —	Reserved
20-0 ISCQW	idleSlopeCredit, Quantum or Weights - idleSlopeCredit When AV feature is enabled, this field contains the idleSlopeCredit value required for the credit-based shaper algorithm for Queue 1. This is the rate of change of credit in bits per cycle (40 ns for 100 Mbps; 8 ns for 1000 Mbps; 3.2 ns for 2500 Mbps) when the credit is increasing. The software should program this field with computed credit in bits per cycle scaled by 1,024. The maximum value is

Table continues on the next page...

Table continued from the previous page...

Field	Function
	portTransmitRate, that is, 0x2000 in 1000/2500 Mbps mode and 0x1000 in 100 Mbps mode. Bits[20:14] must be written to zero. - Quantum When the DCB operation is enabled with DWRR algorithm for Queue 1 traffic, this field contains the quantum value in bytes to be added to credit during every queue scanning cycle. The maximum value is 0x1312D0 bytes. - Weights When DCB operation is enabled with WFQ algorithm for Queue 1 traffic, this field contains the weight for this queue. The maximum value is 0x3FFF where weight of 0 indicates 100% bandwidth. Bits[20:14] must be written to zero. When DCB operation or generic queuing operation is enabled with WRR algorithm for Queue 1 traffic, this field contains the weight for this queue. The maximum value is 0x64. Bits [20:7] must be written to zero. - Note 1: In multiple Queue configuration this field in respective per queue register must be programmed to some non-zero value when multiple queues are enabled or single queue other than Q0 is enabled. This field need not be programmed when only Q0 is enabled. In general, when WRR algorithm is selected a non-zero value must be programmed on both Receive and Transmit. In Receive, the register is MTL_Operation_Mode register. - Note 2: For WFQ algorithm, higher the programmed weights lesser the bandwidth allocated for that Transmit Queue. The finish time is not a function of particular packet alone but it is as per the formula: (previous_finish_time of particular Transmit Queue + (weights*packet_size)) - Note 3: The weights programmed do not correspond to the number of packets but the fraction of bandwidth or time allocated for particular queue w.r.t. total BW or time.

76.17.321 MTL Tx Queue 2 SendSlope Credit (MTL_TxQ2_SendSlopeCredit)

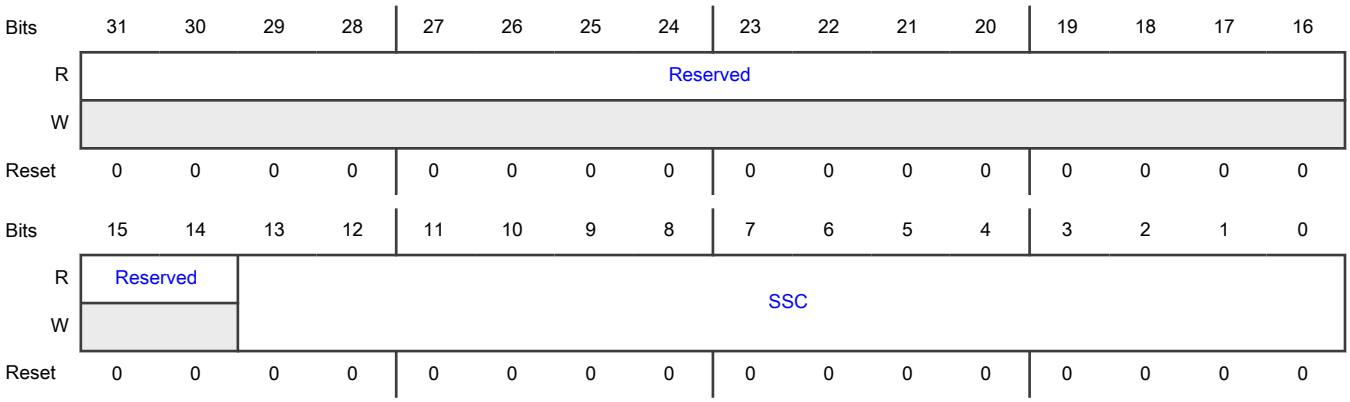
Offset

Register	Offset
MTL_TxQ2_SendSlopeCredit	D9Ch

Function

The sendSlopeCredit register contains the sendSlope credit value required for the credit-based shaper algorithm for the Queue.

Diagram



Fields

Field	Function
31-14 —	Reserved
13-0 SSC	sendSlopeCredit Value When AV operation is enabled, this field contains the sendSlopeCredit value required for credit-based shaper algorithm for Queue 1. This is the rate of change of credit in bits per cycle (40 ns, 8 ns and 3.2 ns for 100 Mbps, 1000 Mbps and 2500 Mbps respectively) when the credit is decreasing. The software should program this field with computed credit in bits per cycle scaled by 1,024. The maximum value is portTransmitRate, that is, 0x2000 in 1000/2500 Mbps mode and 0x1000 in 100 Mbps mode. This field should be programmed with absolute sendSlopeCredit value. The credit-based shaper logic subtracts it from the accumulated credit when Channel 1 is selected for transmission.

76.17.322 MTL Tx Queue 2 Hi Credit (MTL_TxQ2_HiCredit)

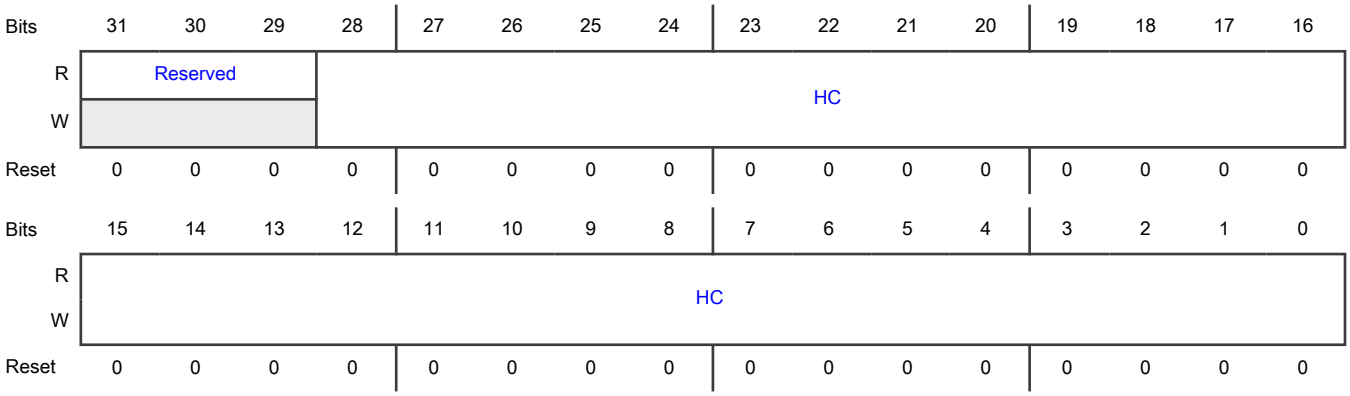
Offset

Register	Offset
MTL_TxQ2_HiCredit	DA0h

Function

The hiCredit register contains the hiCredit value required for the credit-based shaper algorithm for the Queue.

Diagram



Fields

Field	Function
31-29 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
28-0 HC	hiCredit Value When the AV feature is enabled, this field contains the hiCredit value required for the credit-based shaper algorithm. This is the maximum value that can be accumulated in the credit parameter. This is specified in bits scaled by 1,024. The maximum value is maxInterferenceSize, that is, best-effort maximum packet size (16,384 bytes or 131,072 bits). The value to be specified is 131,072 * 1,024 = 134,217,728 or 0x0800_0000.

76.17.323 MTL Tx Queue 2 Lo Credit (MTL_TxQ2_LoCredit)

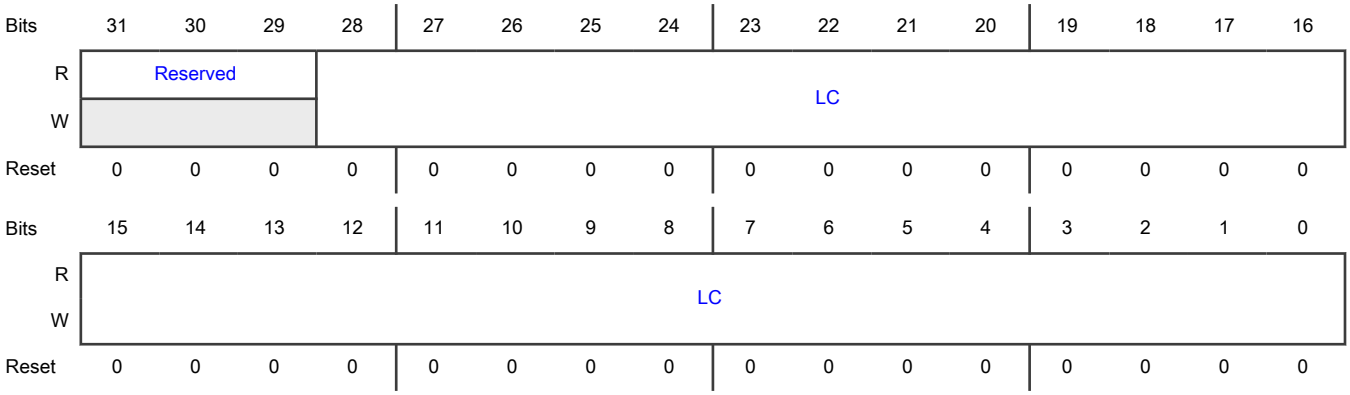
Offset

Register	Offset
MTL_TxQ2_LoCredit	DA4h

Function

The loCredit register contains the loCredit value required for the credit-based shaper algorithm for the Queue.

Diagram



Fields

Field	Function
31-29 —	Reserved
28-0 LC	loCredit Value When AV operation is enabled, this field contains the loCredit value required for the credit-based shaper algorithm. This is the minimum value that can be accumulated in the credit parameter. This is specified in bits scaled by 1,024. The maximum value to be programmed is corresponds to twice the maxFrameSize transmitted from this queue. If the maxFrameSize is 8192 bytes, then (8192*2) * 8 * 1024 = 134,217,728

Table continues on the next page...

Table continued from the previous page...

Field	Function
	or 0x0800_0000. Because it is a negative value, the programmed value is 2's complement of the value, that is, 0x1800_0000.

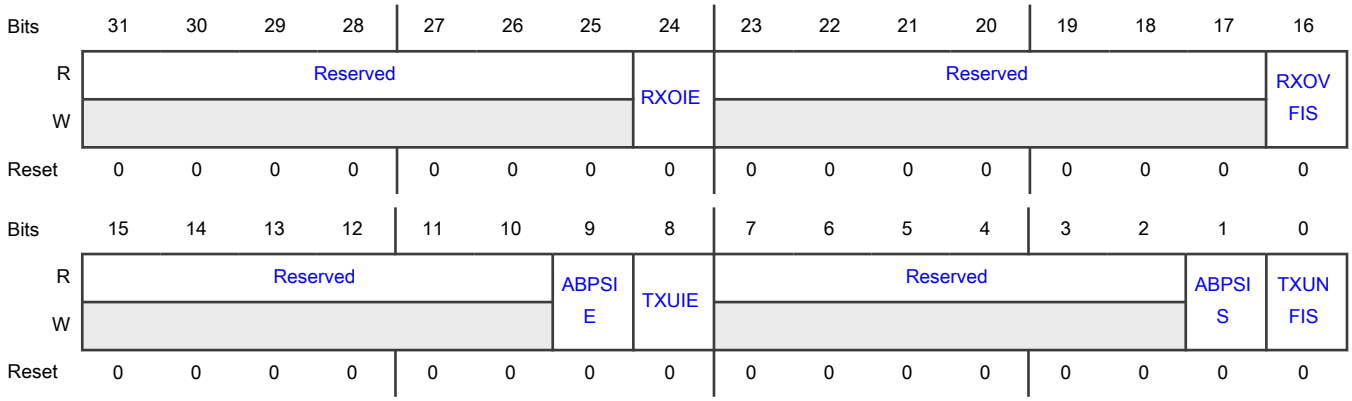
76.17.324 MTL Queue 2 Interrupt Control Status (MTL_Q2_Interrupt_Control_Status)

Offset

Register	Offset
MTL_Q2_Interrupt_Control_Status	DACh

Function
This register contains the interrupt enable and status bits for the queue 2 interrupts.

Diagram



Fields

Field	Function
31-25 —	Reserved
24 RXOIE	Receive Queue Overflow Interrupt Enable When this bit is set, the Receive Queue Overflow interrupt is enabled. When this bit is reset, the Receive Queue Overflow interrupt is disabled. 0b - Receive Queue Overflow Interrupt is disabled 1b - Receive Queue Overflow Interrupt is enabled
23-17	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
16 RXOVFIS	Receive Queue Overflow Interrupt Status This bit indicates that the Receive Queue had an overflow while receiving the packet. If a partial packet is transferred to the application, the overflow status is set in RDES3[21]. This bit is cleared when the application writes 1 to this bit. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **NOTE** This field is not testable by an automated register test. 0b - Receive Queue Overflow Interrupt Status not detected 1b - Receive Queue Overflow Interrupt Status detected
15-10 —	Reserved
9 ABPSIE	Average Bits Per Slot Interrupt Enable When this bit is set, the MAC asserts the sbd_intr_o or mci_intr_o interrupt when the average bits per slot status is updated. When this bit is cleared, the interrupt is not asserted for such an event. 0b - Average Bits Per Slot Interrupt is disabled 1b - Average Bits Per Slot Interrupt is enabled
8 TXUIE	Transmit Queue Underflow Interrupt Enable When this bit is set, the Transmit Queue Underflow interrupt is enabled. When this bit is reset, the Transmit Queue Underflow interrupt is disabled. 0b - Transmit Queue Underflow Interrupt Status is disabled 1b - Transmit Queue Underflow Interrupt Status is enabled
7-2 —	Reserved
1 ABPSIS	Average Bits Per Slot Interrupt Status When set, this bit indicates that the MAC has updated the ABS value. This bit is cleared when the application writes 1 to this bit. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **NOTE** This field is not testable by an automated register test. 0b - Average Bits Per Slot Interrupt Status not detected 1b - Average Bits Per Slot Interrupt Status detected
0	Transmit Queue Underflow Interrupt Status

Table continues on the next page...

Table continued from the previous page...

Field	Function
TXUNFIS	This bit indicates that the Transmit Queue had an underflow while transmitting the packet. Transmission is suspended and an Underflow Error TDES3[2] is set. This bit is cleared when the application writes 1 to this bit. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **NOTE** This field is not testable by an automated register test. 0b - Transmit Queue Underflow Interrupt Status not detected 1b - Transmit Queue Underflow Interrupt Status detected

76.17.325 MTL Rx Queue 2 Operation Mode (MTL_RxQ2_Operation_Mode)

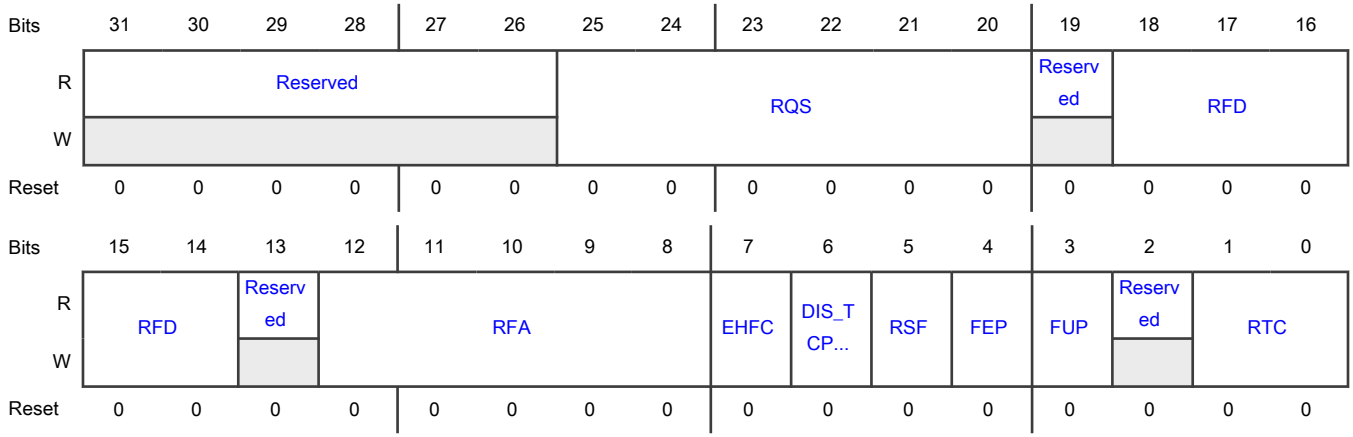
Offset

Register	Offset
MTL_RxQ2_Operation_Mode	DB0h

Function

The Queue 2 Receive Operation Mode register establishes the Receive queue operating modes and command. The RFA and RFD fields are not backward compatible with the RFA and RFD fields of 4.00a release

Diagram



Fields

Field	Function
31-26	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
25-20 RQS	Receive Queue Size This field indicates the size of the allocated Receive queues in blocks of 256 bytes. The RQS field is read-write only if the number of Rx Queues more than one, the reset value is 0x0 and indicates size of 256 bytes. This means that value of 0x0 = 256 bytes, 0x1 = 512 bytes and so on. So, program RQS [5:0] = 6'b001111 to allocate queue size of 4096 (4K) bytes. In general, the size of the Queue = (RQS+1)*256 bytes. When the number of Rx Queues is one, the field is read-only and the configured RX FIFO size in blocks of 256 bytes is reflected in the reset value. The width of this field depends on the Rx memory size selected in your configuration. For example, if the memory size is 2048, the width of this field is 3 bits: LOG2(2048/256) = LOG2(8) = 3 bits
19 —	Reserved
18-14 RFD	Threshold for Deactivating Flow Control (in half-duplex and full-duplex modes) These bits control the threshold (fill-level of Rx queue) at which the flow control is de-asserted after activation: - 0: Full minus 1 KB, that is, FULL 1 KB - 1: Full minus 1.5 KB, that is, FULL 1.5 KB - 2: Full minus 2 KB, that is, FULL 2 KB - 3: Full minus 2.5 KB, that is, FULL 2.5 KB - ... - 30: Full minus 16 KB, that is, FULL 16 KB - 31: Full minus 16.5 KB, that is, FULL 16.5 KB The de-assertion is effective only after flow control is asserted. Note: The value must be programmed in such a way to make sure that the threshold is a positive number. When the EHFC is set high, these values are applicable only when the Rx queue size determined by the RQS field of this register, is equal to or greater than 4 KB. For a given queue size, the values ranges between 0 and the encoding for FULL minus (QSIZE - 0.5 KB) and all other values are illegal. Here the term FULL and QSIZE refers to the queue size determined by the RQS field of this register. The width of this field depends on RX FIFO size selected during the configuration. Remaining bits are reserved and read only.
13 —	Reserved
12-8 RFA	Threshold for Activating Flow Control (in half-duplex and full-duplex These bits control the threshold (fill-level of Rx queue) at which the flow control is activated: For more information on encoding for this field, see RFD.
7 EHFC	Enable Hardware Flow Control When this bit is set, the flow control signal operation, based on the fill-level of Rx queue, is enabled. When reset, the flow control operation is disabled. 0b - Hardware Flow Control is disabled 1b - Hardware Flow Control is enabled
6 DIS_TCP_EF	Disable Dropping of TCP/IP Checksum Error Packets When this bit is set, the MAC does not drop the packets which only have the errors detected by the Receive Checksum Offload engine. Such packets have errors only in the encapsulated payload. There are no errors (including FCS error) in the Ethernet packet received by the MAC. When this bit is reset, all error packets are dropped if the FEP bit is reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Dropping of TCP/IP Checksum Error Packets is enabled 1b - Dropping of TCP/IP Checksum Error Packets is disabled
5 RSF	Receive Queue Store and Forward When this bit is set, the DWC_ether_qos reads a packet from the Rx queue only after the complete packet has been written to it, ignoring the RTC field of this register. When this bit is reset, the Rx queue operates in the Threshold (cut-through) mode, subject to the threshold specified by the RTC field of this register. 0b - Receive Queue Store and Forward is disabled 1b - Receive Queue Store and Forward is enabled
4 FEP	Forward Error Packets When this bit is reset, the Rx queue drops packets with error status (CRC error, GMII_ER, watchdog timeout, or overflow). However, if the start byte (write) pointer of a packet is already transferred to the read controller side (in Threshold mode), the packet is not dropped. When this bit is set, all packets except the runt error packets are forwarded to the application or DMA. If the RSF bit is set and the Rx queue overflows when a partial packet is written, the packet is dropped irrespective of the setting of this bit. However, if the RSF bit is reset and the Rx queue overflows when a partial packet is written, a partial packet might be forwarded to the application or DMA. 0b - Forward Error Packets is disabled 1b - Forward Error Packets is enabled
3 FUP	Forward Undersized Good Packets When this bit is set, the Rx queue forwards the undersized good packets (packets with no error and length less than 64 bytes), including pad-bytes and CRC. When this bit is reset, the Rx queue drops all packets of less than 64 bytes, unless a packet is already transferred because of the lower value of Rx Threshold, for example, RTC = 01. 0b - Forward Undersized Good Packets is disabled 1b - Forward Undersized Good Packets is enabled
2 —	Reserved
1-0 RTC	Receive Queue Threshold Control These bits control the threshold level of the MTL Rx queue (in bytes): The received packet is transferred to the application or DMA when the packet size within the MTL Rx queue is larger than the threshold. In addition, full packets with length less than the threshold are automatically transferred. This field is valid only when the RSF bit is zero. This field is ignored when the RSF bit is set to 1. 00b - 64 01b - 32 10b - 96

Table continues on the next page...

Table continued from the previous page...

Field	Function
	11b - 128

76.17.326 MTL Rx Queue 2 Missed Packet Overflow Counter (MTL_RxQ2_Missed_Packet_Overflow_Cnt)

Offset

Register	Offset
MTL_RxQ2_Missed_Packet_Overflow_Cnt	DB4h

Function

The Queue 2 Missed Packet and Overflow Counter register contains the counter for packets missed because of Receive queue packet flush and packets discarded because of Receive queue overflow.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved				MISCNTO...	MISPKTCNT										
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved				OVFCNTO...	OVFPKTCNT										
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved
27 MISCNTOVF	Missed Packet Counter Overflow Bit When set, this bit indicates that the Rx Queue Missed Packet Counter crossed the maximum limit. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - Missed Packet Counter overflow not detected 1b - Missed Packet Counter overflow detected

Table continues on the next page...

Table continued from the previous page...

Field	Function
26-16 MISPKTCNT	Missed Packet Counter This field indicates the number of packets missed by the DWC_ether_qos because the application asserted ari_pkt_flush_i[] for this queue. This counter is reset when this register is read with mci_be_i[0] at 1b1. This counter is incremented by 1 when the DMA discards the packet because of buffer unavailability. Access restriction applies. Clears on read. Self-set to 1 on internal event.
15-12 —	Reserved
11 OVFCNTOVF	Overflow Counter Overflow Bit When set, this bit indicates that the Rx Queue Overflow Packet Counter field crossed the maximum limit. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - Overflow Counter overflow not detected 1b - Overflow Counter overflow detected
10-0 OVFPKTCNT	Overflow Packet Counter This field indicates the number of packets discarded by the DWC_ether_qos because of Receive queue overflow. This counter is incremented each time the DWC_ether_qos discards an incoming packet because of overflow. This counter is reset when this register is read with mci_be_i[0] at 1'b1. Access restriction applies. Clears on read. Self-set to 1 on internal event.

76.17.327 MTL Rx Queue 2 Debug (MTL_RxQ2_Debug)

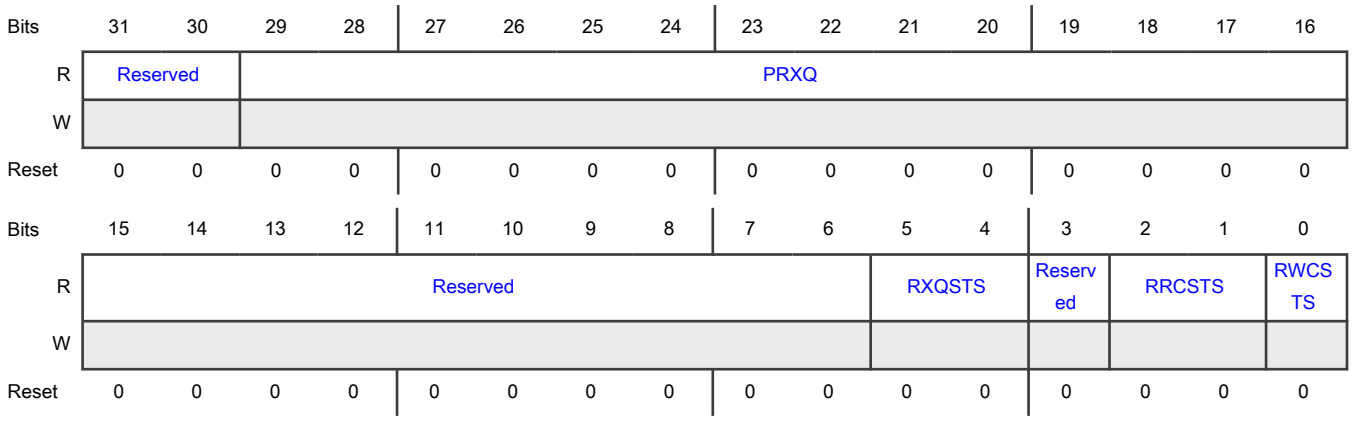
Offset

Register	Offset
MTL_RxQ2_Debug	DB8h

Function

The Queue 2 Receive Debug register gives the debug status of various blocks related to the Receive queue.

Diagram



Fields

Field	Function
31-30 —	Reserved
29-16 PRXQ	Number of Packets in Receive Queue This field indicates the current number of packets in the Rx Queue. The theoretical maximum value for this field is 256KB/16B = 16K Packets, that is, Max_Queue_Size/Min_Packet_Size.
15-6 —	Reserved
5-4 RXQSTS	MTL Rx Queue Fill-Level Status This field gives the status of the fill-level of the Rx Queue: 00b - Rx Queue empty 01b - Rx Queue fill-level below flow-control deactivate threshold 10b - Rx Queue fill-level above flow-control activate threshold 11b - Rx Queue full
3 —	Reserved
2-1 RRCSTS	MTL Rx Queue Read Controller State This field gives the state of the Rx queue Read controller: 00b - Idle state 01b - Reading packet data 10b - Reading packet status (or timestamp) 11b - Flushing the packet data and status
0 RWCSTS	MTL Rx Queue Write Controller Active Status When high, this bit indicates that the MTL Rx queue Write controller is active, and it is transferring a received packet to the Rx Queue. 0b - MTL Rx Queue Write Controller Active Status not detected 1b - MTL Rx Queue Write Controller Active Status detected

76.17.328 MTL Rx Queue 2 Control (MTL_RxQ2_Control)

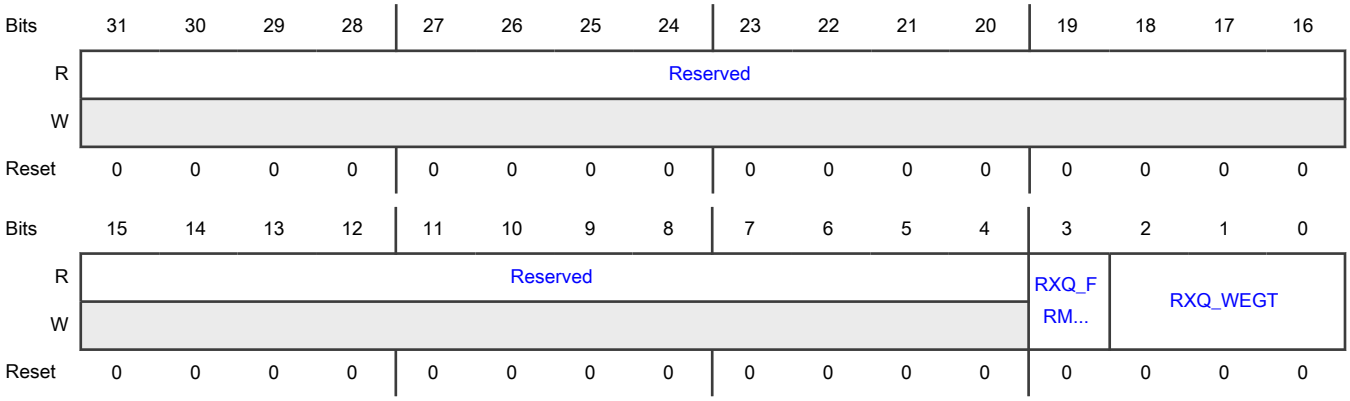
Offset

Register	Offset
MTL_RxQ2_Control	DBCh

Function

The Queue Receive Control register controls the receive arbitration and passing of received packets to the application.

Diagram



Fields

Field	Function
31-4 —	Reserved
3 RXQ_FRM_ARBIT	Receive Queue Packet Arbitration When this bit is set, the DWC_ether_qos drives the packet data to the ARI interface such that the entire packet data of currently-selected queue is transmitted before switching to other queue. When this bit is reset, the DWC_ether_qos drives the packet data to the ARI interface such that the following amount of data of currently-selected queue is transmitted before switching to other queue: - PBL amount of data (indicated by ari_qN_pbl_i[]) or - Complete data of a packet The status and the timestamp are not a part of the PBL data. Therefore, the DWC_ether_qos drives the complete status (including timestamp status) during first PBL request for the packet (in store-and-forward mode) or the last PBL request for the packet (in Threshold mode). 0b - Receive Queue Packet Arbitration is disabled 1b - Receive Queue Packet Arbitration is enabled
2-0 RXQ_WEGT	Receive Queue Weight This field indicates the weight assigned to the Rx Queue 0. This field needs to be programmed with one value less than the required weight, that is, reset value of 0 indicates weight of 1, value of 1 indicates weight of 2, and so on. The weight is used as the number of continuous PBL or packets requests (depending on the RXQ_FRM_ARBIT) allocated to the queue in one arbitration cycle. Note: The change in value of RXQ_WEGT takes effect only after the completion of current service round or when there is change from RAA=SP to RAA=WSP algorithm. This approach is taken so that there is smooth transition. For the RXQ_WEGT value to take effect at the start, the MTL_RxQ(#i)_Control registers must be programmed before the MTL_Operation_Mode register.

76.17.329 DMA Mode (DMA_Mode)

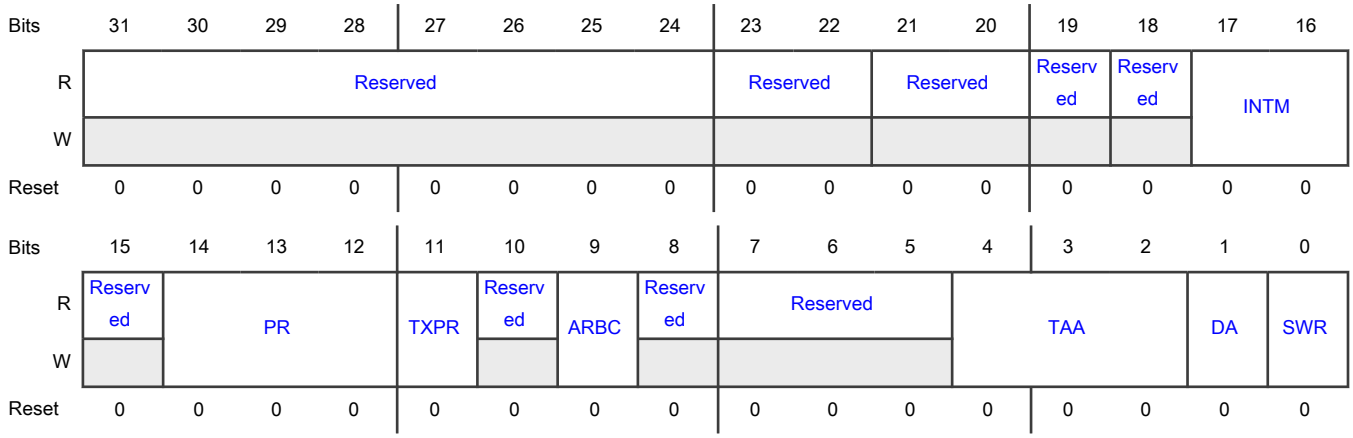
Offset

Register	Offset
DMA_Mode	1000h

Function

The Bus Mode register establishes the bus operating modes for the DMA.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-22 —	Reserved
21-20 —	Reserved
19 —	Reserved
18 —	Reserved
17-16 INTM	Interrupt Mode This field defines the interrupt mode of DWC_ether_qos. The behavior of the following outputs change depending on the following settings: - sbd_perch_tx_intr_o[] (Transmit Per Channel Interrupt)

Table continues on the next page...

Table continued from the previous page...

Field	Function
	- sbd_perch_rx_intr_o[] (Receive Per Channel Interrupt) - sbd_intr_o (Common Interrupt) It also changes the behavior of the RI/TI bits in the DMA_CH0_Status. - 00: sbd_perch_* are pulse signals for each TX/RX packet transfer completion events (irrespective of whether corresponding interrupts are enabled) for which IOC bits are enabled in descriptor. sbd_intr_o is also asserted when corresponding interrupts are enabled and cleared only when software clears the corresponding RI/TI status bits. - 01: sbd_perch_* are level signals asserted on TX/RX packet transfer completion event when corresponding interrupts are enabled and de-asserted when the software clears the corresponding RI/TI status bits. The sbd_intr_o is not asserted for these TX/RX packet transfer completion events. - 10: sbd_perch_* are level signals asserted on TX/RX packet transfer completion event when corresponding interrupts are enabled and de-asserted when the software clears the corresponding RI/TI status bits. However, the signal is asserted again if the same event occurred again before it was cleared. The sbd_intr_o is not asserted for these TX/RX packet transfer completion events. - 11: Reserved For more details, see Table "DWC_ether_qos Transfer Complete Interrupt Behavior". 00b - See above description 01b - See above description 10b - See above description 11b - Reserved
15 — 	Reserved
14-12 PR	Priority Ratio These bits control the priority ratio in weighted round-robin arbitration between the Rx DMA and Tx DMA. These bits are valid only when the DA bit is reset. The priority ratio is Rx:Tx or Tx:Rx depending on whether the TXPR bit is reset or set. 000b - The priority ratio is 1:1 001b - The priority ratio is 2:1 010b - The priority ratio is 3:1 011b - The priority ratio is 4:1 100b - The priority ratio is 5:1 101b - The priority ratio is 6:1 110b - The priority ratio is 7:1 111b - The priority ratio is 8:1
11 TXPR	Transmit Priority When set, this bit indicates that the Tx DMA has higher priority than the Rx DMA during arbitration for the system-side bus or Descriptor reads from DCACHE memory when DWC_EQOS_DCEXT is enabled 0b - Transmit Priority is disabled 1b - Transmit Priority is enabled
10	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
9 ARBC	ARBC ARBC is NXP Reserved, This field must be set to "0". - This field is reserved for NXP Internal use, and must always be set to "0" unless instructed by NXP. - Setting this field to "1" might cause unexpected behavior in the IP. 0b - NXP reserved field disabled 1b - NXP reserved field enabled up on NXP request
8 —	Reserved
7-5 —	Reserved
4-2 TAA	Transmit Arbitration Algorithm This field is used to select the arbitration algorithm for the Transmit side when multiple Tx DMAs are selected. 000b - Fixed priority (Channel 0 has the lowest priority and the last channel has the highest priority) 001b - Weighted Strict Priority (WSP) 010b - Weighted Round-Robin (WRR) 011b - Reserved (for 3'b011 to 3'b111)
1 DA	DMA Tx or Rx Arbitration Scheme This bit specifies the arbitration scheme between the Transmit and Receive paths of all channels: - 0: Weighted Round-Robin with Rx:Tx or Tx:Rx The priority between the paths is according to the priority specified in Bits[14:12] and the priority weight is specified in the TXPR bit. - 1: Fixed Priority The Tx path has priority over the Rx path when the TXPR bit is set. Otherwise, the Rx path has priority over the Tx path. 0b - Weighted Round-Robin with Rx:Tx or Tx:Rx 1b - Fixed Priority
0 SWR	Software Reset When this bit is set, the MAC and the DMA controller reset the logic and all internal registers of the DMA, MTL, and MAC. This bit is automatically cleared after the reset operation is complete in all DWC_ether_qos clock domains. Before reprogramming any DWC_ether_qos register, a value of zero should be read in this bit. This bit must be read at least 4 CSR clock cycles after it is written to 1. Note: The reset operation is complete only when all resets in all active clock domains are de-asserted. Therefore, it is essential that all PHY inputs clocks (applicable for the selected PHY interface) are present for software reset completion. The time to complete the software reset operation depends on the frequency of the slowest active clock. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field is not testable by an automated register test. 0b - Software Reset is disabled 1b - Software Reset is enabled

76.17.330 DMA System Bus Mode (DMA_SysBus_Mode)

Offset

Register	Offset
DMA_SysBus_Mode	1004h

Function

The System Bus mode register controls the behavior of the AHB or AXI master. It mainly controls burst splitting and number of outstanding requests.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved	Reserved	Reserved			Reserved			Reserved					Reserved		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RB	MB	Reserved	AAL	Reserved	Reserved	Reserved		Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	FB
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
29-27 —	Reserved
26-24 —	Reserved
23-19 —	Reserved
18-16 —	Reserved
15 RB	Rebuild INCRx Burst When this bit is set high and the AHB master gets SPLIT, RETRY, or Early Burst Termination (EBT) response, the AHB master interface rebuilds the pending beats of any initiated burst transfer with INCRx and SINGLE transfers. By default, the AHB master interface rebuilds the pending beats of an EBT with an unspecified (INCR) burst. 0b - Rebuild INCRx Burst is disabled 1b - Rebuild INCRx Burst is enabled
14 MB	Mixed Burst When this bit is high and the FB bit is low, the AHB master performs undefined bursts transfers (INCR) for burst length of 16 or more. For burst length of 16 or less, the AHB master performs fixed burst transfers (INCRx and SINGLE). 0b - Mixed Burst is disabled 1b - Mixed Burst is enabled
13 —	Reserved
12 AAL	Address-Aligned Beats When this bit is set to 1, the EQOS-AXI or EQOS-AHB master performs address-aligned burst transfers on Read and Write channels. When this bit is set to 0, the EQOS-AXI or EQOS-AHB master performs burst transfers on Read and Write channels without aligning to address boundaries. 0b - Address-Aligned Beats is disabled 1b - Address-Aligned Beats is enabled
11 —	Reserved
10 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
9-8 —	Reserved
7 —	Reserved
6 —	Reserved
5 —	Reserved
4 —	Reserved
3 —	Reserved
2 —	Reserved
1 —	Reserved
0 FB	Fixed Burst Length - 1: AHB master initiates burst transfers of specified length (INCRx or SINGLE). - 0: AHB master initiates transfers of unspecified length (INCR) or SINGLE transfers. 0b - Fixed Burst Length is disabled 1b - Fixed Burst Length is enabled

76.17.331 DMA Interrupt Status (DMA_Interrupt_Status)

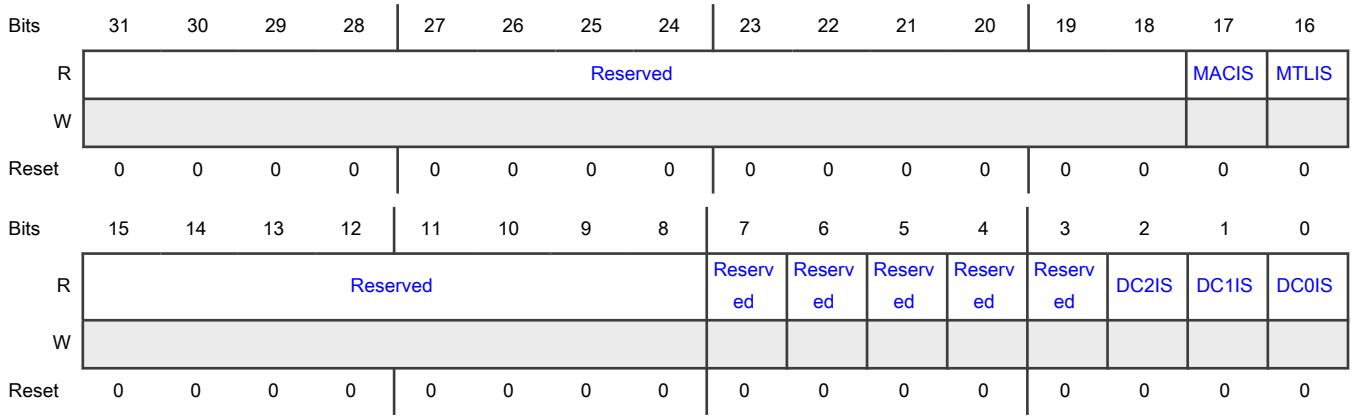
Offset

Register	Offset
DMA_Interrupt_Status	1008h

Function

The application reads this Interrupt Status register during interrupt service routine or polling to determine the interrupt status of DMA channels, MTL queues, and the MAC.

Diagram



Fields

Field	Function
31-18 —	Reserved
17 MACIS	MAC Interrupt Status This bit indicates an interrupt event in the MAC. To reset this bit to 1'b0, the software must read the corresponding register in the MAC to get the exact cause of the interrupt and clear its source. 0b - MAC Interrupt Status not detected 1b - MAC Interrupt Status detected
16 MTLIS	MTL Interrupt Status This bit indicates an interrupt event in the MTL. To reset this bit to 1'b0, the software must read the corresponding register in the MTL to get the exact cause of the interrupt and clear its source. 0b - MTL Interrupt Status not detected 1b - MTL Interrupt Status detected
15-8 —	Reserved
7 —	Reserved
6 —	Reserved
5 —	Reserved
4	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
3 —	Reserved
2 DC2IS	DMA Channel 2 Interrupt Status This bit indicates an interrupt event in DMA Channel 2. To reset this bit to 1'b0, the software must read the corresponding register in DMA Channel 2 to get the exact cause of the interrupt and clear its source. 0b - DMA Channel 2 Interrupt Status not detected 1b - DMA Channel 2 Interrupt Status detected
1 DC1IS	DMA Channel 1 Interrupt Status This bit indicates an interrupt event in DMA Channel 1. To reset this bit to 1'b0, the software must read the corresponding register in DMA Channel 1 to get the exact cause of the interrupt and clear its source. 0b - DMA Channel 1 Interrupt Status not detected 1b - DMA Channel 1 Interrupt Status detected
0 DC0IS	DMA Channel 0 Interrupt Status This bit indicates an interrupt event in DMA Channel 0. To reset this bit to 1'b0, the software must read the corresponding register in DMA Channel 0 to get the exact cause of the interrupt and clear its source. 0b - DMA Channel 0 Interrupt Status not detected 1b - DMA Channel 0 Interrupt Status detected

76.17.332 DMA Debug Status 0 (DMA_Debug_Status0)

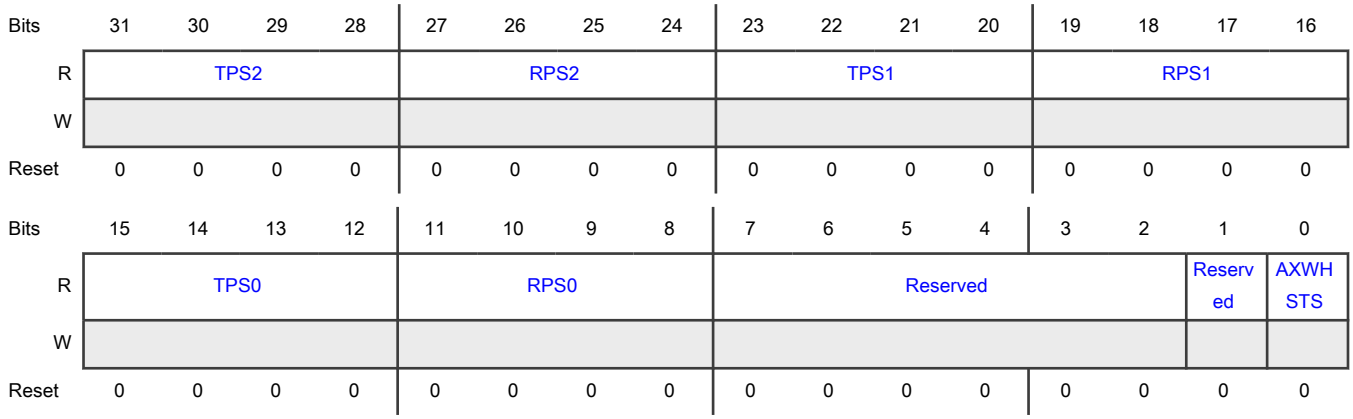
Offset

Register	Offset
DMA_Debug_Status0	100Ch

Function

The Debug Status 0 register gives the Receive and Transmit process status for DMA Channel 0-Channel 2 for debugging purpose.

Diagram



Fields

Field	Function
31-28 TPS2	DMA Channel 2 Transmit Process State This field indicates the Tx DMA FSM state for Channel 2. The MSB of this field always returns 0. This field does not generate an interrupt. 0000b - Stopped (Reset or Stop Transmit Command issued) 0001b - Running (Fetching Tx Transfer Descriptor) 0010b - Running (Waiting for status) 0011b - Running (Reading Data from system memory buffer and queuing it to the Tx buffer (Tx FIFO)) 0100b - Timestamp write state 0101b - Reserved for future use 0110b - Suspended (Tx Descriptor Unavailable or Tx Buffer Underflow) 0111b - Running (Closing Tx Descriptor)
27-24 RPS2	DMA Channel 2 Receive Process State This field indicates the Rx DMA FSM state for Channel 2. The MSB of this field always returns 0. This field does not generate an interrupt. 0000b - Stopped (Reset or Stop Receive Command issued) 0001b - Running (Fetching Rx Transfer Descriptor) 0010b - Reserved for future use 0011b - Running (Waiting for Rx packet) 0100b - Suspended (Rx Descriptor Unavailable) 0101b - Running (Closing the Rx Descriptor) 0110b - Timestamp write state 0111b - Running (Transferring the received packet data from the Rx buffer to the system memory)

Table continues on the next page...

Table continued from the previous page...

Field	Function
23-20 TPS1	DMA Channel 1 Transmit Process State This field indicates the Tx DMA FSM state for Channel 1. The MSB of this field always returns 0. This field does not generate an interrupt. 0000b - Stopped (Reset or Stop Transmit Command issued) 0001b - Running (Fetching Tx Transfer Descriptor) 0010b - Running (Waiting for status) 0011b - Running (Reading Data from system memory buffer and queuing it to the Tx buffer (Tx FIFO)) 0100b - Timestamp write state 0101b - Reserved for future use 0110b - Suspended (Tx Descriptor Unavailable or Tx Buffer Underflow) 0111b - Running (Closing Tx Descriptor)
19-16 RPS1	DMA Channel 1 Receive Process State This field indicates the Rx DMA FSM state for Channel 1. The MSB of this field always returns 0. This field does not generate an interrupt. 0000b - Stopped (Reset or Stop Receive Command issued) 0001b - Running (Fetching Rx Transfer Descriptor) 0010b - Reserved for future use 0011b - Running (Waiting for Rx packet) 0100b - Suspended (Rx Descriptor Unavailable) 0101b - Running (Closing the Rx Descriptor) 0110b - Timestamp write state 0111b - Running (Transferring the received packet data from the Rx buffer to the system memory)
15-12 TPS0	DMA Channel 0 Transmit Process State This field indicates the Tx DMA FSM state for Channel 0. The MSB of this field always returns 0. This field does not generate an interrupt. 0000b - Stopped (Reset or Stop Transmit Command issued) 0001b - Running (Fetching Tx Transfer Descriptor) 0010b - Running (Waiting for status) 0011b - Running (Reading Data from system memory buffer and queuing it to the Tx buffer (Tx FIFO)) 0100b - Timestamp write state 0101b - Reserved for future use 0110b - Suspended (Tx Descriptor Unavailable or Tx Buffer Underflow)

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0111b - Running (Closing Tx Descriptor)
11-8 RPS0	DMA Channel 0 Receive Process State This field indicates the Rx DMA FSM state for Channel 0. The MSB of this field always returns 0. This field does not generate an interrupt. 0000b - Stopped (Reset or Stop Receive Command issued) 0001b - Running (Fetching Rx Transfer Descriptor) 0010b - Reserved for future use 0011b - Running (Waiting for Rx packet) 0100b - Suspended (Rx Descriptor Unavailable) 0101b - Running (Closing the Rx Descriptor) 0110b - Timestamp write state 0111b - Running (Transferring the received packet data from the Rx buffer to the system memory)
7-2 —	Reserved
1 —	Reserved
0 AXWHSTS	AHB Master Status When high, this bit indicates that the AHB master FSMs are in the non-idle state. 0b - AXI Master Write Channel or AHB Master Status not detected 1b - AXI Master Write Channel or AHB Master Status detected

76.17.333 DMA TBS Control 0 (DMA_TBS_CTRL0)

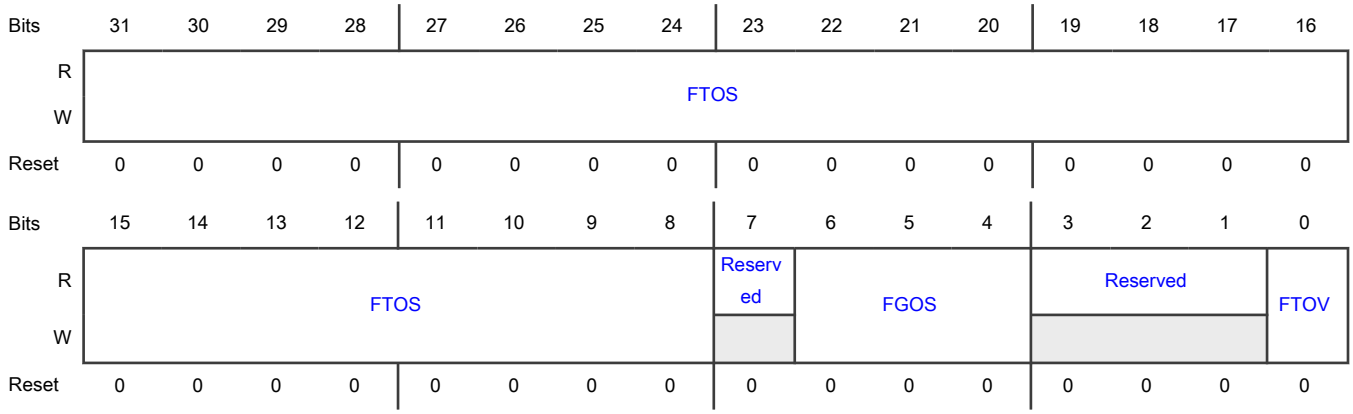
Offset

Register	Offset
DMA_TBS_CTRL0	1050h

Function

This register is used to control the TBS attributes.

Diagram



Fields

Field	Function
31-8 FTOS	Fetch Time Offset The value in units of 256 nanoseconds, that has to be deducted from the Launch time to compute the Fetch Time. Max value: 999,999,999 ns, additionally should be smaller than CTR-1 value when ESTM mode is set since this value is a modulo CTR value.
7 —	Reserved
6-4 FGOS	Fetch GSN Offset The number GSN slots that must be deducted from the Launch GSN to compute the Fetch GSN. Value valid only when FTOV is set.
3-1 —	Reserved
0 FTOV	Fetch Time Offset Valid When set indicates the FTOS field is valid. When not set, indicates the Fetch Offset is not valid and the DMA engine can fetch the frames from host memory without any time restrictions. 0b - Fetch Time Offset is invalid 1b - Fetch Time Offset is valid

76.17.334 DMA TBS Control 1 (DMA_TBS_CTRL1)

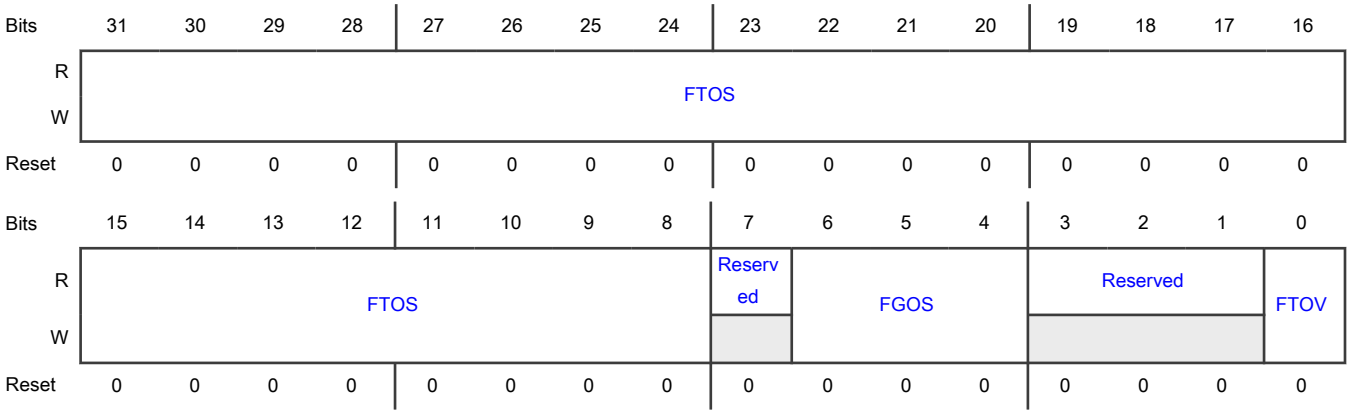
Offset

Register	Offset
DMA_TBS_CTRL1	1054h

Function

This register is used to control the TBS attributes.

Diagram



Fields

Field	Function
31-8 FTOS	Fetch Time Offset The value in units of 256 nanoseconds, that has to be deducted from the Launch time to compute the Fetch Time. Max value: 999,999,999 ns, additionally should be smaller than CTR-1 value when ESTM mode is set since this value is a modulo CTR value.
7 —	Reserved
6-4 FGOS	Fetch GSN Offset The number GSN slots that must be deducted from the Launch GSN to compute the Fetch GSN. Value valid only when FTOV is set.
3-1 —	Reserved
0 FTOV	Fetch Time Offset Valid When set indicates the FTOS field is valid. When not set, indicates the Fetch Offset is not valid and the DMA engine can fetch the frames from host memory without any time restrictions. 0b - Fetch Time Offset is invalid 1b - Fetch Time Offset is valid

76.17.335 DMA TBS Control 2 (DMA_TBS_CTRL2)

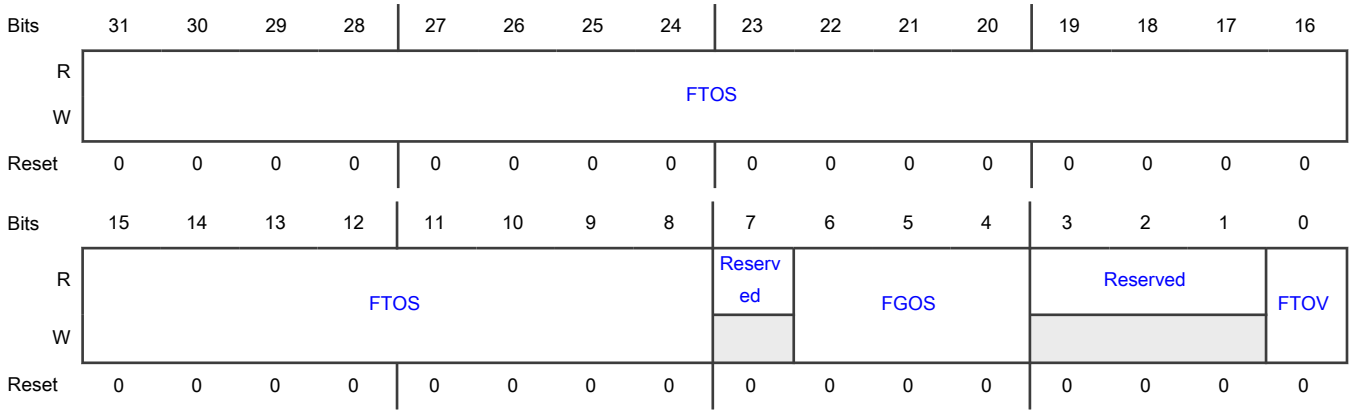
Offset

Register	Offset
DMA_TBS_CTRL2	1058h

Function

This register is used to control the TBS attributes.

Diagram



Fields

Field	Function
31-8 FTOS	Fetch Time Offset The value in units of 256 nanoseconds, that has to be deducted from the Launch time to compute the Fetch Time. Max value: 999,999,999 ns, additionally should be smaller than CTR-1 value when ESTM mode is set since this value is a modulo CTR value.
7 —	Reserved
6-4 FGOS	Fetch GSN Offset The number GSN slots that must be deducted from the Launch GSN to compute the Fetch GSN. Value valid only when FTOV is set.
3-1 —	Reserved
0 FTOV	Fetch Time Offset Valid When set indicates the FTOS field is valid. When not set, indicates the Fetch Offset is not valid and the DMA engine can fetch the frames from host memory without any time restrictions.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Fetch Time Offset is invalid 1b - Fetch Time Offset is valid

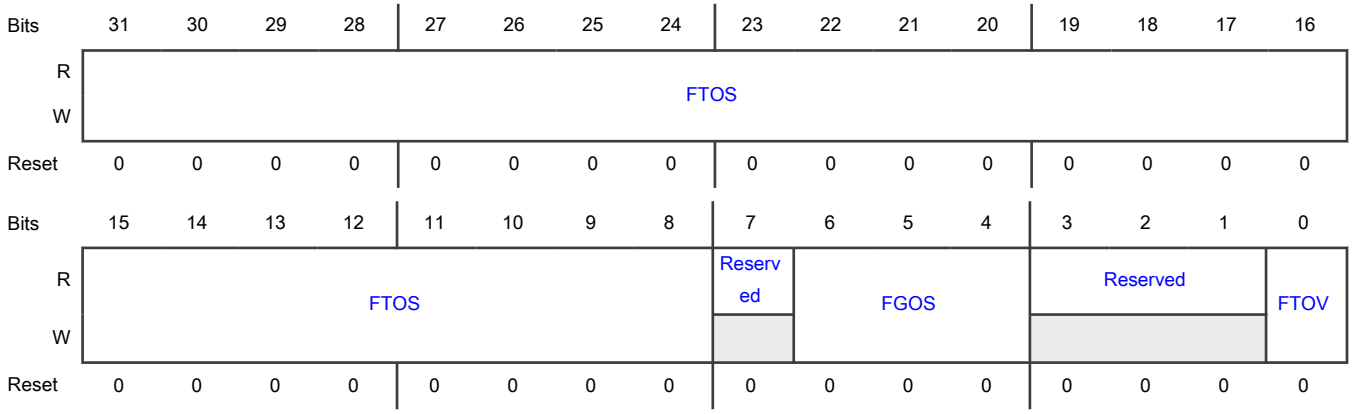
76.17.336 DMA TBS Control 3 (DMA_TBS_CTRL3)

Offset

Register	Offset
DMA_TBS_CTRL3	105Ch

Function
This register is used to control the TBS attributes.

Diagram



Fields

Field	Function
31-8 FTOS	Fetch Time Offset The value in units of 256 nanoseconds, that has to be deducted from the Launch time to compute the Fetch Time. Max value: 999,999,999 ns, additionally should be smaller than CTR-1 value when ESTM mode is set since this value is a modulo CTR value.
7 —	Reserved
6-4 FGOS	Fetch GSN Offset The number GSN slots that must be deducted from the Launch GSN to compute the Fetch GSN. Value valid only when FTOV is set.

Table continues on the next page...

Table continued from the previous page...

Field	Function
3-1 —	Reserved
0 FTOV	Fetch Time Offset Valid When set indicates the FTOS field is valid. When not set, indicates the Fetch Offset is not valid and the DMA engine can fetch the frames from host memory without any time restrictions. 0b - Fetch Time Offset is invalid 1b - Fetch Time Offset is valid

76.17.337 DMA Safety Interrupt Status (DMA_Safety_Interrupt_Status)

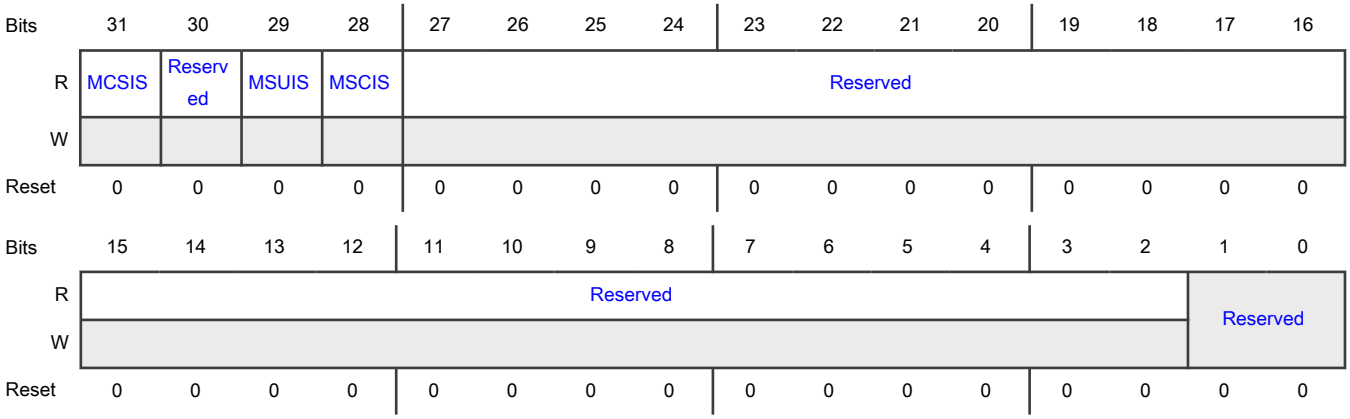
Offset

Register	Offset
DMA_Safety_Interrupt_Status	1080h

Function

This register indicates summary (whether error occured in DMA/MTL/MAC and correctable/uncorrectable) of the Automotive Safety related error interrupts.

Diagram



Fields

Field	Function
31 MCSIS	MAC Safety Uncorrectable Interrupt Status

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Indicates a uncorrectable Safety related Interrupt is set in the MAC module. MAC_DPP_FSM_Interrupt_Status register should be read when this bit is set, to get the cause of the Safety Interrupt in MAC. 0b - MAC Safety Uncorrectable Interrupt Status not detected 1b - MAC Safety Uncorrectable Interrupt Status detected
30 —	Reserved
29 MSUIS	MTL Safety Uncorrectable error Interrupt Status This bit indicates an uncorrectable error interrupt event in MTL. To get exact cause of the interrupt the software should read the MTL_Safety_Interrupt_Status register. 0b - MTL Safety Uncorrectable error Interrupt Status not detected 1b - MTL Safety Uncorrectable error Interrupt Status detected
28 MSCIS	MTL Safety Correctable error Interrupt Status This bit indicates a correctable error interrupt event in MTL. To get exact cause of the interrupt the software should read the MTL_Safety_Interrupt_Status register. 0b - MTL Safety Correctable error Interrupt Status not detected 1b - MTL Safety Correctable error Interrupt Status detected
27-2 —	Reserved
1-0 —	Reserved

76.17.338 DMA Channel 0 Control (DMA_CH0_Control)

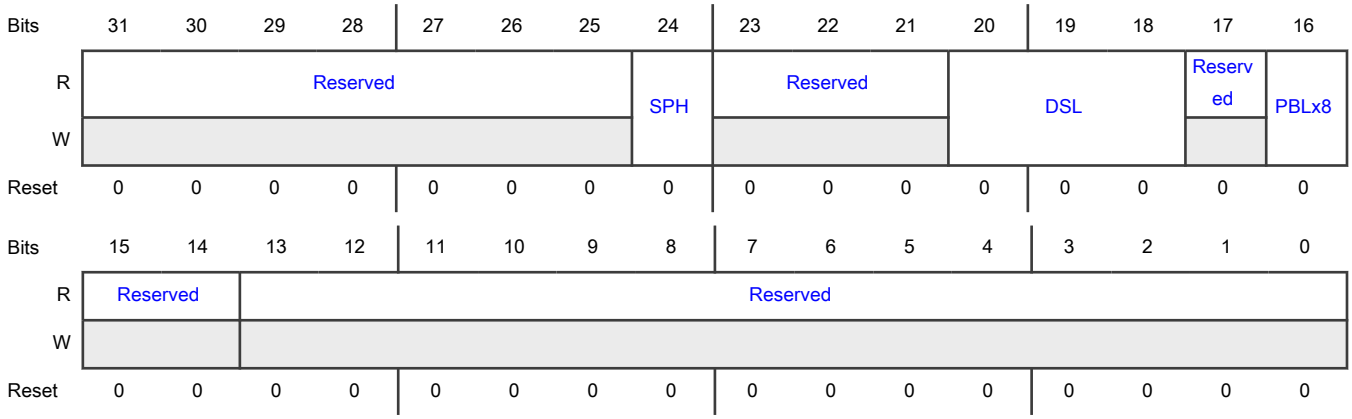
Offset

Register	Offset
DMA_CH0_Control	1100h

Function

The DMA Channeli Control register specifies the MSS value for segmentation, length to skip between two descriptors, and also the features such as header splitting and 8xPBL mode.

Diagram



Fields

Field	Function
31-25 —	Reserved
24 SPH	Split Headers When this bit is set, the DMA splits the header and payload in the Receive path. The DMA writes the header to the Buffer Address1 of RDES0. The DMA writes the payload to the buffer to which the Buffer Address2 is pointing. The software must ensure that the header fits into the Receive buffers. If the header length exceeds the receive buffer size, the DMA does not split the header and payload. This bit is available only if Enable Split Header Structure option is selected. 0b - Split Headers feature is disabled 1b - Split Headers feature is enabled
23-21 —	Reserved
20-18 DSL	Descriptor Skip Length This bit specifies the Word, Dword, or Lword number (depending on the 32-bit, 64-bit, or 128-bit bus) to skip between two unchained descriptors. The address skipping starts from the end of the current descriptor to the start of the next descriptor. When the DSL value is equal to zero, the DMA takes the descriptor table as contiguous.
17 —	Reserved
16 PBLx8	8xPBL mode When this bit is set, the PBL value programmed in Bits[21:16] in DMA_CH(#i)_Tx_Control and Bits[21:16] in DMA_CH(#i)_Rx_Control is multiplied by eight times. Therefore, the DMA transfers the data in 8, 16, 32, 64, 128, and 256 beats depending on the PBL value. 0b - 8xPBL mode is disabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - 8xPBL mode is enabled
15-14 —	Reserved
13-0 —	Reserved

76.17.339 DMA Channel 0 Tx Control (DMA_CH0_Tx_Control)

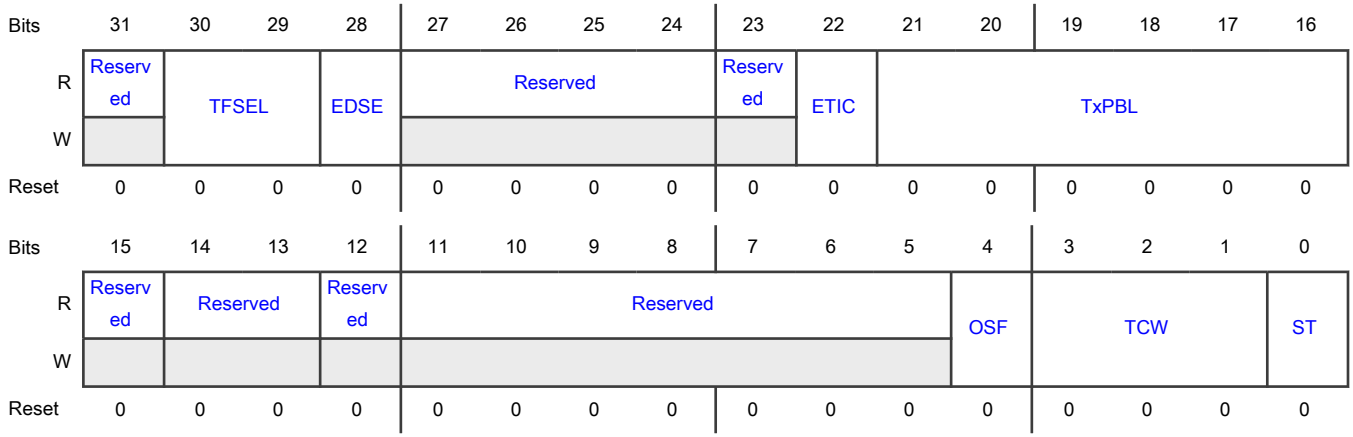
Offset

Register	Offset
DMA_CH0_Tx_Control	1104h

Function

The DMA Channeli Transmit Control register controls the Tx features such as PBL, TCP segmentation, and Tx Channel weights.

Diagram



Fields

Field	Function
31 —	Reserved
30-29	TBS Fetch Select

Table continues on the next page...

Table continued from the previous page...

Field	Function
TFSEL	Select bits for one of the four DMA_TBS_CTRL register fields (FTOS,FGSN,FTOV) for the channel Values: - 2'b00: DMA_TBS_CTRL0 - 2'b01: DMA_TBS_CTRL1 - 2'b10: DMA_TBS_CTRL2 - 2'b11: DMA_TBS_CTRL3 (Reserved if DWC_EQOS_NUM_DMA_TX_CH=3)
28 EDSE	Enhanced Descriptor Enable When this bit is set, the corresponding channel uses Enhanced Descriptors that are 32 Bytes for both Normal and Context Descriptors. When reset, the corresponding channel uses the descriptors that are 16 Bytes. 0b - Enhanced Descriptor is disabled 1b - Enhanced Descriptor is enabled
27-24 —	Reserved
23 —	Reserved
22 ETIC	Early Transmit Interrupt Control When this bit is set, Early Transmit Interrupt (ETI) status is set after completion of transfer of data from buffers of a transmit descriptor in which IOC bit (TDES2[31]) is set. When this bit is reset, ETI is set only after a complete packet is transferred to the MTL TX FIFO memory. 0b - Early Transmit Interrupt is disabled 1b - Early Transmit Interrupt is enabled
21-16 TxPBL	Transmit Programmable Burst Length These bits indicate the maximum number of beats to be transferred in one DMA block data transfer. The DMA always attempts max burst as specified in PBL each time it starts a burst transfer on the application bus. You can program PBL with any of the following values: 1, 2, 4, 8, 16, or 32. Any other value results in undefined behavior. To transfer more than 32 beats, perform the following steps: 1. Set the 8xPBL mode in DMA_CH0_Control register. 2. Set the TxPBL. Note: The maximum value of TxPBL must be less than or equal to half the Tx Queue size (TQS field of MTL_TxQ(#i)_Operation_Mode register) in terms of beats. This is required so that the Tx Queue has space to store at least another Tx PBL worth of data while the MTL Tx Queue Controller is transferring data to MAC. For example, in 64-bit data width configurations the total locations in Tx Queue of size 512 bytes is 64, TxPBL and 8xPBL needs to be programmed to less than or equal to 32.
15 —	Reserved
14-13 —	Reserved
12 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
11-5 —	Reserved
4 OSF	Operate on Second Packet When this bit is set, it instructs the DMA to process the second packet of the Transmit data even before the status for the first packet is obtained. 0b - Operate on Second Packet disabled 1b - Operate on Second Packet enabled
3-1 TCW	Transmit Channel Weight This field indicates the weight assigned to the corresponding Transmit channel. When reset is complete, this field is set to 0 for all channels by default, resulting in equal weights to all channels.
0 ST	Start or Stop Transmission Command When this bit is set, transmission is placed in the Running state. The DMA checks the Transmit list at the current position for a packet to be transmitted. The DMA tries to acquire descriptor from either of the following positions: - The current position in the list This is the base address of the Transmit list set by the DMA_CH0_TxDesc_List_Address register. - The position at which the transmission was previously stopped If the DMA does not own the current descriptor, the transmission enters the Suspended state and the TBU bit of the DMA_CH0_Status register is set. The Start Transmission command is effective only when the transmission is stopped. If the command is issued before setting the DMA_CH0_TxDesc_List_Address register, the DMA behavior is unpredictable. When this bit is reset, the transmission process is placed in the Stopped state after completing the transmission of the current packet. The Next Descriptor position in the Transmit list is saved, and it becomes the current position when the transmission is restarted. To change the list address, you need to program DMA_CH0_TxDesc_List_Address register with a new value when this bit is reset. The new value is considered when this bit is set again. The stop transmission command is effective only when the transmission of the current packet is complete or the transmission is in the Suspended state. **NOTE** This field is not testable by an automated register test. 0b - Stop Transmission Command 1b - Start Transmission Command

76.17.340 DMA Channel 0 Rx Control (DMA_CH0_Rx_Control)

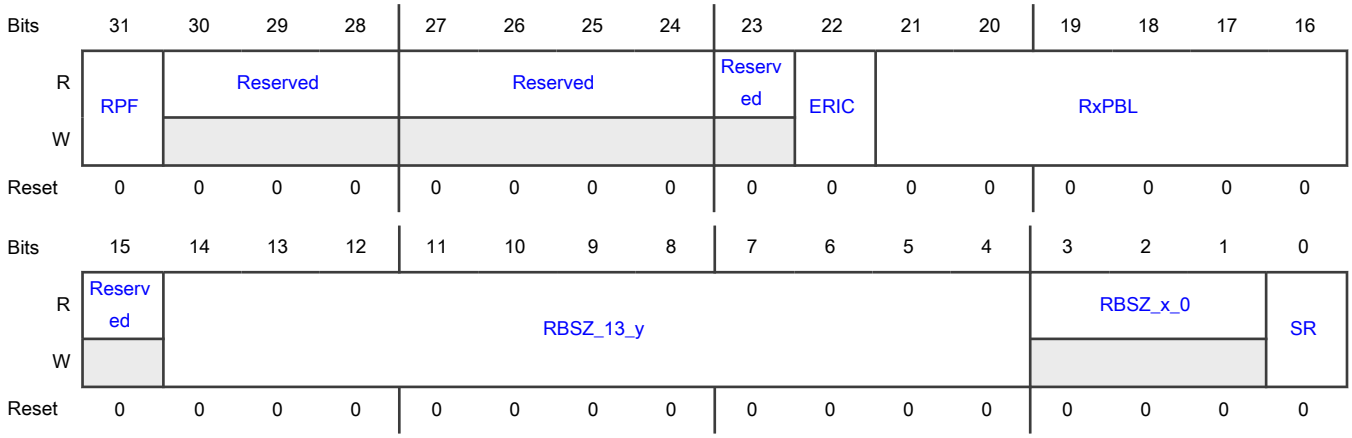
Offset

Register	Offset
DMA_CH0_Rx_Control	1108h

Function

The DMA Channeli Receive Control register controls the Rx features such as PBL, buffer size, and extended status.

Diagram



Fields

Field	Function
31 RPF	Rx Packet Flush. - 1: DWC_ether_qos automatically flushes the packet from the Rx Queues destined to this DMA Rx Channel, when it is stopped. When this bit remains set and the DMA is re-started by the software driver, the packets residing in the Rx Queues that were received when this RxDMA was stopped, are flushed out. The packets that are received by the MAC after the RxDMA is re-started are routed to the RxDMA. The flushing is done on the Read side of the Rx Queue. - 0: DWC_ether_qos does not flush the packet in the Rx Queue destined to this RxDMA Channel when it is in STOP state. This might cause head-of-line blocking in the corresponding RxQueue. Note: The stopping of packet flow from a Rx DMA Channel to the application by setting RPF works only when there is one-to-one mapping of Rx Queue to Rx DMA channels. In Dynamic mapping mode, setting RPF bit in any DMA_CH(#i)_Rx_Control register might flush packets from unintended Rx Queues which are destined to the stopped Rx DMA Channel. 0b - Rx Packet Flush is disabled 1b - Rx Packet Flush is enabled
30-28 —	Reserved
27-24 —	Reserved
23 —	Reserved
22 ERIC	Early Receive Interrupt Control When this bit is set, Early Receive Interrupt (ERI) status is set after the completion of every burst transfer of data from the Rx DMA to the buffer. When this bit is reset, ERI is set only after a complete buffer is filled up by the RxDMA. 0b - Early Receive Interrupt is disabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Early Receive Interrupt is enabled
21-16 RxPBL	Receive Programmable Burst Length These bits indicate the maximum number of beats to be transferred in one DMA block data transfer. The DMA always attempts max burst as specified in PBL each time it starts a burst transfer on the application bus. You can program PBL with any of the following values: 1, 2, 4, 8, 16, or 32. Any other value results in undefined behavior. To transfer more than 32 beats, perform the following steps: 1. Set the 8xPBL mode in the DMA_CH0_Control register. 2. Set the RxPBL. Note: The maximum value of RxPBL must be less than or equal to half the Rx Queue size (RQS field of MTL_RxQ(#i)_Operation_Mode register) in terms of beats. This is required so that the Rx Queue has space to store at least another Rx PBL worth of data while the Rx DMA is transferring a block of data. For example, in 64-bit data width configurations the total locations in Rx Queue of size 512 bytes is 64, so RxPBL and 8xPBL needs to be programmed to less than or equal to 32.
15 —	Reserved
14-4 RBSZ_13_y	Receive Buffer size High RBSZ[13:0] is split into two fields higher RBSZ_13_y and lower RBSZ_x_0. The RBSZ[13:0] field indicates the size of the Rx buffers specified in bytes. The maximum buffer size is limited to 16K bytes. The buffer size is applicable to payload buffers when split headers are enabled. Note: The buffer size must be a multiple of 4, 8, or 16 depending on the data bus widths (32-bit, 64-bit, or 128-bit respectively). This is required even if the value of buffer address pointer is not aligned to data bus width. Hence the lower RBSZ_x_0 bits are read-only and the value is considered as all-zero. Thus the RBSZ_13_y indicates the buffer size in terms of locations (with the width same as bus-width).
3-1 RBSZ_x_0	Receive Buffer size Low RBSZ[13:0] is split into two fields RBSZ_13_y and RBSZ_x_0. The RBSZ_x_0 is the lower field whose width is based on data bus width of the configuration. This field is of width 2, 3, or 4 bits for 32-bit, 64-bit, or 128-bit data bus width respectively. This field is read-only (RO).
0 SR	Start or Stop Receive When this bit is set, the DMA tries to acquire the descriptor from the Receive list and processes the incoming packets. The DMA tries to acquire descriptor from either of the following positions: - The current position in the list This is the address set by the DMA_CH0_RxDesc_List_Address register. - The position at which the Rx process was previously stopped If the DMA does not own the current descriptor, the reception is suspended and the RBU bit of the DMA_CH0_Status register is set. The Start Receive command is effective only when the reception is stopped. If the command is issued before setting the DMA_CH0_RxDesc_List_Address register, the DMA behavior is unpredictable. When this bit is reset, the Rx DMA operation is stopped after the transfer of the current packet. The next descriptor position in the Receive list is saved, and it becomes the current position after the Rx process is restarted. The Stop Receive command is effective only when the Rx process is in the Running (waiting for Rx packet) or Suspended state. **NOTE** This field is not testable by an automated register test.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Stop Receive 1b - Start Receive

76.17.341 DMA Channel 0 Tx Descriptor List Address (DMA_CH0_TxDesc_List_Address)

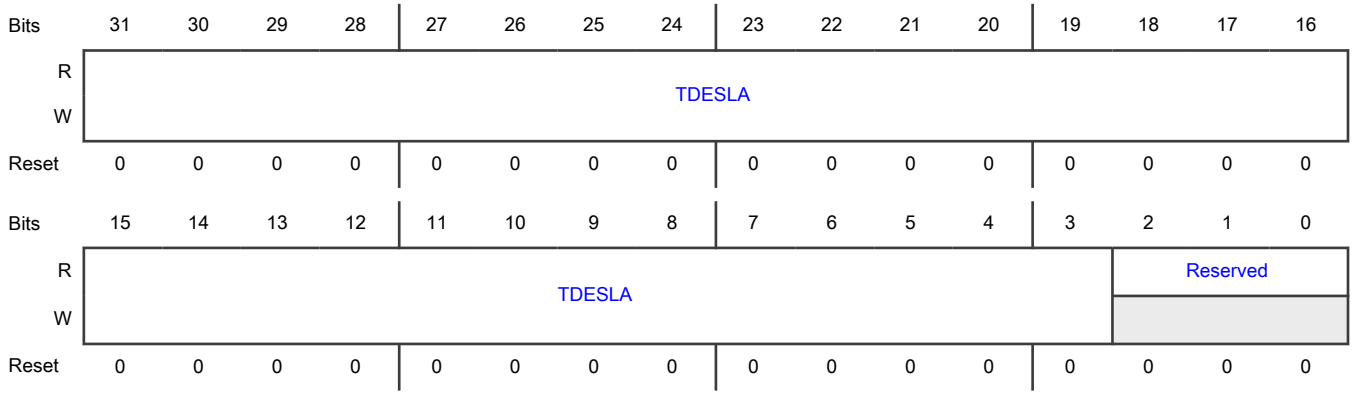
Offset

Register	Offset
DMA_CH0_TxDesc_List_Address	1114h

Function

The Channeli Tx Descriptor List Address register points the DMA to the start of Transmit descriptor list. The descriptor lists reside in the physical memory space of the application and must be Word, Dword, or Lword-aligned (for 32-bit, 64-bit, or 128-bit data bus). The DMA internally converts it to bus width aligned address by making the corresponding LSB to low. You can write to this register only when the Tx DMA has stopped, that is, the ST bit is set to zero in DMA_CH0_Tx_Control register. When stopped, this register can be written with a new descriptor list address. When you set the ST bit to 1, the DMA takes the newly-programmed descriptor base address. If this register is not changed when the ST bit is set to 0, the DMA takes the descriptor address where it was stopped earlier.

Diagram



Fields

Field	Function
31-3 TDESLA	Start of Transmit List This field contains the base address of the first descriptor in the Transmit descriptor list. The DMA ignores the LSB bits (1:0, 2:0, or 3:0) for 32-bit, 64-bit, or 128-bit bus width and internally takes these bits as all-zero. Therefore, these LSB bits are read-only (RO). The width of this field depends on the configuration: - 31:2 for 32-bit configuration - 31:3 for 64-bit configuration - 31:4 for 128-bit configuration

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field is not testable by an automated register test.
2-0 —	Reserved

76.17.342 DMA Channel 0 Rx Descriptor List Address (DMA_CH0_RxDesc_List_Address)

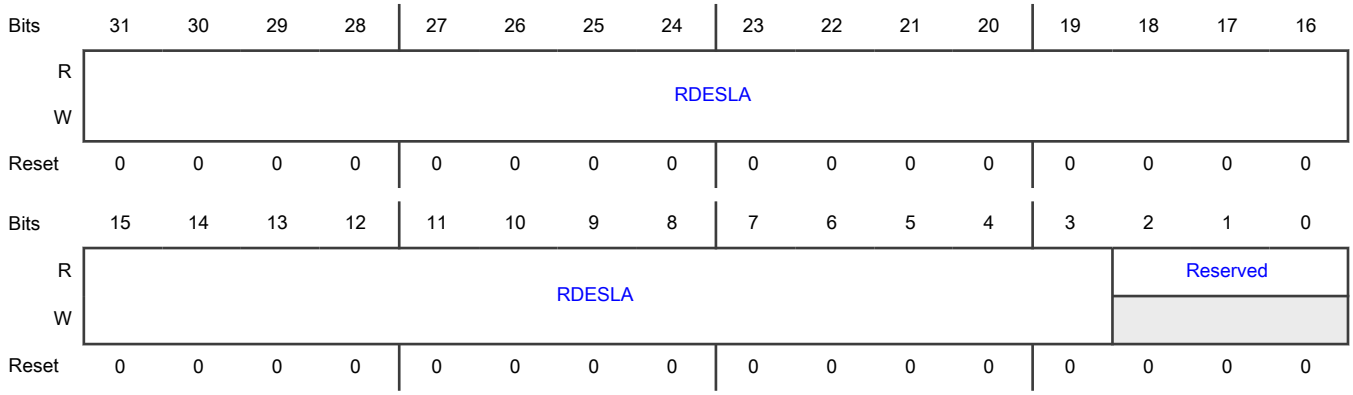
Offset

Register	Offset
DMA_CH0_RxDesc_List_Address	111Ch

Function

The Channeli Rx Descriptor List Address register points the DMA to the start of Receive descriptor list. This register points to the start of the Receive Descriptor List. The descriptor lists reside in the physical memory space of the application and must be Word, Dword, or Lword-aligned (for 32-bit, 64-bit, or 128-bit data bus). The DMA internally converts it to bus width aligned address by making the corresponding LS bits low. Writing to this register is permitted only when reception is stopped. When stopped, this register must be written to before the receive Start command is given. You can write to this register only when Rx DMA has stopped, that is, SR bit is set to zero in DMA_CH0_Rx_Control register. When stopped, this register can be written with a new descriptor list address. When you set the SR bit to 1, the DMA takes the newly programmed descriptor base address.

Diagram



Fields

Field	Function
31-3	Start of Receive List

Table continues on the next page...

Table continued from the previous page...

Field	Function
RDESLA	This field contains the base address of the first descriptor in the Rx Descriptor list. The DMA ignores the LSB bits (1:0, 2:0, or 3:0) for 32-bit, 64-bit, or 128-bit bus width and internally takes these bits as all-zero. Therefore, these LSB bits are read-only (RO). The width of this field depends on the configuration: - 31:2 for 32-bit configuration - 31:3 for 64-bit configuration - 31:4 for 128-bit configuration **NOTE** This field is not testable by an automated register test.
2-0 —	Reserved

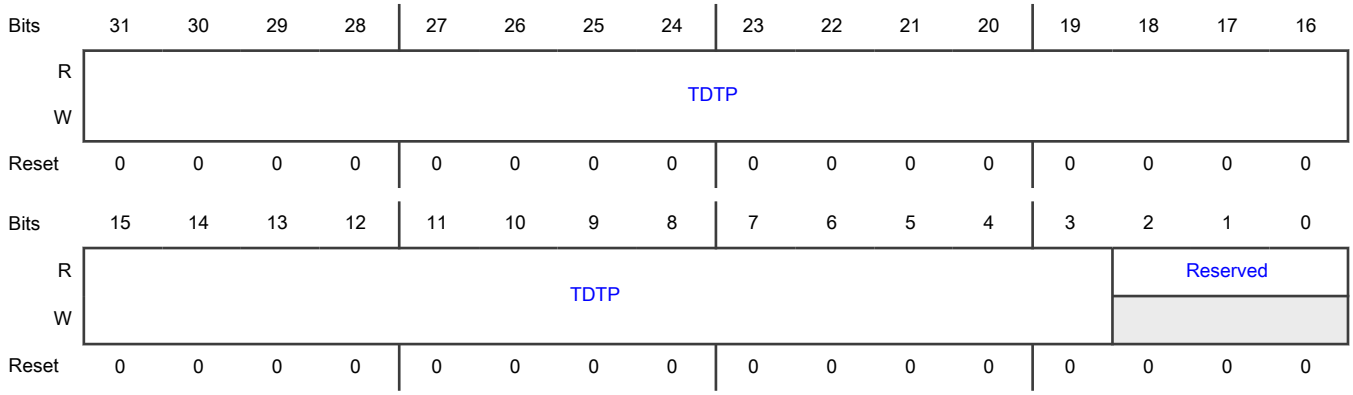
76.17.343 DMA Channel 0 Tx Descriptor Tail Pointer (DMA_CH0_TxDesc_Tail_Pointer)

Offset

Register	Offset
DMA_CH0_TxDesc_Tail_Pointer	1120h

Function
The Channeli Tx Descriptor Tail Pointer register points to an offset from the base and indicates the location of the last valid descriptor.

Diagram



Fields

Field	Function
31-3 TDTP	Transmit Descriptor Tail Pointer

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field contains the tail pointer for the Tx descriptor ring. The software writes the tail pointer to add more descriptors to the Tx channel. The hardware tries to transmit all packets referenced by the descriptors between the head and the tail pointer registers. The width of this field depends on the configuration: - 31:2 for 32-bit configuration - 31:3 for 64-bit configuration - 31:4 for 128-bit configuration
2-0 —	Reserved

76.17.344 DMA Channel 0 Rx Descriptor Tail Pointer (DMA_CH0_RxDesc_Tail_Pointer)

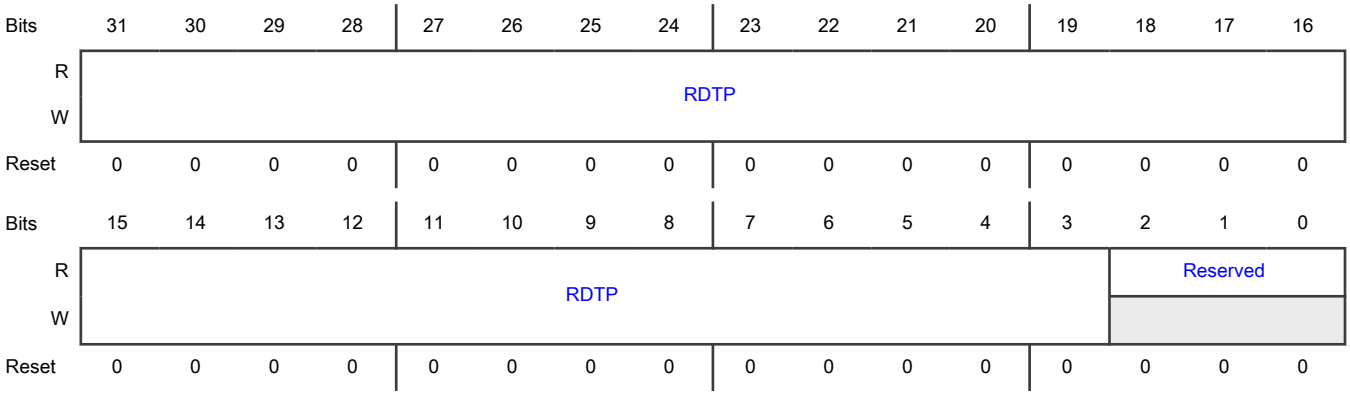
Offset

Register	Offset
DMA_CH0_RxDesc_Tail_Pointer	1128h

Function

The Channeli Rx Descriptor Tail Pointer Points to an offset from the base and indicates the location of the last valid descriptor.

Diagram



Fields

Field	Function
31-3 RDTP	Receive Descriptor Tail Pointer This field contains the tail pointer for the Rx descriptor ring. The software writes the tail pointer to add more descriptors to the Rx channel. The hardware tries to write all received packets to the descriptors referenced between the head and the tail pointer registers. The width of this field depends on the configuration: - 31:2 for 32-bit configuration - 31:3 for 64-bit configuration - 31:4 for 128-bit configuration
2-0 —	Reserved

76.17.345 DMA Channel 0 Tx Descriptor Ring Length (DMA_CH0_TxDesc_Ring_Length)

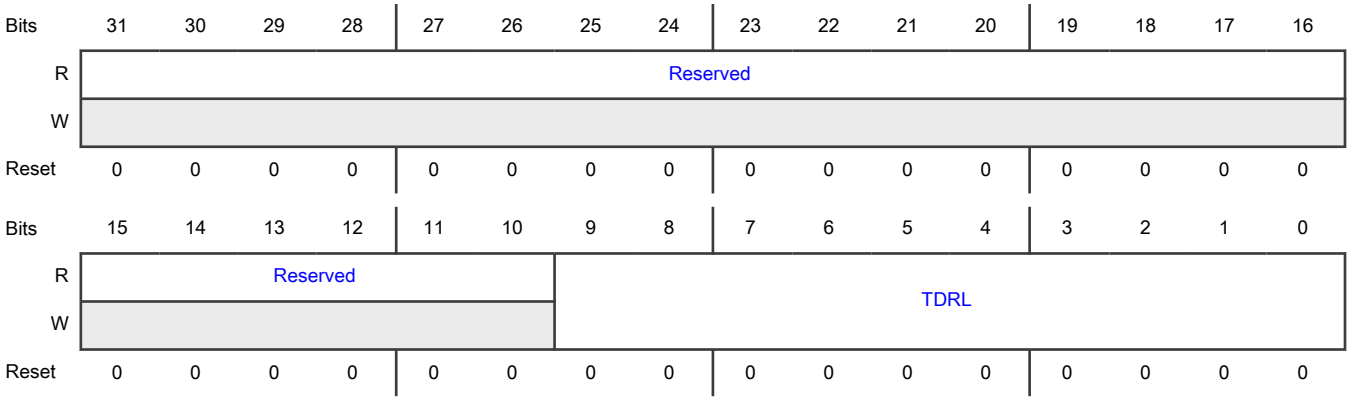
Offset

Register	Offset
DMA_CH0_TxDesc_Ring_Length	112Ch

Function

The Tx Descriptor Ring Length register contains the length of the Transmit descriptor ring.

Diagram



Fields

Field	Function
31-10 —	Reserved
9-0 TDRL	Transmit Descriptor Ring Length Transmit Descriptor Ring Length This field sets the maximum number of Tx descriptors in the circular descriptor ring. The maximum number of descriptors is limited to 1K descriptors. NXP recommends a minimum ring descriptor length of 4. For example, You can program any value up to 0x3FF in this field. This field is 10 bits wide, if you program 0x3FF, you can have 1024 descriptors. If you want to have 10 descriptors, program it to a value of 0x9.

76.17.346 DMA Channel 0 Rx Control 2 (DMA_CH0_Rx_Control2)

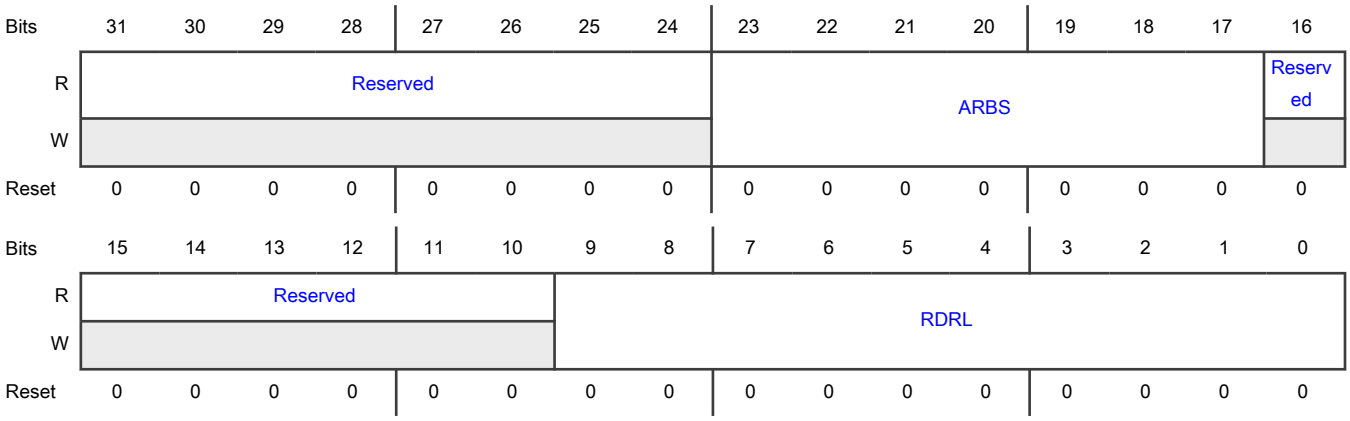
Offset

Register	Offset
DMA_CH0_Rx_Control2	1130h

Function

The Channeli Receive Control register controls the Rx features such as Rx Descriptor Ring Length and Alternate Rx Buffer Size.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-17 ARBS	Alternate Receive Buffer Size Indicates size in bytes for Buffer 1 when ARBS is programmed to a non-zero value (when split header feature is not enabled). When split header feature is enabled, ARBS indicates the buffer size for header data. The maximum alternate buffer is limited to 1020, 1016 or 1008-bytes depending on the data bus widths (32-bit, 64-bit, or 128-bit respectively). When ARBS=0, Rx Buffer1 and Rx Buffer2 sizes are based on RBSZ field of DMA_CH(#i)_Rx_Control. Width of ARBS field is 8, 7 or 6-bits depending on the data bus widths (32-bit, 64-bit, or 128-bit respectively)
16-10 —	Reserved
9-0 RDRL	Receive Descriptor Ring Length This register sets the maximum number of Rx descriptors in the circular descriptor ring. The maximum number of descriptors is limited to 1K descriptors. For example, You can program any value up to 0x3FF in this field. This field is 10 bits wide, if you program 0x3FF, you can have 1024 descriptors. If you want to have 10 descriptors, program it to a value of 0x9.

76.17.347 DMA Channel 0 Interrupt Enable (DMA_CH0_Interrupt_Enable)

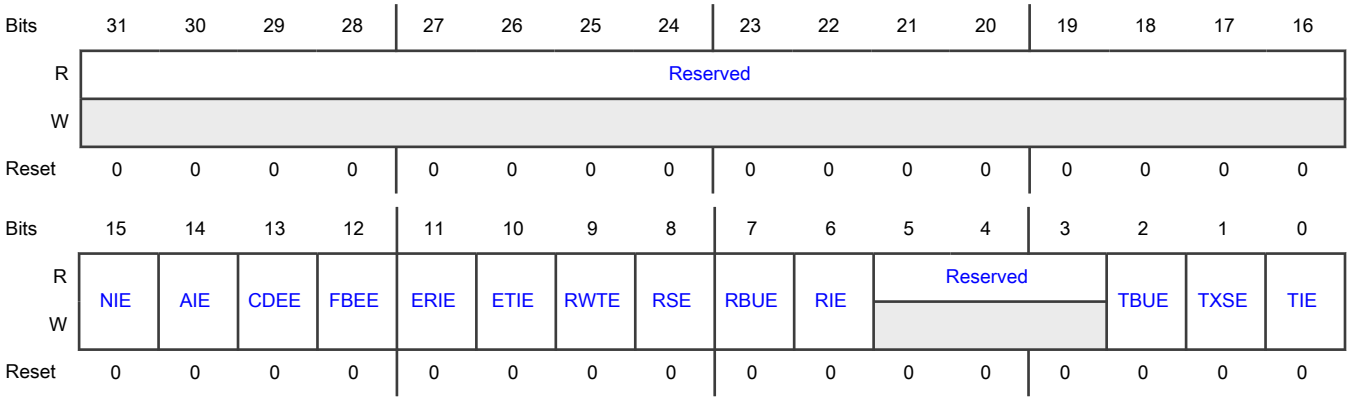
Offset

Register	Offset
DMA_CH0_Interrupt_Enable	1134h

Function

The Channeli Interrupt Enable register enables the interrupts reported by the Status register.

Diagram



Fields

Field	Function
31-16 —	Reserved
15 NIE	Normal Interrupt Summary Enable When this bit is set, the normal interrupt summary is enabled. This bit enables the following interrupts in the DMA_CH0_Status register: - Bit 0: Transmit Interrupt - Bit 2: Transmit Buffer Unavailable - Bit 6: Receive Interrupt - Bit 11: Early Receive Interrupt When this bit is reset, the normal interrupt summary is disabled. 0b - Normal Interrupt Summary is disabled 1b - Normal Interrupt Summary is enabled
14 AIE	Abnormal Interrupt Summary Enable When this bit is set, the abnormal interrupt summary is enabled. This bit enables the following interrupts in the DMA_CH0_Status register: - Bit 1: Transmit Process Stopped - Bit 7: Rx Buffer Unavailable - Bit 8: Receive Process Stopped - Bit 9: Receive Watchdog Timeout - Bit 10: Early Transmit Interrupt - Bit 12: Fatal Bus Error - Bit 13: Context Descriptor Error When this bit is reset, the abnormal interrupt summary is disabled. 0b - Abnormal Interrupt Summary is disabled 1b - Abnormal Interrupt Summary is enabled
13 CDEE	Context Descriptor Error Enable When this bit is set along with the AIE bit, the Descriptor error interrupt is enabled. When this bit is reset, the Descriptor error interrupt is disabled. 0b - Context Descriptor Error is disabled 1b - Context Descriptor Error is enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
12 FBEE	Fatal Bus Error Enable When this bit is set along with the AIE bit, the Fatal Bus error interrupt is enabled. When this bit is reset, the Fatal Bus Error error interrupt is disabled. 0b - Fatal Bus Error is disabled 1b - Fatal Bus Error is enabled
11 ERIE	Early Receive Interrupt Enable When this bit is set along with the NIE bit, the Early Receive interrupt is enabled. When this bit is reset, the Early Receive interrupt is disabled. 0b - Early Receive Interrupt is disabled 1b - Early Receive Interrupt is enabled
10 ETIE	Early Transmit Interrupt Enable When this bit is set along with the AIE bit, the Early Transmit interrupt is enabled. When this bit is reset, the Early Transmit interrupt is disabled. 0b - Early Transmit Interrupt is disabled 1b - Early Transmit Interrupt is enabled
9 RWTE	Receive Watchdog Timeout Enable When this bit is set along with the AIE bit, the Receive Watchdog Timeout interrupt is enabled. When this bit is reset, the Receive Watchdog Timeout interrupt is disabled. 0b - Receive Watchdog Timeout is disabled 1b - Receive Watchdog Timeout is enabled
8 RSE	Receive Stopped Enable When this bit is set along with the AIE bit, the Receive Stopped Interrupt is enabled. When this bit is reset, the Receive Stopped interrupt is disabled. 0b - Receive Stopped is disabled 1b - Receive Stopped is enabled
7 RBUE	Receive Buffer Unavailable Enable When this bit is set along with the AIE bit, the Receive Buffer Unavailable interrupt is enabled. When this bit is reset, the Receive Buffer Unavailable interrupt is disabled. 0b - Receive Buffer Unavailable is disabled 1b - Receive Buffer Unavailable is enabled
6 RIE	Receive Interrupt Enable When this bit is set along with the NIE bit, the Receive Interrupt is enabled. When this bit is reset, the Receive Interrupt is disabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Receive Interrupt is disabled 1b - Receive Interrupt is enabled
5-3 —	Reserved
2 TBUE	Transmit Buffer Unavailable Enable When this bit is set along with the NIE bit, the Transmit Buffer Unavailable interrupt is enabled. When this bit is reset, the Transmit Buffer Unavailable interrupt is disabled. 0b - Transmit Buffer Unavailable is disabled 1b - Transmit Buffer Unavailable is enabled
1 TXSE	Transmit Stopped Enable When this bit is set along with the AIE bit, the Transmission Stopped interrupt is enabled. When this bit is reset, the Transmission Stopped interrupt is disabled. 0b - Transmit Stopped is disabled 1b - Transmit Stopped is enabled
0 TIE	Transmit Interrupt Enable When this bit is set along with the NIE bit, the Transmit Interrupt is enabled. When this bit is reset, the Transmit Interrupt is disabled. 0b - Transmit Interrupt is disabled 1b - Transmit Interrupt is enabled

76.17.348 DMA Channel 0 Rx Interrupt Watchdog Timer (DMA_CH0_Rx_Interrupt_Watchdog_Timer)

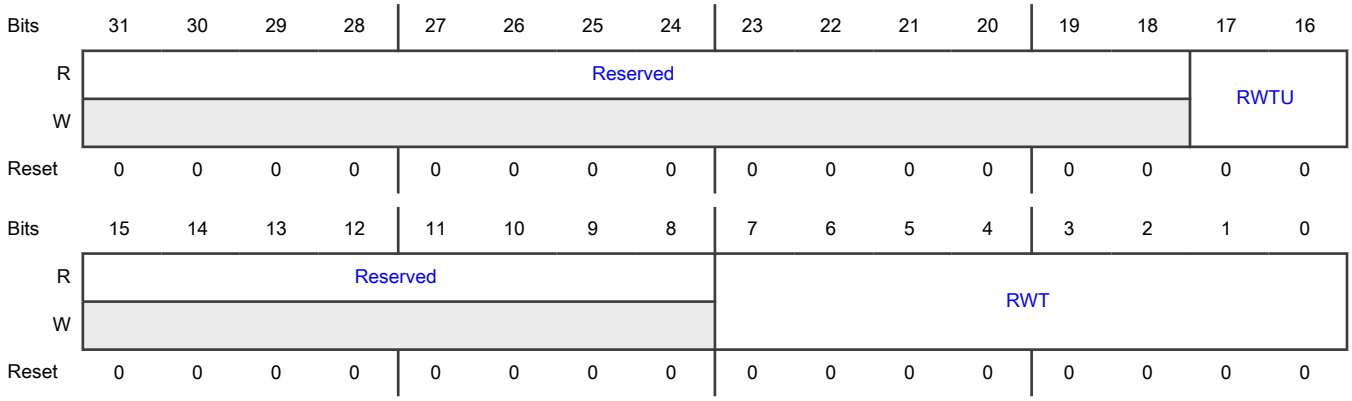
Offset

Register	Offset
DMA_CH0_Rx_Interrupt_Watchdog_Timer	1138h

Function

The Receive Interrupt Watchdog Timer register indicates the watchdog timeout for Receive Interrupt (RI) from the DMA. When this register is written with a non-zero value, it enables the watchdog timer for the RI bit of the DMA_CHi_Status register.

Diagram



Fields

Field	Function
31-18 —	Reserved
17-16 RWTU	Receive Interrupt Watchdog Timer Count Units This fields indicates the number of system clock cycles corresponding to one unit in RWT field. - 2'b00: 256 - 2'b01: 512 - 2'b10: 1024 - 2'b11: 2048 For example, when RWT=2 and RWTU=1, the watchdog timer is set for 2*512=1024 system clock cycles. **NOTE** This field is not testable by an automated register test.
15-8 —	Reserved
7-0 RWT	Receive Interrupt Watchdog Timer Count This field indicates the number of system clock cycles, multiplied by factor indicated in RWTU field, for which the watchdog timer is set. The watchdog timer is triggered with the programmed value after the Rx DMA completes the transfer of a packet for which the RI bit is not set in the DMA_CH(#i)_Status register, because of the setting of Interrupt Enable bit in the corresponding descriptor RDES3[30]. When the watchdog timer runs out, the RI bit is set and the timer is stopped. The watchdog timer is reset when the RI bit is set high because of automatic setting of RI as per the Interrupt Enable bit RDES3[30] of any received packet. **NOTE** This field is not testable by an automated register test.

76.17.349 DMA Channel 0 Slot Function Control Status (DMA_CH0_Slot_Function_Control_Status)

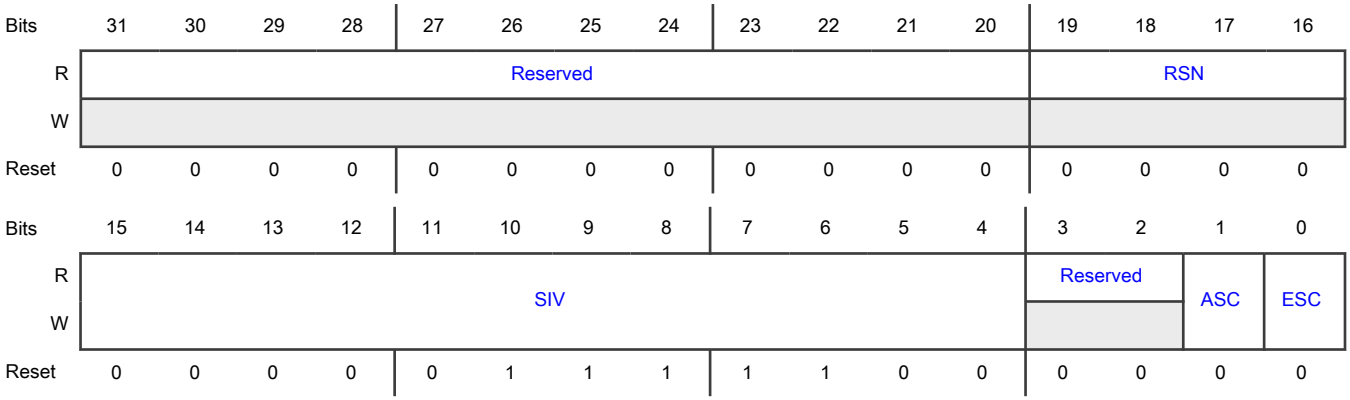
Offset

Register	Offset
DMA_CH0_Slot_Function_Control_Status	113Ch

Function

The Slot Function Control and Status register contains the control bits for slot function and the status for Transmit path.

Diagram



Fields

Field	Function
31-20 —	Reserved
19-16 RSN	Reference Slot Number This field gives the current value of the reference slot number in the DMA. It is used for slot comparison. **NOTE** This field is not testable by an automated register test.
15-4 SIV	Slot Interval Value This field controls the period of the slot interval in which the TxDMA fetches the scheduled packets. A value of 0 specifies the slot interval of 1 us while the maximum value 4095 specifies the slot interval of 4096us. The default/reset value is 0x07C which corresponds to slot interval of 125us
3-2 —	Reserved
1	Advance Slot Check

Table continues on the next page...

Table continued from the previous page...

Field	Function
ASC	When set, this bit enables the DMA to fetch the data from the buffer when the slot number (SLOTNUM) programmed in the Tx descriptor is - equal to the reference slot number given in the RSN field or - ahead of the reference slot number by up to two slots This bit is applicable only when the ESC bit is set. 0b - Advance Slot Check is disabled 1b - Advance Slot Check is enabled
0 ESC	Enable Slot Comparison When set, this bit enables the checking of the slot numbers programmed in the Tx descriptor with the current reference given in the RSN field. The DMA fetches the data from the corresponding buffer only when the slot number is - equal to the reference slot number or - ahead of the reference slot number by one slot When reset, this bit disables the checking of the slot numbers. The DMA fetches the data immediately after the descriptor is processed. Note: The UFO (UDP Fragmentation over IPv4)/TSO/USO should not be enabled along with TBS/AVB Slot number check. The UFO/TSO/USO involves multiple packets/segments/fragments transmission for single packet received from application and the slot number check are applicable for fetching of only first segment/fragment. As a result it might be difficult for software to specify slot number for subsequent packets. 0b - Slot Comparison is disabled 1b - Slot Comparison is enabled

76.17.350 DMA Channel 0 Current Application Transmit Descriptor (DMA_CH0_Current_App_TxDesc)

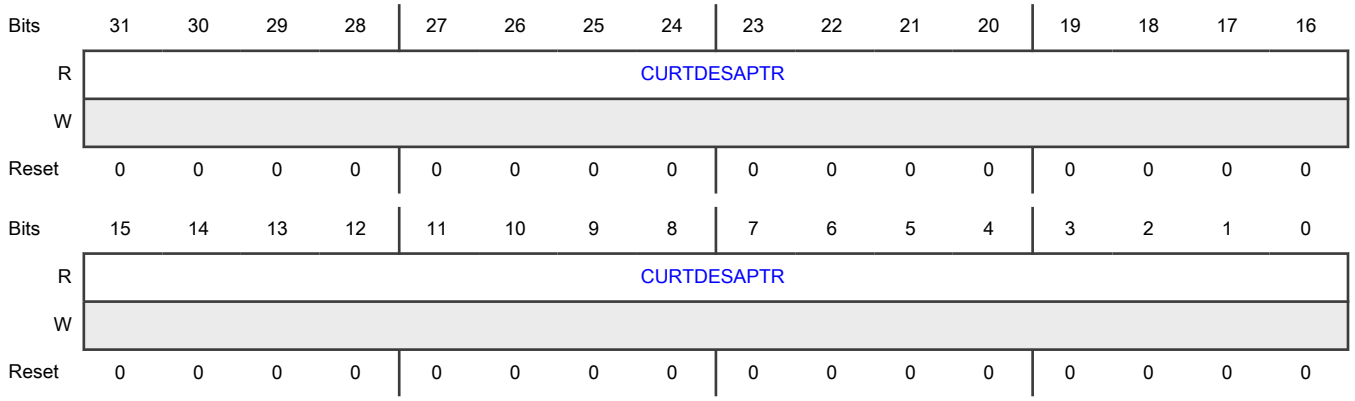
Offset

Register	Offset
DMA_CH0_Current_App_TxDesc	1144h

Function

The Channeli Current Application Transmit Descriptor register points to the current Transmit descriptor read by the DMA.

Diagram



Fields

Field	Function
31-0 CURTDESAPTR	Application Transmit Descriptor Address Pointer The DMA updates this pointer during Tx operation. This pointer is cleared on reset.

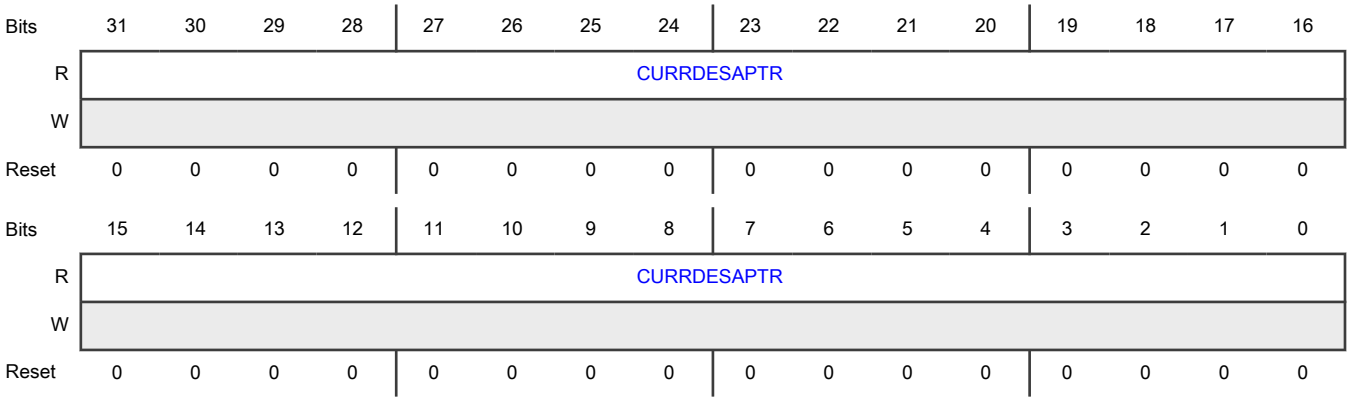
76.17.351 DMA Channel 0 Current Application Receive Descriptor (DMA_CH0_Current_App_RxDesc)

Offset

Register	Offset
DMA_CH0_Current_App_RxDesc	114Ch

Function
The Channeli Current Application Receive Descriptor register points to the current Receive descriptor read by the DMA.

Diagram



Fields

Field	Function
31-0 CURRDESAPTR	Application Receive Descriptor Address Pointer The DMA updates this pointer during Rx operation. This pointer is cleared on reset.

76.17.352 DMA Channel 0 Current Application Transmit Buffer (DMA_CH0_Current_App_TxBuffer)

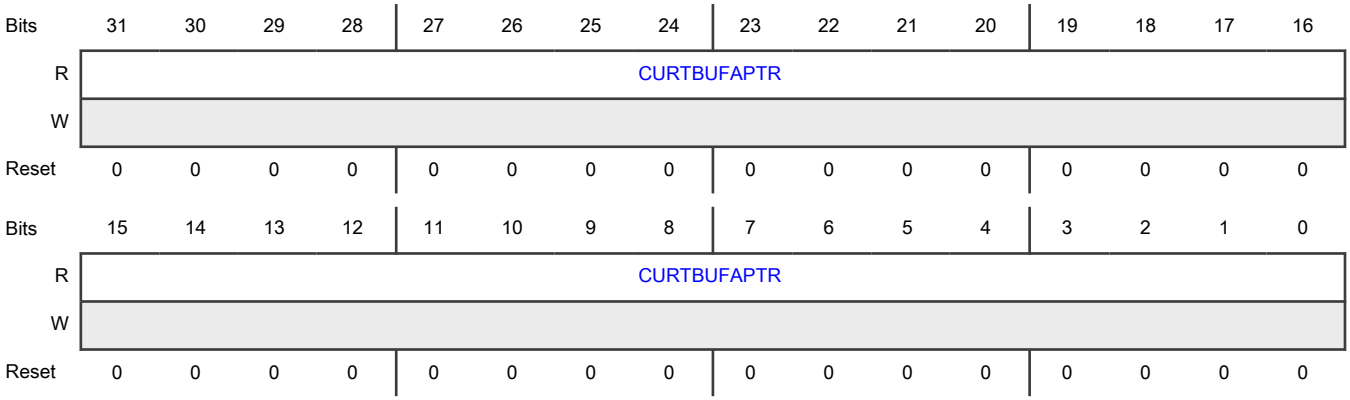
Offset

Register	Offset
DMA_CH0_Current_App_TxBuffer	1154h

Function

The Channeli Current Application Transmit Buffer Address register points to the current Tx buffer address read by the DMA.

Diagram



Fields

Field	Function
31-0 CURTBUFAPTR	Application Transmit Buffer Address Pointer The DMA updates this pointer during Tx operation. This pointer is cleared on reset.

76.17.353 DMA Channel 0 Current Application Receive Buffer (DMA_CH0_Current_App_RxBuffer)

Offset

Register	Offset
DMA_CH0_Current_App_RxBuffer	115Ch

Function

The Channel 0 Current Application Receive Buffer Address register points to the current Rx buffer address read by the DMA.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CURRBUFAPTR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CURRBUFAPTR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 CURRBUFAPTR	Application Receive Buffer Address Pointer The DMA updates this pointer during Rx operation. This pointer is cleared on reset.

76.17.354 DMA Channel 0 Status (DMA_CH0_Status)

Offset

Register	Offset
DMA_CH0_Status	1160h

Function

The software driver (application) reads the Status register during interrupt service routine or polling to determine the status of the DMA. Note: The number of DMA_CH(#i)_Status register in the configuration is the higher of number of Rx DMA Channels and Tx DMA Channels.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved										REB			TEB		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	NIS	AIS	CDE	FBE	ERI	ETI	RWT	RPS	RBU	RI	Reserved			TBU	TPS	TI
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-22 —	Reserved
21-19 REB	Rx DMA Error Bits This field indicates the type of error that caused a Bus Error. For example, error response on the AHB or AXI interface. - Bit 21 -- 1'b1: Error during data transfer by Rx DMA -- 1'b0: No Error during data transfer by Rx DMA - Bit 20 -- 1'b1: Error during descriptor access -- 1'b0: Error during data buffer access - Bit 19 -- 1'b1: Error during read transfer -- 1'b0: Error during write transfer This field is valid only when the FBE bit is set. This field does not generate an interrupt. **NOTE** This field is not testable by an automated register test.
18-16 TEB	Tx DMA Error Bits This field indicates the type of error that caused a Bus Error. For example, error response on the AHB or AXI interface. - Bit 18 -- 1'b1: Error during data transfer by Tx DMA -- 1'b0: No Error during data transfer by Tx DMA - Bit 17 -- 1'b1: Error during descriptor access -- 1'b0: Error during data buffer access - Bit 16 -- 1'b1: Error during read transfer -- 1'b0: Error during write transfer This field is valid only when the FBE bit is set. This field does not generate an interrupt. **NOTE** This field is not testable by an automated register test.
15 NIS	Normal Interrupt Summary Normal Interrupt Summary bit value is the logical OR of the following bits when the corresponding interrupt bits are enabled in the DMA_CH0_Interrupt_Enable register: - Bit 0: Transmit Interrupt - Bit 2: Transmit Buffer Unavailable - Bit 6: Receive Interrupt - Bit 11: Early Receive Interrupt Only unmasked bits (interrupts for which interrupt enable is set in DMA_CH0_Interrupt_Enable register) affect the Normal Interrupt Summary bit. This is a sticky bit. You must clear this bit (by writing 1 to this bit) each time a corresponding bit which causes NIS to be set is cleared. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **NOTE** This field is not testable by an automated register test. 0b - Normal Interrupt Summary status not detected 1b - Normal Interrupt Summary status detected
14 AIS	Abnormal Interrupt Summary Abnormal Interrupt Summary bit value is the logical OR of the following when the corresponding interrupt bits are enabled in the DMA_CH0_Interrupt_Enable register: - Bit 1: Transmit Process Stopped - Bit 7: Receive Buffer Unavailable - Bit 8: Receive Process Stopped - Bit 10: Early Transmit Interrupt - Bit 12: Fatal Bus Error - Bit 13: Context Descriptor Error Only unmasked bits affect the Abnormal Interrupt Summary bit. This is a sticky bit. You must clear this bit (by writing 1 to this bit) each time a corresponding bit, which causes AIS to be set, is cleared. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field is not testable by an automated register test. 0b - Abnormal Interrupt Summary status not detected 1b - Abnormal Interrupt Summary status detected
13 CDE	Context Descriptor Error This bit indicates that the DMA Tx/Rx engine received a descriptor error, which indicates invalid context in the middle of packet flow (intermediate descriptor) or all one's descriptor in Tx case and on Rx side it indicates DMA has read a descriptor with either of the buffer address as ones which is considered to be invalid. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **NOTE** This field is not testable by an automated register test. 0b - Context Descriptor Error status not detected 1b - Context Descriptor Error status detected
12 FBE	Fatal Bus Error This bit indicates that a bus error occurred (as described in the EB field). When this bit is set, the corresponding DMA channel engine disables all bus accesses. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **NOTE** This field is not testable by an automated register test. 0b - Fatal Bus Error status not detected 1b - Fatal Bus Error status detected
11 ERI	Early Receive Interrupt This bit when set indicates that the RxDMA has completed the transfer of packet data to the memory. In configurations supporting ERIC, - ERIC = 0: This bit is set only after the Rx DMA has completely filled up a receive buffer with packet data. - ERIC = 1: This bit is set after every burst transfer of data from the Rx DMA to the buffer. The setting of RI bit automatically clears this bit. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **NOTE** This field is not testable by an automated register test. 0b - Early Receive Interrupt status not detected 1b - Early Receive Interrupt status detected
10 ETI	Early Transmit Interrupt This bit when set indicates that the TxDMA has completed the transfer of packet data to the MTL TXFIFO memory. In configurations supporting ERIC: - ETIC = 0: This bit is set only after the Tx DMA has

Table continues on the next page...

Table continued from the previous page...

Field	Function
	transferred a complete packet to MTL. - ETIC = 1: This bit is set after completion of (partial) packet data transfer from buffers in the Transmit descriptor in which IOC = 1. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **NOTE** This field is not testable by an automated register test. 0b - Early Transmit Interrupt status not detected 1b - Early Transmit Interrupt status detected
9 RWT	Receive Watchdog Timeout This bit is asserted when a packet with length greater than 2,048 bytes (10,240 bytes when Jumbo Packet mode is enabled) is received. **NOTE** This field is not testable by an automated register test. 0b - Receive Watchdog Timeout status not detected 1b - Receive Watchdog Timeout status detected
8 RPS	Receive Process Stopped This bit is asserted when the Rx process enters the Stopped state. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **NOTE** This field is not testable by an automated register test. 0b - Receive Process Stopped status not detected 1b - Receive Process Stopped status detected
7 RBU	Receive Buffer Unavailable This bit indicates that the application owns the next descriptor in the Receive list, and the DMA cannot acquire it. The Rx process is suspended. To resume processing Rx descriptors, the application should change the ownership of the descriptor and issue a Receive Poll Demand command. If this command is not issued, the Rx process resumes when the next recognized incoming packet is received. In ring mode, the application should advance the Receive Descriptor Tail Pointer register of a channel. This bit is set only when the DMA owns the previous Rx descriptor. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **NOTE** This field is not testable by an automated register test. 0b - Receive Buffer Unavailable status not detected 1b - Receive Buffer Unavailable status detected
6 RI	Receive Interrupt

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This bit indicates that the packet reception is complete. When packet reception is complete, Bit 31 of RDES3 is reset in the last descriptor, and the specific packet status information is updated in the descriptor. The reception remains in the Running state. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **NOTE** This field is not testable by an automated register test. 0b - Receive Interrupt status not detected 1b - Receive Interrupt status detected
5-3 —	Reserved
2 TBU	Transmit Buffer Unavailable This bit indicates that the application owns the next descriptor in the Transmit list, and the DMA cannot acquire it. Transmission is suspended. The TPS0 field of the DMA_Debug_Status0 register explains the Transmit Process state transitions. To resume processing the Transmit descriptors, the application should do the following: - Change the ownership of the descriptor by setting Bit 31 of TDES3. - Issue a Transmit Poll Demand command. For ring mode, the application should advance the Transmit Descriptor Tail Pointer register of a channel. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **NOTE** This field is not testable by an automated register test. 0b - Transmit Buffer Unavailable status not detected 1b - Transmit Buffer Unavailable status detected
1 TPS	Transmit Process Stopped This bit is set when the transmission is stopped. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **NOTE** This field is not testable by an automated register test. 0b - Transmit Process Stopped status not detected 1b - Transmit Process Stopped status detected
0 TI	Transmit Interrupt This bit indicates that the packet transmission is complete. When transmission is complete, Bit 31 of TDES3 is reset in the last descriptor, and the specific packet status information is updated in the descriptor. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **NOTE** This field is not testable by an automated register test.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Transmit Interrupt status not detected 1b - Transmit Interrupt status detected

76.17.355 DMA Channel 0 Miss Frame Counter (DMA_CH0_Miss_Frame_Cnt)

Offset

Register	Offset
DMA_CH0_Miss_Frame_Cnt	1164h

Function

This register has the number of packet counter that got dropped by the DMA either due to Bus Error or due to programming RPF field in DMA_CH${i}_Rx_Control register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MFCO	Reserved			MFC											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15 MFCO	Overflow status of the MFC Counter When this bit is set then the MFC counter does not get incremented further. The bit gets cleared when this register is read. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - Miss Frame Counter overflow not occurred 1b - Miss Frame Counter overflow occurred

Table continues on the next page...

Table continued from the previous page...

Field	Function
14-11 —	Reserved
10-0 MFC	Dropped Packet Counters This counter indicates the number of packet counters that are dropped by the DMA either because of bus error or because of programing RPF field in DMA_CH${i}_Rx_Control register. The counter gets cleared when this register is read. Access restriction applies. Clears on read. Self-set to 1 on internal event.

76.17.356 DMA Channel 0 Rx Parser Accept Count (DMA_CH0_RXP_Accept_Cnt)

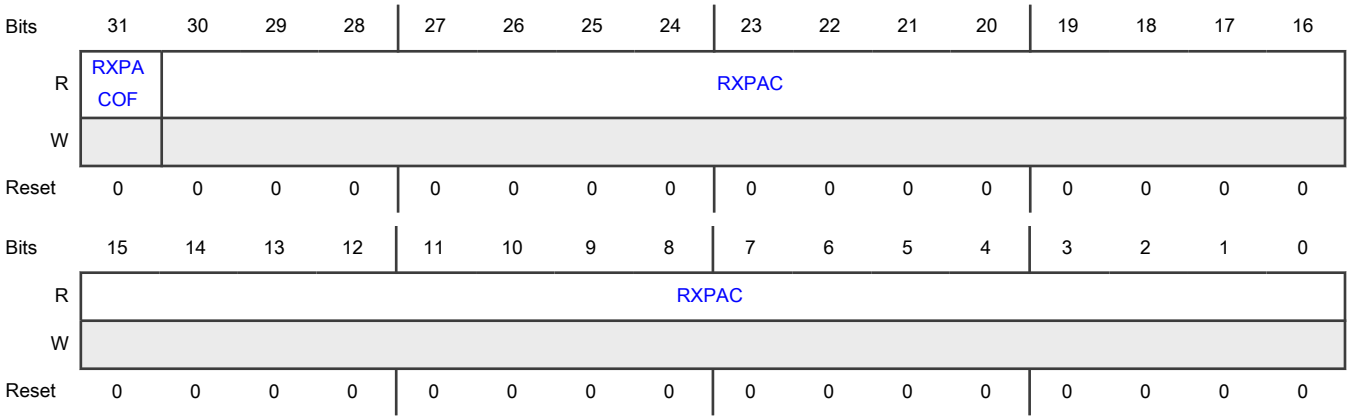
Offset

Register	Offset
DMA_CH0_RXP_Accept_Cnt	1168h

Function

The DMA_CH(#i)_RXP_Accept_Cnt registers provides the count of the number of frames accepted by Rx Parser.

Diagram



Fields

Field	Function
31 RXPACOF	Rx Parser Accept Counter Overflow Bit When set, this bit indicates that the RXPAC Counter field crossed the maximum limit. Access restriction applies. Clears on read. Self-set to 1 on internal event.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Rx Parser Accept Counter overflow not occurred 1b - Rx Parser Accept Counter overflow occurred
30-0 RXPAC	Rx Parser Accept Counter This 31-bit counter is implemented when a Rx Parser Accept a packet due to AF =1. The counter is cleared when the register is read.

76.17.357 DMA Channel 0 Rx ERI Count (DMA_CH0_RX_ERI_Cnt)

Offset

Register	Offset
DMA_CH0_RX_ERI_Cnt	116Ch

Function
The DMA_CH(#i)_RX_ERI_Cnt registers provides the count of the number of times ERI was asserted.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved				ECNT											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-12 —	Reserved
11-0 ECNT	ERI Counter When ERIC bit of DMA_CH(#i)_RX_Control register is set, this counter increments for burst transfer completed by the Rx DMA from the start of packet transfer. This counter is reset at the start of new packet.

76.17.358 DMA Channel 1 Control (DMA_CH1_Control)

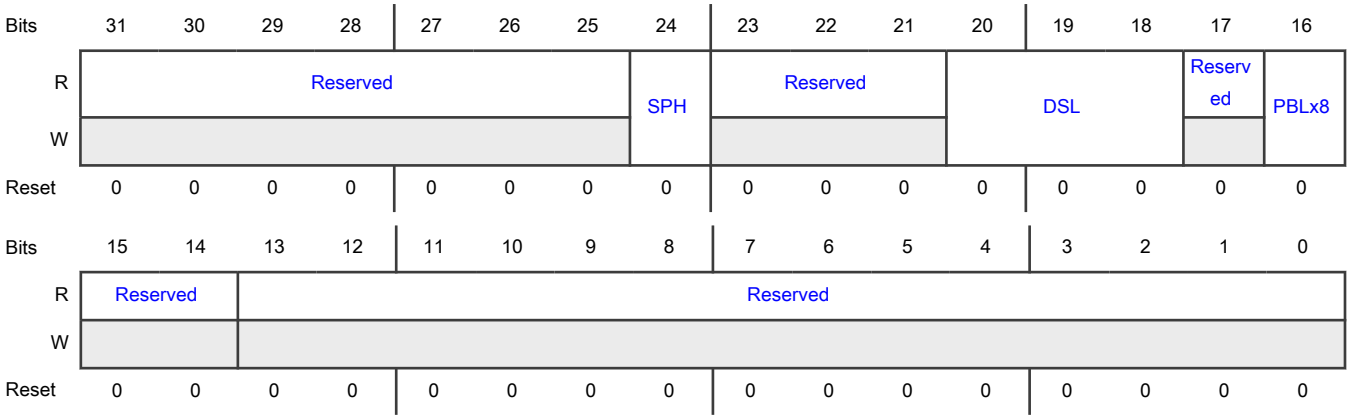
Offset

Register	Offset
DMA_CH1_Control	1180h

Function

The DMA Channeli Control register specifies the MSS value for segmentation, length to skip between two descriptors, and also the features such as header splitting and 8xPBL mode.

Diagram



Fields

Field	Function
31-25 —	Reserved
24 SPH	Split Headers When this bit is set, the DMA splits the header and payload in the Receive path. The DMA writes the header to the Buffer Address1 of RDES0. The DMA writes the payload to the buffer to which the Buffer Address2 is pointing. The software must ensure that the header fits into the Receive buffers. If the header length exceeds the receive buffer size, the DMA does not split the header and payload. This bit is available only if Enable Split Header Structure option is selected. 0b - Split Headers feature is disabled 1b - Split Headers feature is enabled
23-21 —	Reserved
20-18 DSL	Descriptor Skip Length

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This bit specifies the Word, Dword, or Lword number (depending on the 32-bit, 64-bit, or 128-bit bus) to skip between two unchained descriptors. The address skipping starts from the end of the current descriptor to the start of the next descriptor. When the DSL value is equal to zero, the DMA takes the descriptor table as contiguous.
17 —	Reserved
16 PBLx8	8xPBL mode When this bit is set, the PBL value programmed in Bits[21:16] in DMA_CH(#i)_Tx_Control and Bits[21:16] in DMA_CH(#i)_Rx_Control is multiplied by eight times. Therefore, the DMA transfers the data in 8, 16, 32, 64, 128, and 256 beats depending on the PBL value. 0b - 8xPBL mode is disabled 1b - 8xPBL mode is enabled
15-14 —	Reserved
13-0 —	Reserved

76.17.359 DMA Channel 1 Tx Control (DMA_CH1_Tx_Control)

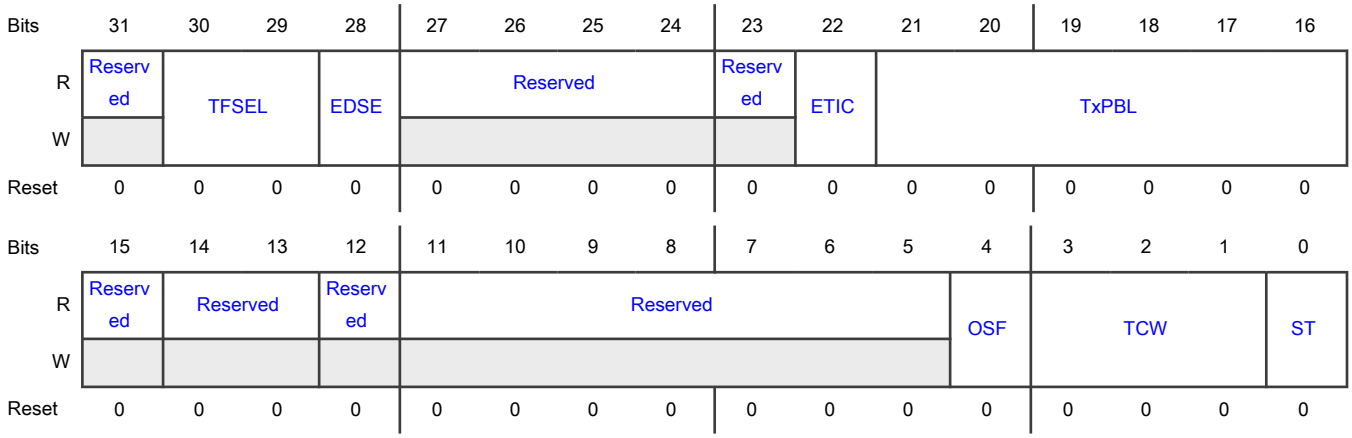
Offset

Register	Offset
DMA_CH1_Tx_Control	1184h

Function

The DMA Channeli Transmit Control register controls the Tx features such as PBL, TCP segmentation, and Tx Channel weights.

Diagram



Fields

Field	Function
31 —	Reserved
30-29 TFSEL	TBS Fetch Select Select bits for one of the four DMA_TBS_CTRL register fields (FTOS,FGSN,FTOV) for the channel Values: - 2'b00: DMA_TBS_CTRL0 - 2'b01: DMA_TBS_CTRL1 - 2'b10: DMA_TBS_CTRL2 - 2'b11: DMA_TBS_CTRL3 (Reserved if DWC_EQOS_NUM_DMA_TX_CH=3)
28 EDSE	Enhanced Descriptor Enable When this bit is set, the corresponding channel uses Enhanced Descriptors that are 32 Bytes for both Normal and Context Descriptors. When reset, the corresponding channel uses the descriptors that are 16 Bytes. 0b - Enhanced Descriptor is disabled 1b - Enhanced Descriptor is enabled
27-24 —	Reserved
23 —	Reserved
22 ETIC	Early Transmit Interrupt Control When this bit is set, Early Transmit Interrupt (ETI) status is set after completion of transfer of data from buffers of a transmit descriptor in which IOC bit (TDES2[31]) is set. When this bit is reset, ETI is set only after a complete packet is transferred to the MTL TX FIFO memory. 0b - Early Transmit Interrupt is disabled 1b - Early Transmit Interrupt is enabled
21-16	Transmit Programmable Burst Length

Table continues on the next page...

Table continued from the previous page...

Field	Function
TxPBL	These bits indicate the maximum number of beats to be transferred in one DMA block data transfer. The DMA always attempts max burst as specified in PBL each time it starts a burst transfer on the application bus. You can program PBL with any of the following values: 1, 2, 4, 8, 16, or 32. Any other value results in undefined behavior. To transfer more than 32 beats, perform the following steps: 1. Set the 8xPBL mode in DMA_CH0_Control register. 2. Set the TxPBL. Note: The maximum value of TxPBL must be less than or equal to half the Tx Queue size (TQS field of MTL_TxQ(#i)_Operation_Mode register) in terms of beats. This is required so that the Tx Queue has space to store at least another Tx PBL worth of data while the MTL Tx Queue Controller is transferring data to MAC. For example, in 64-bit data width configurations the total locations in Tx Queue of size 512 bytes is 64, TxPBL and 8xPBL needs to be programmed to less than or equal to 32.
15 —	Reserved
14-13 —	Reserved
12 —	Reserved
11-5 —	Reserved
4 OSF	Operate on Second Packet When this bit is set, it instructs the DMA to process the second packet of the Transmit data even before the status for the first packet is obtained. 0b - Operate on Second Packet disabled 1b - Operate on Second Packet enabled
3-1 TCW	Transmit Channel Weight This field indicates the weight assigned to the corresponding Transmit channel. When reset is complete, this field is set to 0 for all channels by default, resulting in equal weights to all channels.
0 ST	Start or Stop Transmission Command When this bit is set, transmission is placed in the Running state. The DMA checks the Transmit list at the current position for a packet to be transmitted. The DMA tries to acquire descriptor from either of the following positions: - The current position in the list This is the base address of the Transmit list set by the DMA_CH0_TxDesc_List_Address register. - The position at which the transmission was previously stopped If the DMA does not own the current descriptor, the transmission enters the Suspended state and the TBU bit of the DMA_CH0_Status register is set. The Start Transmission command is effective only when the transmission is stopped. If the command is issued before setting the DMA_CH0_TxDesc_List_Address register, the DMA behavior is unpredictable. When this bit is reset, the transmission process is placed in the Stopped state after completing the transmission of the current packet. The Next Descriptor position in the Transmit list is saved, and it becomes the current position when the transmission is restarted. To change the list address, you need to program DMA_CH0_TxDesc_List_Address register with a new value when this bit is reset. The new value is

Table continues on the next page...

Table continued from the previous page...

Field	Function
	considered when this bit is set again. The stop transmission command is effective only when the transmission of the current packet is complete or the transmission is in the Suspended state. **NOTE** This field is not testable by an automated register test. 0b - Stop Transmission Command 1b - Start Transmission Command

76.17.360 DMA Channel 1 Rx Control (DMA_CH1_Rx_Control)

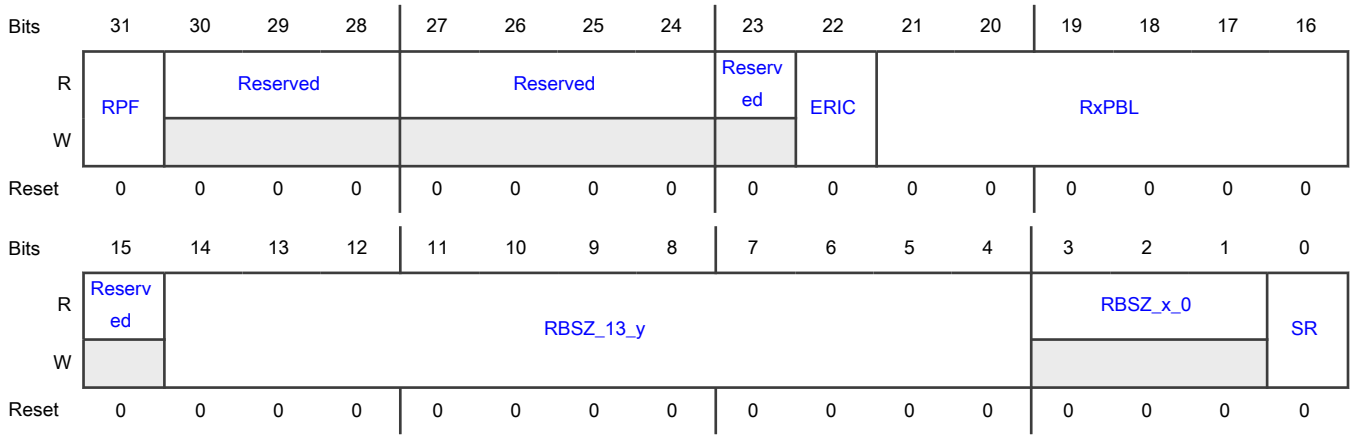
Offset

Register	Offset
DMA_CH1_Rx_Control	1188h

Function

The DMA Channeli Receive Control register controls the Rx features such as PBL, buffer size, and extended status.

Diagram



Fields

Field	Function
31 RPF	Rx Packet Flush. - 1: DWC_ether_qos automatically flushes the packet from the Rx Queues destined to this DMA Rx Channel, when it is stopped. When this bit remains set and the DMA is re-started by the software driver, the packets residing in the Rx Queues that were received when this RxDMA was stopped, are flushed out. The packets that are received by the MAC after the RxDMA is re-started are routed to the RxDMA. The flushing is done on the Read side of the Rx Queue. - 0: DWC_ether_qos does not flush the packet in

Table continues on the next page...

Table continued from the previous page...

Field	Function
	the Rx Queue destined to this RxDMA Channel when it is in STOP state. This might cause head-of-line blocking in the corresponding RxQueue. Note: The stopping of packet flow from a Rx DMA Channel to the application by setting RPF works only when there is one-to-one mapping of Rx Queue to Rx DMA channels. In Dynamic mapping mode, setting RPF bit in any DMA_CH(#i)_Rx_Control register might flush packets from unintended Rx Queues which are destined to the stopped Rx DMA Channel. 0b - Rx Packet Flush is disabled 1b - Rx Packet Flush is enabled
30-28 —	Reserved
27-24 —	Reserved
23 —	Reserved
22 ERIC	Early Receive Interrupt Control When this bit is set, Early Receive Interrupt (ERI) status is set after the completion of every burst transfer of data from the Rx DMA to the buffer. When this bit is reset, ERI is set only after a complete buffer is filled up by the RxDMA. 0b - Early Receive Interrupt is disabled 1b - Early Receive Interrupt is enabled
21-16 RxPBL	Receive Programmable Burst Length These bits indicate the maximum number of beats to be transferred in one DMA block data transfer. The DMA always attempts max burst as specified in PBL each time it starts a burst transfer on the application bus. You can program PBL with any of the following values: 1, 2, 4, 8, 16, or 32. Any other value results in undefined behavior. To transfer more than 32 beats, perform the following steps: 1. Set the 8xPBL mode in the DMA_CH0_Control register. 2. Set the RxPBL. Note: The maximum value of RxPBL must be less than or equal to half the Rx Queue size (RQS field of MTL_RxQ(#i)_Operation_Mode register) in terms of beats. This is required so that the Rx Queue has space to store at least another Rx PBL worth of data while the Rx DMA is transferring a block of data. For example, in 64-bit data width configurations the total locations in Rx Queue of size 512 bytes is 64, so RxPBL and 8xPBL needs to be programmed to less than or equal to 32.
15 —	Reserved
14-4 RBSZ_13_y	Receive Buffer size High RBSZ[13:0] is split into two fields higher RBSZ_13_y and lower RBSZ_x_0. The RBSZ[13:0] field indicates the size of the Rx buffers specified in bytes. The maximum buffer size is limited to 16K bytes. The buffer size is applicable to payload buffers when split headers are enabled. Note: The buffer size must be a multiple of 4, 8, or 16 depending on the data bus widths (32-bit, 64-bit, or 128-bit respectively).

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This is required even if the value of buffer address pointer is not aligned to data bus width. Hence the lower RBSZ_x_0 bits are read-only and the value is considered as all-zero. Thus the RBSZ_13_y indicates the buffer size in terms of locations (with the width same as bus-width).
3-1 RBSZ_x_0	Receive Buffer size Low RBSZ[13:0] is split into two fields RBSZ_13_y and RBSZ_x_0. The RBSZ_x_0 is the lower field whose width is based on data bus width of the configuration. This field is of width 2, 3, or 4 bits for 32-bit, 64-bit, or 128-bit data bus width respectively. This field is read-only (RO).
0 SR	Start or Stop Receive When this bit is set, the DMA tries to acquire the descriptor from the Receive list and processes the incoming packets. The DMA tries to acquire descriptor from either of the following positions: - The current position in the list This is the address set by the DMA_CH0_RxDesc_List_Address register. - The position at which the Rx process was previously stopped If the DMA does not own the current descriptor, the reception is suspended and the RBU bit of the DMA_CH0_Status register is set. The Start Receive command is effective only when the reception is stopped. If the command is issued before setting the DMA_CH0_RxDesc_List_Address register, the DMA behavior is unpredictable. When this bit is reset, the Rx DMA operation is stopped after the transfer of the current packet. The next descriptor position in the Receive list is saved, and it becomes the current position after the Rx process is restarted. The Stop Receive command is effective only when the Rx process is in the Running (waiting for Rx packet) or Suspended state. **NOTE** This field is not testable by an automated register test. 0b - Stop Receive 1b - Start Receive

76.17.361 DMA Channel 1 Tx Descriptor List Address (DMA_CH1_TxDesc_List_Address)

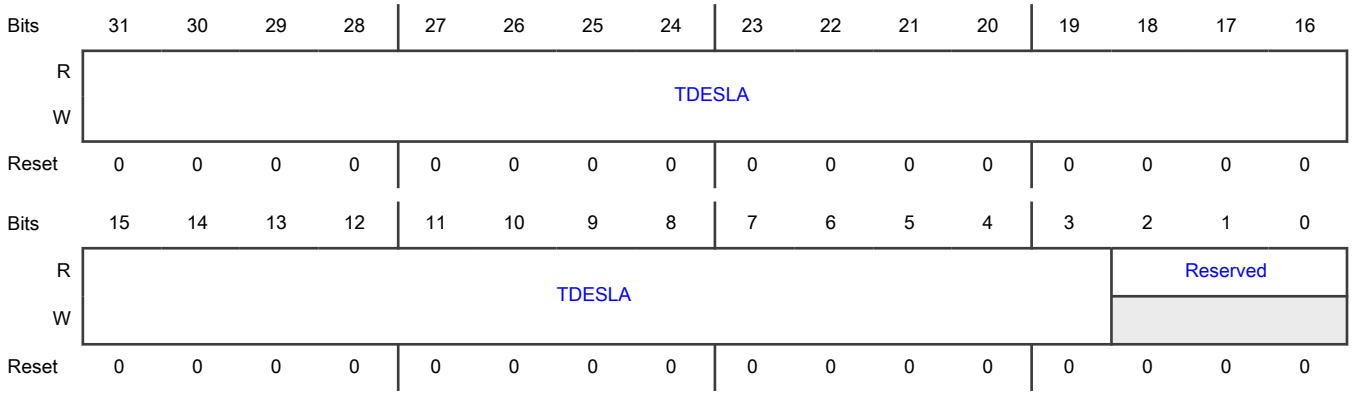
Offset

Register	Offset
DMA_CH1_TxDesc_List_Address	1194h

Function

The Channeli Tx Descriptor List Address register points the DMA to the start of Transmit descriptor list. The descriptor lists reside in the physical memory space of the application and must be Word, Dword, or Lword-aligned (for 32-bit, 64-bit, or 128-bit data bus). The DMA internally converts it to bus width aligned address by making the corresponding LSB to low. You can write to this register only when the Tx DMA has stopped, that is, the ST bit is set to zero in DMA_CH0_Tx_Control register. When stopped, this register can be written with a new descriptor list address. When you set the ST bit to 1, the DMA takes the newly-programmed descriptor base address. If this register is not changed when the ST bit is set to 0, the DMA takes the descriptor address where it was stopped earlier.

Diagram



Fields

Field	Function
31-3 TDESLA	Start of Transmit List This field contains the base address of the first descriptor in the Transmit descriptor list. The DMA ignores the LSB bits (1:0, 2:0, or 3:0) for 32-bit, 64-bit, or 128-bit bus width and internally takes these bits as all-zero. Therefore, these LSB bits are read-only (RO). The width of this field depends on the configuration: - 31:2 for 32-bit configuration - 31:3 for 64-bit configuration - 31:4 for 128-bit configuration **NOTE** This field is not testable by an automated register test.
2-0 —	Reserved

76.17.362 DMA Channel 1 Rx Descriptor List Address (DMA_CH1_RxDesc_List_Address)

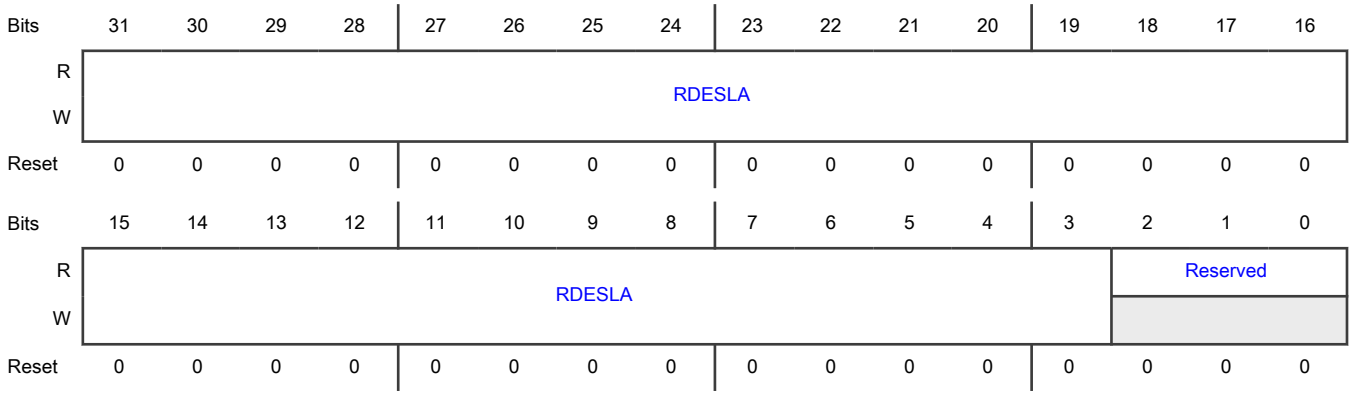
Offset

Register	Offset
DMA_CH1_RxDesc_List_Address	119Ch

Function

The Channeli Rx Descriptor List Address register points the DMA to the start of Receive descriptor list. This register points to the start of the Receive Descriptor List. The descriptor lists reside in the physical memory space of the application and must be Word, Dword, or Lword-aligned (for 32-bit, 64-bit, or 128-bit data bus). The DMA internally converts it to bus width aligned address by making the corresponding LS bits low. Writing to this register is permitted only when reception is stopped. When stopped, this register must be written to before the receive Start command is given. You can write to this register only when Rx DMA has stopped, that is, SR bit is set to zero in DMA_CH0_Rx_Control register. When stopped, this register can be written with a new descriptor list address. When you set the SR bit to 1, the DMA takes the newly programmed descriptor base address.

Diagram



Fields

Field	Function
31-3 RDESLA	Start of Receive List This field contains the base address of the first descriptor in the Rx Descriptor list. The DMA ignores the LSB bits (1:0, 2:0, or 3:0) for 32-bit, 64-bit, or 128-bit bus width and internally takes these bits as all-zero. Therefore, these LSB bits are read-only (RO). The width of this field depends on the configuration: - 31:2 for 32-bit configuration - 31:3 for 64-bit configuration - 31:4 for 128-bit configuration **NOTE** This field is not testable by an automated register test.
2-0 —	Reserved

76.17.363 DMA Channel 1 Tx Descriptor Tail Pointer (DMA_CH1_TxDesc_Tail_Pointer)

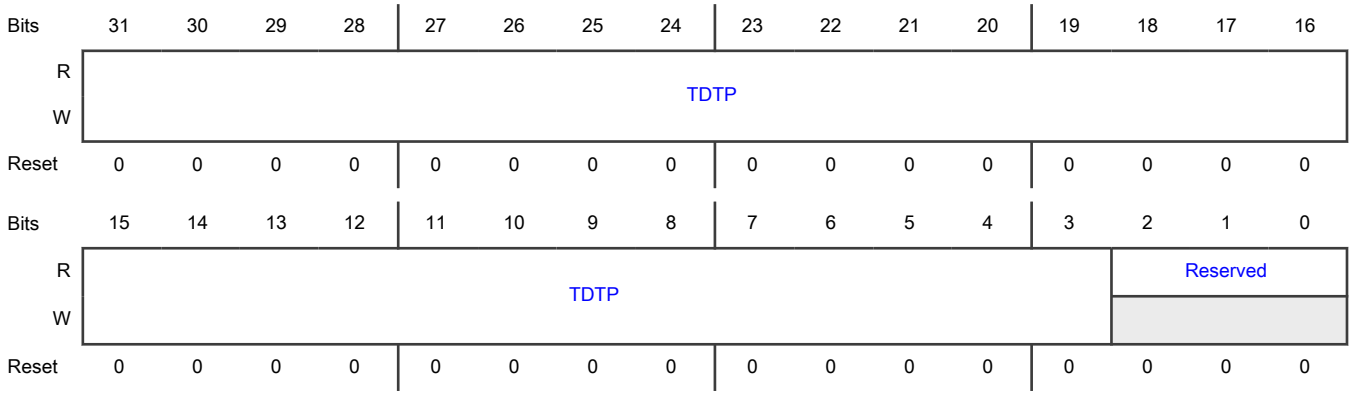
Offset

Register	Offset
DMA_CH1_TxDesc_Tail_Pointer	11A0h

Function

The Channeli Tx Descriptor Tail Pointer register points to an offset from the base and indicates the location of the last valid descriptor.

Diagram



Fields

Field	Function
31-3 TDTP	Transmit Descriptor Tail Pointer This field contains the tail pointer for the Tx descriptor ring. The software writes the tail pointer to add more descriptors to the Tx channel. The hardware tries to transmit all packets referenced by the descriptors between the head and the tail pointer registers. The width of this field depends on the configuration: - 31:2 for 32-bit configuration - 31:3 for 64-bit configuration - 31:4 for 128-bit configuration
2-0 —	Reserved

76.17.364 DMA Channel 1 Rx Descriptor Tail Pointer (DMA_CH1_RxDesc_Tail_Pointer)

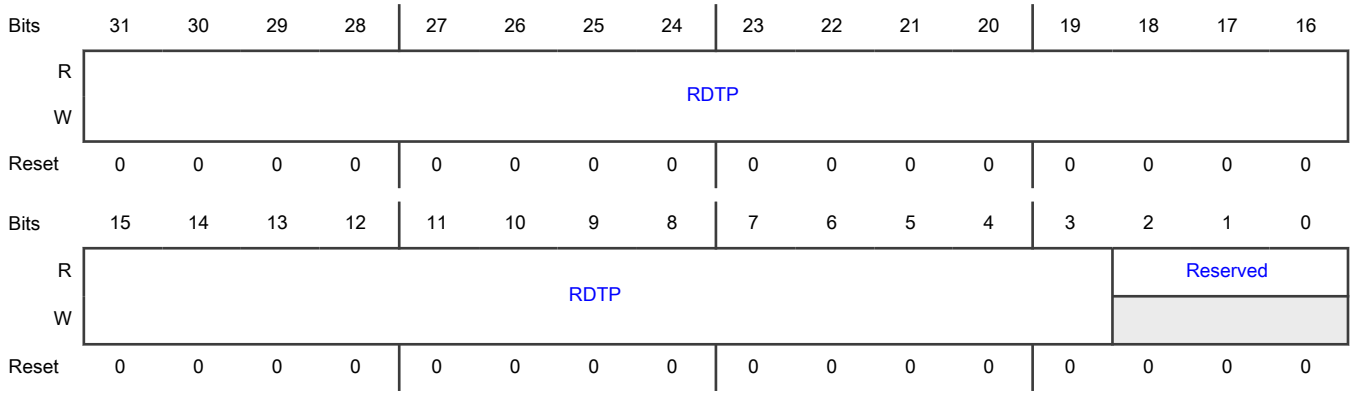
Offset

Register	Offset
DMA_CH1_RxDesc_Tail_Pointer	11A8h

Function

The Channeli Rx Descriptor Tail Pointer Points to an offset from the base and indicates the location of the last valid descriptor.

Diagram



Fields

Field	Function
31-3 RDTP	Receive Descriptor Tail Pointer This field contains the tail pointer for the Rx descriptor ring. The software writes the tail pointer to add more descriptors to the Rx channel. The hardware tries to write all received packets to the descriptors referenced between the head and the tail pointer registers. The width of this field depends on the configuration: - 31:2 for 32-bit configuration - 31:3 for 64-bit configuration - 31:4 for 128-bit configuration
2-0 —	Reserved

76.17.365 DMA Channel 1 Tx Descriptor Ring Length (DMA_CH1_TxDesc_Ring_Length)

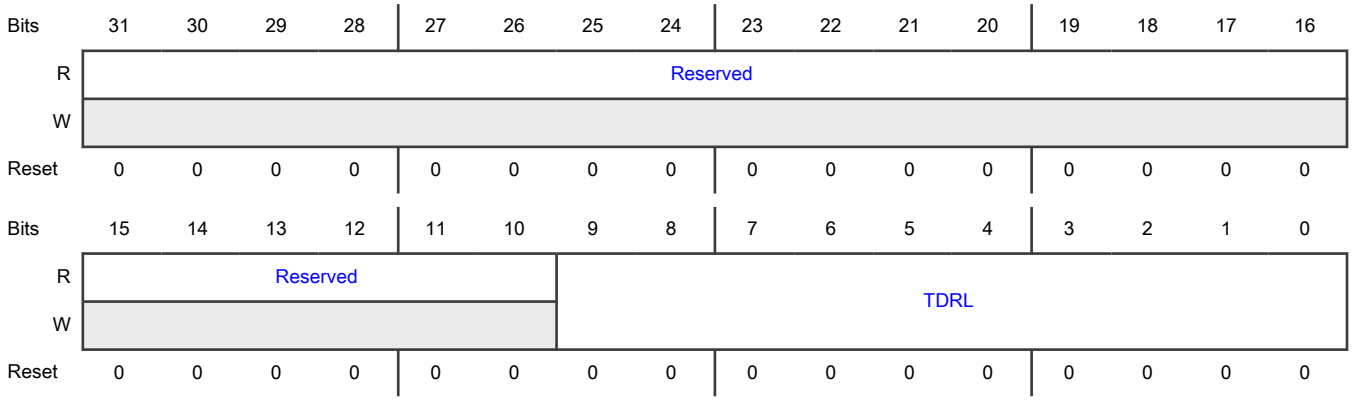
Offset

Register	Offset
DMA_CH1_TxDesc_Ring_Length	11ACh

Function

The Tx Descriptor Ring Length register contains the length of the Transmit descriptor ring.

Diagram



Fields

Field	Function
31-10 —	Reserved
9-0 TDRL	Transmit Descriptor Ring Length Transmit Descriptor Ring Length. This field sets the maximum number of Tx descriptors in the circular descriptor ring. The maximum number of descriptors is limited to 1K descriptors. NXP recommends a minimum ring descriptor length of 4. For example, You can program any value up to 0x3FF in this field. This field is 10 bits wide, if you program 0x3FF, you can have 1024 descriptors. If you want to have 10 descriptors, program it to a value of 0x9.

76.17.366 DMA Channel 1 Rx Control 2 (DMA_CH1_Rx_Control2)

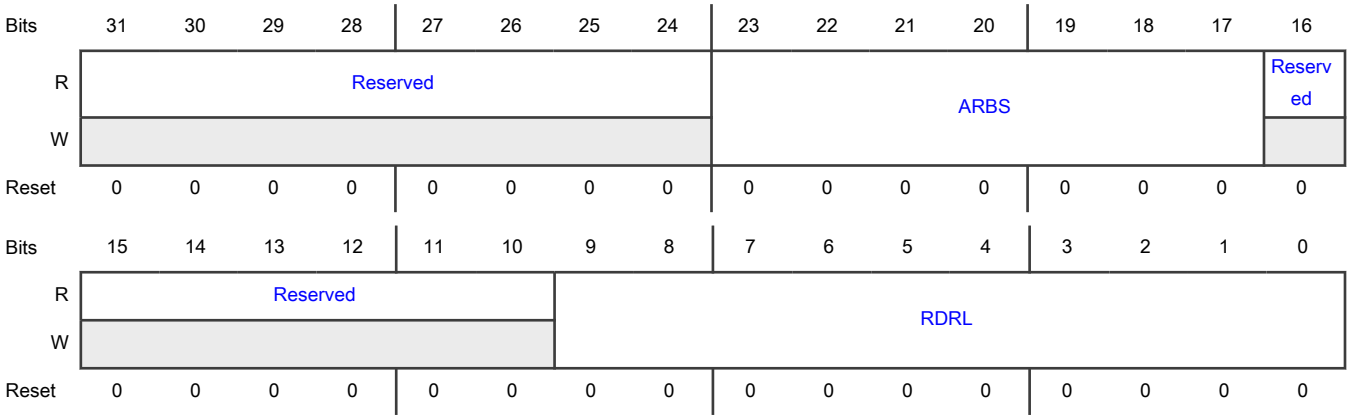
Offset

Register	Offset
DMA_CH1_Rx_Control2	11B0h

Function

The Channeli Receive Control register controls the Rx features such as Rx Descriptor Ring Length and Alternate Rx Buffer Size.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-17 ARBS	Alternate Receive Buffer Size Indicates size in bytes for Buffer 1 when ARBS is programmed to a non-zero value (when split header feature is not enabled). When split header feature is enabled, ARBS indicates the buffer size for header data. The maximum alternate buffer is limited to 1020, 1016 or 1008-bytes depending on the data bus widths (32-bit, 64-bit, or 128-bit respectively). When ARBS=0, Rx Buffer1 and Rx Buffer2 sizes are based on RBSZ field of DMA_CH(#i)_Rx_Control. Width of ARBS field is 8, 7 or 6-bits depending on the data bus widths (32-bit, 64-bit, or 128-bit respectively)
16-10 —	Reserved
9-0 RDRL	Receive Descriptor Ring Length This register sets the maximum number of Rx descriptors in the circular descriptor ring. The maximum number of descriptors is limited to 1K descriptors. For example, You can program any value up to 0x3FF in this field. This field is 10 bits wide, if you program 0x3FF, you can have 1024 descriptors. If you want to have 10 descriptors, program it to a value of 0x9.

76.17.367 DMA Channel 1 Interrupt Enable (DMA_CH1_Interrupt_Enable)

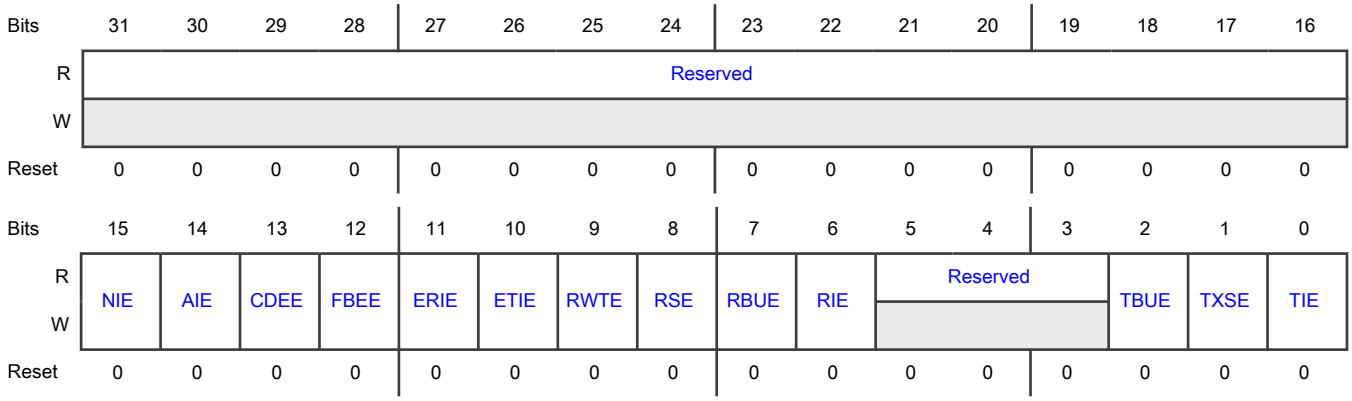
Offset

Register	Offset
DMA_CH1_Interrupt_Enable	11B4h

Function

The Channeli Interrupt Enable register enables the interrupts reported by the Status register.

Diagram



Fields

Field	Function
31-16 —	Reserved
15 NIE	Normal Interrupt Summary Enable When this bit is set, the normal interrupt summary is enabled. This bit enables the following interrupts in the DMA_CH0_Status register: - Bit 0: Transmit Interrupt - Bit 2: Transmit Buffer Unavailable - Bit 6: Receive Interrupt - Bit 11: Early Receive Interrupt When this bit is reset, the normal interrupt summary is disabled. 0b - Normal Interrupt Summary is disabled 1b - Normal Interrupt Summary is enabled
14 AIE	Abnormal Interrupt Summary Enable When this bit is set, the abnormal interrupt summary is enabled. This bit enables the following interrupts in the DMA_CH0_Status register: - Bit 1: Transmit Process Stopped - Bit 7: Rx Buffer Unavailable - Bit 8: Receive Process Stopped - Bit 9: Receive Watchdog Timeout - Bit 10: Early Transmit Interrupt - Bit 12: Fatal Bus Error - Bit 13: Context Descriptor Error When this bit is reset, the abnormal interrupt summary is disabled. 0b - Abnormal Interrupt Summary is disabled 1b - Abnormal Interrupt Summary is enabled
13 CDEE	Context Descriptor Error Enable When this bit is set along with the AIE bit, the Descriptor error interrupt is enabled. When this bit is reset, the Descriptor error interrupt is disabled. 0b - Context Descriptor Error is disabled 1b - Context Descriptor Error is enabled
12 FBEE	Fatal Bus Error Enable When this bit is set along with the AIE bit, the Fatal Bus error interrupt is enabled. When this bit is reset, the Fatal Bus Error error interrupt is disabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Fatal Bus Error is disabled 1b - Fatal Bus Error is enabled
11 ERIE	Early Receive Interrupt Enable When this bit is set along with the NIE bit, the Early Receive interrupt is enabled. When this bit is reset, the Early Receive interrupt is disabled. 0b - Early Receive Interrupt is disabled 1b - Early Receive Interrupt is enabled
10 ETIE	Early Transmit Interrupt Enable When this bit is set along with the AIE bit, the Early Transmit interrupt is enabled. When this bit is reset, the Early Transmit interrupt is disabled. 0b - Early Transmit Interrupt is disabled 1b - Early Transmit Interrupt is enabled
9 RWTE	Receive Watchdog Timeout Enable When this bit is set along with the AIE bit, the Receive Watchdog Timeout interrupt is enabled. When this bit is reset, the Receive Watchdog Timeout interrupt is disabled. 0b - Receive Watchdog Timeout is disabled 1b - Receive Watchdog Timeout is enabled
8 RSE	Receive Stopped Enable When this bit is set along with the AIE bit, the Receive Stopped Interrupt is enabled. When this bit is reset, the Receive Stopped interrupt is disabled. 0b - Receive Stopped is disabled 1b - Receive Stopped is enabled
7 RBUE	Receive Buffer Unavailable Enable When this bit is set along with the AIE bit, the Receive Buffer Unavailable interrupt is enabled. When this bit is reset, the Receive Buffer Unavailable interrupt is disabled. 0b - Receive Buffer Unavailable is disabled 1b - Receive Buffer Unavailable is enabled
6 RIE	Receive Interrupt Enable When this bit is set along with the NIE bit, the Receive Interrupt is enabled. When this bit is reset, the Receive Interrupt is disabled. 0b - Receive Interrupt is disabled 1b - Receive Interrupt is enabled
5-3	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
2 TBUE	Transmit Buffer Unavailable Enable When this bit is set along with the NIE bit, the Transmit Buffer Unavailable interrupt is enabled. When this bit is reset, the Transmit Buffer Unavailable interrupt is disabled. 0b - Transmit Buffer Unavailable is disabled 1b - Transmit Buffer Unavailable is enabled
1 TXSE	Transmit Stopped Enable When this bit is set along with the AIE bit, the Transmission Stopped interrupt is enabled. When this bit is reset, the Transmission Stopped interrupt is disabled. 0b - Transmit Stopped is disabled 1b - Transmit Stopped is enabled
0 TIE	Transmit Interrupt Enable When this bit is set along with the NIE bit, the Transmit Interrupt is enabled. When this bit is reset, the Transmit Interrupt is disabled. 0b - Transmit Interrupt is disabled 1b - Transmit Interrupt is enabled

76.17.368 DMA Channel 1 Rx Interrupt Watchdog Timer (DMA_CH1_Rx_Interrupt_Watchdog_Timer)

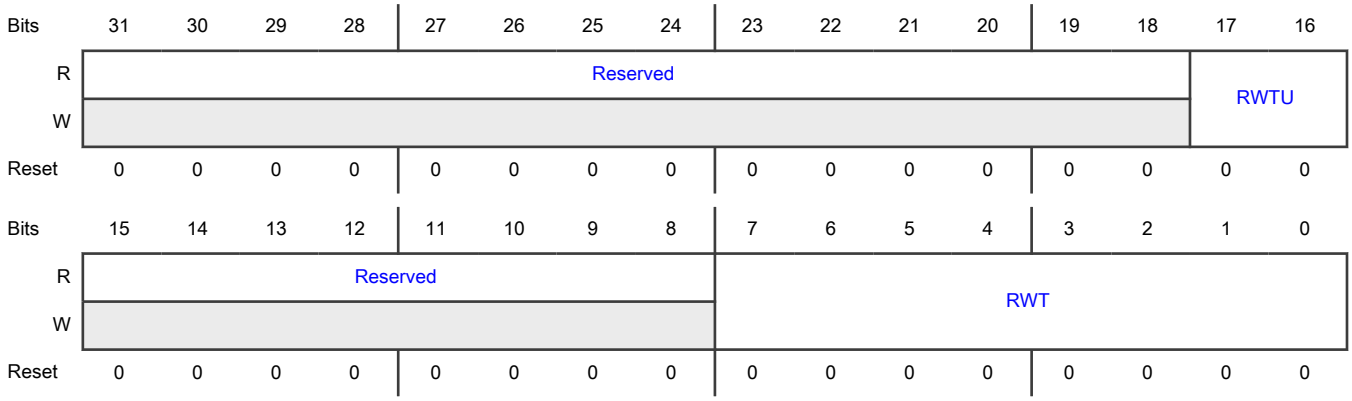
Offset

Register	Offset
DMA_CH1_Rx_Interrupt_Watchdog_Timer	11B8h

Function

The Receive Interrupt Watchdog Timer register indicates the watchdog timeout for Receive Interrupt (RI) from the DMA. When this register is written with a non-zero value, it enables the watchdog timer for the RI bit of the DMA_CHi_Status register.

Diagram



Fields

Field	Function
31-18 —	Reserved
17-16 RWTU	Receive Interrupt Watchdog Timer Count Units This fields indicates the number of system clock cycles corresponding to one unit in RWT field. - 2'b00: 256 - 2'b01: 512 - 2'b10: 1024 - 2'b11: 2048 For example, when RWT=2 and RWTU=1, the watchdog timer is set for 2*512=1024 system clock cycles. **NOTE** This field is not testable by an automated register test.
15-8 —	Reserved
7-0 RWT	Receive Interrupt Watchdog Timer Count This field indicates the number of system clock cycles, multiplied by factor indicated in RWTU field, for which the watchdog timer is set. The watchdog timer is triggered with the programmed value after the Rx DMA completes the transfer of a packet for which the RI bit is not set in the DMA_CH(#i)_Status register, because of the setting of Interrupt Enable bit in the corresponding descriptor RDES3[30]. When the watchdog timer runs out, the RI bit is set and the timer is stopped. The watchdog timer is reset when the RI bit is set high because of automatic setting of RI as per the Interrupt Enable bit RDES3[30] of any received packet. **NOTE** This field is not testable by an automated register test.

76.17.369 DMA Channel 1 Slot Function Control Status (DMA_CH1_Slot_Function_Control_Status)

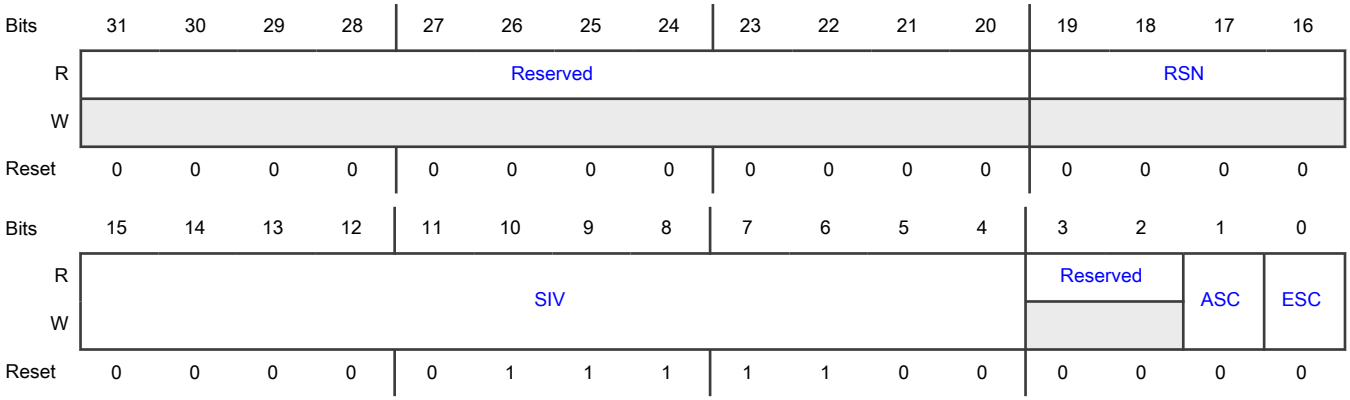
Offset

Register	Offset
DMA_CH1_Slot_Function_Control_Status	11BCh

Function

The Slot Function Control and Status register contains the control bits for slot function and the status for Transmit path.

Diagram



Fields

Field	Function
31-20 —	Reserved
19-16 RSN	Reference Slot Number This field gives the current value of the reference slot number in the DMA. It is used for slot comparison. **NOTE** This field is not testable by an automated register test.
15-4 SIV	Slot Interval Value This field controls the period of the slot interval in which the TxDMA fetches the scheduled packets. A value of 0 specifies the slot interval of 1 us while the maximum value 4095 specifies the slot interval of 4096us. The default/reset value is 0x07C which corresponds to slot interval of 125us
3-2 —	Reserved
1	Advance Slot Check

Table continues on the next page...

Table continued from the previous page...

Field	Function
ASC	When set, this bit enables the DMA to fetch the data from the buffer when the slot number (SLOTNUM) programmed in the Tx descriptor is - equal to the reference slot number given in the RSN field or - ahead of the reference slot number by up to two slots This bit is applicable only when the ESC bit is set. 0b - Advance Slot Check is disabled 1b - Advance Slot Check is enabled
0 ESC	Enable Slot Comparison When set, this bit enables the checking of the slot numbers programmed in the Tx descriptor with the current reference given in the RSN field. The DMA fetches the data from the corresponding buffer only when the slot number is - equal to the reference slot number or - ahead of the reference slot number by one slot When reset, this bit disables the checking of the slot numbers. The DMA fetches the data immediately after the descriptor is processed. Note: The UFO (UDP Fragmentation over IPv4)/TSO/USO should not be enabled along with TBS/AVB Slot number check. The UFO/TSO/USO involves multiple packets/segments/fragments transmission for single packet received from application and the slot number check are applicable for fetching of only first segment/fragment. As a result it might be difficult for software to specify slot number for subsequent packets. 0b - Slot Comparison is disabled 1b - Slot Comparison is enabled

76.17.370 DMA Channel 1 Current Application Transmit Descriptor (DMA_CH1_Current_App_TxDesc)

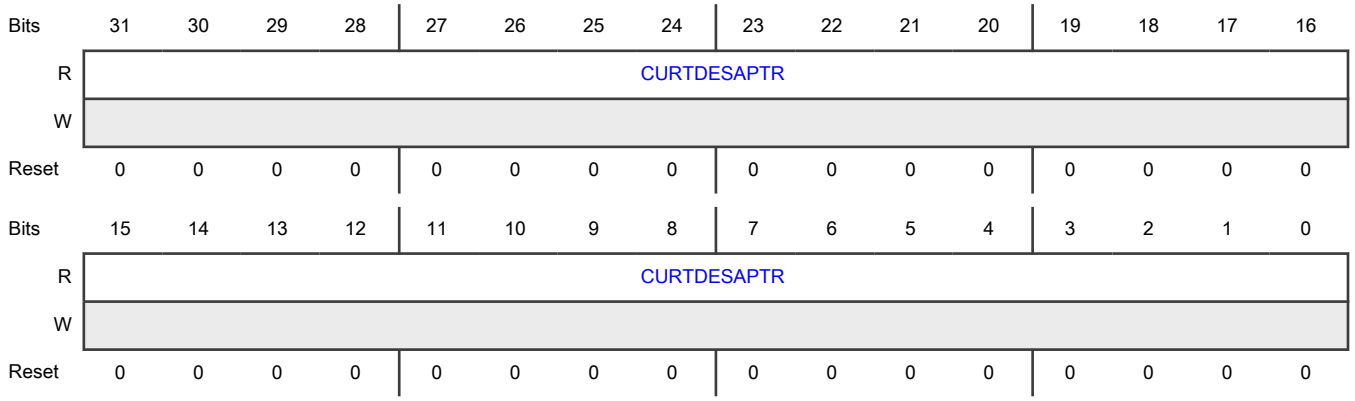
Offset

Register	Offset
DMA_CH1_Current_App_TxDesc	11C4h

Function

The Channeli Current Application Transmit Descriptor register points to the current Transmit descriptor read by the DMA.

Diagram



Fields

Field	Function
31-0 CURTDESAPTR	Application Transmit Descriptor Address Pointer The DMA updates this pointer during Tx operation. This pointer is cleared on reset.

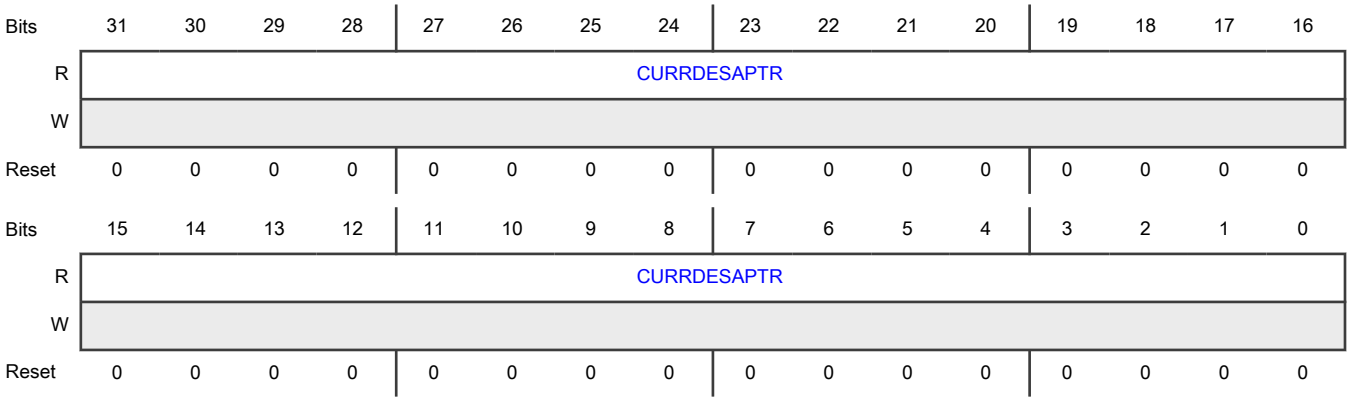
76.17.371 DMA Channel 1 Current Application Receive Descriptor (DMA_CH1_Current_App_RxDesc)

Offset

Register	Offset
DMA_CH1_Current_App_RxDesc	11CCh

Function
The Channeli Current Application Receive Descriptor register points to the current Receive descriptor read by the DMA.

Diagram



Fields

Field	Function
31-0 CURRDESAPTR	Application Receive Descriptor Address Pointer The DMA updates this pointer during Rx operation. This pointer is cleared on reset.

76.17.372 DMA Channel 1 Current Application Transmit Buffer (DMA_CH1_Current_App_TxBuffer)

Offset

Register	Offset
DMA_CH1_Current_App_TxBuffer	11D4h

Function

The Channeli Current Application Transmit Buffer Address register points to the current Tx buffer address read by the DMA.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CURTBUFAPTR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CURTBUFAPTR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 CURTBUFAPTR	Application Transmit Buffer Address Pointer The DMA updates this pointer during Tx operation. This pointer is cleared on reset.

76.17.373 DMA Channel 1 Current Application Receive Buffer (DMA_CH1_Current_App_RxBuffer)

Offset

Register	Offset
DMA_CH1_Current_App_RxBuffer	11DCh

Function

The Channel 0 Current Application Receive Buffer Address register points to the current Rx buffer address read by the DMA.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CURRBUFAPTR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CURRBUFAPTR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 CURRBUFAPTR	Application Receive Buffer Address Pointer The DMA updates this pointer during Rx operation. This pointer is cleared on reset.

76.17.374 DMA Channel 1 Status (DMA_CH1_Status)

Offset

Register	Offset
DMA_CH1_Status	11E0h

Function

The software driver (application) reads the Status register during interrupt service routine or polling to determine the status of the DMA. Note: The number of DMA_CH(#i)_Status register in the configuration is the higher of number of Rx DMA Channels and Tx DMA Channels.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved										REB			TEB		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	NIS	AIS	CDE	FBE	ERI	ETI	RWT	RPS	RBU	RI	Reserved			TBU	TPS	TI
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-22 —	Reserved
21-19 REB	Rx DMA Error Bits This field indicates the type of error that caused a Bus Error. For example, error response on the AHB or AXI interface. - Bit 21 -- 1'b1: Error during data transfer by Rx DMA -- 1'b0: No Error during data transfer by Rx DMA - Bit 20 -- 1'b1: Error during descriptor access -- 1'b0: Error during data buffer access - Bit 19 -- 1'b1: Error during read transfer -- 1'b0: Error during write transfer This field is valid only when the FBE bit is set. This field does not generate an interrupt. **NOTE** This field is not testable by an automated register test.
18-16 TEB	Tx DMA Error Bits This field indicates the type of error that caused a Bus Error. For example, error response on the AHB or AXI interface. - Bit 18 -- 1'b1: Error during data transfer by Tx DMA -- 1'b0: No Error during data transfer by Tx DMA - Bit 17 -- 1'b1: Error during descriptor access -- 1'b0: Error during data buffer access - Bit 16 -- 1'b1: Error during read transfer -- 1'b0: Error during write transfer This field is valid only when the FBE bit is set. This field does not generate an interrupt. **NOTE** This field is not testable by an automated register test.
15 NIS	Normal Interrupt Summary Normal Interrupt Summary bit value is the logical OR of the following bits when the corresponding interrupt bits are enabled in the DMA_CH0_Interrupt_Enable register: - Bit 0: Transmit Interrupt - Bit 2: Transmit Buffer Unavailable - Bit 6: Receive Interrupt - Bit 11: Early Receive Interrupt Only unmasked bits (interrupts for which interrupt enable is set in DMA_CH0_Interrupt_Enable register) affect the Normal Interrupt Summary bit. This is a sticky bit. You must clear this bit (by writing 1 to this bit) each time a corresponding bit which causes NIS to be set is cleared. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **NOTE** This field is not testable by an automated register test. 0b - Normal Interrupt Summary status not detected 1b - Normal Interrupt Summary status detected
14 AIS	Abnormal Interrupt Summary Abnormal Interrupt Summary bit value is the logical OR of the following when the corresponding interrupt bits are enabled in the DMA_CH0_Interrupt_Enable register: - Bit 1: Transmit Process Stopped - Bit 7: Receive Buffer Unavailable - Bit 8: Receive Process Stopped - Bit 10: Early Transmit Interrupt - Bit 12: Fatal Bus Error - Bit 13: Context Descriptor Error Only unmasked bits affect the Abnormal Interrupt Summary bit. This is a sticky bit. You must clear this bit (by writing 1 to this bit) each time a corresponding bit, which causes AIS to be set, is cleared. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field is not testable by an automated register test. 0b - Abnormal Interrupt Summary status not detected 1b - Abnormal Interrupt Summary status detected
13 CDE	Context Descriptor Error This bit indicates that the DMA Tx/Rx engine received a descriptor error, which indicates invalid context in the middle of packet flow (intermediate descriptor) or all one's descriptor in Tx case and on Rx side it indicates DMA has read a descriptor with either of the buffer address as ones which is considered to be invalid. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **NOTE** This field is not testable by an automated register test. 0b - Context Descriptor Error status not detected 1b - Context Descriptor Error status detected
12 FBE	Fatal Bus Error This bit indicates that a bus error occurred (as described in the EB field). When this bit is set, the corresponding DMA channel engine disables all bus accesses. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **NOTE** This field is not testable by an automated register test. 0b - Fatal Bus Error status not detected 1b - Fatal Bus Error status detected
11 ERI	Early Receive Interrupt This bit when set indicates that the RxDMA has completed the transfer of packet data to the memory. In configurations supporting ERIC, - ERIC = 0: This bit is set only after the Rx DMA has completely filled up a receive buffer with packet data. - ERIC = 1: This bit is set after every burst transfer of data from the Rx DMA to the buffer. The setting of RI bit automatically clears this bit. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **NOTE** This field is not testable by an automated register test. 0b - Early Receive Interrupt status not detected 1b - Early Receive Interrupt status detected
10 ETI	Early Transmit Interrupt This bit when set indicates that the TxDMA has completed the transfer of packet data to the MTL TXFIFO memory. In configurations supporting ERIC: - ETIC = 0: This bit is set only after the Tx DMA has

Table continues on the next page...

Table continued from the previous page...

Field	Function
	transferred a complete packet to MTL. - ETIC = 1: This bit is set after completion of (partial) packet data transfer from buffers in the Transmit descriptor in which IOC = 1. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **NOTE** This field is not testable by an automated register test. 0b - Early Transmit Interrupt status not detected 1b - Early Transmit Interrupt status detected
9 RWT	Receive Watchdog Timeout This bit is asserted when a packet with length greater than 2,048 bytes (10,240 bytes when Jumbo Packet mode is enabled) is received. **NOTE** This field is not testable by an automated register test. 0b - Receive Watchdog Timeout status not detected 1b - Receive Watchdog Timeout status detected
8 RPS	Receive Process Stopped This bit is asserted when the Rx process enters the Stopped state. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **NOTE** This field is not testable by an automated register test. 0b - Receive Process Stopped status not detected 1b - Receive Process Stopped status detected
7 RBU	Receive Buffer Unavailable This bit indicates that the application owns the next descriptor in the Receive list, and the DMA cannot acquire it. The Rx process is suspended. To resume processing Rx descriptors, the application should change the ownership of the descriptor and issue a Receive Poll Demand command. If this command is not issued, the Rx process resumes when the next recognized incoming packet is received. In ring mode, the application should advance the Receive Descriptor Tail Pointer register of a channel. This bit is set only when the DMA owns the previous Rx descriptor. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **NOTE** This field is not testable by an automated register test. 0b - Receive Buffer Unavailable status not detected 1b - Receive Buffer Unavailable status detected
6 RI	Receive Interrupt

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This bit indicates that the packet reception is complete. When packet reception is complete, Bit 31 of RDES3 is reset in the last descriptor, and the specific packet status information is updated in the descriptor. The reception remains in the Running state. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **NOTE** This field is not testable by an automated register test. 0b - Receive Interrupt status not detected 1b - Receive Interrupt status detected
5-3 —	Reserved
2 TBU	Transmit Buffer Unavailable This bit indicates that the application owns the next descriptor in the Transmit list, and the DMA cannot acquire it. Transmission is suspended. The TPS0 field of the DMA_Debug_Status0 register explains the Transmit Process state transitions. To resume processing the Transmit descriptors, the application should do the following: - Change the ownership of the descriptor by setting Bit 31 of TDES3. - Issue a Transmit Poll Demand command. For ring mode, the application should advance the Transmit Descriptor Tail Pointer register of a channel. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **NOTE** This field is not testable by an automated register test. 0b - Transmit Buffer Unavailable status not detected 1b - Transmit Buffer Unavailable status detected
1 TPS	Transmit Process Stopped This bit is set when the transmission is stopped. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **NOTE** This field is not testable by an automated register test. 0b - Transmit Process Stopped status not detected 1b - Transmit Process Stopped status detected
0 TI	Transmit Interrupt This bit indicates that the packet transmission is complete. When transmission is complete, Bit 31 of TDES3 is reset in the last descriptor, and the specific packet status information is updated in the descriptor. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **NOTE** This field is not testable by an automated register test.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Transmit Interrupt status not detected 1b - Transmit Interrupt status detected

76.17.375 DMA Channel 1 Miss Frame Counter (DMA_CH1_Miss_Frame_Cnt)

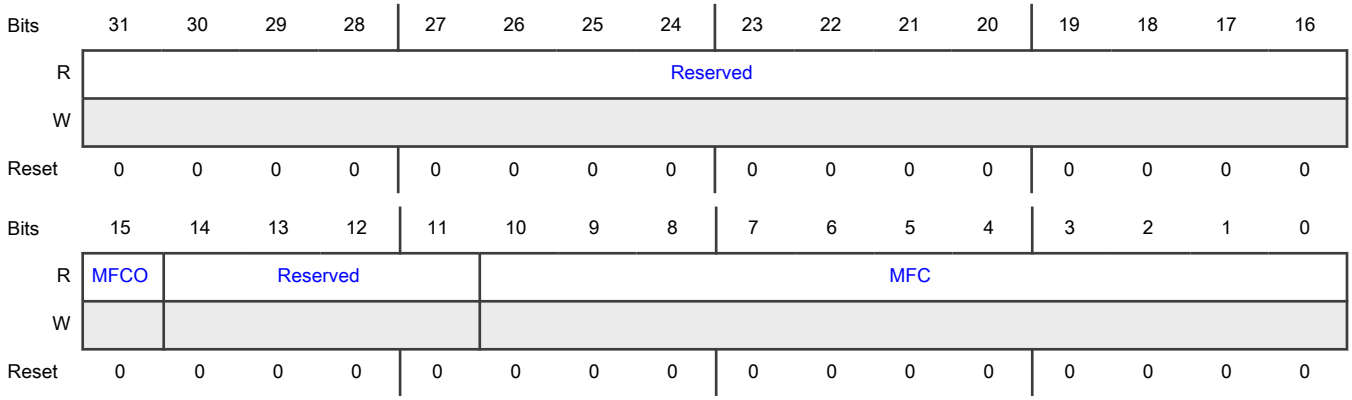
Offset

Register	Offset
DMA_CH1_Miss_Frame_Cnt	11E4h

Function

This register has the number of packet counter that got dropped by the DMA either due to Bus Error or due to programming RPF field in DMA_CH${i}_Rx_Control register.

Diagram



Fields

Field	Function
31-16 —	Reserved
15 MFCO	Overflow status of the MFC Counter When this bit is set then the MFC counter does not get incremented further. The bit gets cleared when this register is read. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - Miss Frame Counter overflow not occurred 1b - Miss Frame Counter overflow occurred

Table continues on the next page...

Table continued from the previous page...

Field	Function
14-11 —	Reserved
10-0 MFC	Dropped Packet Counters This counter indicates the number of packet counters that are dropped by the DMA either because of bus error or because of programing RPF field in DMA_CH${i}_Rx_Control register. The counter gets cleared when this register is read. Access restriction applies. Clears on read. Self-set to 1 on internal event.

76.17.376 DMA Channel 1 Rx Parser Accept Count (DMA_CH1_RXP_Accept_Cnt)

Offset

Register	Offset
DMA_CH1_RXP_Accept_Cnt	11E8h

Function
The DMA_CH(#i)_RXP_Accept_Cnt registers provides the count of the number of frames accepted by Rx Parser.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RXPACOF	RXPAC														
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXPAC															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 RXPACOF	Rx Parser Accept Counter Overflow Bit When set, this bit indicates that the RXPAC Counter field crossed the maximum limit. Access restriction applies. Clears on read. Self-set to 1 on internal event.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Rx Parser Accept Counter overflow not occurred 1b - Rx Parser Accept Counter overflow occurred
30-0 RXPAC	Rx Parser Accept Counter This 31-bit counter is implemented when a Rx Parser Accept a packet due to AF =1. The counter is cleared when the register is read.

76.17.377 DMA Channel 1 Rx ERI Count (DMA_CH1_RX_ERI_Cnt)

Offset

Register	Offset
DMA_CH1_RX_ERI_Cnt	11ECh

Function
The DMA_CH(#i)_RX_ERI_Cnt registers provides the count of the number of times ERI was asserted.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved				ECNT											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-12 —	Reserved
11-0 ECNT	ERI Counter When ERIC bit of DMA_CH(#i)_RX_Control register is set, this counter increments for burst transfer completed by the Rx DMA from the start of packet transfer. This counter is reset at the start of new packet.

76.17.378 DMA Channel 2 Control (DMA_CH2_Control)

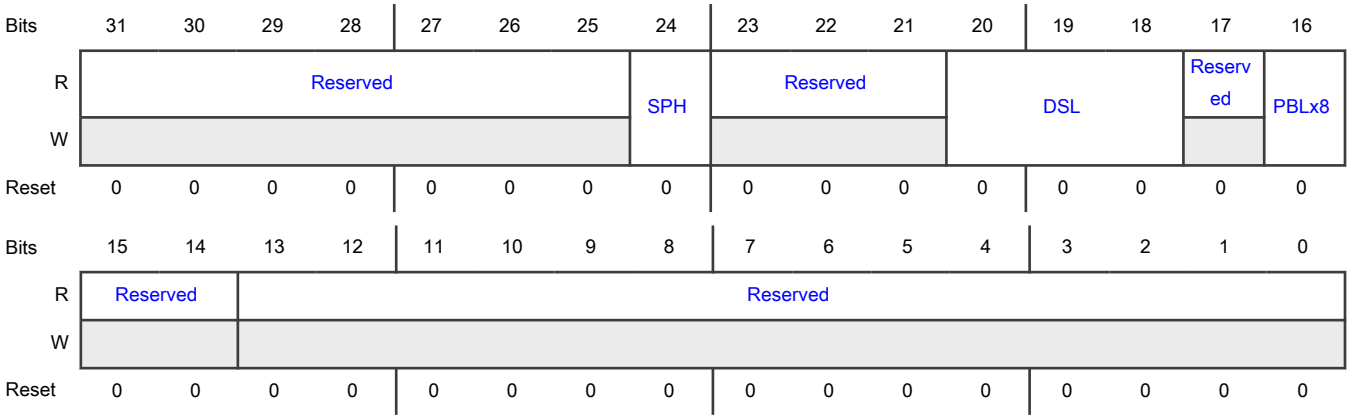
Offset

Register	Offset
DMA_CH2_Control	1200h

Function

The DMA Channeli Control register specifies the MSS value for segmentation, length to skip between two descriptors, and also the features such as header splitting and 8xPBL mode.

Diagram



Fields

Field	Function
31-25 —	Reserved
24 SPH	Split Headers When this bit is set, the DMA splits the header and payload in the Receive path. The DMA writes the header to the Buffer Address1 of RDES0. The DMA writes the payload to the buffer to which the Buffer Address2 is pointing. The software must ensure that the header fits into the Receive buffers. If the header length exceeds the receive buffer size, the DMA does not split the header and payload. This bit is available only if Enable Split Header Structure option is selected. 0b - Split Headers feature is disabled 1b - Split Headers feature is enabled
23-21 —	Reserved
20-18 DSL	Descriptor Skip Length

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This bit specifies the Word, Dword, or Lword number (depending on the 32-bit, 64-bit, or 128-bit bus) to skip between two unchained descriptors. The address skipping starts from the end of the current descriptor to the start of the next descriptor. When the DSL value is equal to zero, the DMA takes the descriptor table as contiguous.
17 —	Reserved
16 PBLx8	8xPBL mode When this bit is set, the PBL value programmed in Bits[21:16] in DMA_CH(#i)_Tx_Control and Bits[21:16] in DMA_CH(#i)_Rx_Control is multiplied by eight times. Therefore, the DMA transfers the data in 8, 16, 32, 64, 128, and 256 beats depending on the PBL value. 0b - 8xPBL mode is disabled 1b - 8xPBL mode is enabled
15-14 —	Reserved
13-0 —	Reserved

76.17.379 DMA Channel 2 Tx Control (DMA_CH2_Tx_Control)

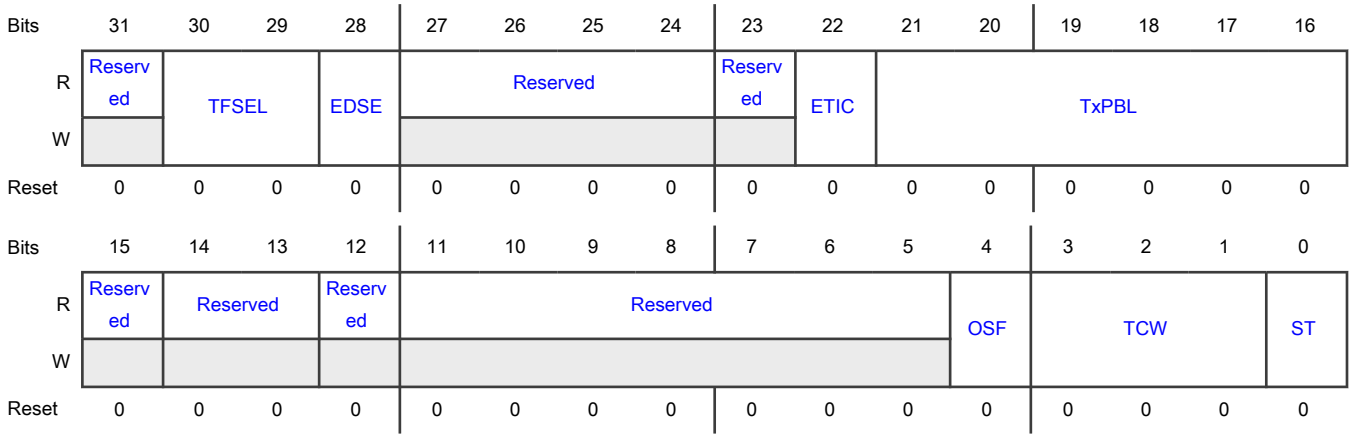
Offset

Register	Offset
DMA_CH2_Tx_Control	1204h

Function

The DMA Channeli Transmit Control register controls the Tx features such as PBL, TCP segmentation, and Tx Channel weights.

Diagram



Fields

Field	Function
31 —	Reserved
30-29 TFSEL	TBS Fetch Select Select bits for one of the four DMA_TBS_CTRL register fields (FTOS,FGSN,FTOV) for the channel Values: - 2'b00: DMA_TBS_CTRL0 - 2'b01: DMA_TBS_CTRL1 - 2'b10: DMA_TBS_CTRL2 - 2'b11: DMA_TBS_CTRL3 (Reserved if DWC_EQOS_NUM_DMA_TX_CH=3)
28 EDSE	Enhanced Descriptor Enable When this bit is set, the corresponding channel uses Enhanced Descriptors that are 32 Bytes for both Normal and Context Descriptors. When reset, the corresponding channel uses the descriptors that are 16 Bytes. 0b - Enhanced Descriptor is disabled 1b - Enhanced Descriptor is enabled
27-24 —	Reserved
23 —	Reserved
22 ETIC	Early Transmit Interrupt Control When this bit is set, Early Transmit Interrupt (ETI) status is set after completion of transfer of data from buffers of a transmit descriptor in which IOC bit (TDES2[31]) is set. When this bit is reset, ETI is set only after a complete packet is transferred to the MTL TX FIFO memory. 0b - Early Transmit Interrupt is disabled 1b - Early Transmit Interrupt is enabled
21-16	Transmit Programmable Burst Length

Table continues on the next page...

Table continued from the previous page...

Field	Function
TxPBL	These bits indicate the maximum number of beats to be transferred in one DMA block data transfer. The DMA always attempts max burst as specified in PBL each time it starts a burst transfer on the application bus. You can program PBL with any of the following values: 1, 2, 4, 8, 16, or 32. Any other value results in undefined behavior. To transfer more than 32 beats, perform the following steps: 1. Set the 8xPBL mode in DMA_CH0_Control register. 2. Set the TxPBL. Note: The maximum value of TxPBL must be less than or equal to half the Tx Queue size (TQS field of MTL_TxQ(#i)_Operation_Mode register) in terms of beats. This is required so that the Tx Queue has space to store at least another Tx PBL worth of data while the MTL Tx Queue Controller is transferring data to MAC. For example, in 64-bit data width configurations the total locations in Tx Queue of size 512 bytes is 64, TxPBL and 8xPBL needs to be programmed to less than or equal to 32.
15 —	Reserved
14-13 —	Reserved
12 —	Reserved
11-5 —	Reserved
4 OSF	Operate on Second Packet When this bit is set, it instructs the DMA to process the second packet of the Transmit data even before the status for the first packet is obtained. 0b - Operate on Second Packet disabled 1b - Operate on Second Packet enabled
3-1 TCW	Transmit Channel Weight This field indicates the weight assigned to the corresponding Transmit channel. When reset is complete, this field is set to 0 for all channels by default, resulting in equal weights to all channels.
0 ST	Start or Stop Transmission Command When this bit is set, transmission is placed in the Running state. The DMA checks the Transmit list at the current position for a packet to be transmitted. The DMA tries to acquire descriptor from either of the following positions: - The current position in the list This is the base address of the Transmit list set by the DMA_CH0_TxDesc_List_Address register. - The position at which the transmission was previously stopped If the DMA does not own the current descriptor, the transmission enters the Suspended state and the TBU bit of the DMA_CH0_Status register is set. The Start Transmission command is effective only when the transmission is stopped. If the command is issued before setting the DMA_CH0_TxDesc_List_Address register, the DMA behavior is unpredictable. When this bit is reset, the transmission process is placed in the Stopped state after completing the transmission of the current packet. The Next Descriptor position in the Transmit list is saved, and it becomes the current position when the transmission is restarted. To change the list address, you need to program DMA_CH0_TxDesc_List_Address register with a new value when this bit is reset. The new value is

Table continues on the next page...

Table continued from the previous page...

Field	Function
	considered when this bit is set again. The stop transmission command is effective only when the transmission of the current packet is complete or the transmission is in the Suspended state. **NOTE** This field is not testable by an automated register test. 0b - Stop Transmission Command 1b - Start Transmission Command

76.17.380 DMA Channel 2 Rx Control (DMA_CH2_Rx_Control)

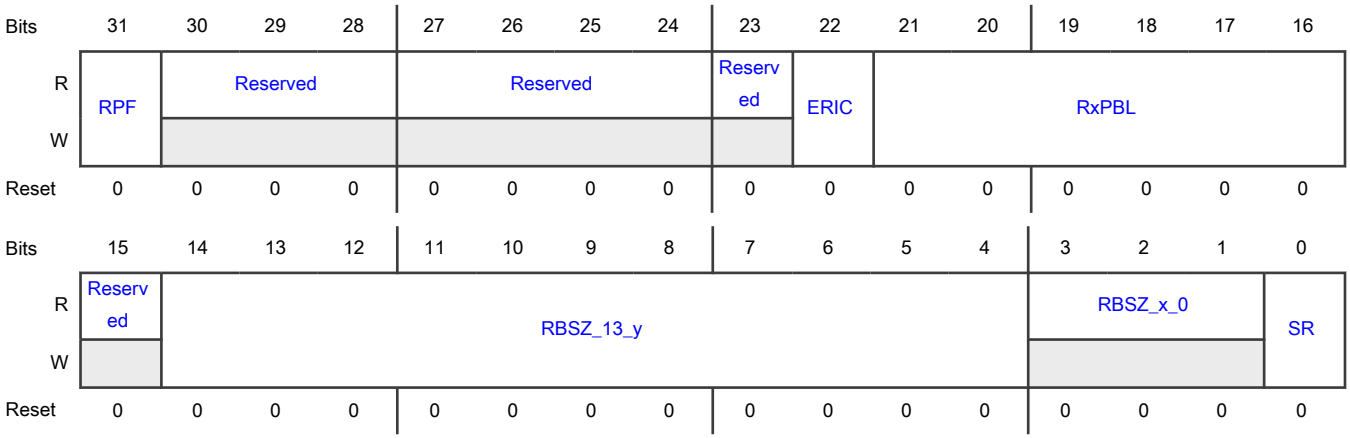
Offset

Register	Offset
DMA_CH2_Rx_Control	1208h

Function

The DMA Channeli Receive Control register controls the Rx features such as PBL, buffer size, and extended status.

Diagram



Fields

Field	Function
31 RPF	Rx Packet Flush. - 1: DWC_ether_qos automatically flushes the packet from the Rx Queues destined to this DMA Rx Channel, when it is stopped. When this bit remains set and the DMA is re-started by the software driver, the packets residing in the Rx Queues that were received when this RxDMA was stopped, are flushed out. The packets that are received by the MAC after the RxDMA is re-started are routed to the RxDMA. The flushing is done on the Read side of the Rx Queue. - 0: DWC_ether_qos does not flush the packet in

Table continues on the next page...

Table continued from the previous page...

Field	Function
	the Rx Queue destined to this RxDMA Channel when it is in STOP state. This might cause head-of-line blocking in the corresponding RxQueue. Note: The stopping of packet flow from a Rx DMA Channel to the application by setting RPF works only when there is one-to-one mapping of Rx Queue to Rx DMA channels. In Dynamic mapping mode, setting RPF bit in any DMA_CH(#i)_Rx_Control register might flush packets from unintended Rx Queues which are destined to the stopped Rx DMA Channel. 0b - Rx Packet Flush is disabled 1b - Rx Packet Flush is enabled
30-28 —	Reserved
27-24 —	Reserved
23 —	Reserved
22 ERIC	Early Receive Interrupt Control When this bit is set, Early Receive Interrupt (ERI) status is set after the completion of every burst transfer of data from the Rx DMA to the buffer. When this bit is reset, ERI is set only after a complete buffer is filled up by the RxDMA. 0b - Early Receive Interrupt is disabled 1b - Early Receive Interrupt is enabled
21-16 RxPBL	Receive Programmable Burst Length These bits indicate the maximum number of beats to be transferred in one DMA block data transfer. The DMA always attempts max burst as specified in PBL each time it starts a burst transfer on the application bus. You can program PBL with any of the following values: 1, 2, 4, 8, 16, or 32. Any other value results in undefined behavior. To transfer more than 32 beats, perform the following steps: 1. Set the 8xPBL mode in the DMA_CH0_Control register. 2. Set the RxPBL. Note: The maximum value of RxPBL must be less than or equal to half the Rx Queue size (RQS field of MTL_RxQ(#i)_Operation_Mode register) in terms of beats. This is required so that the Rx Queue has space to store at least another Rx PBL worth of data while the Rx DMA is transferring a block of data. For example, in 64-bit data width configurations the total locations in Rx Queue of size 512 bytes is 64, so RxPBL and 8xPBL needs to be programmed to less than or equal to 32.
15 —	Reserved
14-4 RBSZ_13_y	Receive Buffer size High RBSZ[13:0] is split into two fields higher RBSZ_13_y and lower RBSZ_x_0. The RBSZ[13:0] field indicates the size of the Rx buffers specified in bytes. The maximum buffer size is limited to 16K bytes. The buffer size is applicable to payload buffers when split headers are enabled. Note: The buffer size must be a multiple of 4, 8, or 16 depending on the data bus widths (32-bit, 64-bit, or 128-bit respectively).

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This is required even if the value of buffer address pointer is not aligned to data bus width. Hence the lower RBSZ_x_0 bits are read-only and the value is considered as all-zero. Thus the RBSZ_13_y indicates the buffer size in terms of locations (with the width same as bus-width).
3-1 RBSZ_x_0	Receive Buffer size Low RBSZ[13:0] is split into two fields RBSZ_13_y and RBSZ_x_0. The RBSZ_x_0 is the lower field whose width is based on data bus width of the configuration. This field is of width 2, 3, or 4 bits for 32-bit, 64-bit, or 128-bit data bus width respectively. This field is read-only (RO).
0 SR	Start or Stop Receive When this bit is set, the DMA tries to acquire the descriptor from the Receive list and processes the incoming packets. The DMA tries to acquire descriptor from either of the following positions: - The current position in the list This is the address set by the DMA_CH0_RxDesc_List_Address register. - The position at which the Rx process was previously stopped If the DMA does not own the current descriptor, the reception is suspended and the RBU bit of the DMA_CH0_Status register is set. The Start Receive command is effective only when the reception is stopped. If the command is issued before setting the DMA_CH0_RxDesc_List_Address register, the DMA behavior is unpredictable. When this bit is reset, the Rx DMA operation is stopped after the transfer of the current packet. The next descriptor position in the Receive list is saved, and it becomes the current position after the Rx process is restarted. The Stop Receive command is effective only when the Rx process is in the Running (waiting for Rx packet) or Suspended state. **NOTE** This field is not testable by an automated register test. 0b - Stop Receive 1b - Start Receive

76.17.381 DMA Channel 2 Tx Descriptor List Address (DMA_CH2_TxDesc_List_Address)

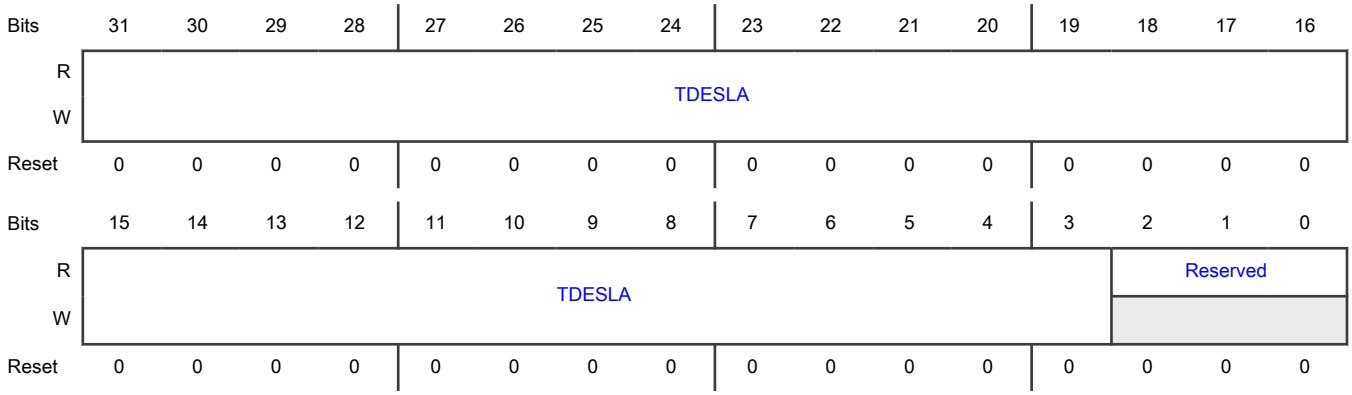
Offset

Register	Offset
DMA_CH2_TxDesc_List_Address	1214h

Function

The Channeli Tx Descriptor List Address register points the DMA to the start of Transmit descriptor list. The descriptor lists reside in the physical memory space of the application and must be Word, Dword, or Lword-aligned (for 32-bit, 64-bit, or 128-bit data bus). The DMA internally converts it to bus width aligned address by making the corresponding LSB to low. You can write to this register only when the Tx DMA has stopped, that is, the ST bit is set to zero in DMA_CH0_Tx_Control register. When stopped, this register can be written with a new descriptor list address. When you set the ST bit to 1, the DMA takes the newly-programmed descriptor base address. If this register is not changed when the ST bit is set to 0, the DMA takes the descriptor address where it was stopped earlier.

Diagram



Fields

Field	Function
31-3 TDESLA	Start of Transmit List This field contains the base address of the first descriptor in the Transmit descriptor list. The DMA ignores the LSB bits (1:0, 2:0, or 3:0) for 32-bit, 64-bit, or 128-bit bus width and internally takes these bits as all-zero. Therefore, these LSB bits are read-only (RO). The width of this field depends on the configuration: - 31:2 for 32-bit configuration - 31:3 for 64-bit configuration - 31:4 for 128-bit configuration **NOTE** This field is not testable by an automated register test.
2-0 —	Reserved

76.17.382 DMA Channel 2 Rx Descriptor List Address (DMA_CH2_RxDesc_List_Address)

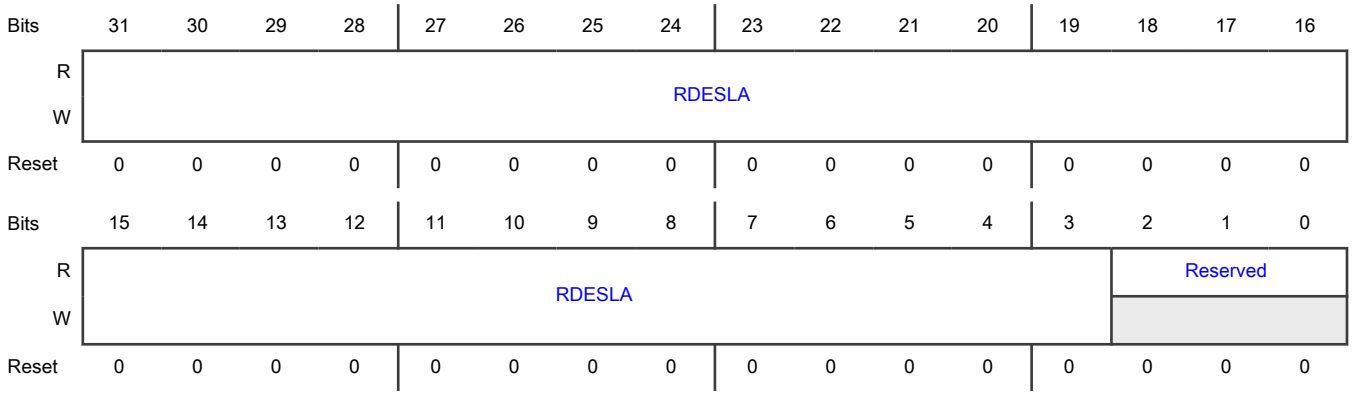
Offset

Register	Offset
DMA_CH2_RxDesc_List_Address	121Ch

Function

The Channeli Rx Descriptor List Address register points the DMA to the start of Receive descriptor list. This register points to the start of the Receive Descriptor List. The descriptor lists reside in the physical memory space of the application and must be Word, Dword, or Lword-aligned (for 32-bit, 64-bit, or 128-bit data bus). The DMA internally converts it to bus width aligned address by making the corresponding LS bits low. Writing to this register is permitted only when reception is stopped. When stopped, this register must be written to before the receive Start command is given. You can write to this register only when Rx DMA has stopped, that is, SR bit is set to zero in DMA_CH0_Rx_Control register. When stopped, this register can be written with a new descriptor list address. When you set the SR bit to 1, the DMA takes the newly programmed descriptor base address.

Diagram



Fields

Field	Function
31-3 RDESLA	Start of Receive List This field contains the base address of the first descriptor in the Rx Descriptor list. The DMA ignores the LSB bits (1:0, 2:0, or 3:0) for 32-bit, 64-bit, or 128-bit bus width and internally takes these bits as all-zero. Therefore, these LSB bits are read-only (RO). The width of this field depends on the configuration: - 31:2 for 32-bit configuration - 31:3 for 64-bit configuration - 31:4 for 128-bit configuration **NOTE** This field is not testable by an automated register test.
2-0 —	Reserved

76.17.383 DMA Channel 2 Tx Descriptor Tail Pointer (DMA_CH2_TxDesc_Tail_Pointer)

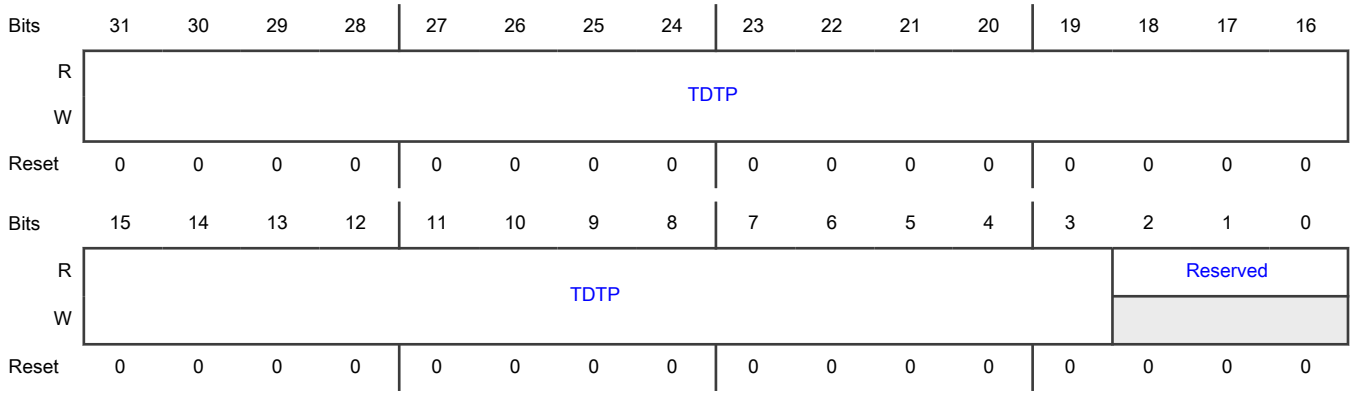
Offset

Register	Offset
DMA_CH2_TxDesc_Tail_Pointer	1220h

Function

The Channeli Tx Descriptor Tail Pointer register points to an offset from the base and indicates the location of the last valid descriptor.

Diagram



Fields

Field	Function
31-3 TDTP	Transmit Descriptor Tail Pointer This field contains the tail pointer for the Tx descriptor ring. The software writes the tail pointer to add more descriptors to the Tx channel. The hardware tries to transmit all packets referenced by the descriptors between the head and the tail pointer registers. The width of this field depends on the configuration: - 31:2 for 32-bit configuration - 31:3 for 64-bit configuration - 31:4 for 128-bit configuration
2-0 —	Reserved

76.17.384 DMA Channel 2 Rx Descriptor Tail Pointer (DMA_CH2_RxDesc_Tail_Pointer)

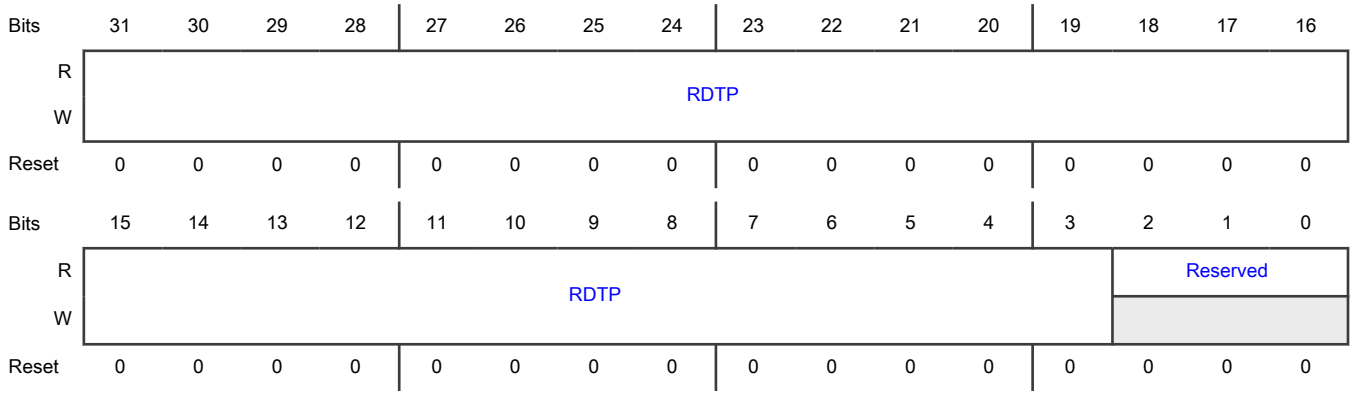
Offset

Register	Offset
DMA_CH2_RxDesc_Tail_Pointer	1228h

Function

The Channeli Rx Descriptor Tail Pointer Points to an offset from the base and indicates the location of the last valid descriptor.

Diagram



Fields

Field	Function
31-3 RDTP	Receive Descriptor Tail Pointer This field contains the tail pointer for the Rx descriptor ring. The software writes the tail pointer to add more descriptors to the Rx channel. The hardware tries to write all received packets to the descriptors referenced between the head and the tail pointer registers. The width of this field depends on the configuration: - 31:2 for 32-bit configuration - 31:3 for 64-bit configuration - 31:4 for 128-bit configuration
2-0 —	Reserved

76.17.385 DMA Channel 2 Tx Descriptor Ring Length (DMA_CH2_TxDesc_Ring_Length)

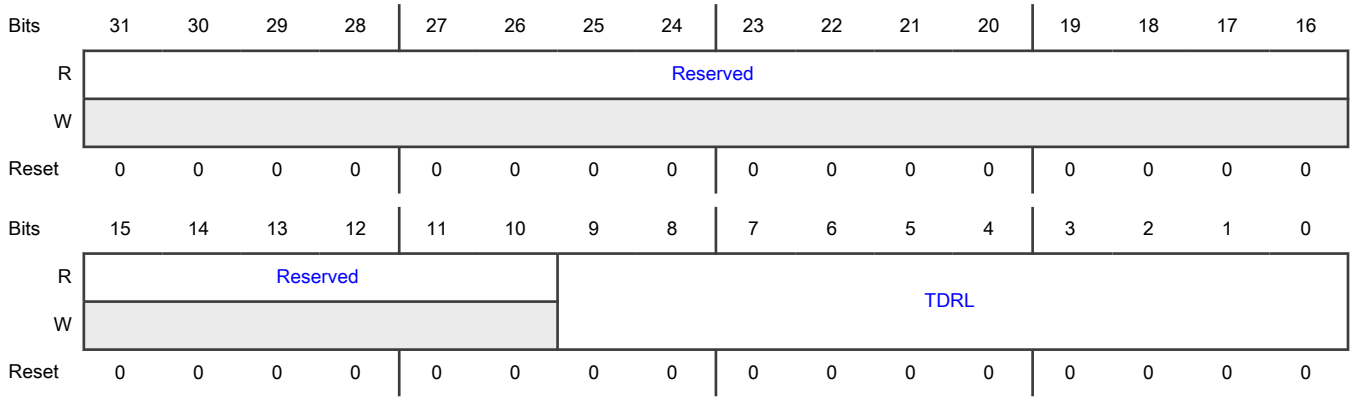
Offset

Register	Offset
DMA_CH2_TxDesc_Ring_Length	122Ch

Function

The Tx Descriptor Ring Length register contains the length of the Transmit descriptor ring.

Diagram



Fields

Field	Function
31-10 —	Reserved
9-0 TDRL	Transmit Descriptor Ring Length Transmit Descriptor Ring Length. This field sets the maximum number of Tx descriptors in the circular descriptor ring. The maximum number of descriptors is limited to 1K descriptors. NXP recommends a minimum ring descriptor length of 4. For example, You can program any value up to 0x3FF in this field. This field is 10 bits wide, if you program 0x3FF, you can have 1024 descriptors. If you want to have 10 descriptors, program it to a value of 0x9.

76.17.386 DMA Channel 2 Rx Control 2 (DMA_CH2_Rx_Control2)

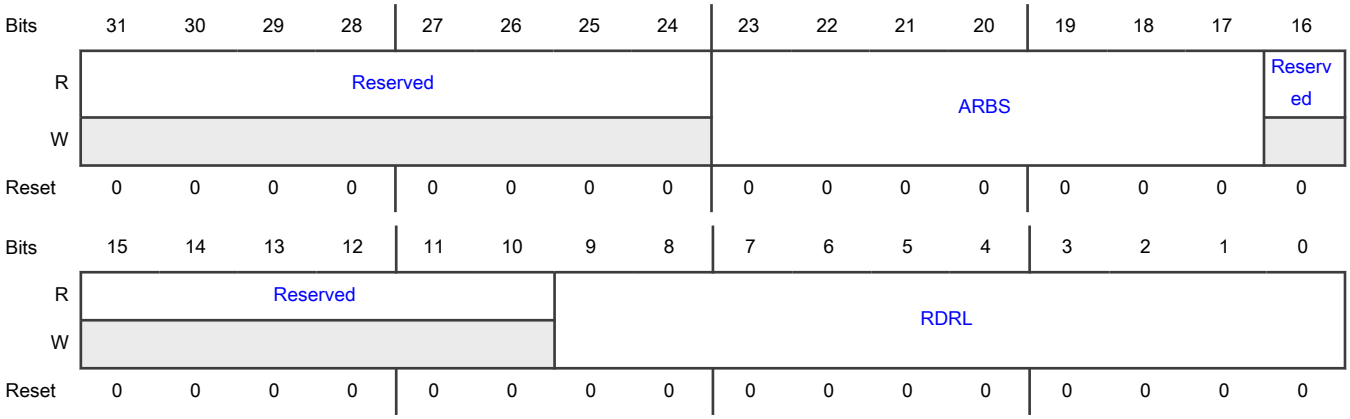
Offset

Register	Offset
DMA_CH2_Rx_Control2	1230h

Function

The Channeli Receive Control register controls the Rx features such as Rx Descriptor Ring Length and Alternate Rx Buffer Size.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-17 ARBS	Alternate Receive Buffer Size Indicates size in bytes for Buffer 1 when ARBS is programmed to a non-zero value (when split header feature is not enabled). When split header feature is enabled, ARBS indicates the buffer size for header data. The maximum alternate buffer is limited to 1020, 1016 or 1008-bytes depending on the data bus widths (32-bit, 64-bit, or 128-bit respectively). When ARBS=0, Rx Buffer1 and Rx Buffer2 sizes are based on RBSZ field of DMA_CH(#i)_Rx_Control. Width of ARBS field is 8, 7 or 6-bits depending on the data bus widths (32-bit, 64-bit, or 128-bit respectively)
16-10 —	Reserved
9-0 RDRL	Receive Descriptor Ring Length This register sets the maximum number of Rx descriptors in the circular descriptor ring. The maximum number of descriptors is limited to 1K descriptors. For example, You can program any value up to 0x3FF in this field. This field is 10 bits wide, if you program 0x3FF, you can have 1024 descriptors. If you want to have 10 descriptors, program it to a value of 0x9.

76.17.387 DMA Channel 2 Interrupt Enable (DMA_CH2_Interrupt_Enable)

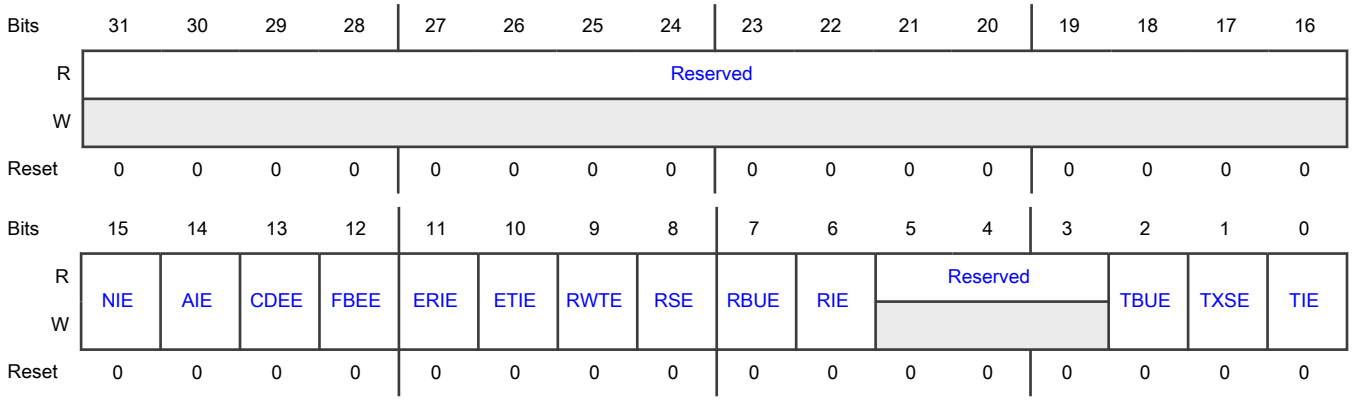
Offset

Register	Offset
DMA_CH2_Interrupt_Enable	1234h

Function

The Channeli Interrupt Enable register enables the interrupts reported by the Status register.

Diagram



Fields

Field	Function
31-16 —	Reserved
15 NIE	Normal Interrupt Summary Enable When this bit is set, the normal interrupt summary is enabled. This bit enables the following interrupts in the DMA_CH0_Status register: - Bit 0: Transmit Interrupt - Bit 2: Transmit Buffer Unavailable - Bit 6: Receive Interrupt - Bit 11: Early Receive Interrupt When this bit is reset, the normal interrupt summary is disabled. 0b - Normal Interrupt Summary is disabled 1b - Normal Interrupt Summary is enabled
14 AIE	Abnormal Interrupt Summary Enable When this bit is set, the abnormal interrupt summary is enabled. This bit enables the following interrupts in the DMA_CH0_Status register: - Bit 1: Transmit Process Stopped - Bit 7: Rx Buffer Unavailable - Bit 8: Receive Process Stopped - Bit 9: Receive Watchdog Timeout - Bit 10: Early Transmit Interrupt - Bit 12: Fatal Bus Error - Bit 13: Context Descriptor Error When this bit is reset, the abnormal interrupt summary is disabled. 0b - Abnormal Interrupt Summary is disabled 1b - Abnormal Interrupt Summary is enabled
13 CDEE	Context Descriptor Error Enable When this bit is set along with the AIE bit, the Descriptor error interrupt is enabled. When this bit is reset, the Descriptor error interrupt is disabled. 0b - Context Descriptor Error is disabled 1b - Context Descriptor Error is enabled
12 FBEE	Fatal Bus Error Enable When this bit is set along with the AIE bit, the Fatal Bus error interrupt is enabled. When this bit is reset, the Fatal Bus Error error interrupt is disabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Fatal Bus Error is disabled 1b - Fatal Bus Error is enabled
11 ERIE	Early Receive Interrupt Enable When this bit is set along with the NIE bit, the Early Receive interrupt is enabled. When this bit is reset, the Early Receive interrupt is disabled. 0b - Early Receive Interrupt is disabled 1b - Early Receive Interrupt is enabled
10 ETIE	Early Transmit Interrupt Enable When this bit is set along with the AIE bit, the Early Transmit interrupt is enabled. When this bit is reset, the Early Transmit interrupt is disabled. 0b - Early Transmit Interrupt is disabled 1b - Early Transmit Interrupt is enabled
9 RWTE	Receive Watchdog Timeout Enable When this bit is set along with the AIE bit, the Receive Watchdog Timeout interrupt is enabled. When this bit is reset, the Receive Watchdog Timeout interrupt is disabled. 0b - Receive Watchdog Timeout is disabled 1b - Receive Watchdog Timeout is enabled
8 RSE	Receive Stopped Enable When this bit is set along with the AIE bit, the Receive Stopped Interrupt is enabled. When this bit is reset, the Receive Stopped interrupt is disabled. 0b - Receive Stopped is disabled 1b - Receive Stopped is enabled
7 RBUE	Receive Buffer Unavailable Enable When this bit is set along with the AIE bit, the Receive Buffer Unavailable interrupt is enabled. When this bit is reset, the Receive Buffer Unavailable interrupt is disabled. 0b - Receive Buffer Unavailable is disabled 1b - Receive Buffer Unavailable is enabled
6 RIE	Receive Interrupt Enable When this bit is set along with the NIE bit, the Receive Interrupt is enabled. When this bit is reset, the Receive Interrupt is disabled. 0b - Receive Interrupt is disabled 1b - Receive Interrupt is enabled
5-3	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
2 TBUE	Transmit Buffer Unavailable Enable When this bit is set along with the NIE bit, the Transmit Buffer Unavailable interrupt is enabled. When this bit is reset, the Transmit Buffer Unavailable interrupt is disabled. 0b - Transmit Buffer Unavailable is disabled 1b - Transmit Buffer Unavailable is enabled
1 TXSE	Transmit Stopped Enable When this bit is set along with the AIE bit, the Transmission Stopped interrupt is enabled. When this bit is reset, the Transmission Stopped interrupt is disabled. 0b - Transmit Stopped is disabled 1b - Transmit Stopped is enabled
0 TIE	Transmit Interrupt Enable When this bit is set along with the NIE bit, the Transmit Interrupt is enabled. When this bit is reset, the Transmit Interrupt is disabled. 0b - Transmit Interrupt is disabled 1b - Transmit Interrupt is enabled

76.17.388 DMA Channel 2 Rx Interrupt Watchdog Timer (DMA_CH2_Rx_Interrupt_Watchdog_Timer)

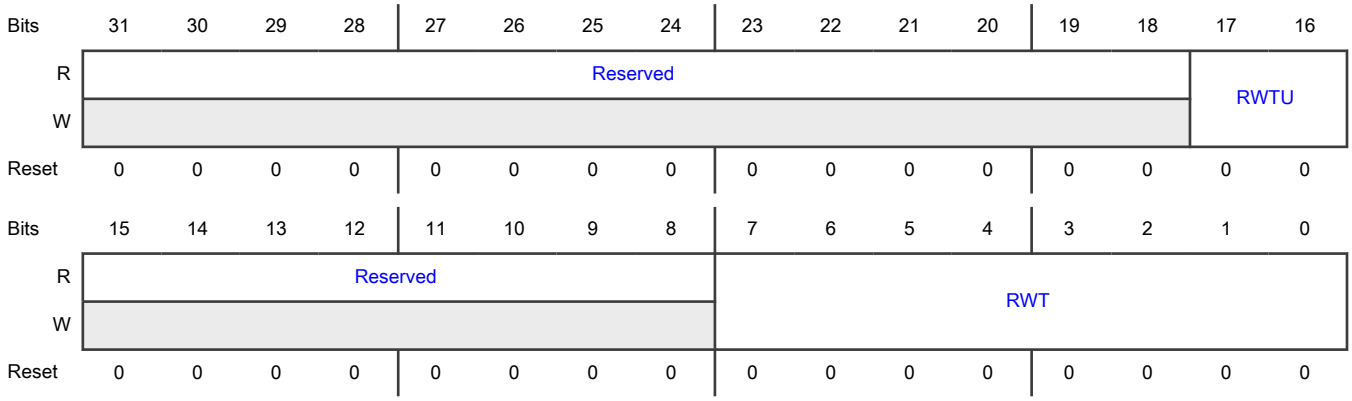
Offset

Register	Offset
DMA_CH2_Rx_Interrupt_Watchdog_Timer	1238h

Function

The Receive Interrupt Watchdog Timer register indicates the watchdog timeout for Receive Interrupt (RI) from the DMA. When this register is written with a non-zero value, it enables the watchdog timer for the RI bit of the DMA_CHi_Status register.

Diagram



Fields

Field	Function
31-18 —	Reserved
17-16 RWTU	Receive Interrupt Watchdog Timer Count Units This fields indicates the number of system clock cycles corresponding to one unit in RWT field. - 2'b00: 256 - 2'b01: 512 - 2'b10: 1024 - 2'b11: 2048 For example, when RWT=2 and RWTU=1, the watchdog timer is set for 2*512=1024 system clock cycles. **NOTE** This field is not testable by an automated register test.
15-8 —	Reserved
7-0 RWT	Receive Interrupt Watchdog Timer Count This field indicates the number of system clock cycles, multiplied by factor indicated in RWTU field, for which the watchdog timer is set. The watchdog timer is triggered with the programmed value after the Rx DMA completes the transfer of a packet for which the RI bit is not set in the DMA_CH(#i)_Status register, because of the setting of Interrupt Enable bit in the corresponding descriptor RDES3[30]. When the watchdog timer runs out, the RI bit is set and the timer is stopped. The watchdog timer is reset when the RI bit is set high because of automatic setting of RI as per the Interrupt Enable bit RDES3[30] of any received packet. **NOTE** This field is not testable by an automated register test.

76.17.389 DMA Channel 2 Slot Function Control Status (DMA_CH2_Slot_Function_Control_Status)

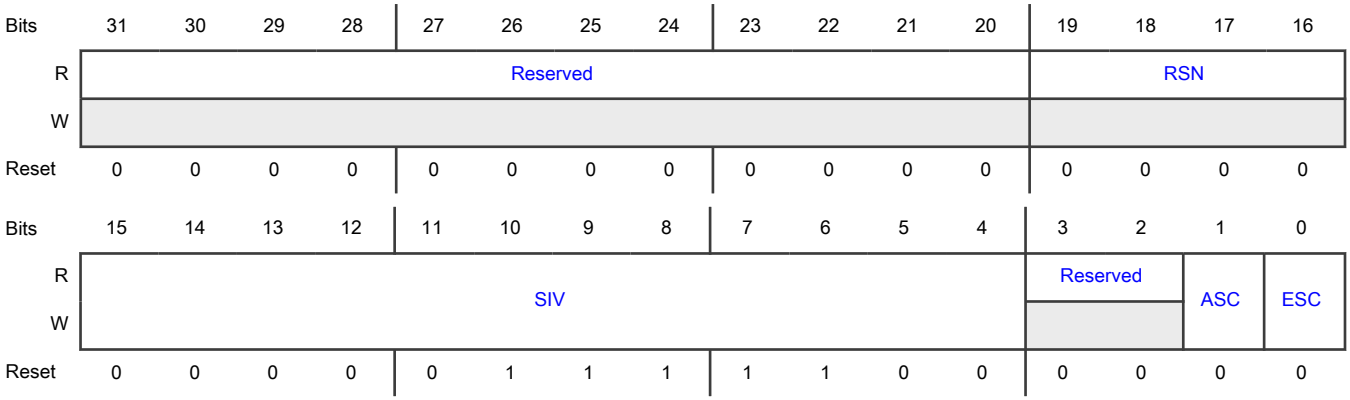
Offset

Register	Offset
DMA_CH2_Slot_Function_Control_Status	123Ch

Function

The Slot Function Control and Status register contains the control bits for slot function and the status for Transmit path.

Diagram



Fields

Field	Function
31-20 —	Reserved
19-16 RSN	Reference Slot Number This field gives the current value of the reference slot number in the DMA. It is used for slot comparison. **NOTE** This field is not testable by an automated register test.
15-4 SIV	Slot Interval Value This field controls the period of the slot interval in which the TxDMA fetches the scheduled packets. A value of 0 specifies the slot interval of 1 us while the maximum value 4095 specifies the slot interval of 4096us. The default/reset value is 0x07C which corresponds to slot interval of 125us
3-2 —	Reserved
1	Advance Slot Check

Table continues on the next page...

Table continued from the previous page...

Field	Function
ASC	When set, this bit enables the DMA to fetch the data from the buffer when the slot number (SLOTNUM) programmed in the Tx descriptor is - equal to the reference slot number given in the RSN field or - ahead of the reference slot number by up to two slots This bit is applicable only when the ESC bit is set. 0b - Advance Slot Check is disabled 1b - Advance Slot Check is enabled
0 ESC	Enable Slot Comparison When set, this bit enables the checking of the slot numbers programmed in the Tx descriptor with the current reference given in the RSN field. The DMA fetches the data from the corresponding buffer only when the slot number is - equal to the reference slot number or - ahead of the reference slot number by one slot When reset, this bit disables the checking of the slot numbers. The DMA fetches the data immediately after the descriptor is processed. Note: The UFO (UDP Fragmentation over IPv4)/TSO/USO should not be enabled along with TBS/AVB Slot number check. The UFO/TSO/USO involves multiple packets/segments/fragments transmission for single packet received from application and the slot number check are applicable for fetching of only first segment/fragment. As a result it might be difficult for software to specify slot number for subsequent packets. 0b - Slot Comparison is disabled 1b - Slot Comparison is enabled

76.17.390 DMA Channel 2 Current Application Transmit Descriptor (DMA_CH2_Current_App_TxDesc)

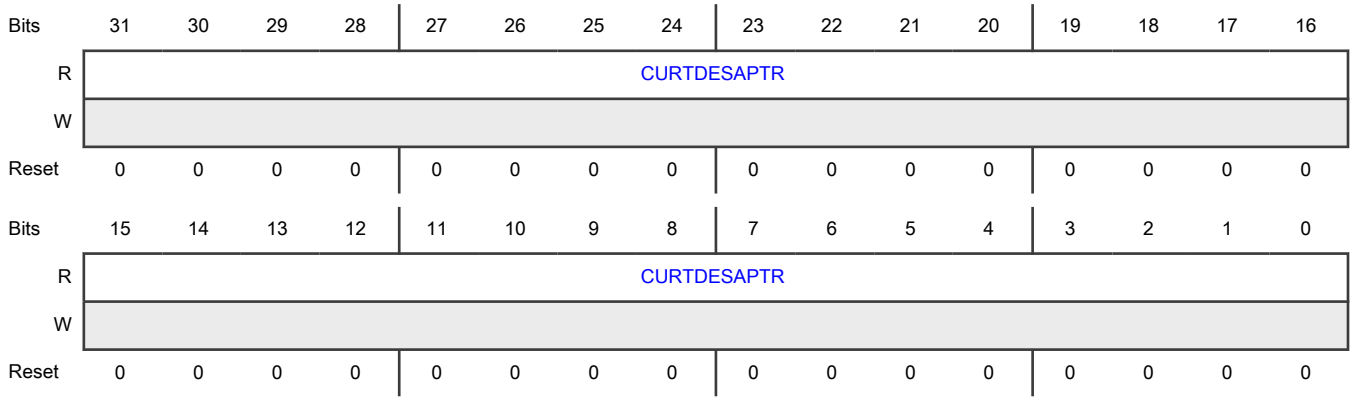
Offset

Register	Offset
DMA_CH2_Current_App_TxDesc	1244h

Function

The Channeli Current Application Transmit Descriptor register points to the current Transmit descriptor read by the DMA.

Diagram



Fields

Field	Function
31-0 CURTDESAPTR	Application Transmit Descriptor Address Pointer The DMA updates this pointer during Tx operation. This pointer is cleared on reset.

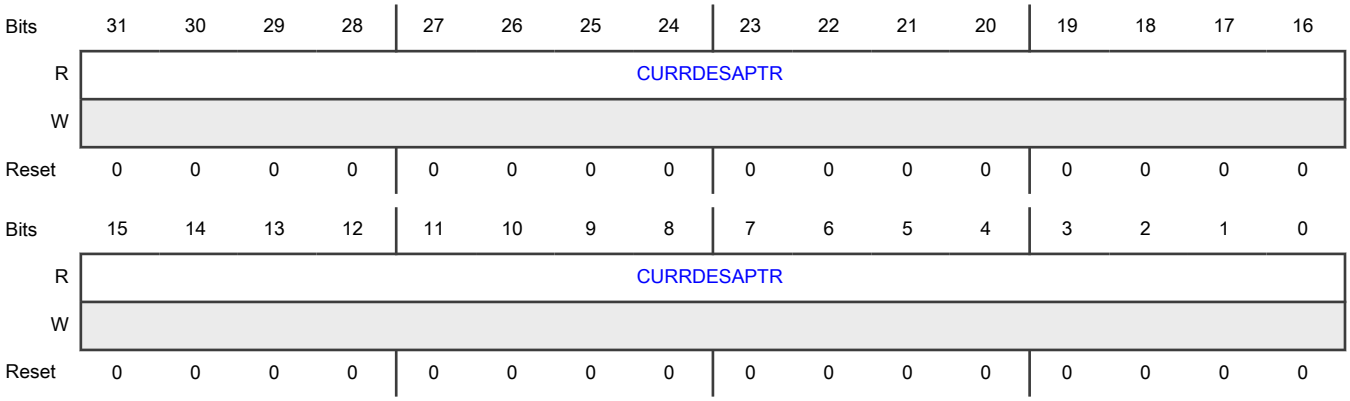
76.17.391 DMA Channel 2 Current Application Receive Descriptor (DMA_CH2_Current_App_RxDesc)

Offset

Register	Offset
DMA_CH2_Current_App_RxDesc	124Ch

Function
The Channeli Current Application Receive Descriptor register points to the current Receive descriptor read by the DMA.

Diagram



Fields

Field	Function
31-0 CURRDESAPTR	Application Receive Descriptor Address Pointer The DMA updates this pointer during Rx operation. This pointer is cleared on reset.

76.17.392 DMA Channel 2 Current Application Transmit Buffer (DMA_CH2_Current_App_TxBuffer)

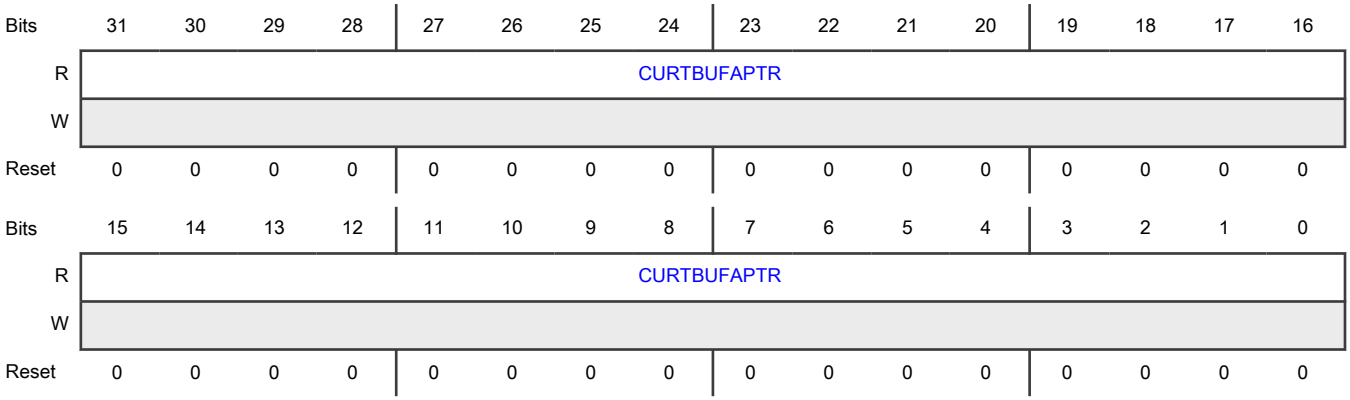
Offset

Register	Offset
DMA_CH2_Current_App_TxBuffer	1254h

Function

The Channeli Current Application Transmit Buffer Address register points to the current Tx buffer address read by the DMA.

Diagram



Fields

Field	Function
31-0 CURTBUFAPTR	Application Transmit Buffer Address Pointer The DMA updates this pointer during Tx operation. This pointer is cleared on reset.

76.17.393 DMA Channel 2 Current Application Receive Buffer (DMA_CH2_Current_App_RxBuffer)

Offset

Register	Offset
DMA_CH2_Current_App_RxBuffer	125Ch

Function

The Channel 0 Current Application Receive Buffer Address register points to the current Rx buffer address read by the DMA.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CURRBUFAPTR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CURRBUFAPTR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 CURRBUFAPTR	Application Receive Buffer Address Pointer The DMA updates this pointer during Rx operation. This pointer is cleared on reset.

76.17.394 DMA Channel 2 Status (DMA_CH2_Status)

Offset

Register	Offset
DMA_CH2_Status	1260h

Function

The software driver (application) reads the Status register during interrupt service routine or polling to determine the status of the DMA. Note: The number of DMA_CH(#i)_Status register in the configuration is the higher of number of Rx DMA Channels and Tx DMA Channels.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved										REB			TEB		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	NIS	AIS	CDE	FBE	ERI	ETI	RWT	RPS	RBU	RI	Reserved			TBU	TPS	TI
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-22 —	Reserved
21-19 REB	Rx DMA Error Bits This field indicates the type of error that caused a Bus Error. For example, error response on the AHB or AXI interface. - Bit 21 -- 1'b1: Error during data transfer by Rx DMA -- 1'b0: No Error during data transfer by Rx DMA - Bit 20 -- 1'b1: Error during descriptor access -- 1'b0: Error during data buffer access - Bit 19 -- 1'b1: Error during read transfer -- 1'b0: Error during write transfer This field is valid only when the FBE bit is set. This field does not generate an interrupt. **NOTE** This field is not testable by an automated register test.
18-16 TEB	Tx DMA Error Bits This field indicates the type of error that caused a Bus Error. For example, error response on the AHB or AXI interface. - Bit 18 -- 1'b1: Error during data transfer by Tx DMA -- 1'b0: No Error during data transfer by Tx DMA - Bit 17 -- 1'b1: Error during descriptor access -- 1'b0: Error during data buffer access - Bit 16 -- 1'b1: Error during read transfer -- 1'b0: Error during write transfer This field is valid only when the FBE bit is set. This field does not generate an interrupt. **NOTE** This field is not testable by an automated register test.
15 NIS	Normal Interrupt Summary Normal Interrupt Summary bit value is the logical OR of the following bits when the corresponding interrupt bits are enabled in the DMA_CH0_Interrupt_Enable register: - Bit 0: Transmit Interrupt - Bit 2: Transmit Buffer Unavailable - Bit 6: Receive Interrupt - Bit 11: Early Receive Interrupt Only unmasked bits (interrupts for which interrupt enable is set in DMA_CH0_Interrupt_Enable register) affect the Normal Interrupt Summary bit. This is a sticky bit. You must clear this bit (by writing 1 to this bit) each time a corresponding bit which causes NIS to be set is cleared. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **NOTE** This field is not testable by an automated register test. 0b - Normal Interrupt Summary status not detected 1b - Normal Interrupt Summary status detected
14 AIS	Abnormal Interrupt Summary Abnormal Interrupt Summary bit value is the logical OR of the following when the corresponding interrupt bits are enabled in the DMA_CH0_Interrupt_Enable register: - Bit 1: Transmit Process Stopped - Bit 7: Receive Buffer Unavailable - Bit 8: Receive Process Stopped - Bit 10: Early Transmit Interrupt - Bit 12: Fatal Bus Error - Bit 13: Context Descriptor Error Only unmasked bits affect the Abnormal Interrupt Summary bit. This is a sticky bit. You must clear this bit (by writing 1 to this bit) each time a corresponding bit, which causes AIS to be set, is cleared. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field is not testable by an automated register test. 0b - Abnormal Interrupt Summary status not detected 1b - Abnormal Interrupt Summary status detected
13 CDE	Context Descriptor Error This bit indicates that the DMA Tx/Rx engine received a descriptor error, which indicates invalid context in the middle of packet flow (intermediate descriptor) or all one's descriptor in Tx case and on Rx side it indicates DMA has read a descriptor with either of the buffer address as ones which is considered to be invalid. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **NOTE** This field is not testable by an automated register test. 0b - Context Descriptor Error status not detected 1b - Context Descriptor Error status detected
12 FBE	Fatal Bus Error This bit indicates that a bus error occurred (as described in the EB field). When this bit is set, the corresponding DMA channel engine disables all bus accesses. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **NOTE** This field is not testable by an automated register test. 0b - Fatal Bus Error status not detected 1b - Fatal Bus Error status detected
11 ERI	Early Receive Interrupt This bit when set indicates that the RxDMA has completed the transfer of packet data to the memory. In configurations supporting ERIC, - ERIC = 0: This bit is set only after the Rx DMA has completely filled up a receive buffer with packet data. - ERIC = 1: This bit is set after every burst transfer of data from the Rx DMA to the buffer. The setting of RI bit automatically clears this bit. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **NOTE** This field is not testable by an automated register test. 0b - Early Receive Interrupt status not detected 1b - Early Receive Interrupt status detected
10 ETI	Early Transmit Interrupt This bit when set indicates that the TxDMA has completed the transfer of packet data to the MTL TXFIFO memory. In configurations supporting ERIC: - ETIC = 0: This bit is set only after the Tx DMA has

Table continues on the next page...

Table continued from the previous page...

Field	Function
	transferred a complete packet to MTL. - ETIC = 1: This bit is set after completion of (partial) packet data transfer from buffers in the Transmit descriptor in which IOC = 1. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **NOTE** This field is not testable by an automated register test. 0b - Early Transmit Interrupt status not detected 1b - Early Transmit Interrupt status detected
9 RWT	Receive Watchdog Timeout This bit is asserted when a packet with length greater than 2,048 bytes (10,240 bytes when Jumbo Packet mode is enabled) is received. **NOTE** This field is not testable by an automated register test. 0b - Receive Watchdog Timeout status not detected 1b - Receive Watchdog Timeout status detected
8 RPS	Receive Process Stopped This bit is asserted when the Rx process enters the Stopped state. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **NOTE** This field is not testable by an automated register test. 0b - Receive Process Stopped status not detected 1b - Receive Process Stopped status detected
7 RBU	Receive Buffer Unavailable This bit indicates that the application owns the next descriptor in the Receive list, and the DMA cannot acquire it. The Rx process is suspended. To resume processing Rx descriptors, the application should change the ownership of the descriptor and issue a Receive Poll Demand command. If this command is not issued, the Rx process resumes when the next recognized incoming packet is received. In ring mode, the application should advance the Receive Descriptor Tail Pointer register of a channel. This bit is set only when the DMA owns the previous Rx descriptor. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **NOTE** This field is not testable by an automated register test. 0b - Receive Buffer Unavailable status not detected 1b - Receive Buffer Unavailable status detected
6 RI	Receive Interrupt

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This bit indicates that the packet reception is complete. When packet reception is complete, Bit 31 of RDES3 is reset in the last descriptor, and the specific packet status information is updated in the descriptor. The reception remains in the Running state. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **NOTE** This field is not testable by an automated register test. 0b - Receive Interrupt status not detected 1b - Receive Interrupt status detected
5-3 —	Reserved
2 TBU	Transmit Buffer Unavailable This bit indicates that the application owns the next descriptor in the Transmit list, and the DMA cannot acquire it. Transmission is suspended. The TPS0 field of the DMA_Debug_Status0 register explains the Transmit Process state transitions. To resume processing the Transmit descriptors, the application should do the following: - Change the ownership of the descriptor by setting Bit 31 of TDES3. - Issue a Transmit Poll Demand command. For ring mode, the application should advance the Transmit Descriptor Tail Pointer register of a channel. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **NOTE** This field is not testable by an automated register test. 0b - Transmit Buffer Unavailable status not detected 1b - Transmit Buffer Unavailable status detected
1 TPS	Transmit Process Stopped This bit is set when the transmission is stopped. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **NOTE** This field is not testable by an automated register test. 0b - Transmit Process Stopped status not detected 1b - Transmit Process Stopped status detected
0 TI	Transmit Interrupt This bit indicates that the packet transmission is complete. When transmission is complete, Bit 31 of TDES3 is reset in the last descriptor, and the specific packet status information is updated in the descriptor. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **NOTE** This field is not testable by an automated register test.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Transmit Interrupt status not detected 1b - Transmit Interrupt status detected

76.17.395 DMA Channel 2 Miss Frame Count (DMA_CH2_Miss_Frame_Cnt)

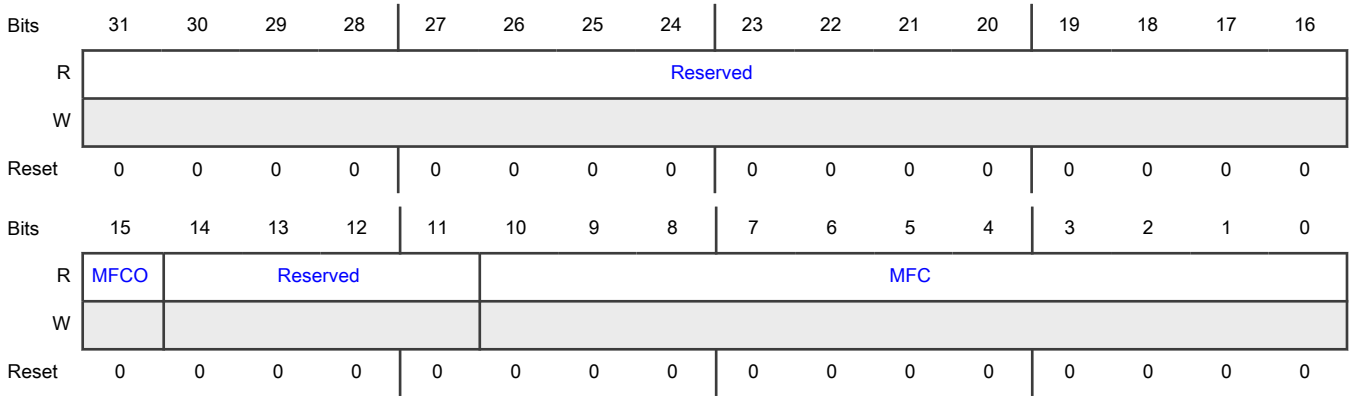
Offset

Register	Offset
DMA_CH2_Miss_Frame_Cnt	1264h

Function

This register has the number of packet counter that got dropped by the DMA either due to Bus Error or due to programming RPF field in DMA_CH${i}_Rx_Control register.

Diagram



Fields

Field	Function
31-16 —	Reserved
15 MFCO	Overflow status of the MFC Counter When this bit is set then the MFC counter does not get incremented further. The bit gets cleared when this register is read. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0b - Miss Frame Counter overflow not occurred 1b - Miss Frame Counter overflow occurred

Table continues on the next page...

Table continued from the previous page...

Field	Function
14-11 —	Reserved
10-0 MFC	Dropped Packet Counters This counter indicates the number of packet counters that are dropped by the DMA either because of bus error or because of programing RPF field in DMA_CH${i}_Rx_Control register. The counter gets cleared when this register is read. Access restriction applies. Clears on read. Self-set to 1 on internal event.

76.17.396 DMA Channel 2 Rx Parser Accept Count (DMA_CH2_RXP_Accept_Cnt)

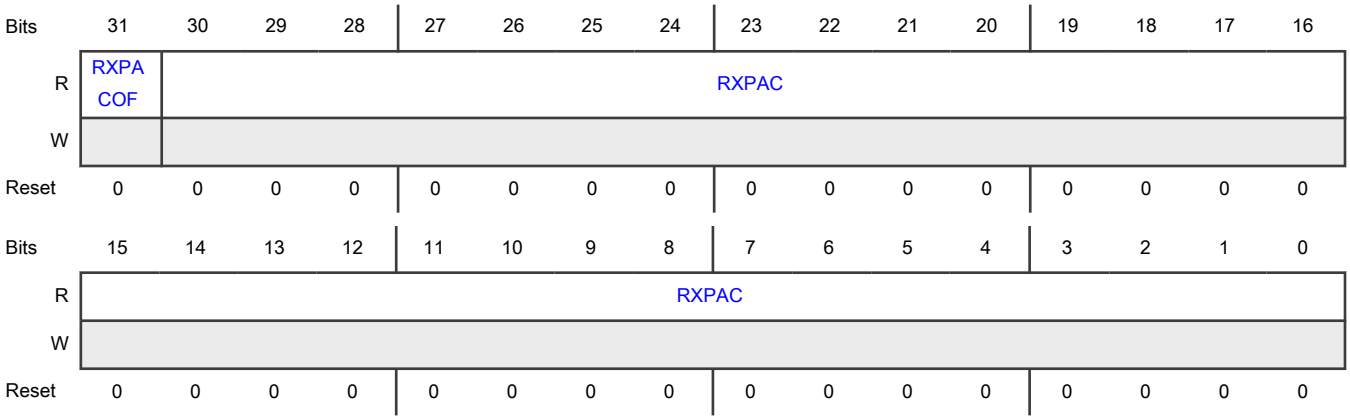
Offset

Register	Offset
DMA_CH2_RXP_Accept_Cnt	1268h

Function

The DMA_CH(#i)_RXP_Accept_Cnt registers provides the count of the number of frames accepted by Rx Parser.

Diagram



Fields

Field	Function
31 RXPACOF	Rx Parser Accept Counter Overflow Bit When set, this bit indicates that the RXPAC Counter field crossed the maximum limit. Access restriction applies. Clears on read. Self-set to 1 on internal event.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Rx Parser Accept Counter overflow not occurred 1b - Rx Parser Accept Counter overflow occurred
30-0 RXPAC	Rx Parser Accept Counter This 31-bit counter is implemented when a Rx Parser Accept a packet due to AF =1. The counter is cleared when the register is read.

76.17.397 DMA Channel 2 Rx ERI Count (DMA_CH2_RX_ERI_Cnt)

Offset

Register	Offset
DMA_CH2_RX_ERI_Cnt	126Ch

Function
The DMA_CH(#i)_RX_ERI_Cnt registers provides the count of the number of times ERI was asserted.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved				ECNT											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-12 —	Reserved
11-0 ECNT	ERI Counter When ERIC bit of DMA_CH(#i)_RX_Control register is set, this counter increments for burst transfer completed by the Rx DMA from the start of packet transfer. This counter is reset at the start of new packet.

76.18 Glossary

AFM Address filtering module

ARI Application receive interface

ATI Application transmit interface

AVB Audio video bridging

AXI Advanced extensible interface

BTR Base time register

CPT Current presentation time

CRC Cyclic redundancy check

DA Destination address

DCB Data center bridging

DUT Device under test

EOP End of packet

FCS A 32-bit cyclic redundancy check used to detect an in-transit corruption of data. Also marks the end of an Ethernet frame.

FIFO First in first out

GCL Gate control list

GMII Gigabit media independent interface

MAC Media access controller

MCI MAC control interface

MDC Management data clock

MDIO Management data input/output

MII Media independent interface

MRI MAC receive interface

MTI MAC transmit interface

MTL MAC transmit layer

PBL Programmable burst length

PHY The physical interface transceiver that implements the OSI physical layer for an ethernet network. The IEEE 802.3 standard defines it.

RGMII Reduced gigabit media independent interface

RMII Reduced media independent interface

RMON Remote network monitoring

SA Source address

SFD Start frame delimiter

SGMII Serial gigabit media independent interface

SMA Station management agent

TCP Transmission control protocol

UDP	User datagram protocol
VLAN	Virtual local area network

Chapter 77
Low Power Universal Asynchronous Receiver/ Transmitter (LPUART)

77.1 Chip-specific LPUART information

77.1.1 LPUART instances and configuration

Table 685. LPUART instances

Instance	S32K314/S32K324/S32K44/S32K358/S32K356/ S32K348/S32K338/S32K328/S32K388/ S32K389	S32K310/S32K311/S32K322/ S32K342/S32K341	S32K312
LPUART_0	Yes	Yes	Yes
LPUART_1	Yes	Yes	Yes
LPUART_2	Yes	Yes	Yes
LPUART_3	Yes	Yes	Yes
LPUART_4	Yes	No	Yes
LPUART_5	Yes	No	Yes
LPUART_6	Yes	No	Yes
LPUART_7	Yes	No	Yes
LPUART_8	Yes	No	No
LPUART_9	Yes	No	No
LPUART_10	Yes	No	No
LPUART_11	Yes	No	No
LPUART_12	Yes	No	No
LPUART_13	Yes	No	No
LPUART_14	Yes	No	No
LPUART_15	Yes	No	No

Table 686. LPUART Configuration

Feature	Configuration		
	S32K344/S32K324/ S32K314/S32K312	S32K342/ S32K322/S32K341/ S32K311/S32K310	S32K358/S32K356/ S32K348/S32K338/S32K328/ S32K388/S32K389
TX FIFO size	4 Words	16 Words (for instance 0,1) 4 Words (for instance 2,3)	16 Words (for instance 0,1) 4 Words (for instance 2 to 15)
RX FIFO size	4 Words	16 Words (for instance 0,1)	16 Words (for instance 0,1)

Table continues on the next page...

Table 686. LPUART Configuration (continued)

Feature	Configuration		
		4 Words (for instance 2,3)	4 Words (for instance 2 to 15)
Functionality supported	Standard LPUART functionality with MODEM/IrDA	• Standard LPUART functionality with MODEM/IrDA • MODBUS and PPROFIBUS support[1]	
	LIN master and slave operation		

1. Only supported for instance LPUART_0 and LPUART_1.

NOTE

LPUART instances are not available during Standby.

LPUART[0:2]_trg_input is coming from TRGMUX and you should take care of the pulse width of trigger. It should follow the requirement mentioned in section "Peripheral Triggers". These triggers are not present in S32K314, S32K324, S32K344.

77.2 Overview

LPUART provides asynchronous, serial communication capabilities with external devices. It supports the non-return-to-zero (NRZ) encoding format and infrared data association (IrDA)-compatible, low-speed serial infrared (SIR) protocol. LPUART can continue operating when the processor is in Low-Power mode, if an appropriate peripheral clock is available.

77.2.1 Block diagram

Figure 487 shows the transmitter portion of LPUART.

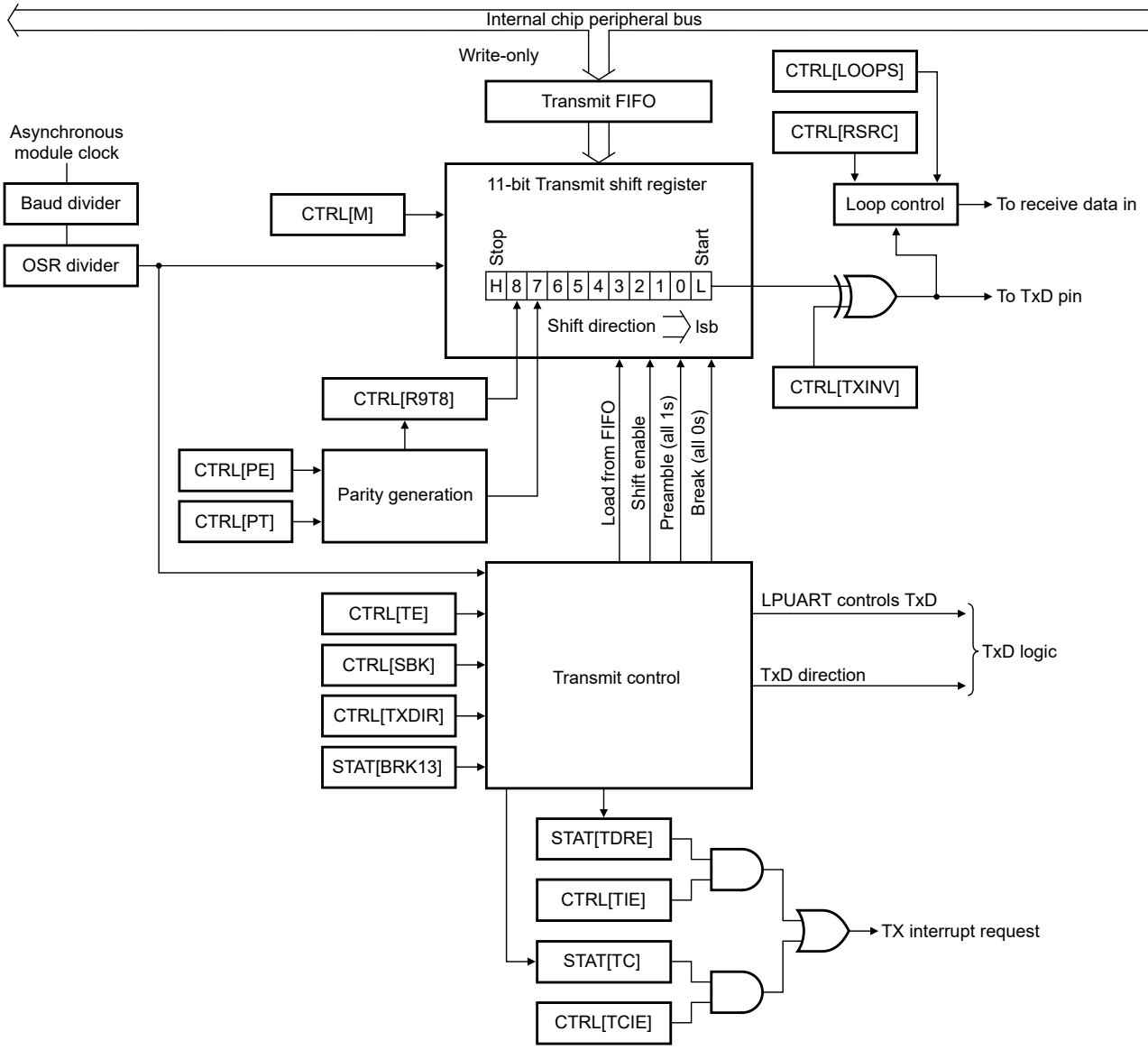


Figure 487. Transmitter block diagram

Figure 488 shows the receiver portion of LPUART.

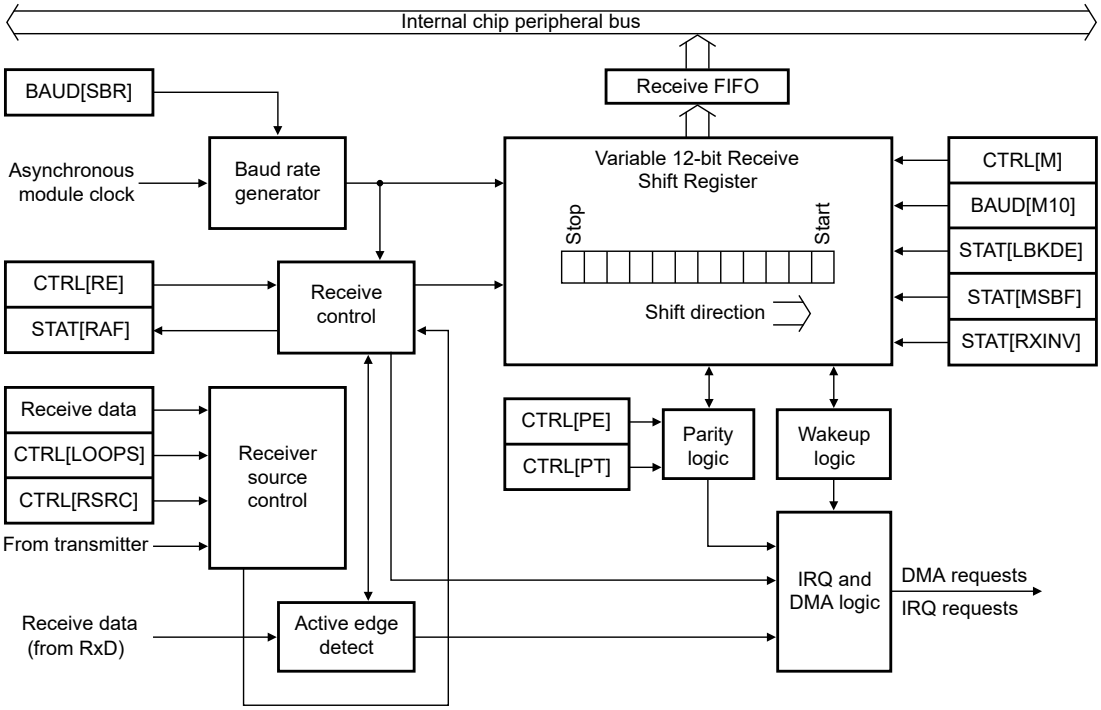


Figure 488. Receiver block diagram

77.2.2 Features

- Full-duplex, standard NRZ format
- Programmable baud rates (13-bit modulo divider) with a configurable oversampling ratio (OSR) from 4× to 32×
- Asynchronous operation of transmit and receive baud rates with respect to the bus clock:
  - Baud rate can be configured independently of the bus clock frequency.
  - Operation in Low-Power modes is supported.
- Interrupt, DMA, or polled operations:
  - Transmit data empty and transmission complete
  - Receive data full
  - Receive overrun, parity error, framing error, and noise error
  - Idle receiver detect
  - Active edge on receive pin
  - Break detect supporting LIN
  - Receive data match
- Hardware parity generation and checking
- Programmable 7-bit, 8-bit, 9-bit, or 10-bit character length
- Programmable 1-bit or 2-bit stop bits
- Support for three receiver wake-up methods:
  - Idle line wake-up

  - Address mark wake-up
  - Receive data match
- Automatic address matching to reduce ISR overhead:
  - Address mark matching
  - Idle line address matching
  - Address match start, address match end
- Optional 13-bit and 11-bit break character generation
- Configurable idle length detection supporting 1, 2, 4, 8, 16, 32, 64, or 128 idle characters
- Selectable transmitter output and receiver input polarity
- Hardware flow control support for request to send (RTS) and clear to send (CTS) signals
- Selectable IrDA 1.4 return-to-zero-inverted (RZI) format with a programmable pulse width
- Independent FIFO structure for transmit and receive functions:
  - Separate configurable watermarks for receive and transmit requests
  - Option for receiver to assert request after a configurable number of idle characters, if receive FIFO is not empty

77.3 Functional description

LPUART supports full-duplex, asynchronous, NRZ serial communication and comprises a baud rate generator, transmitter, and receiver block. The transmitter and receiver operate independently, although they use the same baud rate generator. The following sections describe all LPUART blocks.

77.3.1 Baud rate generation

A 13-bit modulus counter in the baud rate generator derives the baud rate for both the receiver and transmitter. The value, ranging from 1 to 8191, written to BAUD[SBR] determines the baud clock divisor for the asynchronous LPUART baud clock. The baud rate clock drives the receiver, while a bit clock, generated from the baud rate clock divided by the OSR, drives the transmitter. Depending on the OSR, the receiver has an acquisition rate of 4 to 32 samples per bit time. LPUART requires BAUD[SBR] and BAUD[OSR] to accurately match the baud rate requirement. There is a relationship between the required baud rate and the input clock frequency, SBR, and OSR. See the below figure for details.

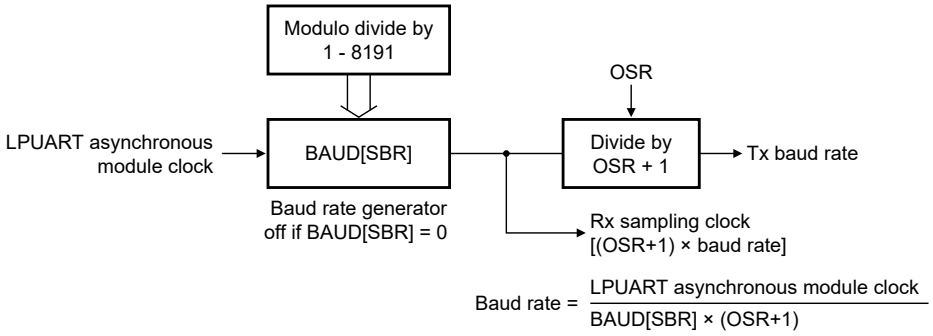


Figure 489. Baud rate generation

Baud rate generation is subject to these sources of error:

- Integer division of the asynchronous LPUART baud clock may not give the exact target frequency.
- Synchronization with the asynchronous LPUART baud clock can lead to a phase shift.

Baud rate generation is a free-running counter that continues whenever the transmitter or receiver is enabled. The transmitter bit clock continues whenever the transmitter is enabled; each transmitted character aligns to the next edge of the transmit bit clock.

In general, configuring OSR for a higher ratio and/or sampling on both edges of the clock slightly improves LPUART's tolerance to baud rate mismatch between the received data and LPUART configured baud rate. However, the three data samples in each bit (see Data sampling technique) are also closer together, which may impact noise sensitivity.

77.3.2 Baud rate tolerance

A transmitting device may operate at a baud rate below or above that of the receiver.

Accumulated bit time misalignment can cause one of the three stop bit data samples to fall outside the actual stop bit. A noise error will occur if the three samples are not all the same logical values. A framing error will occur if the receiver clock is misaligned in such a way that the majority of the three stop bit samples are a logic zero.

As the receiver samples an incoming frame, it may resynchronize the oversampling clock on any valid falling edge within the frame. Resynchronization within frames will correct a misalignment between transmitter bit times and receiver bit times.

In general, increasing the number of samples per bit will increase the baud rate tolerance and decreasing the number of samples per bit will reduce the baud rate tolerance. Note that since LPUART implements triple voting on consecutive receive data samples, increasing the number of samples per bit will move those samples closer together which would reduce the width of noise that can be filtered by the triple voting logic.

77.3.3 Calculating baud rate tolerance

Using the following definitions:

- SAM is the number of sample points per bit (valid range from 8 to 32; equal to (OSR + 1) × (BOTHEDGE + 1)).
- BIT is the number of bits in a character including start, data and stop bits (valid range from 9 to 13).

The ideal baud rate tolerance can be calculated as follows:

- Slow data rate tolerance = ((SAM ÷ 2) - 1) ÷ ((SAM × BIT) - (SAM ÷ 2) + 2)
- Fast data rate tolerance = ((SAM ÷ 2) - 2) ÷ (SAM × BIT)

As an example, if configured for 8-bit data, 1 stop bit (BIT = 10) and with OSR=0x7 and BOTHEDGE = 1 (SAM = 16):

- Slow data rate tolerance = (8 - 1) ÷ (160 - 8 + 2) = 7 ÷ 154 = 4.54%
- Fast data rate tolerance = (8 - 2) ÷ 160 = 6 ÷ 160 = 3.75%

If configured for 9-bit data with 1 stop bit (BIT = 11) with same oversampling configuration, then:

- Slow data rate tolerance = (8 - 1) ÷ (176 - 8 + 2) = 7 ÷ 170 = 4.12%
- Fast data rate tolerance = (8 - 2) ÷ 176 = 6 ÷ 176 = 3.41%

NOTE

Additional factors can contribute to a lower baud rate tolerance than the ideal. These include clock uncertainty or jitter on the LPUART functional clock source, differences in rise and fall times on the transmitter output and synchronization of the external receive pin to the local LPUART functional clock.

77.3.4 Transmitter functional description

This section describes the functioning of the LPUART transmitter, as shown in the transmitter portion of Block diagram, as well as specialized functions for sending break and idle characters.

The transmitter output (TXD) idle state defaults to logic high; the transmitter output is inverted when you write 1 to CTRL[TXINV], which becomes 0 following reset. You can enable the transmitter by writing 1 to CTRL[TE]. This queues a preamble character that is one full character frame of the Idle state. The transmitter then remains idle until data is available in the transmit FIFO and programs store data in the transmit FIFO by writing to Data (DATA).

The central element of the LPUART transmitter is the transmit shift register that is 9-bit to 13-bit long depending on the settings of CTRL[M], CTRL[M7], BAUD[M10], and BAUD[SBNS]. Going forward in this discussion, assume that CTRL[M], CTRL[M7], BAUD[M10], and BAUD[SBNS] are 0, selecting the normal 8-bit Data mode, in which the shift register holds a start bit, eight data bits, and a stop bit. When the transmit shift register is available for a new character, the value waiting in transmit FIFO is transferred to the transmit shift register, synchronized with the baud rate clock, and STAT[TDRE] becomes 1 to indicate that another character may be written to the transmit FIFO at Data (DATA).

If no new character is waiting in the transmit FIFO after a stop bit is shifted out of the TXD pin, the transmitter sets the transmit complete flag and enters an idle mode, with TXD high, waiting for more characters to transmit.

Writing 0 to CTRL[TE] does not immediately disable the transmitter. The current transmit activity in progress must first be completed (that could include a data character, idle character, or break character), although the transmitter does not start transmitting another character.

77.3.4.1 Break character length

CTRL[SBK] sends break characters, originally used to gain the attention of old teletype receivers. Break characters are a full character time of logic 0, 9-bit to 12-bit times, including the start and stop bits. You can enable a longer break of 13-bit times by writing 1 to STAT[BRK13]. Normally, a program waits for STAT[TDRE] to become 1 to indicate that the last character of a message has moved to the transmit shifter. Next, the program writes 1 and then writes 0 to CTRL[SBK]. This action queues a break character to be sent as soon as the shifter is available. If CTRL[SBK] remains 1 when the queued break moves into the shifter, synchronized with the baud rate clock, an additional break character is queued. When LPUART is the receiving module, it receives a break character as 0s in all data bits and a framing error (STAT[FE] = 1) is detected.

You can also transmit a break character by writing to Data (DATA) with DATA[FRETSC] = 1 and the data bits clear. This supports transmitting the break character as part of the normal data stream and also allows DMA to transmit a break character.

When idle line wake-up is used, a full character time of idle (logic 1) is needed between messages to wake up any sleeping receivers. Normally, a program waits for STAT[TDRE] to become 1 to indicate that the last character of a message has moved to the transmit shifter. Next, write 0 and then write 1 to CTRL[TE]. This action queues an idle character to be sent as soon as the shifter is available. As long as the character in the shifter does not finish while CTRL[TE] becomes 0, the LPUART transmitter does not release control of the TXD pin.

You can also write to Data (DATA) to transmit an idle character, with DATA[FRETSC] and DATA[R9T9] = 1 and the values of all the other fields = 0. This supports transmitting the idle character as part of the normal data stream and also allows DMA to transmit an idle character.

As shown in the following table, STAT[BRK13], CTRL[M], CTRL[M7], BAUD[M10], and BAUD[SBNS] affect the length of the break character.

Table 687. Break character length

STAT[BRK13]	CTRL[M]	BAUD[M10]	CTRL[M7]	BAUD[SBNS]	Break character length (in bit times)
0	0	0	0	0	10
0	0	0	0	1	11
0	0	0	1	0	9
0	0	0	1	1	10
0	1	0	—	0	11
0	1	0	—	1	12
0	—	1	—	0	12
0	—	1	—	1	13
1	0	0	0	0	13

Table continues on the next page...

Table 687. Break character length (continued)

STAT[BRK13]	CTRL[M]	BAUD[M10]	CTRL[M7]	BAUD[SBNS]	Break character length (in bit times)
1	0	0	0	1	13
1	0	0	1	0	12
1	0	0	1	1	12
1	1	0	—	0	14
1	1	0	—	1	14
1	—	1	—	0	15
1	—	1	—	1	15

77.3.4.2 Hardware flow control

The transmitter supports hardware flow control by gating the transmission with the value of CTS_B. If the CTS operation is enabled, the character is transmitted when CTS_B is asserted. If CTS_B is deasserted in the middle of a transmission with characters remaining in the transmitter FIFO, the character in the transmit shift register is complete. Any characters in the FIFO wait for CTS_B to assert again, and TXD remains in the mark state (idle state) until CTS_B is reasserted. The CTS_B pin must assert for longer than one bit period to guarantee that a new transmission is started when the transmitter is idle with CTS.

If the CTS operation is disabled, the transmitter ignores the state of CTS_B.

The transmitter's CTS_B signal can be enabled even if the same LPUART receiver's RTS_B signal is disabled.

77.3.4.3 Transceiver driver enable

The transmitter can use RTS_B as an enable signal for the driver of an external transceiver. See Transceiver driver enable using RTS_B for details. If the RTS operation is enabled, when a character is placed into an empty transmit shift register, RTS_B asserts 1-bit time before the start bit is transmitted. RTS_B remains asserted for the whole time that the transmit shift register has any characters. RTS_B deasserts 1-bit time after all characters in the transmit FIFO and shift register are completely sent, including the last stop bit. In other words, when RTS_B is used as a transceiver enable, RTS_B asserts 1-bit time before the transmitter starts transmitting and negates 1-bit time after the transmitter goes idle.

Transmitting a break character also asserts RTS_B, with the same assertion and deassertion timing as having a character in the transmit shift register.

The transmitter's RTS_B signal asserts only when the transmitter is enabled. However, the transmitter's RTS_B signal is unaffected by its CTS_B signal. RTS_B remains asserted until the transfer is complete, even if the transmitter is disabled mid-way through a data transfer.

You can configure HDCR[RTSEXT] to the desired length by delaying the transmitter's RTS_B negation by up to 256-bit clock (baud rate) after the last stop bit.

77.3.4.4 Transceiver driver enable using RTS_B

RS-485 is a multiple drop communication protocol in which the LPUART transceiver's driver is three-stated unless LPUART is driving. The transmitter can use the RTS_B signal to enable the driver of a transceiver. The polarity of RTS_B can be matched to the polarity of the transceiver's driver enable signal.

The following figure shows the receiver enable signal asserted. This connection can also connect RTS_B to both DE and RE_B. The transceiver's receiver is disabled when driving. A pullup can pull RXD to a nonfloating value during this time. You can refine this option further by operating LPUART in Single-Wire mode, freeing the RXD pin for other uses.

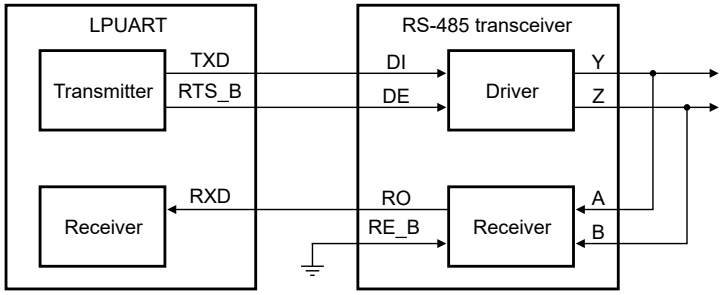


Figure 490. Transceiver driver enable using RTS_B

77.3.5 Receiver functional description

This section discusses the functioning of the LPUART receiver, as shown in the receiver portion of Block diagram. The section also discusses:

- The data sampling technique used to reconstruct receiver data.
- Different variations of the receiver wake-up function.

You can invert the receiver input by writing 1 to STAT[RXINV] and enable the receiver by writing 1 to CTRL[RE]. Character frames consist of a start bit of logic 0, along with N (7, 8, 9, 10) bits (MSB or LSB first), and one or two stop bits of logic 1. For information about 7-bit, 9-bit, or 10-bit Data mode, see Data modes. Going forward in this discussion, assume that LPUART is configured for a normal 8-bit Data mode.

After receiving the stop bit into the receive shifter, and provided the receive data register is not already full (STAT[RDRF] = 0), the data character is transferred to the receive FIFO, resulting in STAT[RDRF] becoming 1. However, if STAT[RDRF] is already 1, indicating that the receive data buffer is already full, STAT[OR] becomes 1 and the new data is lost.

Because the LPUART receiver is separate from the receive FIFO, the receive shift register can receive the next word when the receive FIFO is full, and it is only at the end of the character that the next data is written into the receive FIFO, potentially triggering the overrun flag if the FIFO is full.

When a program detects that the receive data register is full (STAT[RDRF] = 1), it gets the data from the FIFO by reading Data (DATA). See Interrupts for details about flag clearing.

77.3.5.1 Data sampling technique

The LPUART receiver supports a configurable oversampling rate of between 4× and 32× of the baud rate clock for sampling. The receiver starts by considering logic level samples at the oversampling rate times the baud rate to search for a falling edge on the RXD serial data input pin. A falling edge is defined as a logic 0 sample after three consecutive logic 1 samples. The oversampling baud rate clock divides the bit time into 4 to 32 segments from 1 to OSR (where OSR is the configured oversampling ratio). When a falling edge is located, three more samples are taken at (OSR ÷ 2), (OSR ÷ 2) + 1, and (OSR ÷ 2) + 2 to ensure that this is a real start bit and not merely noise. If at least two of these three samples are 0, the receiver assumes they are synchronized to a received character. If another falling edge is detected before the receiver is considered synchronized, the receiver restarts sampling from the first segment.

The receiver then samples each bit time, including the start and stop bits, at (OSR ÷ 2), (OSR ÷ 2) + 1, and (OSR ÷ 2) + 2, to determine the logic level for that bit. The logic level is interpreted to be that of the majority of the samples taken during the bit time. If any sample in any bit time, including the start and stop bits, in a character frame fails to agree with the logic level for that bit, noise flag (STAT[NF]) becomes 1 when the received character is transferred to the receive FIFO.

When the LPUART receiver is configured to sample on both edges of the baud rate clock (that is, when BAUD[BOTHEDGE] = 1), the number of segments in each received bit is effectively doubled (from 1 to OSR× 2). The start and data bits are then sampled at OSR, OSR + 1, and OSR + 2. You must enable sampling on both edges of the clock for oversampling rates of 4× to 7×. This sampling is optional for higher oversampling rates.

The synchronization feature of LPUART synchronizes the internal oversampling counter with a detected falling edge on the receive signal, and to adjust the data sampling window. The falling edge detection needs three consecutive 1s prior to the "1->0" (one to zero) transition. After the initial falling edge detection for the start bit, the circuit continuously monitors the next falling edge, and resets the counter after another falling edge is detected. This synchronization to the start bit is termed as resynchronization.

When BAUD[RESYNCDIS] is 0, you perform this falling edge detection and resynchronization not only for the start bit but also for the rest of the character reception after the start bit.

When BAUD[RESYNCDIS] is 1, you perform the falling edge detection and resynchronization only for the start bit. The use case for disabling the resynchronization is protocols that require this (for example, LIN 2.1 prohibits resynchronization within a byte).

The following table and figure explain LPUART resynchronization.

Table 688. LPUART resynchronization settings

Resynchronization	BAUD[RESYNCDIS] = 0	BAUD[RESYNCDIS] = 1
For the starting bit falling edge	Yes	Yes
For all falling edges after the start bit	Yes	No

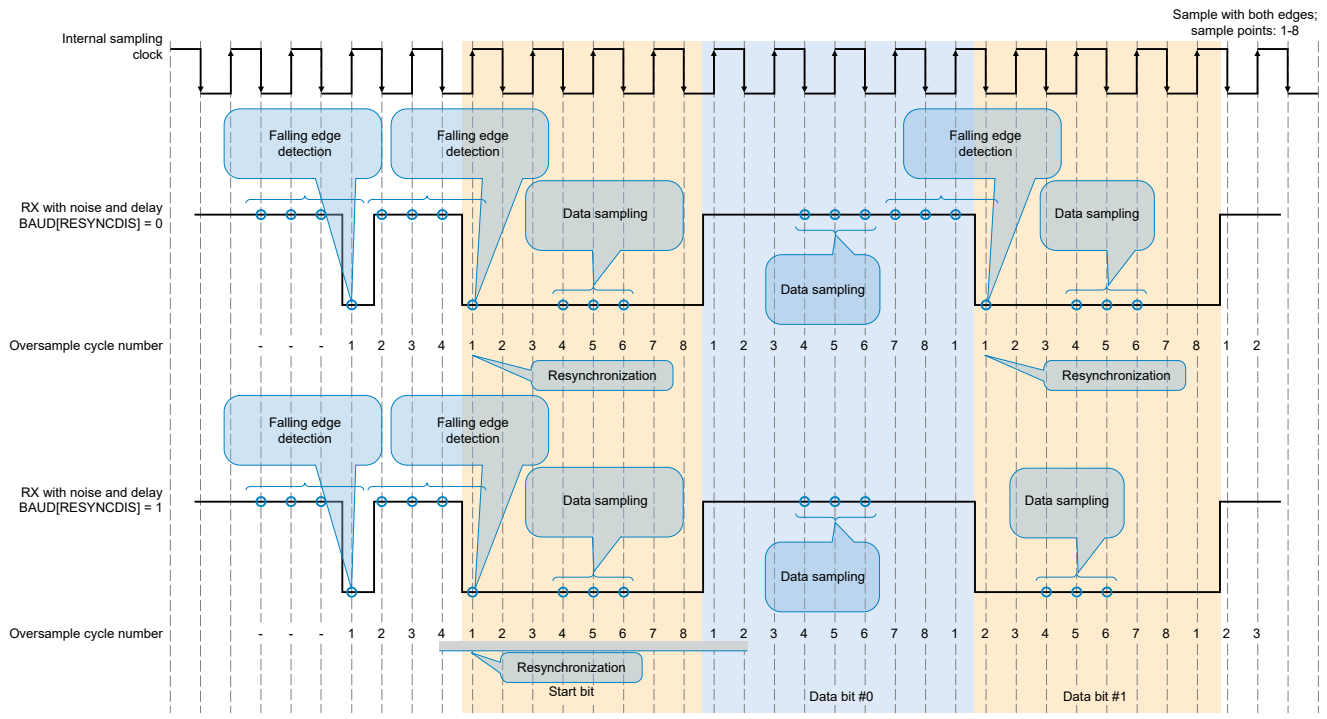


Figure 491. LPUART resynchronization diagram

77.3.5.2 Receiver wake-up operation

Receiver wake-up and receiver address matching are hardware mechanisms that allow an LPUART receiver to ignore the characters in a message intended for a different receiver.

During receiver wake-up, all receivers evaluate the first character(s) of each message, and as soon as they determine the message is intended for a different receiver, they write 1 to CTRL[RWU].

When CTRL[RWU] and STAT[RWUID] are 1, the status fields associated with the receiver, with the exception of STAT[IDLE], are inhibited from becoming 1, thus eliminating the software overhead for handling the unimportant message characters. At the end of a message, all receivers automatically force CTRL[RWU] to become 0. This results in all receivers waking up in time to look at the first character(s) of the next message.

During receiver address matching, the address matching is performed in hardware and the LPUART receiver ignores all characters that do not meet the address match requirements.

Table 689. Receiver wake-up options

CTRL[RWU]	BAUD[MAEN1] | BAUD[MAEN2]	BAUD[MATCFG]	CTRL[WAKE]: STAT[RWUID]	Receiver wake-up
0	0	X	X	Normal operation
1	0	00	00	Receiver wake-up on idle line; STAT[IDLE] = 0
1	0	00	01	Receiver wake-up on idle line; STAT[IDLE] = 1
1	0	00	10	Receiver wake-up on address mark
1	1	11	10	Receiver wake-up on data match
0	1	00	X0	Address mark address match; STAT[IDLE] = 0 for discarded characters
0	1	00	X1	Address mark address match; STAT[IDLE] = 1 for discarded characters
0	1	01	X0	Idle line address match
0	1	10	X0	Match on and match off; STAT[IDLE] = 0 for discarded characters
0	1	10	X1	Match on and match off; STAT[IDLE] = 1 for discarded characters

77.3.5.2.1 Idle line wake-up

When CTRL[WAKE] is 0, you can configure the receiver for an idle line wake-up. In this mode, CTRL[RWU] becomes 0 automatically when the receiver detects a full character time of the idle-line level.

CTRL[M], CTRL[M7], and BAUD[M10] select 7-bit to 10-bit Data mode and BAUD[SBNS] selects a 1-bit or 2-bit stop bit number that determines how many bit times of idle are needed to constitute a full character time, 9 to 13 bit times because of the start and stop bits.

When CTRL[RWU] is 1 and STAT[RWUID] is 0, the idle condition that wakes up the receiver does not lead to STAT[IDLE] becoming 1. The receiver wakes up and waits for the first data character of the next message that leads to STAT[RDRF] becoming 1 and generates an interrupt if enabled. When STAT[RWUID] is 1, any idle condition leads to STAT[IDLE] becoming 1 and generates an interrupt if enabled, regardless of whether CTRL[RWU] is 0 or 1.

These are the ways to detect an idle line:

- When CTRL[ILT] is 0, the idle bit counter starts after the start bit so that the stop bit and any logic 1s at the end of a character count to calculate the full character time of idle.
- When CTRL[ILT] is 1, the idle bit counter does not start until after the stop bit time so that the data in the last character of the previous message does not impact the idle detection.

77.3.5.2.2 Address mark wake-up

When CTRL[WAKE] is 1, you can configure the receiver for an address mark wake-up. In this mode, CTRL[RWU] becomes 0 automatically when the receiver detects a logic 1 in the most significant bit of the received character. When parity is enabled, the second most significant bit is used for address mark wake-up.

Address mark wake-up allows messages to contain idle characters, but requires one bit to be reserved for use in address frames. The logic 1 in the most significant bit (or second most significant bit when parity is enabled) of an address frame writes 0 to CTRL[RWU] and writes 1 to STAT[RDRF]. In this case, the character with the address mark bit is received even if the receiver is sleeping during most of this character time.

77.3.5.2.3 Data match wake-up

When CTRL[RWU] and CTRL[WAKE] are 1, and BAUD[MATCFG] equals 11, the receiver is configured for a data match wake-up. In this mode, CTRL[RWU] becomes 0 automatically when the receiver detects a character that matches MATCH[MA1] when BAUD[MAEN1] is 1, or that matches MATCH[MA2] when BAUD[MAEN2] is 1.

77.3.5.2.4 Address match operation

You can enable the address match operation when either BAUD[MAEN1] or BAUD[MAEN2] is 1 and BAUD[MATCFG] is 0. In this function, a character that the RXD pin receives with a logic 1 in the most significant bit (or the second most significant bit when parity is enabled) is considered an address and is compared to the associated MATCH[MA1] or MATCH[MA2]. The character is only transferred to the receive buffer, and STAT[RDRF] becomes 1 if the comparison matches. All subsequent characters received with a logic 0 in the most significant bit (or the second most significant bit when parity is enabled) are considered to be data associated with the address and are transferred to the receive FIFO. If no marked address match occurs, no transfer is made to the receive FIFO, and all the characters that follow, with logic 0 in the most significant bit (or second most significant bit when parity is enabled), are also discarded. If both BAUD[MAEN1] and BAUD[MAEN2] are 0, the receiver operates normally, and all the received data is transferred to the receive FIFO.

The address match operation functions in the same way for both MATCH[MA1] and MATCH[MA2]:

- If either BAUD[MAEN1] or BAUD[MAEN2] is 1, a marked address is compared only to the associated Match Address (MATCH) and data is transferred to the receive FIFO only on a match.
- If both BAUD[MAEN1] and BAUD[MAEN2] are 1, a marked address is compared to both MATCH[MA1] and MATCH[MA2] and data is transferred only on a match with either of these fields.

77.3.5.2.5 Idle match operation

You can enable the idle match operation when either BAUD[MAEN1] or BAUD[MAEN2] is 1 and BAUD[MATCFG] is 1. In this function, the first character that the RXD pin receives after an idle line condition is considered an address and is compared to the associated MATCH[MA1] or MATCH[MA2]. The character is transferred only to the receive buffer, and STAT[RDRF] becomes 1, if the comparison matches. All subsequent characters are considered to be data associated with the address and are transferred to the receive FIFO until the next idle line condition is detected. If no address match occurs, no transfer is made to the receive FIFO, and all the frames that follow, until the next idle condition, are also discarded. If both BAUD[MAEN1] and BAUD[MAEN2] are 0, the receiver operates normally, and all the received data is transferred to the receive FIFO.

An idle match operation functions in the same way for both MATCH[MA1] and MATCH[MA2]:

- If either BAUD[MAEN1] or BAUD[MAEN2] is 1, the first character after an idle line is compared only to the associated Data (DATA) and data is transferred to the receive FIFO only on a match.

- If both BAUD[MAEN1] and BAUD[MAEN2] are 1, the first character after an idle line is compared to both MATCH[MA1] and MATCH[MA2] and data is transferred only on a match with either of these fields.

77.3.5.2.6 Match on, match off operation

The match on, match off operation is enabled when both BAUD[MAEN1] and BAUD[MAEN2] are 1 and BAUD[MATCFG] = 10. In this function, a character that the RXD pin receives matches MATCH[MA1] and is transferred to the receive buffer, and STAT[RDRF] becomes 1. All subsequent characters are considered to be data and are also transferred to the receive FIFO, until a character that matches MATCH[MA2] is received. The character that matches MATCH[MA2], along with all subsequent characters, is discarded; and this continues until another character that matches MATCH[MA1] is received. If both BAUD[MAEN1] and BAUD[MAEN2] are 0, the receiver operates normally, and all the received data is transferred to the receive FIFO.

NOTE

The match on, match off operation requires both BAUD[MAEN1] and BAUD[MAEN2] to be 1.

77.3.5.3 Hardware flow control

To support hardware flow control, you can program the receiver to automatically assert and deassert RTS_B:

- RTS_B remains asserted until the transfer is complete, even if the transmitter is disabled midway through a data transfer. See Transceiver driver enable using RTS_B for more information.
- If the receiver RTS functionality is enabled, the receiver automatically deasserts RTS_B if STAT[RDRF] is 1 or a start bit is detected that causes STAT[RDRF] to become 1.
- The receiver asserts RTS_B when STAT[RDRF] is 0 and has not detected a start bit that causes STAT[RDRF] to become 1. There is no impact if STAT[RDRF] is 1 already.
- Even if RTS_B is deasserted, the receiver continues to receive characters until the receive FIFO is overrun.
- If the receiver RTS functionality is disabled, the receiver's RTS_B remains deasserted.
- When RTS is driven by receiver, it will be negated when the receiver is disabled irrespective of the FIFO level.

77.3.6 Additional LPUART functions

77.3.6.1 Data modes

You can configure the LPUART transmitter and receiver to operate in 7-bit Data mode by writing 1 to CTRL[M7], 9-bit Data mode by writing 1 to CTRL[M], or 10-bit Data mode by writing 1 to BAUD[M10]. In 9-bit Data mode, there exists a ninth data bit and in 10-bit mode, there exists a tenth data bit.

When performing 8-bit writes to the transmit FIFO, the ninth and tenth bits are pushed into the FIFO from CTRL[T8] and CTRL[T9]. For coherent 8-bit writes, you must write to CTRL[T8] and CTRL[T9] before writing to Data (DATA)[7:0]. However, if the values in CTRL[T8] or CTRL[T9] do not need to change, it is not necessary to update CTRL[T8] and CTRL[T9] before every 8-bit write to Data (DATA).

When performing 16-bit or 32-bit writes to the transmit FIFO, all 10 bits are pushed into the transmit FIFO from the write data.

When performing 8-bit reads of the receive FIFO, the ninth and tenth bits are held in CTRL[R8] and CTRL[R9] but you must read them before reading Data (DATA). A 16-bit or 32-bit read of the receive FIFO returns all 10 bits in Data (DATA).

The 9-bit Data mode is typically used with parity to allow eight bits of data plus the parity in the ninth bit, or it is used with the address mark wake-up so that the ninth data bit can serve as the wake-up bit. The 10-bit Data mode is typically used with parity and address mark wake-up so that the ninth data bit can serve as the wake-up bit and the tenth bit can serve as the parity bit. In custom protocols, the ninth and/or tenth bits can also serve as software-controlled markers.

77.3.6.2 Idle length

An idle character is one where the start bit, all data bits, and stop bits are in the mark position (idle state, generally logic 1). You can configure CTRL[ILT] to start detecting an idle character from the previous start bit (any data bits and stop bits count for idle character detection) or from the previous stop bit.

You can also use CTRL[IDLECFG] to configure the number of idle characters that must be received before an idle line condition is detected. This field configures the number of idle characters that must be received before STAT[IDLE] becomes 1, STAT[RAF] becomes 0, and DATA[IDLINE] becomes 1 with the next received character.

CTRL[IDLECFG] also affects the idle line wake-up and idle match operations. When either the address match or match on/off operation is enabled, writing 1 to STAT[RWUID] causes any discarded characters to be treated as idle characters.

After the extended idle time is enabled for the receiver, you can configure an idle line condition by using REIR[IDTIME], which specifies the number of bits (baud rate) since the last stop bit that is required for an idle condition to be detected. This replaces the configuration of CTRL[ILT] and CTRL[IDLECFG].

The transmitter can also enable the extended idle time. In this case, any idle character queued through the transmit FIFO forces the transmitter to be idled for the configured number of bit clocks before the idle character is read from the FIFO and the transmitter continues.

After you enable the transmitter extended idle time, the transmitter does not automatically queue an idle character whenever it is enabled.

77.3.6.3 Loop mode

Enable Loop mode by setting CTRL[LOOPS] = 1 and CTRL[RSRC] = 0. You, sometimes, use Loop mode to check software, independent of connections in the external system, to help isolate system problems. In this mode, the transmitter output is internally connected to the receiver input and LPUART does not use the RXD pin.

Loop mode also internally connects the RTS_B output to the CTS_B input and the DTR_B output to the DSR_B input.

77.3.6.4 Single-Wire mode

Enable Single-Wire mode by setting CTRL[LOOPS] = 1 and CTRL[RSRC] = 1. Single-Wire mode implements a half-duplex serial connection. The receiver is internally connected to the transmitter output and TXD pin (the RXD pin is not used).

In Single-Wire mode, CTRL[TXDIR] controls the direction of serial data on the TXD pin. When CTRL[TXDIR] becomes 0, the TXD pin is an input to the receiver and the transmitter is temporarily disconnected from the TXD pin so that an external device can send serial data to the receiver. When CTRL[TXDIR] = 1, the TXD pin is an output that the transmitter drives. The internal loop back connection is disabled, and as a result, the receiver is unable to receive characters that the transmitter sends out.

Half Duplex Control (HDCR) replaces the implementation of CTRL[LOOPS] and CTRL[RSRC], and you can use this register to configure various options for both Single-Wire and Half-Duplex operations, using independent RXD and TXD pins:

- HDCR[TXSTALL] replaces the CTRL[TXDIR] functionality and prevents the transmitter from becoming busy or asserting the RTS_B transmitter if STAT[RAF] is 1.
- You can select the TXD pin, as the source for the receiver, to configure HDCR[RXSEL] for a single-wire operation. If HDCR[RXSEL] is 1, you must configure the TXD pin for an open-drain operation.
- HDCR[RXMSK] masks the receiver input when the RTS_B transmitter is asserted (this applies even if you have not configured RTS_B as an output).
- HDCR[RXWRMSK] blocks storage of the receive data in the receive FIFO when the RTS_B transmitter is asserted. This setting does not affect the receiver idle functionality.
- HDCR[RTSEXT] delays the negation of the RTS_B transmitter by the configured number of bit clocks (baud rate).

77.3.6.5 Timeout counter

LPUART implements four general-purpose timeout counters; counters 0 and 1 are used to monitor the receiver and counters 2 and 3 are used to monitor the transmitter. When enabled, you can configure each counter to monitor one of the following conditions:

- Idle time in number of bits, starting to increment when first enabled and an idle condition is detected. The counter restarts whenever a character is received or transmitted.
- Idle time in number of bits, starting to increment after the next character is received or transmitted. The counter restarts whenever a character is received or transmitted.
- Idle time is more than the timeout interval, in number of bits, but less than the extended idle timeout. The counter restarts whenever a character is received or transmitted. You can use this to detect a gap between characters that is greater than a threshold (timeout) but less than the configured extended idle time. This configuration requires the extended idle feature for the transmitter or receiver to be enabled for a proper operation.
- Number of characters received or transmitted is equal to the configured timeout. The counter asserts at the start of the character that equals the timeout value.

The timeout counters restart counting whenever the counter is disabled and then enabled. These timeout counters are disabled when the corresponding STAT[TSF] field is 1. If the timeout enable is still set after STAT[TSF] becomes 0, the timeout counter restarts as if the counter had been disabled and then enabled. You can measure the idle time in bit times from the end of the last stop bit and until a start bit is validated.

77.3.7 Peripheral triggers

The connection of the LPUART peripheral triggers with other peripherals is chip-specific.

77.3.7.1 Output triggers

LPUART generates the following output triggers that can be connected to other peripherals on the chip:

- The transmit word trigger asserts at the end of each transmitted word and negates after 1-bit period.
- The transmit data trigger is identical to the TXD pin output, but without support for input trigger modulation.
- The receive word trigger asserts at the end of each received word that is written to the receive FIFO, for one oversampling clock period.
- The receive idle trigger asserts when STAT[IDLE] becomes 1, and negates when the next valid start bit is detected.

77.3.7.2 Input trigger

LPUART supports a peripheral input trigger that you can configure in one of the following ways:

- By enabling the CTS function: You can connect the input trigger instead of the CTS_B pin input. The input trigger must assert for longer than 1-bit clock period when the transmitter is idle, with data to send, to guarantee a new transmission.
- By making the input trigger modulate the transmit data output (trigger is logically ANDed with the TXD output): The input trigger is expected to be a free-running clock (carrier signal) that generates from a timer or PWM source with a frequency that is greater than the bit-clock frequency. The carrier signal must not toggle faster than the maximum supported bit time.
- By connecting the input trigger instead of the RXD pin input: The input trigger is expected to be generated from a receive data source, such as an analog comparator or external pin.

77.3.8 Infrared (IR) interface

LPUART provides the capability of transmitting narrow pulses to an IR LED and receiving narrow pulses, transforming them to serial bits, which are then sent to LPUART. The IrDA physical layer specification defines a half-duplex IR communication link for exchanging data. The full standard includes data rates up to 16 Mbit/s. The LPUART IrDA support is limited to SIR mode that supports data rates only between 2.4 kbit/s and 115.2 kbit/s.

LPUART has an infrared transmit encoder and a receive decoder. The infrared decoder converts the received character from the IrDA format to the NRZ format, which the receiver uses. It also has an OSR oversampling baud rate clock counter that filters noise and indicates when a 1 is received. LPUART transmits serial bits of data, which the infrared submodule encodes, to transmit a narrow pulse for every zero bit. No pulse is transmitted for every single bit. When receiving data, an IR photo diode (external to LPUART) detects the IR pulses. The IR receive decoder transforms them to CMOS levels. The infrared receive decoder then stretches the narrow pulses to get back to a serial bit stream that LPUART receives. You can invert the polarity of transmitted

pulses and expected receive pulses so that a direct connection can be made to external IrDA transceiver modules that use active-high pulses.

The IR submodule receives its clock sources from LPUART. The submodule selects one of these clocks to generate either 1 ÷ OSR, 2 ÷ OSR, 3 ÷ OSR, or 4 ÷ OSR narrow pulses during transmission.

77.3.8.1 Infrared transmit encoder

The infrared transmit encoder converts serial bits of data from the transmit shift register to the TXD signal. A narrow pulse is transmitted for a 0 bit and no pulse is transmitted for a 1 bit. The narrow pulse is sent at the start of the bit with a duration of 1 ÷ OSR, 2 ÷ OSR, 3 ÷ OSR, or 4 ÷ OSR of a bit time. A narrow low pulse is transmitted for a 0 bit when CTRL[TXINV] is 0, while a narrow high pulse is transmitted for a 0 bit when CTRL[TXINV] is 1.

77.3.8.2 Infrared receive decoder

The infrared receive block converts data from the RXD signal to the receive shift register. A narrow pulse is expected for each 0 received and no pulse is expected for each 1 received. A narrow low pulse is expected for a 0 bit when STAT[RXINV] is 0, while a narrow high pulse is expected for a 0 bit when STAT[RXINV] is 1. This receive decoder meets the edge jitter requirement as defined by the IrDA serial infrared physical layer specification.

77.3.8.3 Start-bit detection

When STAT[RXINV] is 0, the first falling edge of the received character corresponds to the start bit. The infrared decoder resets its counter. At this time, the receiver also begins its start bit detection process. After the start bit is detected, the receiver synchronizes its bit times to this start bit time. For the rest of the character reception, the infrared decoder's counter and the receiver's bit time counter count independently of each other.

77.3.8.4 Noise filtering

The decoder ignores any rising edges detected during the first half of the infrared decoder counter, and can leave any pulses less than one oversampling baud clock as undetected. This is regardless of whether the pulse is seen in the first or second half of the count.

77.3.8.5 Low-bit detection

During the second half of the decoder count, a rising edge is decoded as 0, which is sent to the receiver. The decoder counter is also reset.

77.3.8.6 High-bit detection

At OSR oversampling baud rate clocks after the previous rising edge, if a rising edge is not seen, the decoder sends a 1 to the receiver.

If the next bit is 0, which arrives late, a low bit is detected according to Low-bit detection. The value sent to the receiver is changed from 1 to 0. Then, if a noise pulse occurs outside the receiver's bit time sampling period, the delay of a 0 is not recorded as noise.

77.3.9 MODBUS protocol

77.3.9.1 MODBUS frame structure

MODBUS is an application layer messaging protocol, positioned at level 7 of the OSI model. It provides client and server communication between devices connected on different types of buses or networks.

The MODBUS protocol defines a simple protocol data unit (PDU) independent of the underlying communication layers. The mapping of the MODBUS protocol on specific buses or network can introduce some additional fields on the application data unit (ADU).

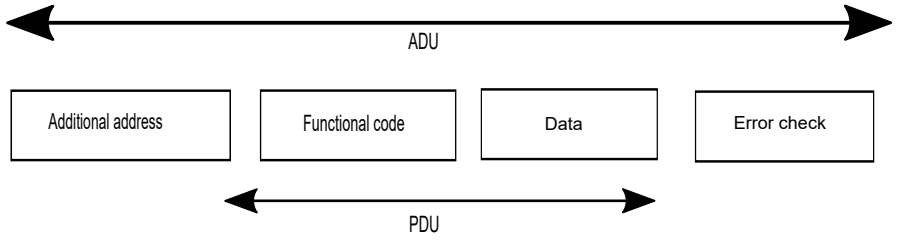


Figure 492. MODBUS frame structure

The function code field informs the server about which action to perform, as requested by the client (only the client can initiate the communication), where the data field contains additional information that the server uses to take the action defined by the function code. MODBUS PDU for serial line communication = 256 - server address (1 byte) - CRC (2 bytes) = 253 bytes.

You can set up a MODBUS controller to communicate on standard MODBUS networks using either of the two transmission modes: ASCII or RTU. RTU mode allows better character density and throughput for the same baud rate as the ASCII format allows 7 bits to represent a character where RTU keeps 8 bits for data representation in each packet.

77.3.9.2 MODBUS frame for LPUART

Though the protocol resides in the application layer, it can also be checked at the physical layer using LPUART by considering the frame structure shown in the following figure, following RTU mode of data transmission.

Start	Address	Function	Data	CRC	End
3.5 Char time	8 Bit	8 Bit	N * 8Bit	16 Bit	3.5 Char time

Figure 493. MODBUS frame for LPUART

An entire message frame of MODBUS must contain start character, address, function code, data, and end character. LPUART does not support CRC calculation for TX and RX, instead, each LPUART packet containing frame information is transmitted with its individual parity, and the same is checked in RX. Following the last transmitted character, a similar interval of at least 3.5 character times marks the end of the message and a new message can begin after this interval. The entire message frame must be transmitted as a continuous stream. If a silent interval of more than 1.5 character times occurs before the completion of the frame, the receiving device flushes the incomplete message, and assumes that the next byte is the address field of a new message. The correctness of timeout logic ensures the correctness of the MODBUS frame using LPUART.

Registers such as REIR, TEIR, TOCR, TOSR, TIMEOUT[0 - 3], TCBR, and TDBR are useful for realizing MODBUS using LPUART. The TX IDLE time is configured by using TEIR[IDTIME].

For higher baud rates, MODBUS recommends adopting fixed t1.5 and t3.5 times of 750us ms and 1.75 ms. The receiver idle line wake-up can be used by a MODBUS device to only wake-up on a matching address or broadcast address received after a valid idle time. The receiver end-of-packet DMA transfer can be used to offload the reception of a MODBUS packet onto the DMA controller.

77.3.9.3 MODBUS TX and RX programming sequence

The following programming sequences can be useful for sending and receiving MODBUS frames using LPUART.

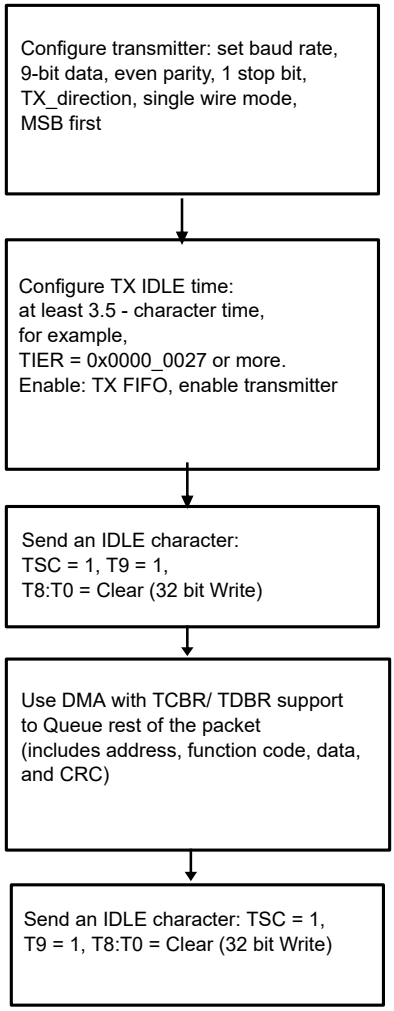


Figure 494. MODBUS TX programming

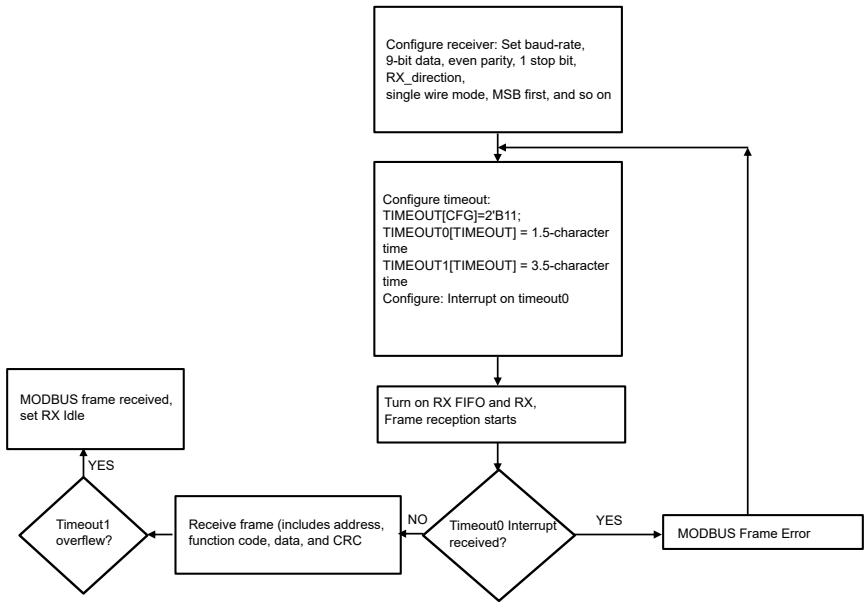


Figure 495. MODBUS RX programming

77.3.10 Modes of operation

77.3.10.1 Low-Power modes

NOTE

See the chip-specific information for specific low-power modes available on your chip.

77.3.10.2 Debug mode

LPUART remains functional in Debug mode.

77.3.11 Clocking

Table 690. Types of clocks

Clock	Description
Functional	Is asynchronous to the bus clock and can remain enabled in Low-Power modes to support transmit and/or receive functions, including low-power wake-up.
Bus	Is only used for bus accesses to the control and configuration registers. The bus clock frequency must be sufficient to support the data bandwidth requirements of the LPUART transmit and receive registers, including the FIFOs.

77.3.12 Reset

Table 691. Types of resets

Reset	Description
Chip	Enables the logic and registers for the LPUART transmitter and receiver to reset to their default states.

Table continues on the next page...

Table 691. Types of resets (continued)

Reset	Description
Software	Resets the LPUART logic and registers to their default states, except for Global (GLOBAL). GLOBAL[RST] controls the LPUART software reset.
FIFO	Implements write-only control fields that reset the transmit FIFO (FIFO[TXFLUSH]) and receive FIFO (FIFO[RXFLUSH]). After a FIFO is reset, that FIFO becomes empty.

77.3.13 Interrupts

The LPUART transmitter has two status fields that can optionally generate hardware interrupt requests. If STAT[TDRE] is 1, it indicates that there is room in the transmit FIFO to write another transmit character to Data (DATA). If CTRL[TIE] is 1, a hardware interrupt is requested when STAT[TDRE] is 1.

STAT[TC] indicates that the transmitter is finished transmitting all data, preamble, and break characters and is idle with TXD at the inactive level. This field is often used in systems with modems to determine when it is safe to turn off the modem. If CTRL[TCIE] is 1, a hardware interrupt is requested when STAT[TC] is 1. Instead of hardware interrupts, software polling may be used to monitor STAT[TDRE] and STAT[TC] if the corresponding CTRL[TIE] or CTRL[TCIE] field is 0.

When a program detects that STAT[RDRF] is 1, it gets the data from this field by reading Data (DATA). The field becomes 0 by reading Data (DATA).

STAT[IDLE] includes logic that prevents it from becoming 1 repeatedly when the RXD line remains idle for an extended period of time. STAT[IDLE] becomes 0 when you write 1 to it, and cannot become 1 again until the receiver has received at least one new character and has 1 as the value of STAT[RDRF].

If the associated error is detected in the received character that caused STAT[RDRF] to become 1, STAT[NF], STAT[FE], and STAT[PF] become 1 at the same time STAT[RDRF] becomes 1. These flags do not become 1 in overrun cases.

If STAT[RDRF] is already 1 when a new character is ready to be transferred from the receive shifter to the receive FIFO, STAT[OR] becomes 1, instead of the data along with any associated STAT[NF], STAT[FE], or STAT[PF] condition getting lost.

If the received character matches the contents of MATCH[MA1] and/or MATCH[MA2], then STAT[MA1F] and/or STAT[MA2F] become 1 at the same time that STAT[RDRF] becomes 1.

At any time, an active edge on the RXD serial data input pin causes STAT[RXEDGIF] to become 1. STAT[RXEDGIF] becomes 0 when you write 1 to it. This function depends on the receiver being enabled (the value of CTRL[RE] being 1).

MODEM Status (MSR) can generate an interrupt from a configured status field, which STAT[MSF] indicates.

Timeout Status (TOSR) can generate an interrupt from a configured status field, which STAT[TSF] indicates.

77.3.14 DMA

77.3.14.1 DMA burst support

To support efficient DMA transfers to the transmit FIFO, two alias regions are implemented to support incrementing 8-bit, 16-bit, or 32-bit write accesses to the transmit FIFO:

- Transmit Command Burst (TCBR0 - TCBR127) is a 512-byte region that supports pushing 16-bit data into the transmit FIFO.
- Transmit Data Burst (TDBR0 - TDBR255) is a 1024-byte region that supports pushing zero extended 8-bit data into the transmit FIFO.

The aforementioned regions are contiguous, so a DMA transfer can start in Transmit Command Burst (TCBR0 - TCBR127) to initialize the transfer including address mark, idle word, or break character, and then complete the transfer in Transmit Data Burst (TDBR0 - TDBR255) with the data to transmit, without changing the transfer size.

The transmit FIFO block writes overflow the FIFO, but that does not signal an error. Do not perform 32-bit writes to Transmit Data Burst (TDBR0 - TDBR255) unless there are four empty slots in the transmit FIFO, and do not perform 16-bit writes to this register unless there are two empty slots in the transmit FIFO.

77.4 External signals

Table 692. External signals

Signal	Description	I/O
TXD	Transmit data: This pin is normally an output, but is an input (tristated) in Single-Wire mode whenever the transmitter is disabled or the transmit direction is configured for receive data.	I/O
RXD	Receive data	I
CTS_B	Clear-to-send	I
RTS_B	Request-to-send	O
DTR_B	Data terminal ready	O
DSR_B	Data set ready	I
DCD_B	Data carrier detect	I
RIN_B	Ring indicator	I

77.5 Initialization

This module does not require initialization.

77.6 LPUART register descriptions

LPUART includes registers to control baud rate, select options, report status, and store transmit and receive data. Access to an address outside the valid memory map generates a bus error.

NOTE

Writing to a read-only (RO) register or reading a write-only (WO) register can cause bus errors. LPUART does not verify whether programmed values in the registers are correct; you must write valid values to them.

77.6.1 LPUART memory map

LPUART_0 base address: 4032_8000h

LPUART_1 base address: 4032_C000h

LPUART_2 base address: 4033_0000h

LPUART_3 base address: 4033_4000h

LPUART_4 base address: 4033_8000h

LPUART_5 base address: 4033_C000h

LPUART_6 base address: 4034_0000h

LPUART_7 base address: 4034_4000h

LPUART_8 base address: 4048_C000h

LPUART_9 base address: 4049_0000h

LPUART_10 base address: 4049_4000h

LPUART_11 base address: 4049_8000h

LPUART_12 base address: 4049_C000h

LPUART_13 base address: 404A_0000h

LPUART_14 base address: 404A_4000h

LPUART_15 base address: 404A_8000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID (VERID)	32	R	See section
4h	Parameter (PARAM)	32	R	See section
8h	Global (GLOBAL)	32	RW	0000_0000h
Ch	Pin Configuration (PINCFG)	32	RW	0000_0000h
10h	Baud Rate (BAUD)	32	RW	0F00_0004h
14h	Status (STAT)	32	RW	00C0_0000h
18h	Control (CTRL)	32	RW	0000_0000h
1Ch	Data (DATA)	32	RW	0000_1000h
20h	Match Address (MATCH)	32	RW	0000_0000h
24h	MODEM IrDA (MODIR)	32	RW	0000_0000h
28h	FIFO (FIFO)	32	RW	See section
2Ch	Watermark (WATER)	32	RW	0000_0000h
30h	Data Read-Only (DATARO)	32	R	0000_1000h
40h	MODEM Control (MCR)	32	RW	0000_0000h
44h	MODEM Status (MSR)	32	RW	0000_0000h
48h	Receiver Extended Idle (REIR)	32	RW	0000_0000h
4Ch	Transmitter Extended Idle (TEIR)	32	RW	0000_0000h
50h	Half Duplex Control (HDCR)	32	RW	0000_0000h
58h	Timeout Control (TOCR)	32	RW	0000_0000h
5Ch	Timeout Status (TOSR)	32	RW	0000_000Fh
60h - 6Ch	Timeout N (TIMEOUT0 - TIMEOUT3)	32	RW	0000_0000h
200h - 3FCh	Transmit Command Burst (TCBR0 - TCBR127)	32	W	0000_0000h
400h - 7FCh	Transmit Data Burst (TDBR0 - TDBR255)	32	W	0000_0000h

77.6.2 Version ID (VERID)

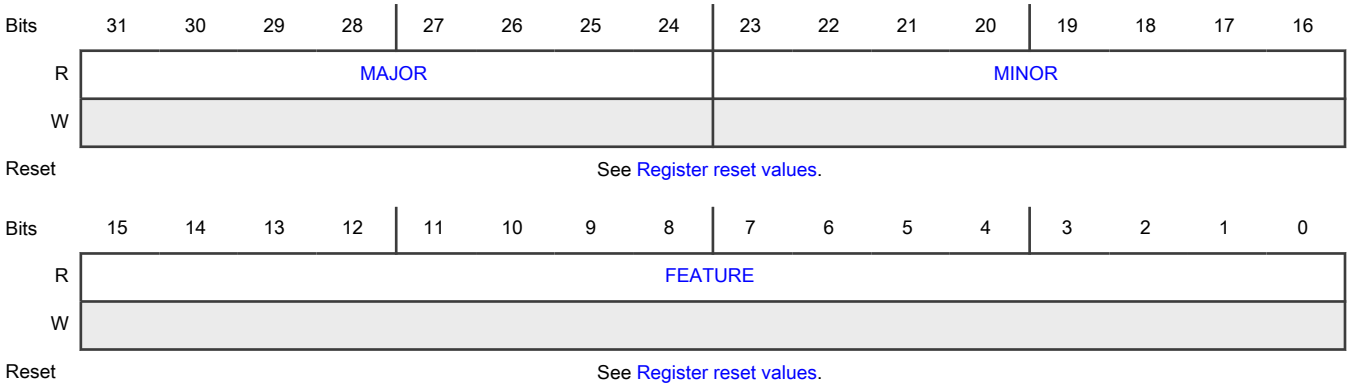
Offset

Register	Offset
VERID	0h

Function

Indicates the version integrated for this instance on the chip and also specifies the inclusion and exclusion of several optional features.

Diagram



Register reset values

Register	Reset value
VERID	LPUART_0,LPUART_1: 0404_0007h LPUART_2–LPUART_15: 0404_0003h

Fields

Field	Function
31-24 MAJOR	Major Version Number Indicates the major version number for the module specification.
23-16 MINOR	Minor Version Number Indicates the minor version number for the module specification.
15-0 FEATURE	Feature Identification Number Indicates the feature set number.

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
NOTE
The descriptions of the field settings vary by module instance.
<table>
<tr><th>Instance</th><th>Field value and description</th></tr>
<tr><td>LPUART_0
LPUART_1</td><td>0000_0000_0000_0001b - Standard feature set
0000_0000_0000_0011b - Standard feature set with MODEM and IrDA support
0000_0000_0000_0111b - Enhanced feature set with full MODEM, IrDA, and enhanced idle detection</td></tr>
<tr><td>LPUART_2
LPUART_3
LPUART_4
LPUART_5
LPUART_6
LPUART_7
LPUART_8
LPUART_9
LPUART_10
LPUART_11
LPUART_12
LPUART_13
LPUART_14
LPUART_15</td><td>0000_0000_0000_0001b - Standard feature set
0000_0000_0000_0011b - Standard feature set with MODEM and IrDA support</td></tr>
</table>
</td></tr>
</table>

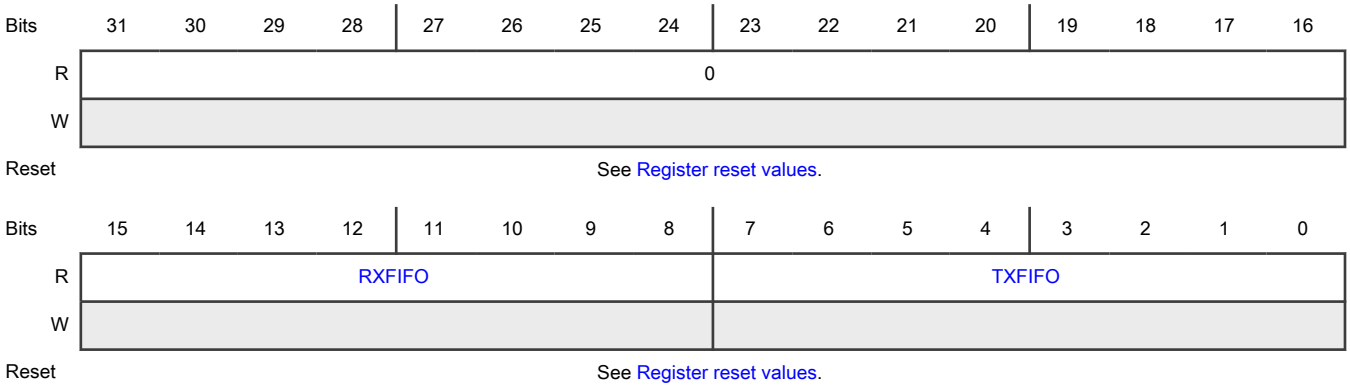
77.6.3 Parameter (PARAM)

Offset

Register	Offset
PARAM	4h

Function
Indicates the parameter configuration for this instance on the chip.

Diagram



Register reset values

Register	Reset value
PARAM	LPUART_0,LPUART_1: 0000_0404h LPUART_2–LPUART_15: 0000_0202h

Fields

Field	Function
31-16 —	Reserved
15-8 RXFIFO	Receive FIFO Size Indicates the number of characters in the receive FIFO, which is 2^RXFIFO.
7-0 TXFIFO	Transmit FIFO Size Indicates the number of characters in the transmit FIFO, which is 2^TXFIFO.

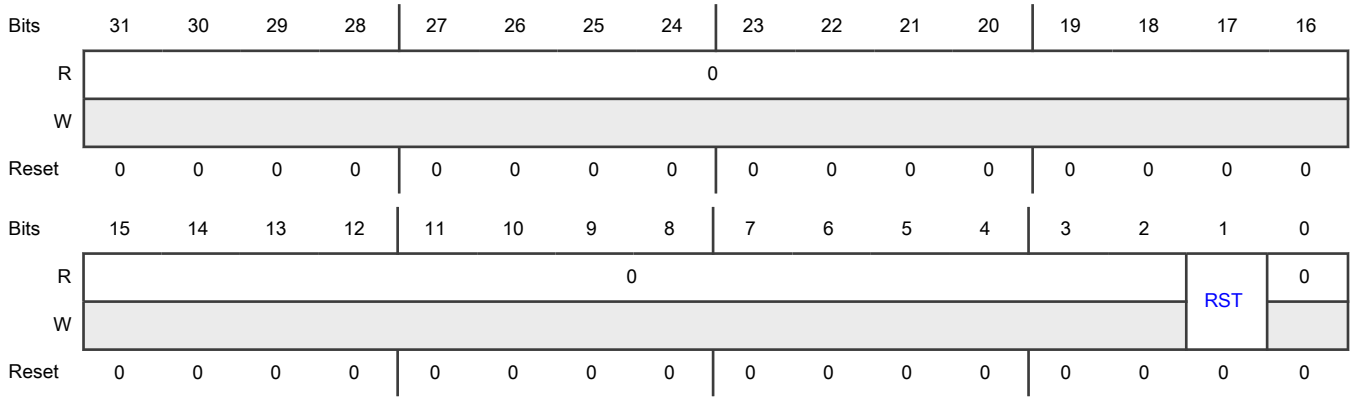
77.6.4 Global (GLOBAL)

Offset

Register	Offset
GLOBAL	8h

Function
Performs global functions.

Diagram



Fields

Field	Function
31-2 —	Reserved
1 RST	Software Reset Specifies whether the module is reset. This field resets all internal logic and registers, except Global (GLOBAL). The reset takes effect immediately and remains asserted until you negate it. There is no minimum delay required before clearing the software reset. 0b - Not reset 1b - Reset
0 —	Reserved

77.6.5 Pin Configuration (PINCFG)

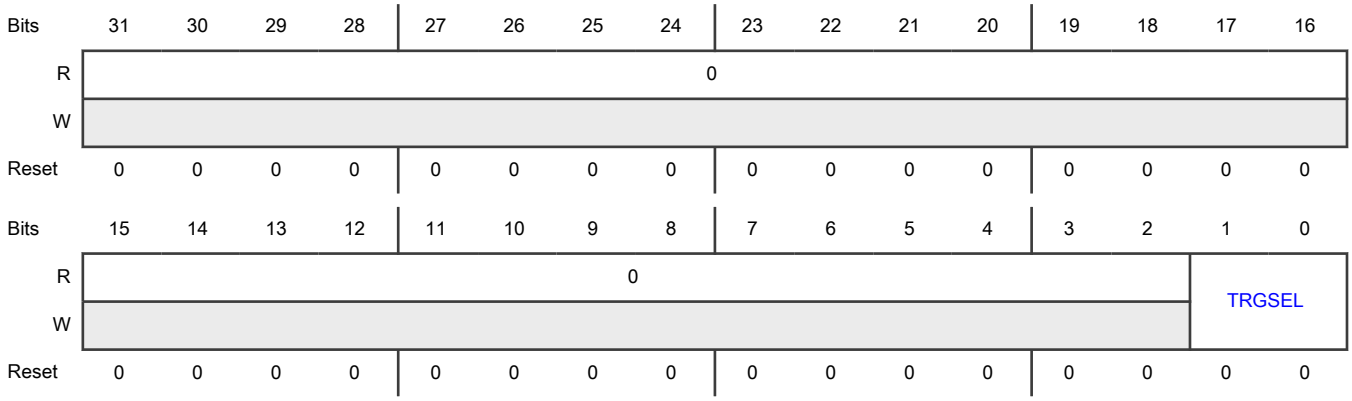
Offset

Register	Offset
PINCFG	Ch

Function

Enables the selection of input pins.

Diagram



Fields

Field	Function
31-2 —	Reserved
1-0 TRGSEL	Trigger Select Configures the input trigger usage. You must change the value of this field only when both the transmitter and receiver are disabled. 00b - Input trigger disabled 01b - Input trigger used instead of the RXD pin input 10b - Input trigger used instead of the CTS_B pin input 11b - Input trigger used to modulate the TXD pin output, which (after TXINV configuration) is internally ANDed with the input trigger

77.6.6 Baud Rate (BAUD)

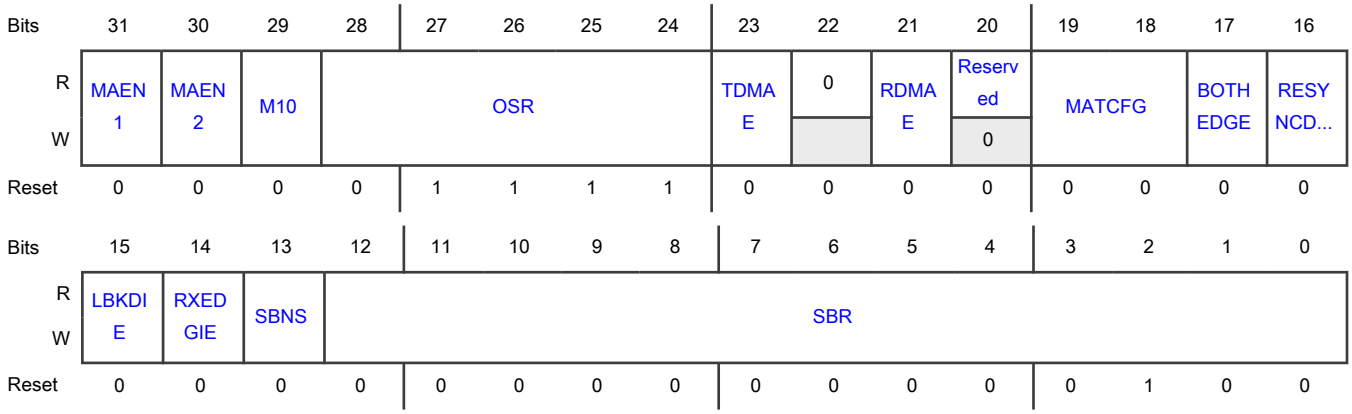
Offset

Register	Offset
BAUD	10h

Function

Configures the baud rate.

Diagram



Fields

Field	Function
31 MAEN1	Match Address Mode Enable 1 Enables automatic address matching or data matching mode for MATCH[MA1]. If this field is 0, normal operation takes place. 0b - Disable 1b - Enable
30 MAEN2	Match Address Mode Enable 2 Enables automatic address matching or data matching mode for MATCH[MA2]. If this field is 0, normal operation takes place. 0b - Disable 1b - Enable
29 M10	10-Bit Mode Select Causes the tenth bit to be a part of the serial transmission. You must change the value of this field only when both the transmitter and receiver are disabled. 0b - Receiver and transmitter use 7-bit to 9-bit data characters 1b - Receiver and transmitter use 10-bit data characters
28-24 OSR	Oversampling Ratio Configures the OSR of the receiver. You must change the value of this field only when both the transmitter and receiver are disabled. **NOTE** BAUD[OSR] results in an OSR of BAUD[OSR] + 1, for example, BAUD[OSR] = 0_0101b results in a final division by 6. 0_0000b - Results in an OSR of 16

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0_0001b - Reserved 0_0010b - Reserved 0_0011b - Results in an OSR of 4 (requires BAUD[BOTHEDGE] to be 1) 0_0100b - Results in an OSR of 5 (requires BAUD[BOTHEDGE] to be 1) 0_0101b - Results in an OSR of 6 (requires BAUD[BOTHEDGE] to be 1) 0_0110b - Results in an OSR of 7 (requires BAUD[BOTHEDGE] to be 1) 0_0111b - Results in an OSR of 8 0_1000b - Results in an OSR of 9 0_1001b - Results in an OSR of 10 0_1010b - Results in an OSR of 11 0_1011b - Results in an OSR of 12 0_1100b - Results in an OSR of 13 0_1101b - Results in an OSR of 14 0_1110b - Results in an OSR of 15 0_1111b - Results in an OSR of 16 1_0000b - Results in an OSR of 17 1_0001b - Results in an OSR of 18 1_0010b - Results in an OSR of 19 1_0011b - Results in an OSR of 20 1_0100b - Results in an OSR of 21 1_0101b - Results in an OSR of 22 1_0110b - Results in an OSR of 23 1_0111b - Results in an OSR of 24 1_1000b - Results in an OSR of 25 1_1001b - Results in an OSR of 26 1_1010b - Results in an OSR of 27 1_1011b - Results in an OSR of 28 1_1100b - Results in an OSR of 29 1_1101b - Results in an OSR of 30 1_1110b - Results in an OSR of 31 1_1111b - Results in an OSR of 32
23 TDMAE	Transmitter DMA Enable Enables STAT[TDRE] to generate a DMA request.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disable 1b - Enable
22 —	Reserved
21 RDMAE	Receiver Full DMA Enable Enables STAT[RDRF] to generate a DMA request. 0b - Disable 1b - Enable
20 —	Reserved
19-18 MATCFG	Match Configuration Configures the match addressing mode used. You must change the value of this field only when both the transmitter and receiver are disabled. 00b - Address match wake-up 01b - Idle match wake-up 10b - Match on and match off 11b - Enables RWU on data match and match on or off for the transmitter CTS input
17 BOTHEDGE	Both Edge Sampling Enables sampling of the received data on both edges of the baud rate clock, effectively doubling the number of times the receiver samples the input data for a given OSR. This field must be 1 for OSRs between x4 and x7 and is optional for higher OSRs. You must change the value of this field only when the receiver is disabled. If this field is 0, the receiver samples input data using the rising edge of the baud rate clock. If this field is 1, the receiver samples input data using the rising and falling edges of the baud rate clock. 0b - Rising edge 1b - Both rising and falling edges
16 RESYNCDIS	Resynchronization Disable Disables resynchronization of the received data word when a data one followed by data zero transition is detected. You must change the value of this field only when the receiver is disabled. 0b - Enable 1b - Disable
15	LIN Break Detect Interrupt Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
LBKDIE	Enables STAT[LBKDIF] to generate hardware interrupt requests. If this field is 0, hardware interrupts from STAT[LBKDIF] (uses polling) are disabled. If this field is 1, hardware interrupts are requested when STAT[LBKDIF] is 1. 0b - Disable 1b - Enable
14 RXEDGIE	RX Input Active Edge Interrupt Enable Enables STAT[RXEDGIF] to generate interrupt requests. If this field is 0, hardware interrupts from STAT[RXEDGIF] are disabled. If this field is 1, hardware interrupts are requested when STAT[RXEDGIF] is 1. Changing the value of CTRL[LOOPS] or CTRL[RSRC] when this field (RXEDGIE) is 1 can cause STAT[RXEDGIF] to become 1. 0b - Disable 1b - Enable
13 SBNS	Stop Bit Number Select Determines whether data characters include one or two stop bits. You must change the value of this field only when both the transmitter and receiver are disabled. 0b - One stop bit 1b - Two stop bits
12-0 SBR	Baud Rate Modulo Divisor Sets the modulo divide rate for the baud rate generator. • If SBR is 0, baud rate generator is disabled. • If SBR is 1–8191, baud rate = baud clock ÷ ((OSR + 1) × SBR). You must update the 13-bit baud rate setting [SBR12:SBR0] only when both the transmitter and receiver are disabled (both CTRL[RE] and CTRL[TE] are 0).

77.6.7 Status (STAT)

Offset

Register	Offset
STAT	14h

Function
Provides the module status.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LBKDIF	RXEDGIF	MSBF	RXINV	RWUID	BRK13	LBKDE	RAF	TDRE	TC	RDRF	IDLE	OR	NF	FE	PF
W	W1C	W1C										W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MA1F	MA2F	0				TSF	MSF	0						AME	LBKFE
W	W1C	W1C														
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 LBKDIF	LIN Break Detect Interrupt Flag Indicates whether a LIN break character is detected. This field becomes 1 when the LIN break detect circuitry is enabled and a LIN break character is detected. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not detected 1b - Detected When writing 0b - No effect 1b - Clear the flag
30 RXEDGIF	RXD Pin Active Edge Interrupt Flag Indicates whether an active edge on the receive pin has occurred. This field becomes 1 whenever the receiver is enabled and an active edge (falling if STAT[RXINV] is 0; rising if STAT[RXINV] is 1) on the RXD pin occurs. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not occurred 1b - Occurred When writing 0b - No effect

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Clear the flag
29 MSBF	MSB First Specifies the first bit that is transmitted after the start bit. If this field is 0, LSB (bit 0) is the first bit transmitted after the start bit (which means, the first bit received after the start bit is identified as bit 0). If this field is 1, MSB (identified as bit 9, bit 8, bit 7, or bit 6) is the first bit that is transmitted, after the start bit, depending on the settings of CTRL[M], CTRL[PE], and BAUD[M10]. Writing 1 to this field reverses the order of the bits that are transmitted and received on the wire. This field does not affect the polarity of the bits, the location of the parity bit, or the location of the start or stop bits. You must change the value of this field only when both the transmitter and receiver are disabled. 0b - LSB 1b - MSB
28 RXINV	Receive Data Inversion Specifies whether receive data is inverted. Writing 1 to this field reverses the polarity of the received data input. You must change the value of this field only when the receiver is disabled. **NOTE** Writing 1 to this field inverts the RXD input for all cases: data bits, start and stop bits, break, and idle. 0b - Inverted 1b - Not inverted
27 RWUID	Receive Wake Up Idle Detect Controls, for CTRL[RWU] on idle character detection, whether the idle character that wakes up the receiver writes 1 to STAT[IDLE]. For address match wake-up, this field controls whether STAT[IDLE] = 1 when the address does not match. You must change the value of this field only when the receiver is disabled. If this field is 0, during the Receive Standby state (CTRL[RWU] = 1), STAT[IDLE] does not become 1 upon detection of an idle character. During address match wake-up, STAT[IDLE] does not become 1 when an address does not match. If this field is 1, during the Receive Standby state (CTRL[RWU] = 1), STAT[IDLE] becomes 1 upon detection of an idle character. During address match wake-up, STAT[IDLE] becomes 1 when an address does not match. 0b - STAT[IDLE] does not become 1 1b - STAT[IDLE] becomes 1
26 BRK13	Break Character Generation Length Selects the longer transmitted break character length.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	The state of this field does not affect the detection of a framing error. You must change the value of this field only when the transmitter is disabled. You can send a break character by writing 1 to CTRL[SBK], or by writing the transmit FIFO when DATA[FRETSC] is 1 and DATA[R9T9] is 0. 0b - 9 to 13 bit times 1b - 12 to 15 bit times
25 LBKDE	LIN Break Detection Enable Enables LIN break detection. If this field is 0, LIN break detect is disabled, and only a normal break character can be detected. If this field is 1, LIN break detect is enabled and the LIN break character is detected at a length of 11 bit times (if CTRL[M] is 0), 12 bit times (if CTRL[M] is 1), or 13 bit times (if BAUD[M10] is 1). This field selects a longer break character detection length. When the field is 1, receive data is not stored in the receive FIFO. **NOTE** This field enables the LIN break detect circuit and disables writing receive data to FIFO. Therefore, it ignores all characters except a LIN break. 0b - Disable 1b - Enable
24 RAF	Receiver Active Flag Indicates whether the LPUART receiver is idle or active. This field becomes 1 when the receiver detects the beginning of a valid start bit, and the field becomes 0 automatically when the receiver detects an idle line. 0b - Idle, waiting for a start bit 1b - Receiver active (RXD pin input not idle)
23 TDRE	Transmit Data Register Empty Flag Indicates whether the transmit FIFO level is greater than, equal to, or less than the watermark. After the transmit FIFO is enabled, this field becomes 1 when the number of datawords in the transmit FIFO is equal to, or less than the number that WATER[TXWATER] indicates. To make the value of this field 0, write to it until the number of words in the transmit FIFO is greater than the number that WATER[TXWATER] indicates. After the transmit FIFO is disabled, this field becomes 1 to indicate that the FIFO level is less than the watermark. To make the value of this field 0 again, write to Data (DATA). This register is not affected by a character that is in the process of being transmitted; it is updated at the start of each transmitted character. 0b - Greater than watermark 1b - Equal to or less than watermark
22	Transmission Complete Flag Indicates whether the transmitter is active.

Table continues on the next page...

Table continued from the previous page...

Field	Function
TC	This field becomes 0 when a transmission is in progress or a preamble or break character is loaded; in other words, when the transmitter is active (sending data, a preamble, or a break). The field becomes 1 when the transmit buffer is empty and no data, preamble, or break character is being transmitted; in other words, when the transmission activity is complete. When this happens, the transmit data output signal becomes idle (logic 1). This field becomes 0 after you write to Data (DATA) to transmit new data, queuing a preamble by first writing 0 and then writing 1 to CTRL[TE], queuing a break character by writing 1 to CTRL[SBK]. 0b - Transmitter active 1b - Transmitter idle
21 RDRF	Receive Data Register Full Flag Indicates whether the receive FIFO level is less than, equal to, or greater than the watermark. This field becomes 1 when the number of datawords in the receive buffer is greater than the number that WATER[RXWATER] indicates and the receive FIFO is enabled. To write 0 to this field, read Data (DATA) until the number of datawords in the receive FIFO is equal to, or less than the number that WATER[RXWATER] indicates. When the receive FIFO is disabled, this field (RDRF) becomes 1 if the receive buffer (Data (DATA)) is full. To make this field 0, read Data (DATA). A character that is in the process of being received does not cause a change in this field until the entire character is received. Even if this field is 1, the character continues to be received until an overrun condition occurs after the entire character is received. 0b - Equal to or less than watermark 1b - Greater than watermark
20 IDLE	Idle Line Flag Indicates whether an idle line is detected. This field becomes 1 when the LPUART receive line becomes idle for a full character time after a period of activity. When CTRL[ILT] is 0, the receiver starts counting idle bit times after the start bit. If the receive character is all 1s, these bit times and the stop bit time count towards the full character time of logic high, 10 to 13 bit times, needed for the receiver to detect an idle line. After CTRL[ILT] becomes 1, the receiver does not start counting idle bit times until after the stop bits. The stop bits and any logic high bit times at the end of the previous character do not count towards the full character time of logic high needed for the receiver to detect an idle line. For this field to become 0, write 1 to it. After the field becomes 0, you cannot write 1 to it again until after a new character is stored in the receive buffer or a LIN break character writes 1 to STAT[LBKDIF]. This field becomes 1 only once, even if the receive line remains idle for an extended period. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Idle line detected 1b - Idle line not detected When writing

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No effect 1b - Clear the flag
19 OR	Receiver Overrun Flag Indicates whether there is receive overrun. This field becomes 1 when you cannot prevent STAT[RDRF] from overflowing with data. The field becomes 1 immediately after the stop bit is completely received for the dataword that overflows the buffer and all the other error fields (STAT[FE], STAT[NF], and STAT[PF]) are prevented from becoming 1. The data in the shift register is lost, but the data already in the LPUART data registers is not affected. If STAT[LBKDE] is enabled and a LIN break is detected, this field becomes 1 if STAT[LBKDIF] is not 0 before the next data character is received. When this field is 1, no additional data is stored in the receive FIFO even if sufficient room exists. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No overrun 1b - Receive overrun (new LPUART data is lost) When writing 0b - No effect 1b - Clear the flag
18 NF	Noise Flag Indicates whether noise is detected in the received character of Data (DATA). The advanced sampling technique used in the receiver takes three samples in each of the received bits. If some of these samples disagree with the rest of the samples within any bit time in the frame, then noise is detected for that character. This field becomes 1 whenever the next character to be read from Data (DATA) is received with noise detected within the character. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No noise detected 1b - Noise detected When writing 0b - No effect 1b - Clear the flag
17 FE	Framing Error Flag Indicates whether a framing error is detected.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field becomes 1 whenever the next character to be read from Data (DATA) is received with logic 0 detected where a stop bit was expected. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No framing error detected (this does not guarantee that the framing is correct) 1b - Framing error detected When writing 0b - No effect 1b - Clear the flag
16 PF	Parity Error Flag Indicates whether a parity error is detected. This field becomes 1 whenever the next character to be read from Data (DATA) is received when parity is enabled (CTRL[PE] is 1) and the parity bit in the received character does not agree with the expected parity value. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No parity error detected 1b - Parity error detected When writing 0b - No effect 1b - Clear the flag
15 MA1F	Match 1 Flag Indicates whether the received data is equal to MATCH[MA1]. This field becomes 1 whenever the next character to be read from Data (DATA) matches the value of MATCH[MA1]. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not equal to MA1 1b - Equal to MA1 When writing

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No effect 1b - Clear the flag
14 MA2F	Match 2 Flag Indicates whether the received data is equal to MATCH[MA2]. This field becomes 1 whenever the next character to be read from Data (DATA) matches the value of MATCH[MA2]. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not equal to MA2 1b - Equal to MA2 When writing 0b - No effect 1b - Clear the flag
13-10 —	Reserved
9 TSF	Timeout Status Flag Indicates whether a field in Timeout Status (TOSR) is 1 and configured to generate an interrupt. **NOTE** This field is not supported in every instance. The following table includes only supported registers.

Instance	Field supported in	Field not supported in
LPUART_0	STAT	—
LPUART_1	STAT	—
LPUART_2	—	STAT
LPUART_3	—	STAT
LPUART_4	—	STAT
LPUART_5	—	STAT
LPUART_6	—	STAT

Table continues on the next page...

Table continued from the previous page...

Field	Function
	<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>LPUART_7</td><td>—</td><td>STAT</td></tr><tr><td>LPUART_8</td><td>—</td><td>STAT</td></tr><tr><td>LPUART_9</td><td>—</td><td>STAT</td></tr><tr><td>LPUART_10</td><td>—</td><td>STAT</td></tr><tr><td>LPUART_11</td><td>—</td><td>STAT</td></tr><tr><td>LPUART_12</td><td>—</td><td>STAT</td></tr><tr><td>LPUART_13</td><td>—</td><td>STAT</td></tr><tr><td>LPUART_14</td><td>—</td><td>STAT</td></tr><tr><td>LPUART_15</td><td>—</td><td>STAT</td></tr></table> 0b - Field is 0 1b - Field is 1
8 MSF	MODEM Status Flag Indicates whether a field in MODEM Status (MSR) is 1 and configured to generate an interrupt. **NOTE** This field is not supported in every instance. The following table includes only supported registers. <table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>LPUART_0</td><td>STAT</td><td>—</td></tr><tr><td>LPUART_1</td><td>STAT</td><td>—</td></tr><tr><td>LPUART_2</td><td>—</td><td>STAT</td></tr><tr><td>LPUART_3</td><td>—</td><td>STAT</td></tr><tr><td>LPUART_4</td><td>—</td><td>STAT</td></tr><tr><td>LPUART_5</td><td>—</td><td>STAT</td></tr><tr><td>LPUART_6</td><td>—</td><td>STAT</td></tr></table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>LPUART_7</td><td>—</td><td>STAT</td></tr><tr><td>LPUART_8</td><td>—</td><td>STAT</td></tr><tr><td>LPUART_9</td><td>—</td><td>STAT</td></tr><tr><td>LPUART_10</td><td>—</td><td>STAT</td></tr><tr><td>LPUART_11</td><td>—</td><td>STAT</td></tr><tr><td>LPUART_12</td><td>—</td><td>STAT</td></tr><tr><td>LPUART_13</td><td>—</td><td>STAT</td></tr><tr><td>LPUART_14</td><td>—</td><td>STAT</td></tr><tr><td>LPUART_15</td><td>—</td><td>STAT</td></tr></table> 0b - Field is 0 1b - Field is 1
7-2 —	Reserved
1 AME	Address Mark Enable Configures the location of the address mark when configured for MSB first transfers. This field has no effect when configured for LSB first and you must change the value of this field only when both the transmitter and receiver are disabled. If this field is 0, address mark in character is MSB. If this field is 1, the address mark is stored in Data (DATA) at MSB (or MSB-1 when the parity bit is enabled). In other words, the address mark in character is the last bit before the stop bit (or parity bit when enabled). 0b - Disable 1b - Enable
0 LBKFE	LIN Break Flag Enable Enables the LIN break flag to assert whenever a LIN break character is detected. Unlike STAT[LBKDE], this does not impact data being stored in the receive data buffer, but does cause STAT[LBKDIF] to become 1 whenever a LIN break is detected. Because a LIN break is longer than a normal character, the LIN break triggers a write to STAT[RDRF] with the data fields as 0 and STAT[FE] as 1. The character following the LIN break has DATA[LINBRK] as 1 to indicate that the previous character was a LIN break. You must change the value of this field only when both the transmitter and receiver are disabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	If this field is 1, the LIN break character is detected at a length of 11-bit times (if CTRL[M] is 0), 12 (if CTRL[M] is 1), or 13 (if BAUD[M10] is 1). 0b - Disable 1b - Enable

77.6.8 Control (CTRL)

Offset

Register	Offset
CTRL	18h

Function

Controls various optional features of the LPUART system.

You must write to the fields of this register only when both the transmitter and receiver are disabled.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	R8T9	R9T8	TXDIR	TXINV	ORIE	NEIE	FEIE	PEIE	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MA1IE	MA2IE	0	0	M7	IDLECFG			LOOPS	Reserved	RSRC	M	WAKE	ILT	PE	PT
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 R8T9	Receive Bit 8 Transmit Bit 9 Contains R8 and T9 that correspond to different functions. R8 is the ninth data bit received after you configure LPUART for 9-bit or 10-bit data formats. When reading 9-bit or 10-bit data, read R8 before reading Data (DATA). T9 is the tenth data bit transmitted after you configure LPUART for 10-bit data formats. When writing 10-bit data, write T9 before writing to Data (DATA). If T9 does not need to change from its previous value, such

Table continues on the next page...

Table continued from the previous page...

Field	Function
	as when it is used to generate address mark or parity, then you need not write to it each time you write to Data (DATA). **NOTE** R8 is a read-only bit and T9 is a write-only bit; the value read is different from the value written.
30 R9T8	Receive Bit 9 Transmit Bit 8 Contains R9 and T8 that correspond to different functions. R9 is the tenth data bit received after you configure LPUART for 10-bit data formats. When reading 10-bit data, read R9 before reading Data (DATA). T8 is the ninth data bit transmitted after you configure LPUART for 9-bit or 10-bit data formats. When writing 9-bit or 10-bit data, write T8 before writing to Data (DATA). If T8 does not need to change from its previous value, such as when it is used to generate address mark or parity, then you need not write to it each time you write to Data (DATA). **NOTE** R9 is a read-only field and T8 is a write-only field; the value read is different from the value written.
29 TXDIR	TXD Pin Direction in Single-Wire Mode Determines the direction of data at the TXD pin, in Single-Wire mode, when LPUART is configured for a single-wire half-duplex operation (CTRL[LOOPS] and CTRL[RSRC] are 1). When writing 0 to this field, the transmitter finishes transmitting the current character (if any) before the receiver starts receiving data from the TXD pin. 0b - Input 1b - Output
28 TXINV	Transmit Data Inversion Specifies whether transmit data is inverted. Writing 1 to this field reverses the polarity of the transmitted data output. This action inverts the TXD output for all cases: data bits, start and stop bits, break, and idle. 0b - Not inverted 1b - Inverted
27 ORIE	Overrun Interrupt Enable Enables STAT[OR] to generate hardware interrupt requests. When this field is 1, a hardware interrupt is requested. Use polling when OR interrupts are disabled. 0b - Disable 1b - Enable
26	Noise Error Interrupt Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
NEIE	Enables STAT[NF] to generate hardware interrupt requests. When this field is 1, a hardware interrupt is requested. Use polling when NF interrupts are disabled. 0b - Disable 1b - Enable
25 FEIE	Framing Error Interrupt Enable Enables STAT[FE] to generate hardware interrupt requests. When this field is 1, a hardware interrupt is requested. Use polling when FE interrupts are disabled. 0b - Disable 1b - Enable
24 PEIE	Parity Error Interrupt Enable Enables STAT[PF] to generate hardware interrupt requests. When this field is 1, a hardware interrupt is requested. Use polling when PF interrupts are disabled. 0b - Disable 1b - Enable
23 TIE	Transmit Interrupt Enable Enables STAT[TDRE] to generate interrupt requests if STAT[TDRE] is 1. 0b - Disable 1b - Enable
22 TCIE	Transmission Complete Interrupt Enable Enables STAT[TC] to generate interrupt requests if STAT[TC] is 1. 0b - Disable 1b - Enable
21 RIE	Receiver Interrupt Enable Enables STAT[RDRF] to generate hardware interrupt requests if STAT[RDRF] is 1. 0b - Disable 1b - Enable
20 ILIE	Idle Line Interrupt Enable Enables hardware interrupts. This field enables STAT[IDLE] to generate interrupt requests. If this field is 0, hardware interrupts from STAT[IDLE] are disabled and polling is used, and if this field is 1, hardware interrupts are enabled when STAT[IDLE] is 1. 0b - Disable 1b - Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
19 TE	Transmitter Enable Enables the LPUART transmitter. Using this field, you can also queue an idle preamble by first writing 0 and then writing 1 to this field. After this field becomes 0, the field reads 1 until the transmitter has completed the current character and the TXD pin is tristated. You can also queue a single idle character by writing to the transmit FIFO with DATA[FRETSC] and DATA[R9T9] = 1. 0b - Disable 1b - Enable
18 RE	Receiver Enable Enables the LPUART receiver. After you write 0 to this field, this field remains 1 until the receiver finishes receiving the current character (if any). 0b - Disable 1b - Enable
17 RWU	Receiver Wake-Up Control Specifies whether the LPUART receiver in standby is waiting for a wake-up condition. You can write 1 to this field to place the LPUART receiver in a Standby state. The field becomes 0 automatically when an RWU event occurs, that is, in case of an idle event when CTRL[WAKE] is 0 or an address match when CTRL[WAKE] is 1 and STAT[RWUID] is 0. **NOTE** You must write 1 to this field only when CTRL[WAKE] is 0 (wake-up on idle), if the channel is currently not idle. You can determine this by the value of STAT[RAF]. If the field is 1 to wake up an idle event and the channel is already idle, LPUART, possibly, discards the data. This is because the data must be received or a LIN break is detected after an Idle condition is detected before the IDLE flag is allowed to be reasserted. 0b - Normal receiver operation 1b - LPUART receiver in standby, waiting for a wake-up condition
16 SBK	Send Break Specifies whether queue break character(s) are to be sent. Writing 1 and then 0 to this field queues a break character in the transmit data stream. Additional break characters of 9 to 13 bits, or 12 to 15 bits if STAT[BRK13] is 1, and bit times of logic 0 are queued as long as this field is 1. Depending on the timing when this field is 1 and 0, relative to the character currently being transmitted, a second break character may be queued before you write 0 to this field. If the time taken to write 0 to this field is too long, for example, if the field does not become 0 by the end of the first break character, a second break character is sent. This is compared to queuing a break character through the transmit FIFO that guarantees only one break character is sent.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	You can also queue a single break character by writing to the transmit FIFO when DATA[FRETSC] is 1 and DATA[R9T9] is 0. 0b - Normal transmitter operation 1b - Queue break character(s) to be sent
15 MA1IE	Match 1 (MA1F) Interrupt Enable Enables the MA1F interrupt. 0b - Disable 1b - Enable
14 MA2IE	Match 2 (MA2F) Interrupt Enable Enables the MA2F interrupt. 0b - Disable 1b - Enable
13 —	Reserved
12 —	Reserved
11 M7	7-Bit Mode Select Specifies the data characters that the receiver and transmitter use. You must change the value of this field only after both the transmitter and receiver are disabled. 0b - 8-bit to 10-bit 1b - 7-bit
10-8 IDLECFG	Idle Configuration Configures the number of idle characters that must be received before you write 1 to STAT[IDLE]. 000b - 1 001b - 2 010b - 4 011b - 8 100b - 16 101b - 32 110b - 64 111b - 128
7	Loop Mode Select

Table continues on the next page...

Table continued from the previous page...

Field	Function
LOOPS	Selects Loop mode. After this field becomes 1, the RXD pin is disconnected from LPUART and the transmitter output is internally connected to the receiver input. The transmitter and receiver must be enabled to use the loop function. In Loop mode or Single-Wire mode, the transmitter outputs are internally connected to the receiver input (see CTRL[RSRC]). 0b - Normal operation: RXD and TXD use separate pins 1b - Loop mode or Single-Wire mode
6 —	Reserved
5 RSRC	Receiver Source Select Determines the source of the receiver shift register input if CTRL[LOOPS] is 1. This field has no effect unless CTRL[LOOPS] is 1. If this field is 0, internal Loopback mode is selected. LPUART does not use the RXD pin. Additionally, the CTS_B pin is not used and internally driven by the RTS_B output. If this field is 1, single-wire LPUART mode is selected where the TXD pin is connected to the transmitter output and receiver input. 0b - Internal Loopback mode 1b - Single-wire mode
4 M	9-Bit Or 8-Bit Mode Select Specifies the data characters that the receiver and transmitter use. 0b - 8-bit 1b - 9-bit
3 WAKE	Receiver Wake-Up Method Select Determines which condition wakes up LPUART when CTRL[RWU] = 1 and BAUD[MATCFG] = 0 (this field must be 1 when BAUD[MATCFG] = 11): • Address mark in the bit preceding the stop bit (or bit preceding the parity bit when parity is enabled) of the received data character • An idle condition on the receive pin input signal If this field is 0, CTRL[RWU] is configured for idle line wake-up, and if this field is 1, CTRL[RWU] is configured with address mark wake-up. 0b - Idle 1b - Mark
2 ILT	Idle Line Type Select Determines when the receiver starts counting logic 1s as idle character bits.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	The count begins either after a valid start bit or the stop bit. If the count begins after the start bit, a string of logic 1s preceding the stop bit can cause false recognition of an idle character. Beginning the count after the stop bit avoids false idle character recognition, but requires properly synchronized transmissions. **NOTE** In case you write 1 to this field, a logic 0 is automatically shifted after a received stop bit, therefore resetting the idle count. 0b - After the start bit 1b - After the stop bit
1 PE	Parity Enable Enables hardware parity generation and checking. If parity is enabled, the bit immediately before the stop bit is treated as the parity bit. 0b - Disable 1b - Enable
0 PT	Parity Type Selects the type of parity, even or odd, if parity is enabled (CTRL[PE] = 1): • Odd parity means that the total number of logic 1 bits in the data character, including the parity bit, is odd. • Even parity means that the total number of 1s in the data character, including the parity bit, is even. 0b - Even parity 1b - Odd parity

77.6.9 Data (DATA)

Offset

Register	Offset
DATA	1Ch

Function

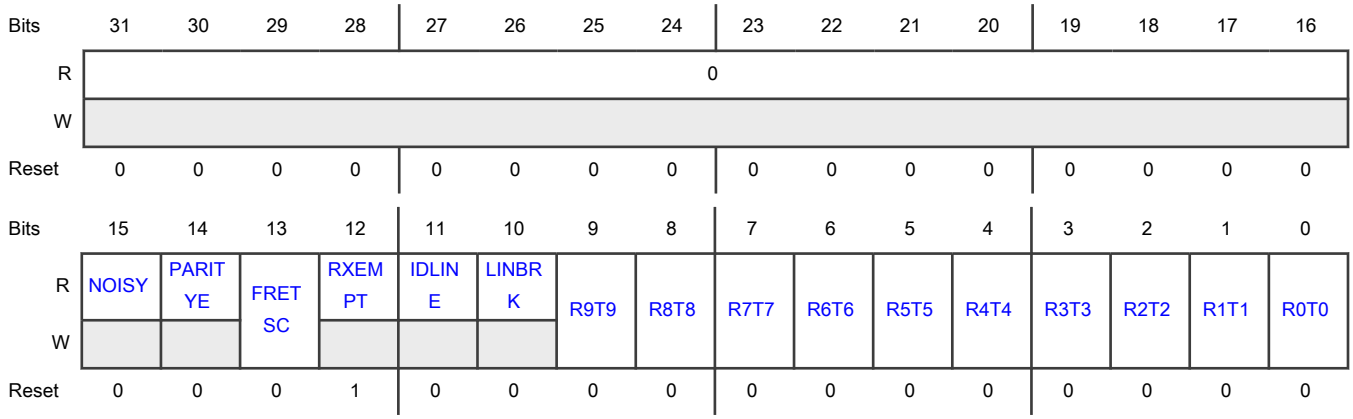
Supports 8-bit, 16-bit, or 32-bit writes, each type of write performing a separate function. An 8-bit write to DATA[7:0] pushes {CTRL[R8T9], CTRL[R9T8], DATA[7:0]} the transmit FIFO with TSC clear. A 16-bit or 32-bit write pushes the data written into the FIFO and does not update the value of CTRL[R8T9] or CTRL[R9T8].

Reads and writes of this register are also involved in the automatic flag clearing mechanisms for some of the LPUART status fields.

NOTE

Reads return the contents of the read-only receive FIFO and writes go to the write-only transmit FIFO, making this register work as a set of two separate registers.

Diagram



Fields

Field	Function
31-16 —	Reserved
15 NOISY	Noisy Data Received Indicates whether the current received dataword contained in DATA[R9:R0] is received with noise. 0b - Received without noise 1b - Received with noise
14 PARITYE	Parity Error Indicates whether the current received dataword contained in DATA[R9:R0] is received with a parity error. 0b - Received without a parity error 1b - Received with a parity error
13 FRETSC	Frame Error Transmit Special Character Specifies the way the dataword is received. For reads, this field indicates that the current received dataword contained in DATA[R9:R0] is received with a frame error. For writes, the field indicates that a break or idle character is to be transmitted instead of the contents in DATA[T9:T0]. T9 indicates a break character when it is 0 and indicates an idle character when it is 1. The contents of DATA[T8:T0] must be 0. 0b - Received without a frame error on reads or transmits a normal character on writes 1b - Received with a frame error on reads or transmits an idle or break character on writes
12 RXEMPT	Receive Buffer Empty Indicates whether the receive buffer contains valid data. This field becomes 1 when there is no data in the receive buffer. The field does not consider data in the receive shift register.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Valid data 1b - Invalid data and empty
11 IDLINE	Idle Line Indicates whether the receiver line was idle before receiving the character in DATA[9:0]. It can be read as “1” on the first character when the receiver is first enabled. The difference between this field and STAT[IDLE] is that, STAT[IDLE] flag does not set on an idle line after the receiver is first enabled, it needs to receive a character before it can become set, whereas this field does not have this limitation and can be set on the first character received if an idle line is detected beforehand. 0b - Not idle 1b - Idle
10 LINBRK	LIN Break Indicates whether the receiver line detected a LIN break before receiving the character in DATA[9:0]. This field requires the value of STAT[LBKDIF] to be 1. If this field is 0, the LIN break detect circuitry is disabled. 0b - Not detected 1b - Detected
9 R9T9	Read receive FIFO bit 9 or write transmit FIFO bit 9
8 R8T8	Read receive FIFO bit 8 or write transmit FIFO bit 8
7 R7T7	Read receive FIFO bit 7 or write transmit FIFO bit 7
6 R6T6	Read receive FIFO bit 6 or write transmit FIFO bit 6
5 R5T5	Read receive FIFO bit 5 or write transmit FIFO bit 5
4 R4T4	Read receive FIFO bit 4 or write transmit FIFO bit 4
3 R3T3	Read receive FIFO bit 3 or write transmit FIFO bit 3
2 R2T2	Read receive FIFO bit 2 or write transmit FIFO bit 2

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 R1T1	Read receive FIFO bit 1 or write transmit FIFO bit 1
0 R0T0	Read receive FIFO bit 0 or write transmit FIFO bit 0

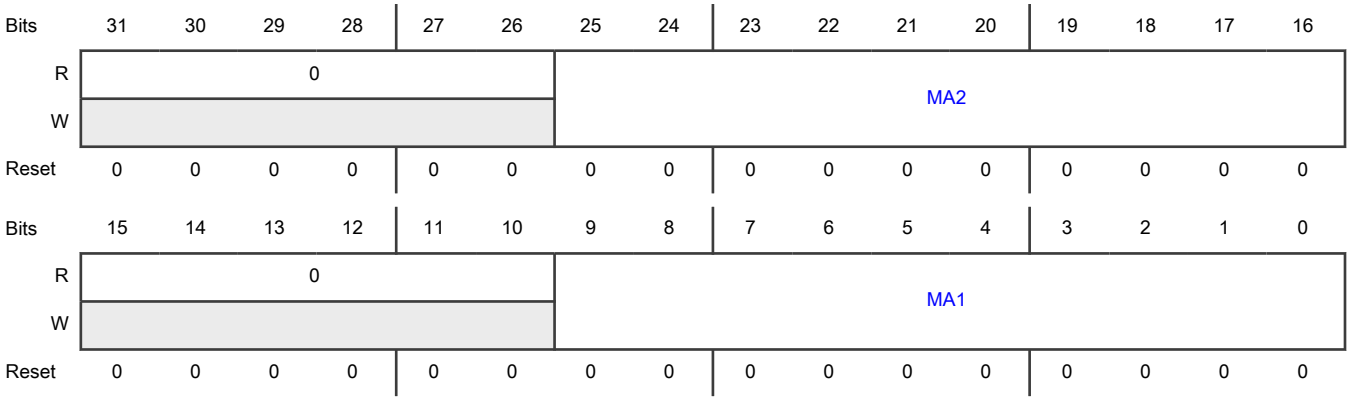
77.6.10 Match Address (MATCH)

Offset

Register	Offset
MATCH	20h

Function
Provides addresses for address matching during the receiver operation.

Diagram



Fields

Field	Function
31-26 —	Reserved
25-16 MA2	Match Address 2 Is compared to input data addresses when the most significant bit is 1 and the associated Baud Rate (BAUD) field is 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	If a match occurs, the data that follows is transferred to Data (DATA). If a match fails, the data that follows is discarded. You must write to MATCH[MA1] and MATCH[MA2] only when the associated Baud Rate (BAUD) field is 0.
15-10 —	Reserved
9-0 MA1	Match Address 1 Is compared to input data addresses when the most significant bit is 1 and the associated Baud Rate (BAUD) field is 1. If a match occurs, the data that follows is transferred to Data (DATA). If a match fails, the data that follows is discarded. You must write to MATCH[MA1] and MATCH[MA2] fields only when the associated field in Baud Rate (BAUD) is 0.

77.6.11 MODEM IrDA (MODIR)

Offset

Register	Offset
MODIR	24h

Function
Controls options for setting the MODEM configuration.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0													IREN	TNP	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				RTSWATER				0		TXCTSSRC	TXCTSC	RXRTSE	TXRTSPOL	TXRTSE	TXCTSE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-19	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
18 IREN	IR Enable Enables IR modulation and demodulation. You must change the value of this field only when both the transmitter and receiver are disabled. 0b - Disable 1b - Enable
17-16 TNP	Transmitter Narrow Pulse Specifies whether LPUART transmits a 1 ÷ OSR, 2 ÷ OSR, 3 ÷ OSR, or 4 ÷ OSR narrow pulse when the IR pulse is enabled. You must change the value of this field only when both the transmitter and receiver are disabled. The IR pulse width must be configured to less than half of the OSR. Common pulse widths are 3 ÷ 16, 1 ÷ 16, 1 ÷ 32, or 1 ÷ 4 of the bit length. You can configure these by selecting the appropriate OSR and pulse width. 00b - 1 ÷ OSR 01b - 2 ÷ OSR 10b - 3 ÷ OSR 11b - 4 ÷ OSR
15-12 —	Reserved
11-8 RTSWATER	Receive RTS Configuration Configures the assertion and negation of the receiver's RTS_B output. The receiver's RTS_B output negates when the number of empty words in the receive FIFO is greater or equal to the value of this field. If this field is 0, the RTS_B pin negates when the receive FIFO is full. For the purpose of receive RTS_B generation, the number of words in the receive FIFO updates when a start bit is detected. This supports additional latency between RTS_B negation and the external transmitter ceasing transmission. If both receive RTS_B and address or data matching is enabled, RTS_B could assert at the end of a character if there exists no match. You must change the value of this field only when the receiver is disabled. **NOTE** This field is not supported in every instance. The following table includes only supported registers.

Instance	Field supported in	Field not supported in
LPUART_0	MODIR	—
LPUART_1	MODIR	—

Table continued from the previous page...

Field	Function

Instance	Field supported in	Field not supported in
LPUART_2	MODIR[9–8]	MODIR[11–10]
LPUART_3	MODIR[9–8]	MODIR[11–10]
LPUART_4	MODIR[9–8]	MODIR[11–10]
LPUART_5	MODIR[9–8]	MODIR[11–10]
LPUART_6	MODIR[9–8]	MODIR[11–10]
LPUART_7	MODIR[9–8]	MODIR[11–10]
LPUART_8	MODIR[9–8]	MODIR[11–10]
LPUART_9	MODIR[9–8]	MODIR[11–10]
LPUART_10	MODIR[9–8]	MODIR[11–10]
LPUART_11	MODIR[9–8]	MODIR[11–10]
LPUART_12	MODIR[9–8]	MODIR[11–10]
LPUART_13	MODIR[9–8]	MODIR[11–10]
LPUART_14	MODIR[9–8]	MODIR[11–10]
LPUART_15	MODIR[9–8]	MODIR[11–10]

7-6 —	Reserved
5 TXCTSSRC	Transmit CTS Source Configures the source of the CTS input. 0b - The CTS_B pin 1b - An internal connection to the receiver address match result
4 TXCTSC	Transmit CTS Configuration Configures whether the CTS state or input is checked or sampled at the start of each character or only when the transmitter is idle. 0b - Sampled at the start of each character 1b - Sampled when the transmitter is idle

Table continues on the next page...

Table continued from the previous page...

Field	Function
3 RXRTSE	Receiver RTS Enable Allows the RTS output to control the CTS input of the transmitting device to prevent receiver overrun. You must change the value of this field only when the receiver is disabled. If this field is 0, the receiver has no effect on RTS. If this field is 1, RTS is deasserted if STAT[RDRF] is 1 or a start bit is detected that causes STAT[RDRF] to become 1. RTS is asserted if STAT[RDRF] is 0 and has not detected a start bit that causes STAT[RDRF] to become 1. **NOTE** Do not write 1 to both MODIR[RXRTSE] and MODIR[TXRTSE]. 0b - Disable 1b - Enable
2 TXRTSPOL	Transmitter RTS Polarity Controls the polarity of the transmitter RTS. This field does not affect the polarity of the receiver RTS that remains negated in the active-low state unless MODIR[TXRTSE] is 1. You must change the value of this field only when the transmitter is disabled. 0b - Active low 1b - Active high
1 TXRTSE	Transmitter RTS Enable Controls the operation of RTS before and after a transmission. You must change the value of this field only when the transmitter is disabled. If this field is 0, the transmitter has no effect on RTS, and if this field is 1, a character is placed into an empty transmit shift register. RTS asserts 1-bit time before the start bit is transmitted and deasserts 1-bit time after all characters in the transmitter FIFO and shift register are completely sent, including the last stop bit. 0b - Disable 1b - Enable
0 TXCTSE	Transmitter CTS Enable Enables the operation of the transmitter. You can write 1 to this field irrespective of the states of MODIR[TXRTSE] and MODIR[RXRTSE]. If this field is 1, the transmitter checks the state of the CTS signal each time it is ready to send a character. If CTS is asserted, the character is sent. If CTS is deasserted, the TXD signal remains in the mark state and transmission is delayed until CTS is asserted. Changes in CTS, when a character is being sent, do not affect its transmission. 0b - Disable 1b - Enable

77.6.12 FIFO (FIFO)

Offset

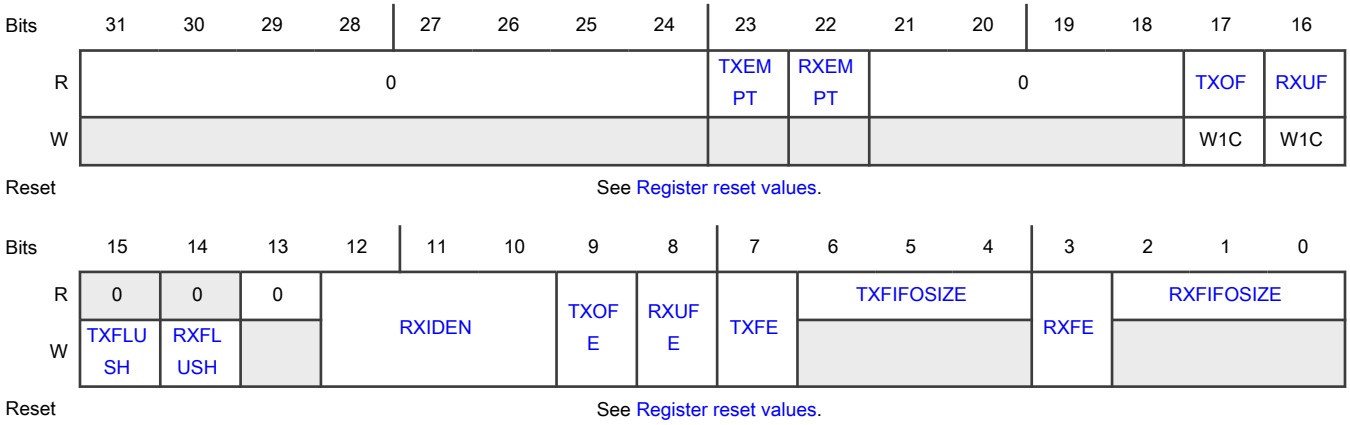
Register	Offset
FIFO	28h

Function

Provides you the ability to turn on and turn off the FIFO functionality.

This register also provides you the size of the FIFO that has been implemented. You can read this register at any time and must write to it only when CTRL[RE] and CTRL[TE] are 0 and the FIFO is empty.

Diagram



Register reset values

Register	Reset value
FIFO	LPUART_0,LPUART_1: 00C0_0033h LPUART_2–LPUART_15: 00C0_0011h

Fields

Field	Function
31-24 —	Reserved
23 TXEMPT	Transmit FIFO Or Buffer Empty Indicates whether the transmit buffer is empty. This field becomes 1 when there is no data in the transmit FIFO or buffer. The field does not consider data in the transmit shift register.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Not empty 1b - Empty
22 RXEMPT	Receive FIFO Or Buffer Empty Indicates whether the receive buffer is empty. This field becomes 1 when there is no data in the receive FIFO or buffer. The field does not consider data in the receive shift register. 0b - Not empty 1b - Empty
21-18 —	Reserved
17 TXOF	Transmitter FIFO Overflow Flag Indicates whether more data has been written to the transmit FIFO than it can hold. If this field is 0, no transmit FIFO overflow has occurred since the last time the field was cleared, and if this field is 1, at least one transmit FIFO overflow has occurred since the last time the field was cleared. This field becomes 1 regardless of the value of FIFO[TXOFE]. However, an interrupt is issued to the host only if FIFO[TXOFE] is 1. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No overflow 1b - Overflow When writing 0b - No effect 1b - Clear the flag
16 RXUF	Receiver FIFO Underflow Flag Indicates whether more data has been read from the receive FIFO than was present. If this field is 0, no receive FIFO underflow has occurred since the last time the field was cleared, and if this field is 1, at least one receive FIFO underflow has occurred since the last time the field was cleared. This field becomes 1 regardless of the value of FIFO[RXUFE]. However, an interrupt is issued to the host only if FIFO[RXUFE] is 1. **NOTE** This field behaves differently for register reads and writes. When reading

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No underflow 1b - Underflow When writing 0b - No effect 1b - Clear the flag
15 TXFLUSH	Transmit FIFO Flush Causes all data that is stored in the transmit FIFO to be flushed. If you write 0 to this field, no flush operation occurs, and if you write 1 to this field, all data in the transmit FIFO or buffer clears out. This does not affect data in the transmit shift register. 0b - No effect 1b - All data flushed out
14 RXFLUSH	Receive FIFO Flush Causes all data that is stored in the receive FIFO to be flushed. If you write 0 to this field, no flush operation occurs, and if you write 1 to this field, all data in the receive FIFO or buffer clears out. This does not affect data in the receive shift register. 0b - No effect 1b - All data flushed out
13 —	Reserved
12-10 RXIDEN	Receiver Idle Empty Enable Enables STAT[RDRF] to become 1 when the receiver is idle for a number of idle characters and the FIFO is not empty. This feature is not supported when the receiver extended idle time is enabled. 000b - Disable STAT[RDRF] to become 1 because of partially filled FIFO when the receiver is idle 001b - Enable STAT[RDRF] to become 1 because of partially filled FIFO when the receiver is idle for one character 010b - Enable STAT[RDRF] to become 1 because of partially filled FIFO when the receiver is idle for two characters 011b - Enable STAT[RDRF] to become 1 because of partially filled FIFO when the receiver is idle for four characters 100b - Enable STAT[RDRF] to become 1 because of partially filled FIFO when the receiver is idle for eight characters

Table continues on the next page...

Table continued from the previous page...

Field	Function
	101b - Enable STAT[RDRF] to become 1 because of partially filled FIFO when the receiver is idle for 16 characters 110b - Enable STAT[RDRF] to become 1 because of partially filled FIFO when the receiver is idle for 32 characters 111b - Enable STAT[RDRF] to become 1 because of partially filled FIFO when the receiver is idle for 64 characters
9 TXOFE	Transmit FIFO Overflow Interrupt Enable Enables FIFO[TXOF] to generate an interrupt to the host. 0b - Disable 1b - Enable
8 RXUFE	Receive FIFO Underflow Interrupt Enable Enables FIFO[RXUF] to generate an interrupt to the host. 0b - Disable 1b - Enable
7 TXFE	Transmit FIFO Enable Enables the transmit FIFO. If this field is 0, the transmit buffer operates as a FIFO of depth equal to 1 dataword, regardless of the value in FIFO[TXFIFOSIZE]. Both CTRL[TE] and CTRL[RE] must be 0 before you change the value of this field. If this field is 1, the built-in FIFO structure for the transmit buffer is enabled. FIFO[TXFIFOSIZE] indicates the size of the FIFO structure. 0b - Disable 1b - Enable
6-4 TXFIFOSIZE	Transmit FIFO Buffer Depth Indicates the maximum number of transmit datawords (transmit FIFO buffer depth) that can be stored in the transmit buffer. 000b - 1 001b - 4 010b - 8 011b - 16 100b - 32 101b - 64 110b - 128 111b - 256
3	Receive FIFO Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
RXFE	Enables the receive FIFO. If this field is 0, the receive buffer operates as a FIFO of depth equal to 1 dataword, regardless of the value in FIFO[RXFIFOSIZE]. Both CTRL[RE] and CTRL[TE] must be 0 before you change the value of this field. If this field is 1, the built-in FIFO structure for the receive buffer is enabled. FIFO[RXFIFOSIZE] indicates the size of the FIFO structure. 0b - Disable 1b - Enable
2-0 RXFIFOSIZE	Receive FIFO Buffer Depth Indicates the maximum number of receive datawords (receive FIFO buffer depth) that can be stored in the receive buffer before an overrun occurs. 000b - 1 001b - 4 010b - 8 011b - 16 100b - 32 101b - 64 110b - 128 111b - 256

77.6.13 Watermark (WATER)

Offset

Register	Offset
WATER	2Ch

Function

Provides the ability to set a programmable threshold for notification, or sets the programmable thresholds to indicate that transmit data can be written or receive data can be read.

You may read this register at any time but must write to it only when CTRL[TE] is 0.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				RXCOUNT				0				RXWATER			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				TXCOUNT				0				TXWATER			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	Reserved
28-24 RXCOUNT	Receive Counter Indicates the number of datawords in the receive FIFO or buffer. If a dataword is being received in the receive shift register, it is not included in the count. This value may be used in conjunction with FIFO[RXFIFOSIZE] to calculate the room left in the receive FIFO or buffer. **NOTE** This field is not supported in every instance. The following table includes only supported registers.

Instance	Field supported in	Field not supported in
LPUART_0	WATER	—
LPUART_1	WATER	—
LPUART_2	WATER[26–24]	WATER[28–27]
LPUART_3	WATER[26–24]	WATER[28–27]
LPUART_4	WATER[26–24]	WATER[28–27]
LPUART_5	WATER[26–24]	WATER[28–27]
LPUART_6	WATER[26–24]	WATER[28–27]
LPUART_7	WATER[26–24]	WATER[28–27]
LPUART_8	WATER[26–24]	WATER[28–27]
LPUART_9	WATER[26–24]	WATER[28–27]

Field	Function

Instance	Field supported in	Field not supported in
LPUART_10	WATER[26–24]	WATER[28–27]
LPUART_11	WATER[26–24]	WATER[28–27]
LPUART_12	WATER[26–24]	WATER[28–27]
LPUART_13	WATER[26–24]	WATER[28–27]
LPUART_14	WATER[26–24]	WATER[28–27]
LPUART_15	WATER[26–24]	WATER[28–27]

Field	Function
23-20 —	Reserved
19-16 RXWATER	Receive Watermark Generates an interrupt or a DMA request if the number of datawords in the receive FIFO or buffer is greater than the value of this field. For proper operation, the value of this field must be less than the size of the receive FIFO or buffer, as indicated by FIFO[RXFIFOSIZE] and FIFO[RXFE]. **NOTE** This field is not supported in every instance. The following table includes only supported registers.

Instance	Field supported in	Field not supported in
LPUART_0	WATER	—
LPUART_1	WATER	—
LPUART_2	WATER[17–16]	WATER[19–18]
LPUART_3	WATER[17–16]	WATER[19–18]
LPUART_4	WATER[17–16]	WATER[19–18]
LPUART_5	WATER[17–16]	WATER[19–18]
LPUART_6	WATER[17–16]	WATER[19–18]
LPUART_7	WATER[17–16]	WATER[19–18]
LPUART_8	WATER[17–16]	WATER[19–18]

Table continued from the previous page...

Field	Function

Instance	Field supported in	Field not supported in
LPUART_9	WATER[17–16]	WATER[19–18]
LPUART_10	WATER[17–16]	WATER[19–18]
LPUART_11	WATER[17–16]	WATER[19–18]
LPUART_12	WATER[17–16]	WATER[19–18]
LPUART_13	WATER[17–16]	WATER[19–18]
LPUART_14	WATER[17–16]	WATER[19–18]
LPUART_15	WATER[17–16]	WATER[19–18]

Field	Function
15-13 —	Reserved
12-8 TXCOUNT	Transmit Counter Indicates the number of datawords in the transmit FIFO or buffer. If a dataword is being transmitted to the transmit shift register, it is not included in the count. This value may be used in conjunction with the value of FIFO[TXFIFOSIZE] to calculate the room left in the transmit FIFO or buffer. **NOTE** This field is not supported in every instance. The following table includes only supported registers.

Instance	Field supported in	Field not supported in
LPUART_0	WATER	—
LPUART_1	WATER	—
LPUART_2	WATER[10–8]	WATER[12–11]
LPUART_3	WATER[10–8]	WATER[12–11]
LPUART_4	WATER[10–8]	WATER[12–11]
LPUART_5	WATER[10–8]	WATER[12–11]
LPUART_6	WATER[10–8]	WATER[12–11]

Table continues on the next page...

Table continued from the previous page...

Field	Function

Instance	Field supported in	Field not supported in
LPUART_7	WATER[10–8]	WATER[12–11]
LPUART_8	WATER[10–8]	WATER[12–11]
LPUART_9	WATER[10–8]	WATER[12–11]
LPUART_10	WATER[10–8]	WATER[12–11]
LPUART_11	WATER[10–8]	WATER[12–11]
LPUART_12	WATER[10–8]	WATER[12–11]
LPUART_13	WATER[10–8]	WATER[12–11]
LPUART_14	WATER[10–8]	WATER[12–11]
LPUART_15	WATER[10–8]	WATER[12–11]

Field	Function
7-4 —	Reserved
3-0 TXWATER	Transmit Watermark Generates an interrupt or a DMA request when the number of datawords in the transmit FIFO or buffer is equal to or less than the value of this field. For proper operation, the value of this field must be less than the size of the transmit buffer or FIFO, as indicated by FIFO[TXFIFOSIZE] and FIFO[TXFE]. **NOTE** This field is not supported in every instance. The following table includes only supported registers.

Instance	Field supported in	Field not supported in
LPUART_0	WATER	—
LPUART_1	WATER	—
LPUART_2	WATER[1–0]	WATER[3–2]
LPUART_3	WATER[1–0]	WATER[3–2]
LPUART_4	WATER[1–0]	WATER[3–2]

Table continues on the next page...

Table continued from the previous page...

Field	Function		
	Instance	Field supported in	Field not supported in
	LPUART_5	WATER[1–0]	WATER[3–2]
	LPUART_6	WATER[1–0]	WATER[3–2]
	LPUART_7	WATER[1–0]	WATER[3–2]
	LPUART_8	WATER[1–0]	WATER[3–2]
	LPUART_9	WATER[1–0]	WATER[3–2]
	LPUART_10	WATER[1–0]	WATER[3–2]
	LPUART_11	WATER[1–0]	WATER[3–2]
	LPUART_12	WATER[1–0]	WATER[3–2]
	LPUART_13	WATER[1–0]	WATER[3–2]
	LPUART_14	WATER[1–0]	WATER[3–2]
	LPUART_15	WATER[1–0]	WATER[3–2]

77.6.14 Data Read-Only (DATARO)

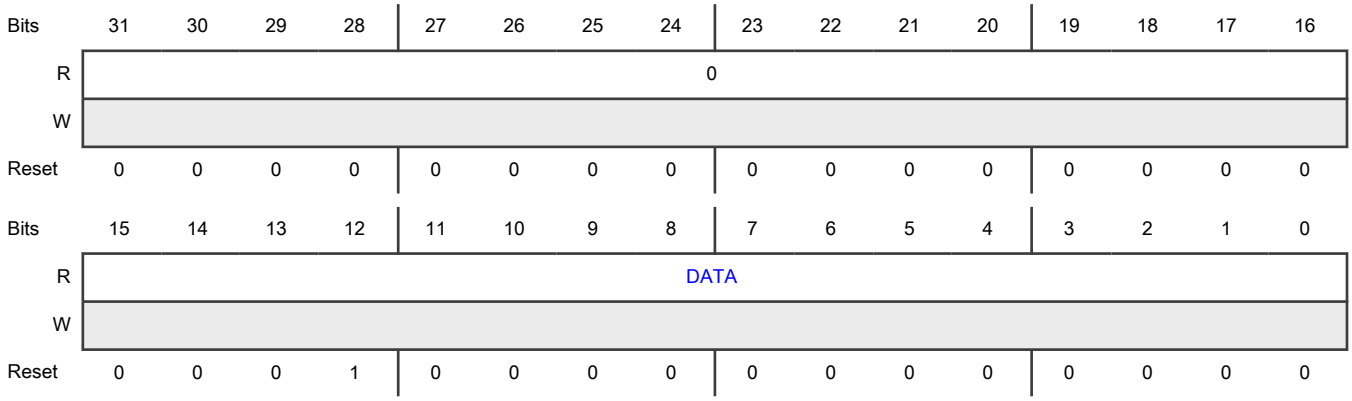
Offset

Register	Offset
DATARO	30h

Function

Indicates the first entry in the receive FIFO, but does not pull data from the FIFO.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 DATA	Receive Data Indicates the first entry from the receive FIFO. This register has the same functionality as that of Data (DATA).

77.6.15 MODEM Control (MCR)

Offset

Register	Offset
MCR	40h

Function

Controls the operation of the MODEM pins.

NOTE

Each module instance supports a different number of registers.

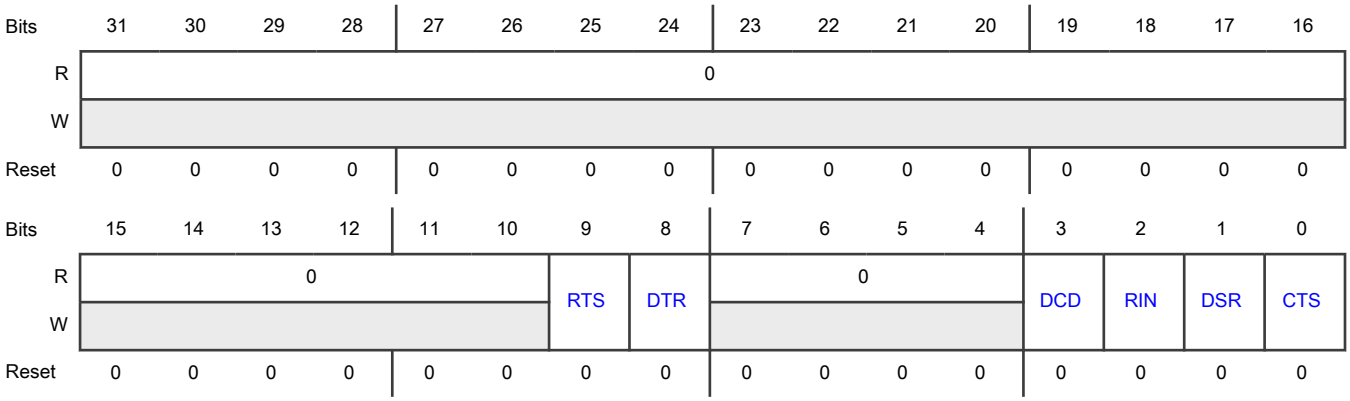
Instance	Register supported	Register not supported
LPUART_0	MCR	—
LPUART_1	MCR	—
LPUART_2	—	MCR
LPUART_3	—	MCR

Table continues on the next page...

Table continued from the previous page...

Instance	Register supported	Register not supported
LPUART_4	—	MCR
LPUART_5	—	MCR
LPUART_6	—	MCR
LPUART_7	—	MCR
LPUART_8	—	MCR
LPUART_9	—	MCR
LPUART_10	—	MCR
LPUART_11	—	MCR
LPUART_12	—	MCR
LPUART_13	—	MCR
LPUART_14	—	MCR
LPUART_15	—	MCR

Diagram



Fields

Field	Function
31-10 —	Reserved
9	Request To Send

Table continues on the next page...

Table continued from the previous page...

Field	Function
RTS	Configures the default state of the RTS_B pin when the function is disabled. 0b - Logic one 1b - Logic zero
8 DTR	Data Terminal Ready Configures the default state of the DTR_B pin. 0b - Logic one 1b - Logic zero
7-4 —	Reserved
3 DCD	Data Carrier Detect Configures the interrupt, DCD_B, for change of state of the DCD_B pin. 0b - Disable interrupt 1b - Enable interrupt
2 RIN	Ring Indicator Configures the interrupt, RIN_B, for change of state on the RIN_B pin. 0b - Disable interrupt 1b - Enable interrupt
1 DSR	Data Set Ready Configures the interrupt, DSR_B, for change of state on the DSR_B pin. 0b - Disable interrupt 1b - Enable interrupt
0 CTS	Clear To Send Configures the interrupt, CTS_B, for change of state on the CTS_B pin. 0b - Disable interrupt 1b - Enable interrupt

77.6.16 MODEM Status (MSR)

Offset

Register	Offset
MSR	44h

Function

Indicates the status of the MODEM pins.

NOTE

You must appropriately configure the PAD connecting to DCD_B, RIN_B, DSR_B, and CTS_B inputs to get appropriate reset values for the MSR register fields.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
LPUART_0	MSR	—
LPUART_1	MSR	—
LPUART_2	—	MSR
LPUART_3	—	MSR
LPUART_4	—	MSR
LPUART_5	—	MSR
LPUART_6	—	MSR
LPUART_7	—	MSR
LPUART_8	—	MSR
LPUART_9	—	MSR
LPUART_10	—	MSR
LPUART_11	—	MSR
LPUART_12	—	MSR
LPUART_13	—	MSR
LPUART_14	—	MSR
LPUART_15	—	MSR

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								DCD	RIN	DSR	CTS	DDCD	DRI	DDSR	DCTS
W													W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-8 —	Reserved
7 DCD	Data Carrier Detect Indicates the state of the DCD_B pin. 0b - Logic one 1b - Logic zero
6 RIN	Ring Indicator Indicates the state of the RIN_B pin. 0b - Logic one 1b - Logic zero
5 DSR	Data Set Ready Indicates the state of the DSR_B pin. 0b - Logic one 1b - Logic zero
4 CTS	Clear To Send Indicates the state of the CTS_B pin. 0b - Logic one 1b - Logic zero
3 DDCD	Delta Data Carrier Detect Indicates whether the DCD_B pin changed state since the last time this field was 0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field behaves differently for register reads and writes. When reading 0b - Did not change state 1b - Changed state When writing 0b - No effect 1b - Clear the flag
2 DRI	Delta Ring Indicator Indicates whether the RIN_B pin changed state since the last time this field was 0. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Did not change state 1b - Changed state When writing 0b - No effect 1b - Clear the flag
1 DDSR	Delta Data Set Ready Indicates whether the DSR_B pin changed state since the last time this field was 0. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Did not change state 1b - Changed state When writing 0b - No effect 1b - Clear the flag
0 DCTS	Delta Clear To Send Indicates whether the CTS_B pin changed state since the last time this field was 0. **NOTE** This field behaves differently for register reads and writes.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When reading 0b - Did not change state 1b - Changed state When writing 0b - No effect 1b - Clear the flag

77.6.17 Receiver Extended Idle (REIR)

Offset

Register	Offset
REIR	48h

Function

Configures the receiver extended idle functionality. You must not change this value when the receiver is enabled.

NOTE

Each module instance supports a different number of registers.

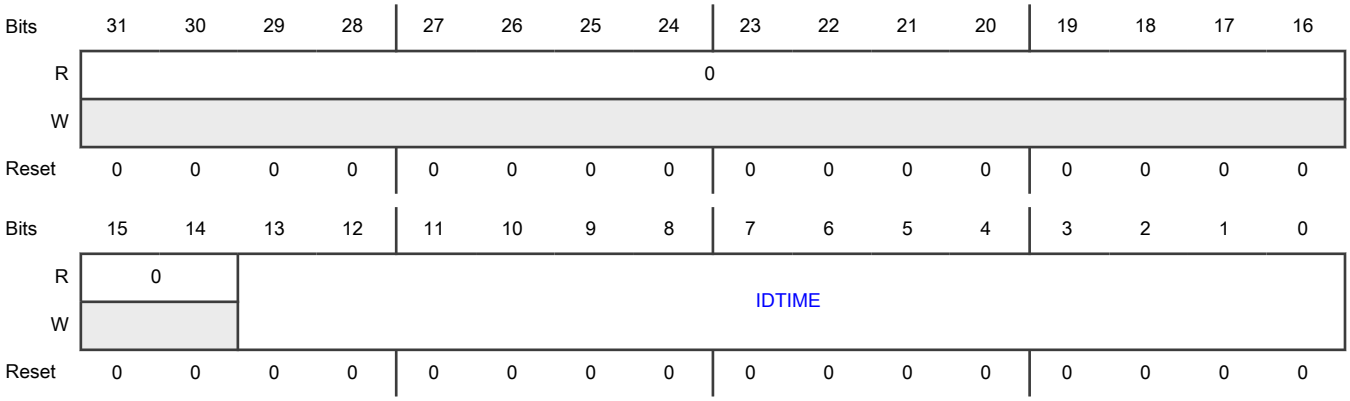
Instance	Register supported	Register not supported
LPUART_0	REIR	—
LPUART_1	REIR	—
LPUART_2	—	REIR
LPUART_3	—	REIR
LPUART_4	—	REIR
LPUART_5	—	REIR
LPUART_6	—	REIR
LPUART_7	—	REIR
LPUART_8	—	REIR
LPUART_9	—	REIR

Table continues on the next page...

Table continued from the previous page...

Instance	Register supported	Register not supported
LPUART_10	—	REIR
LPUART_11	—	REIR
LPUART_12	—	REIR
LPUART_13	—	REIR
LPUART_14	—	REIR
LPUART_15	—	REIR

Diagram



Fields

Field	Function
31-14 —	Reserved
13-0 IDTIME	Idle Time Configures the idle length in number of bits (baud rate) since the end of the last stop bit. This affects the behavior of the idle wake-up, STAT[IDLE], DATA[IDLINE], and STAT[RAF]. The minimum supported extended idle time is equal to one idle character. The extended idle feature is disabled when this field is 0.

77.6.18 Transmitter Extended Idle (TEIR)

Offset

Register	Offset
TEIR	4Ch

Function

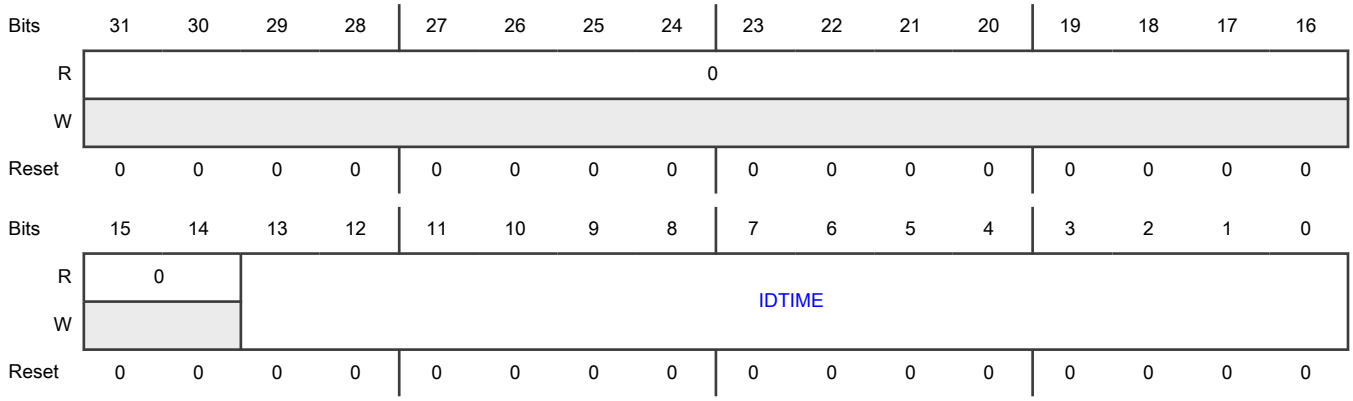
Configures the transmitter extended idle functionality. You must not change this value when the transmitter is enabled.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
LPUART_0	TEIR	—
LPUART_1	TEIR	—
LPUART_2	—	TEIR
LPUART_3	—	TEIR
LPUART_4	—	TEIR
LPUART_5	—	TEIR
LPUART_6	—	TEIR
LPUART_7	—	TEIR
LPUART_8	—	TEIR
LPUART_9	—	TEIR
LPUART_10	—	TEIR
LPUART_11	—	TEIR
LPUART_12	—	TEIR
LPUART_13	—	TEIR
LPUART_14	—	TEIR
LPUART_15	—	TEIR

Diagram



Fields

Field	Function
31-14 —	Reserved
13-0 IDTIME	Idle Time Configures the transmitter idle time in number of bits (baud rate) whenever an idle character is queued through the transmit FIFO. An idle character is not automatically queued whenever the transmitter is enabled. The minimum supported extended idle time equals one idle character. The extended idle feature is disabled when this field is 0.

77.6.19 Half Duplex Control (HDCR)

Offset

Register	Offset
HDCR	50h

Function

Provides control for half-duplex-related operations.

You can use this register instead of CTRL[LOOPS], CTRL[RSRC], and CTRL[TXDIR] functions, although you can use CTRL[LOOPS] to loop-back the transmitter outputs to the receiver.

NOTE

Each module instance supports a different number of registers.

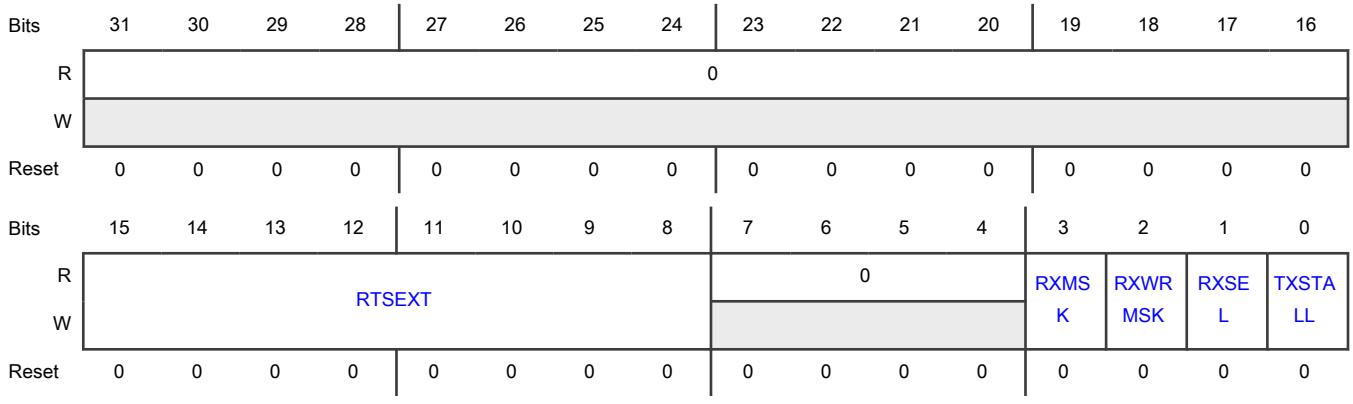
Instance	Register supported	Register not supported
LPUART_0	HDCR	—

Table continues on the next page...

Table continued from the previous page...

Instance	Register supported	Register not supported
LPUART_1	HDCR	—
LPUART_2	—	HDCR
LPUART_3	—	HDCR
LPUART_4	—	HDCR
LPUART_5	—	HDCR
LPUART_6	—	HDCR
LPUART_7	—	HDCR
LPUART_8	—	HDCR
LPUART_9	—	HDCR
LPUART_10	—	HDCR
LPUART_11	—	HDCR
LPUART_12	—	HDCR
LPUART_13	—	HDCR
LPUART_14	—	HDCR
LPUART_15	—	HDCR

Diagram



Fields

Field	Function
31-16 —	Reserved
15-8 RTSEXT	RTS Extended Specifies RTS extension. The transmit RTS_B remains asserted for (RTSEXT + 1) bit times after the end of the last stop bit. This applies even when the transmitter's RTS_B output is disabled and is only used internally to mask the receiver.
7-4 —	Reserved
3 RXMSK	Receive Mask Specifies whether the transmitter RTS_B masks the receive data pin. When enabled, the transmitter RTS_B masks the receive data pin. 0b - Do not mask 1b - Mask
2 RXWRMSK	Receive FIFO Write Mask Specifies whether the transmitter RTS_B masks writes to the receive FIFO. When enabled, the transmitter RTS_B masks receive FIFO writes, but the idle flag is not affected. 0b - Do not mask 1b - Mask
1 RXSEL	Receive Select Specifies the receive data pin. When enabled, the receive data is sourced from the TXD pin. 0b - RXD 1b - TXD
0 TXSTALL	Transmit Stall Specifies whether the transmitter becomes busy when the receiver is active. When enabled, the transmitter does not become busy or asserts transmitter RTS_B when the receiver is active (STAT[RAF] is 1). 0b - No effect 1b - Does not become busy

77.6.20 Timeout Control (TOCR)

Offset

Register	Offset
TOCR	58h

Function

Configures the behavior of the timeout logic. Timeouts 0 and 1 are used to monitor the receiver and timeouts 2 and 3 are used to monitor the transmitter.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
LPUART_0	TOCR	—
LPUART_1	TOCR	—
LPUART_2	—	TOCR
LPUART_3	—	TOCR
LPUART_4	—	TOCR
LPUART_5	—	TOCR
LPUART_6	—	TOCR
LPUART_7	—	TOCR
LPUART_8	—	TOCR
LPUART_9	—	TOCR
LPUART_10	—	TOCR
LPUART_11	—	TOCR
LPUART_12	—	TOCR
LPUART_13	—	TOCR
LPUART_14	—	TOCR
LPUART_15	—	TOCR

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				TOIE				0				TOEN			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-12 —	Reserved
11-8 TOIE	Timeout Interrupt Enable Enables the corresponding timeout flag to generate an interrupt.
7-4 —	Reserved
3-0 TOEN	Timeout Enable Enables the corresponding timeout counter.

77.6.21 Timeout Status (TOSR)

Offset

Register	Offset
TOSR	5Ch

Function

Indicates the status of the timeout logic. Timeouts 0 and 1 are used to monitor the receiver and timeouts 2 and 3 are used to monitor the transmitter.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
LPUART_0	TOSR	—
LPUART_1	TOSR	—
LPUART_2	—	TOSR
LPUART_3	—	TOSR
LPUART_4	—	TOSR
LPUART_5	—	TOSR
LPUART_6	—	TOSR
LPUART_7	—	TOSR
LPUART_8	—	TOSR
LPUART_9	—	TOSR
LPUART_10	—	TOSR
LPUART_11	—	TOSR
LPUART_12	—	TOSR
LPUART_13	—	TOSR
LPUART_14	—	TOSR
LPUART_15	—	TOSR

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				TOF				0				TOZ			
W					W1C											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1

Fields

Field	Function
31-12 —	Reserved
11-8 TOF	Timeout Flag Indicates whether the corresponding timeout occurred. The timeout counter is disabled when this field is 1. **NOTE** This field behaves differently for register reads and writes. When reading 0000b - Not occurred 0001b - Occurred When writing 0000b - No effect 0001b - Clear the flag
7-4 —	Reserved
3-0 TOZ	Timeout Zero Indicates whether the corresponding timeout counter equals 0.

77.6.22 Timeout N (TIMEOUT0 - TIMEOUT3)

Offset

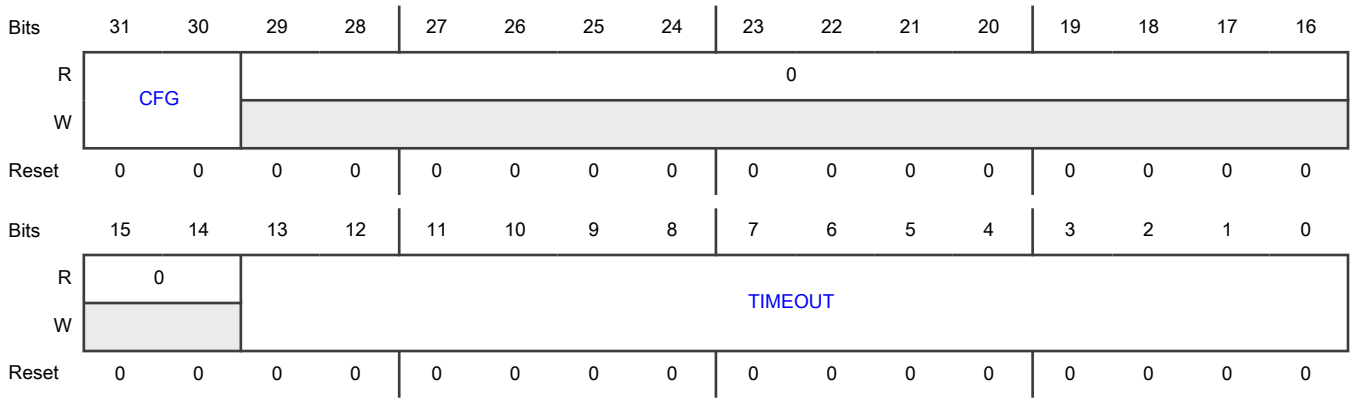
Register	Offset
TIMEOUT0	60h
TIMEOUT1	64h
TIMEOUT2	68h
TIMEOUT3	6Ch

Function
Configures the corresponding timeout counter and status field. Timeouts 0 and 1 are used to monitor the receiver and timeouts 2 and 3 are used to monitor the transmitter. You must write to this register only when the corresponding timeout is disabled or when the value of the corresponding timeout field is 1.

NOTE
Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
LPUART_0	TIMEOUT0–TIMEOUT3	—
LPUART_1	TIMEOUT0–TIMEOUT3	—
LPUART_2	—	TIMEOUT0–TIMEOUT3
LPUART_3	—	TIMEOUT0–TIMEOUT3
LPUART_4	—	TIMEOUT0–TIMEOUT3
LPUART_5	—	TIMEOUT0–TIMEOUT3
LPUART_6	—	TIMEOUT0–TIMEOUT3
LPUART_7	—	TIMEOUT0–TIMEOUT3
LPUART_8	—	TIMEOUT0–TIMEOUT3
LPUART_9	—	TIMEOUT0–TIMEOUT3
LPUART_10	—	TIMEOUT0–TIMEOUT3
LPUART_11	—	TIMEOUT0–TIMEOUT3
LPUART_12	—	TIMEOUT0–TIMEOUT3
LPUART_13	—	TIMEOUT0–TIMEOUT3
LPUART_14	—	TIMEOUT0–TIMEOUT3
LPUART_15	—	TIMEOUT0–TIMEOUT3

Diagram



Fields

Field	Function
31-30 CFG	Idle Configuration Configures the behavior of TIMEOUT[TIMEOUT]. 00b - Becomes 1 after timeout characters are received 01b - Becomes 1 when idle for timeout bit clocks 10b - Becomes 1 when idle for timeout bit clocks following the next character 11b - Becomes 1 when idle for at least timeout bit clocks, but a new character is detected before the extended idle timeout is reached
29-14 —	Reserved
13-0 TIMEOUT	Timeout Value Configures the timeout value.

77.6.23 Transmit Command Burst (TCBR0 - TCBR127)

Offset

For a = 0 to 127:

Register	Offset
TCBRa	200h + (a × 4h)

Function

Acts as an alias of Data (DATA), designed to support incrementing burst transfers to the transmit FIFO by a DMA controller, using aligned 8-bit, 16-bit, or 32-bit writes. The size of this register is 512 bytes:

- An aligned 32-bit write in this region pushes one entry into the transmit FIFO.
- An aligned 16-bit write in this region to TCBR*x*[15:0] pushes one entry into the transmit FIFO.
- An 8-bit write in this region to TCBR*x*[7:0] updates DATA[7:0], but does not push the data into the transmit FIFO.
- An 8-bit write in this region to TCBR*x*[15:8] pushes the data written to DATA[15:8] plus the previously written DATA[7:0] into the transmit FIFO.
- An 8-bit or 16-bit write in this region to TXBR*x*[31:16] is ignored.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
LPUART_0	TCBR0–TCBR127	—

Table continues on the next page...

Table continued from the previous page...

Instance	Register supported	Register not supported
LPUART_1	TCBR0–TCBR127	—
LPUART_2	—	TCBR0–TCBR127
LPUART_3	—	TCBR0–TCBR127
LPUART_4	—	TCBR0–TCBR127
LPUART_5	—	TCBR0–TCBR127
LPUART_6	—	TCBR0–TCBR127
LPUART_7	—	TCBR0–TCBR127
LPUART_8	—	TCBR0–TCBR127
LPUART_9	—	TCBR0–TCBR127
LPUART_10	—	TCBR0–TCBR127
LPUART_11	—	TCBR0–TCBR127
LPUART_12	—	TCBR0–TCBR127
LPUART_13	—	TCBR0–TCBR127
LPUART_14	—	TCBR0–TCBR127
LPUART_15	—	TCBR0–TCBR127

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	DATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-0 DATA	Data Enables writing data to Data (DATA).

77.6.24 Transmit Data Burst (TDBR0 - TDBR255)

Offset

For a = 0 to 255:

Register	Offset
TDBRa	400h + (a × 4h)

Function

Acts as an alias of Data (DATA), designed to support incrementing burst transfers to the transmit FIFO by a DMA controller using 8-bit, 16-bit, or 32-bit writes. The size of this register is 1024 bytes:

- An aligned 32-bit write in this region pushes four DATA byte entries into the transmit FIFO (DATA0 byte first). The register access is extended by three wait states.
- An aligned 16-bit write in this region to either half of a 32-bit word pushes two DATA byte entries into the transmit FIFO (DATA0 or DATA2 first). The register access is extended by one wait state.
- An 8-bit write in this region pushes one DATA byte entry into the transmit FIFO.

Byte writes to Data (DATA) use the contents of CTRL[R9T8] for the ninth data bit, CTRL[R8T9] for the tenth data bit, and zero extend DATA[FRETSC].

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
LPUART_0	TDBR0–TDBR255	—
LPUART_1	TDBR0–TDBR255	—
LPUART_2	—	TDBR0–TDBR255
LPUART_3	—	TDBR0–TDBR255
LPUART_4	—	TDBR0–TDBR255
LPUART_5	—	TDBR0–TDBR255

Table continues on the next page...

Table continued from the previous page...

Instance	Register supported	Register not supported
LPUART_6	—	TDBR0–TDBR255
LPUART_7	—	TDBR0–TDBR255
LPUART_8	—	TDBR0–TDBR255
LPUART_9	—	TDBR0–TDBR255
LPUART_10	—	TDBR0–TDBR255
LPUART_11	—	TDBR0–TDBR255
LPUART_12	—	TDBR0–TDBR255
LPUART_13	—	TDBR0–TDBR255
LPUART_14	—	TDBR0–TDBR255
LPUART_15	—	TDBR0–TDBR255

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	DATA3								DATA2							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	DATA1								DATA0							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 DATA3	Data3 Enables writing of data to Data (DATA).
23-16 DATA2	Data2 Enables writing of data to Data (DATA).
15-8	Data1

Table continues on the next page...

Table continued from the previous page...

Field	Function
DATA1	Enables writing of data to Data (DATA).
7-0 DATA0	Data0 Enables writing of data to Data (DATA).

77.7 Glossary

Baud rate Number of bits per second that LPUART transmits or receives

Break character Break character is generated when the transmitter is holding the data line at the space level for at least one character time

Oversampling Number of times the receive circuitry samples the receive input per baud period (that is, per data bit)

Chapter 78
Quad Serial Peripheral Interface (QuadSPI) for S32K322, S32K342, S32K341, S32K314, S32K324, and S32K344

78.1 Chip-specific QuadSPI information

78.1.1 QuadSPI configuration

Table 693. QuadSPI instances

Instance	S32K322/S32K342/S32K341/S32K314/S32K324/S32K344	S32K310/S32K311/S32K312
QuadSPI	Yes	No

Table 694. QuadSPI configuration details

Configuration	S32K312/S32K322/S32K342/S32K344/ S32K324/S32K314
QuadSPI Tx FIFO size	32 words
QuadSPI Rx FIFO size	32 words
Look Up Table Size	4 words
Rx Buffer	256 Bytes

For supported data rates, see device Datasheet.

NOTE

- Boot from QuadSPI is not supported but execution from external memory is supported.
- SFP is not supported in S32K344, S32K324, S32K314, S32K342, and S32K341.
- SoC generate secure access from all CM7 core to QuadSPI.

Table 695. Features supported

Feature	S32K322/S32K342/S32K341/S32K344/S32K324/S32K314
AHB Write	No
Data learning feature	No
DLL	No
OTFAD (On-the-fly-AES-decryption engine)	No
DDR mode	No
HyperRAM	No
HyperFlash	No
External DQS (Data Strobe)	No
Boot from QuadSPI interface	No

The following block diagram depicts the QuadSPI Flash A Interface.

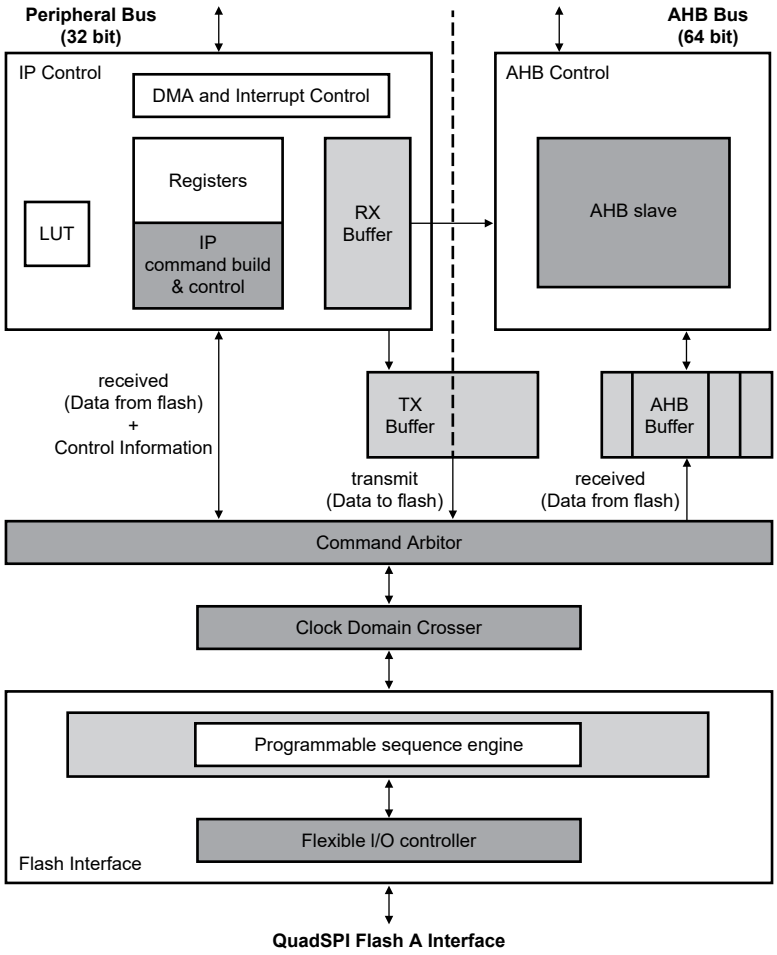


Figure 496. Block diagram

78.1.2 Supported read modes

The table below provides an overview of the QuadSPI read modes.

Table 696. QuadSPI read modes

Read modes		SDR support (QuadSPI_MCR [DDR_EN]=0)	QuadSPI_MCR [DQS_EN]	QuadSPI_MCR [DQS_FA_SEL]	Data learning support	DLL
DQS sampling method	Pad loopback	Yes	1	01	No	No

78.1.3 QuadSPI initialization sequence

Following initialization sequence should be followed for proper QuadSPI operation:

- Enable QuadSPI module by MC_ME peripheral clock enable (register PRTNx_COFBy_CLKEN present within MC_ME). Refer MC_ME chapter for peripheral mapping.
- Configure the SIUL2 registers MSCR[OBE] as 1 and MSCR[SSS] as 0 for QuadSPI_SCKFA pin.

- Initialize QuadSPI SCKFA by writing a sequence of 1010 to the SIUL2 register GPDO[PDO_a] for QuadSPI_SCKFA pin.
- Configure the SIUL2 register MSCR[OBE] back as 0 for the pins.
- Initiate a dummy flash read to reset all DQS flops by itself and any crossing from DQS domain to IPS/AHB is taken care by CDC logic.
- Initiate a peripheral software reset to QuadSPI controller by writing to QuadSPI controller's MCR.
- Post this initialization sequence, the QuadSPI will work in intended deterministic manner.

NOTE

QuadSPI initialization is to be done before using QuadSPI after each functional reset.

78.1.4 Pad clock loopback

This chip supports pad clock loopback. The QuadSPI can be configured to use clock loopback to sample input data. SCK is delayed by the SCK pin output delay, plus the SCK pin input delay using pad loopback, and is configured by setting QuadSPI config registers SOCCR[SOCCFG] and MCR[DQS_FA_SEL]. Enabling the loopback version of SCK can improve the setup time of the input data from the Flash.

For details of these register, see QuadSPI register descriptions.

78.1.5 QuadSPI SOC Configuration register SOCCR[SOCCFG] implementation

The QuadSPI SOC Configuration register QuadSPI_SOCCR[SOCCFG] register is used to control dummy loopback pads and obe_pull_timing_relax_b. Below is the description of it's bits:

Table 697. SOCCR[SOCCFG] implementation

Bit	Description
Bit[0]	obe_pull_timing_relax_b : enables the timing relaxation by pulling obe for pad 1 for half cycle, if 0 then enabled else disabled.
Bit[1]	ibe of QSPIA_SCK_DUMMY pad. 0: Disable input receiver 1: Enable input receiver.
Bit[2]	obe of QSPIA_SCK_DUMMY pad. 0: Disable output driver 1: Enable output driver.
Bit[3]	dse of QSPIA_SCK_DUMMY pad. 0: Disable drive strength 1: Enable drive strength.
Bit[4]	pue of QSPIA_SCK_DUMMY pad. 0: Disable internal pullup or pulldown resistor 1: Enable internal pullup or pulldown resistor.
Bit[5]	pus of QSPIA_SCK_DUMMYpad. 0: Enable internal pulldown resistor if pue is set 1: Enable internal pullup resistor if pue is set.
Bit[6]	sre of QSPIA_SCK_DUMMY pad. 0: Disable slew rate control 1: Enable slew rate control.
Bit[31:7]	Reserved

To Enable the Quadspi dummy PAD loopback use following settings

For Flash-A: MCR[DQS_FB_SEL] = 0x2

SOCCR[SOCCFG] = 0x0000000E (ibe=1, obe=1, dse=1 and sre=0)

NOTE

S32K3xx SoCs does not support dual die flashes. Hence, the signal PCSFA2 from QuadSPI is not used.

78.1.6 QuadSPI AHB Buffer write access control

In S32K344/S32K324/S32K314 the QuadSPI AHB buffer does not support writes. A write to QuadSPI AHB buffer results in error response from QuadSPI.

QuadSPI might behave unexpectedly in case if its AHB buffer is written with QuadSPI MCR[MDIS] =1. In cases wherein QuadSPI is disabled with above MDIS control bit, XRDC must also be appropriately configured to block the accesses to QSPI AHB buffer and indicate an error response.

78.2 Introduction

The QuadSPI module acts as an interface to a single serial flash memory device, with up to four bidirectional data lines.

78.2.1 Features

QuadSPI supports the following features:

- Flexible sequence engine to support various flash memory vendor devices. As there is no specific standard, the module supports various kinds of flash memories from different vendors. See Serial flash memory devices for example sequences.
- Single, dual, and quad modes of operation supported for Quad flash memories
- Support for HyperFlash memory
- AHB master to read RX buffer data through AMBA AHB (64-bit width interface) or IPS registers space (32-bit access) and fill TX buffer via IPS registers space (32-bit access)
  - AHB master can be a DMA with a configurable inner loop size
- Multi-master accesses are allowed
  - Flexible and configurable buffer for each master—total available buffer size is 256 bytes.
- All AHB accesses to flash/RAM memory devices are directly memory mapped to the chip system memory
- Programmable sequence engine to cater to future command/protocol changes and ability to support all existing vendor commands and operations. The software needs to select the corresponding sequence according to the connected flash memory device.
  - Support for 3-byte and 4-byte addressing

78.2.2 RX buffer push event

To add the valid entries into the RX buffer

By default, each buffer push event adds two entries to the RX buffer because the interface to the serial clock domain is 64 bits in width. Depending on the number of bytes read from the serial flash memory device, it is possible for the very last buffer push event that only one entry is added.

RBSR[RDBFL] is incremented by the number of entries added to the RX buffer.

78.2.3 RX buffer POP event

To remove valid entries from the RX buffer

Each buffer POP event removes (RBCT[WMRK] + 1) valid entries from the buffer. BSR[RDBFL] is decremented by the same number and RBSR[RDCTR] is incremented accordingly.

78.2.4 Block diagram

The following figure shows a block diagram of the QuadSPI module.

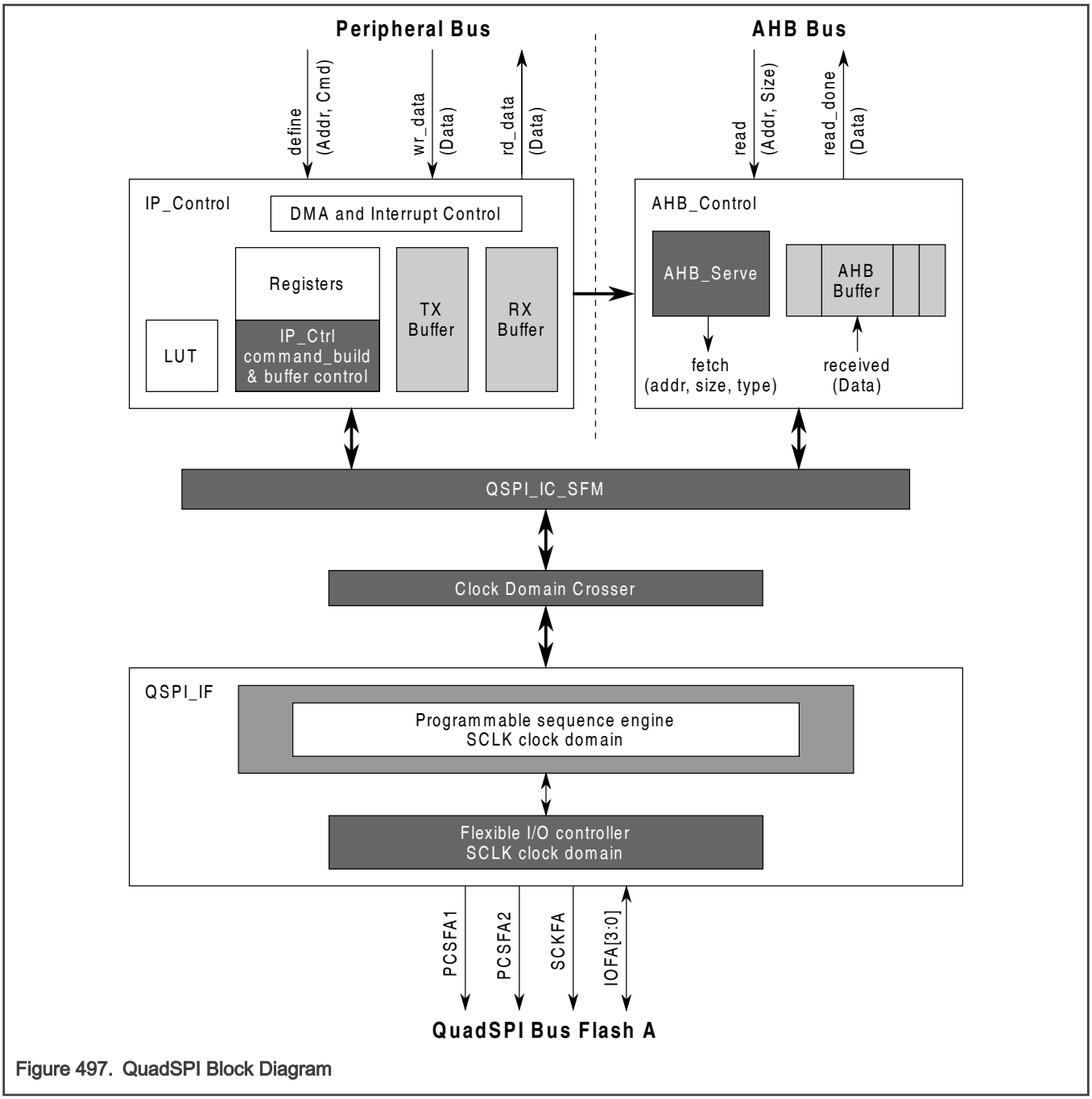


Figure 497. QuadSPI Block Diagram

78.2.5 QuadSPI modes of operation

QuadSPI supports the following modes of operation:

- Normal mode: You can use this mode for write or read accesses to an external serial flash memory device. See Normal mode for details.
  - Serial flash memory write: You can program data into the flash memory through the IP interface only. See Flash memory programming for details.
  - Serial flash memory read: Read the contents of the serial flash memory device. Two separate read channels are available through the RX buffer and AHB buffer. See Flash memory read for details.

- Module Disable mode: You can use this mode for disabling serial flash memory clock and AHB command. The clock to the non-memory mapped logic in QuadSPI can be stopped in the Module Disable mode. The module enters the mode by setting MCR[MDIS].

78.3 External signal description

This section provides the external signal information for the QuadSPI module.

The following table lists the external signals belonging to the module in conjunction with the different modes of operation.

Table 698. Signal properties

Signal name	Function	Direction	Description
PCSFA1	Peripheral Chip Select Flash Memory A1	O	This signal is the chip select for the serial flash memory device A1 that represents the first of the two flash memory devices that share IOFA.
PCSFA2	Peripheral Chip Select Flash Memory A2	O	This signal is the chip select for the serial flash memory device A2 that represents the the second of the two flash memory devices that share IOFA.
SCKFA	Serial Clock Flash Memory A	O	This signal is the serial clock output to the serial flash memory device A.
IOFA[3:0]	Serial I/O Flash memory A	I/O	These signals are the data I/O lines to/from the serial flash memory device A. See Driving external signals for details about the signal drive and timing behavior. Note that the signal pins of the serial flash memory device may change their function according to the SFM Command executed, leaving them as control inputs when single and dual instructions are executed. The module supports driving these inputs to dedicated values. In single I/O mode, QuadSPI drives data on IOFA[0] and expects data on IOFA[1].

NOTE

Please refer to chip specific information to check the configuration of QuadSPI block.

78.3.1 Driving external signals

Single/dual/quad instructions

Depending on the serial flash memory device connected to the QuadSPI module, there are instructions using a different number of data lines:

- Single pad: Single line I/O with one data out and one data in line to/from the serial flash memory device
- Dual pad: Dual line I/O with two bidirectional I/O lines, driven alternatively by the serial flash memory device or the QuadSPI module
- Quad pad: Quad line I/O with four bidirectional I/O lines, driven alternatively by the serial flash memory device or the QuadSPImodule

The different phases of the serial flash memory access scheme are shown in the following figure.

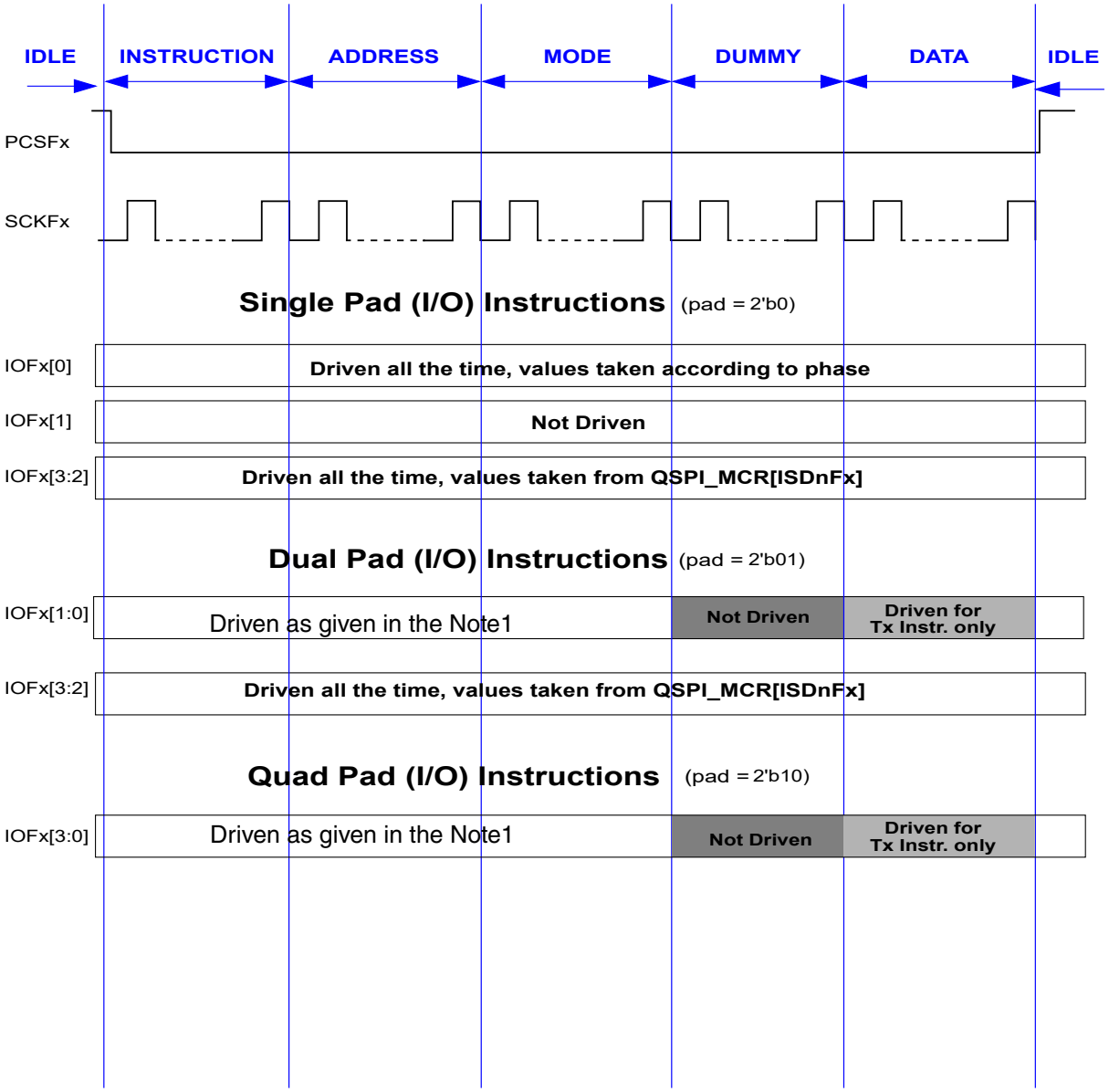


Figure 498. Serial flash memory access scheme

Note1:The IOs are driven from QuadSPI as per the number of pads configured for ongoing phase.

Note:The lines status will change based on command mode in case of instruction, address and mode phases. It can be either 1,2 or 4 lines

Following are the different phases and the I/O driving characteristics of the QuadSPI module:

- Idle: Serial flash memory device not selected – no interaction with the serial flash memory device and the IOFx signals are driven.
- Instruction: Serial flash memory device selected – the instruction is sent to the serial device and all the IOFx signals are driven.
- Address: Serial flash memory address is sent to the device – all the IOFx signals are driven and this phase is not applicable for all SFM commands.
- Mode: Mode bytes are sent to the serial flash memory device – all the IOFx signals are driven and this phase is not applicable for all SFM commands.

- Dummy: Dummy clocks are provided to the serial flash memory device. See Figure 498 for the IOFx signals driven. The actual data lines required for the SFM command executed are not driven for data read commands.

NOTE

— This phase is not applicable for all the SFM commands.

— All read commands in Dual pad, Quad pad modes must use dummy phase before read phase. Note that this restriction is not applicable to Single-pad mode.

- Data: Serial flash memory data are sent to or received from the serial flash memory device. See the preceding figure for the IOFx signals driven. The actual data lines required for the SFM command executed are not driven for data read commands.

NOTE

This phase is not applicable for all the SFM commands.

The PCSFx and SCKFx signals are driven permanently throughout all the phases. In the individual flash memory mode, this applies to the selected flash memory device.

Access to a single, individual serial flash memory device

See Serial flash memory access schemes for details.

Read access to two serial flash memory devices attached to the QuadSPI module in parallel. See Serial flash memory access schemes for details.

78.4 Functional description

This section provides a functional description of the QuadSPI module.

78.4.1 Serial flash memory access schemes

In the individual flash memory mode, all supported commands are available.

78.4.2 Normal mode

This mode allows communication with an external serial flash memory device. Compared to the standard SPI protocol, this communication method uses up to four bidirectional data lines operating at high-data rates. The communication to the external serial flash memory device consists of an instruction code and optional address, mode, dummy, and data transfers. The flexible programmable core engine described below is immune to a wide variety of command or protocol differences in the serial flash memory devices provided by various flash memory vendors.

78.4.2.1 Programmable sequence engine

The core of the QuadSPI module is a programmable sequence engine that works on "instruction-operand" pairs. The core controller executes each programmed instruction sequentially. The complete list of instructions and the corresponding operands are provided in the following table.

Table 699. Instruction set

Instruction	Instruction encoding	Pins	Operand	Action on serial flash memories
CMD	1d	N={0,1,2}d 0d - One pad 1d - Two pads 2d - Four pads	8-bit command value	Provides the serial flash memory with the SFM command operand (Encoded) on the number of pads specified in STR mode.
ADDR	2d		Number of address bits	Provide the serial flash memory with address cycles according to the operand on the number of pads specified

Table continues on the next page...

Table 699. Instruction set (continued)

Instruction	Instruction encoding	Pins	Operand	Action on serial flash memories
			to be sent (for example, 24d => 24 address bits required)	The actual address to be provided is derived from the incoming address in case of AHB-initiated transactions and the value of SFAR in case of IPS-initiated transactions.
DUMMY	3d		Number of dummy clock cycles (should be <= 64 and > 2 cycles)	Provide the serial flash memory with dummy cycles according the operand The PAD information defines the number of pads in input mode. For example, one pad implies that pad 1 is not driven, rest all are driven.
MODE	4d		8-bit mode value	Provide the serial flash memory with 8-bit operand on the number of pads specified
MODE2	5d	N={0,1}d	2-bit mode value	Provide the serial flash memory with 2-bit operand on the number of pads[1] specified
MODE4	6d	N={0,1,2}d	4-bit mode value	Provide the serial flash memory with 4-bit operand on the number of pads[2] specified
READ	7d	N={0,1,2}d 0d - One pad 1d - Two pads 2d - Four pads	Read data size in bytes (for AHB transactions, your application should ensure that data size is a multiple of 8 bytes)	Read data from flash memory on the number of pads specified The data size can be overwritten by writing to the ADATSZ field of the BUFxCR registers for AHB-initiated transactions and to the IDATSZ field of the IP Configuration Register (IPCR) for IPS-initiated transactions.
WRITE	8		Write data size in bytes	Write data on the number of pads specified The data size can be overwritten by writing to the IDATSZ field of IP Configuration Register (IPCR).
JMP_ON_CS[3]	9d	NA	Instruction number	Every time the CS is deasserted, jump to the instruction pointed to by the operand. This instruction allows the programmer to specify the behavior of the controller when a new read transaction is initiated following a CS deassertion.
STOP[3]	0d	NA	NA	Stop execution; deassert CS

1. For a one-pad instruction, MODE2 takes two serial flash memory clock cycles on the flash memory interface.
2. For a one-pad instruction, MODE4 takes four serial flash memory clock cycles on the flash memory interface. For a four-pad instruction, MODE4 takes one serial flash memory clock cycle on the flash memory interface.
3. Sequence ending with this instruction must have all remaining bits as 0s after it.

The programmable sequence engine allows you to configure the QuadSPI module according to the serial flash memory connected on board. This flexible structure is compatible with new command or protocol changes from different vendors.

78.4.2.2 Flexible read xAHB buffers

To reduce the latency of the reads for AHB masters, the data read from serial flash memory is buffered in flexible AHB buffers. There are four such flexible buffers. The size of each of these buffers is configurable with the minimum size being 0 bytes and maximum size being the size of the complete buffer instantiated (256 bytes). The size of buffer 0 ranges from 0 to BUF0IND. The size of buffer 1 ranges from BUF0IND to BUF1IND, buffer2 from BUF1IND to BUF2IND and, buffer 3 ranges from BUF2IND to the size of the complete buffer (256 bytes).

Each flexible AHB buffer is associated with the following:

- An AHB master: Optionally, buffer3 may be configured as an "all master" buffer by writing 1 to BUF3CR[ALLMST]. When buffer3 is configured in such a way, any access from a master not associated with any other buffer is routed to buffer3.
- A datasize field representing the amount of data to be fetched from the flash memory on every missed access.

The master ID of every incoming request is checked and the data is returned or fetched into the corresponding associated buffer. See the chip-specific QuadSPI information for details about master IDs and their corresponding components. Every missed access to the buffer causes the controller to clear the buffer and fetch the BUFxCR[ADATSZ] amount of data from the serial flash memory. As such, you need not configure the buffer size to be greater than ADATSZ because the locations greater than ADATSZ are never used. For any AHB access, the sequence pointed to by BFGENCR[SEQID] is used for the initiated flash memory transaction. The data is returned to the master as soon as the requested amount is read from the serial flash memory. The controller; however, continues to prefetch the rest of the data in anticipation of a next consecutive request. See Figure 499 that shows flexible AHB buffers.

BFGENCR[SEQID] points to an index of the LUT. See LUT for details.

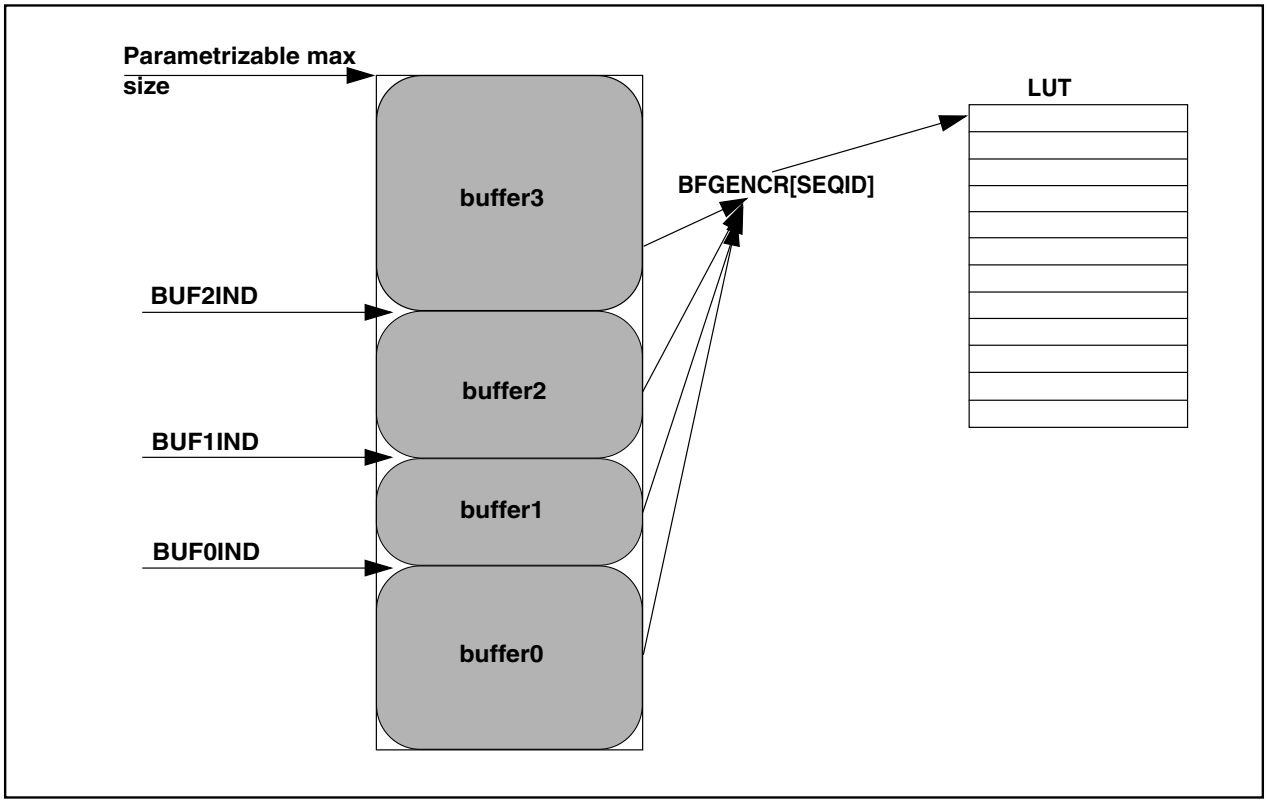


Figure 499. Flexible read AHB buffers

78.4.2.3 Abort mechanism during AHB read

Any ongoing read transaction is aborted if a request from the same master arrives for a location other than the location at which the transaction is going on. The abort can happen at any point of time.

78.4.2.4 HBURST support with AHB read

QuadSPI controller supports HBURST and HSIZE on the AHB read interface. HBURST indicates if the transfer forms part of a burst. Four, eight, and sixteen beat bursts are supported and the bursts might either be incrementing or wrapping. HSIZE indicates the size of the transfer, and supports 8-, 16-, 32-, and 64-bit data sizes. In case of WRAP accesses, QuadSPI generates aligned accesses to serial flash memory if there is no buffer hit for any incoming, non-sequential AHB read access. In case there is a buffer hit, the incoming address in the haddr line is latched as it is. If the total burst size is more than the data prefetch size, an error response is generated and the value of FR[AIBSEF] is configured as 1. The data prefetch size can be defined by BUFxCR[ADATSZ] or data size mentioned in the sequence pointed to by the SEQID field when ADATSZ is programmed as 0. In case of wrap burst, data prefetch size must be greater than or equal to the wrap burst size + 32 bytes. A few examples are shown in the figure below:

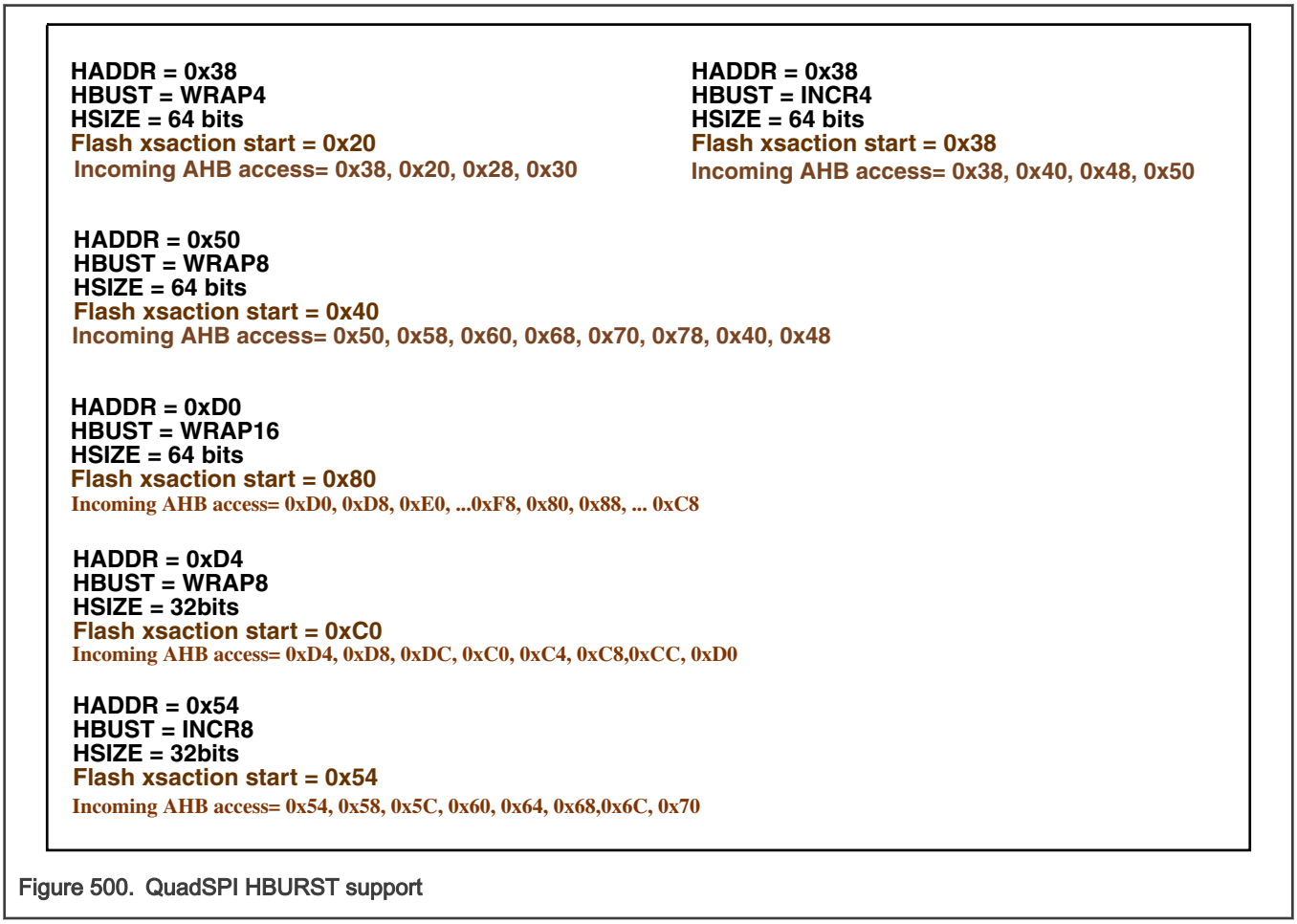


Figure 500. QuadSPI HBURST support

NOTE

The software must take care that the prefetch size should never be set less than the minimum data needed by any external interface to start processing.

NOTE

Whenever a core accesses QuadSPI memory with cache enabled, the prefetch size must be configured as equal or more than the cache line size; otherwise, FR[AIBSEF] error appears.

78.4.2.5 LUT

The LUT consists of a number of pre-programmed sequences. Each sequence is basically a sequence of instruction-operand pairs, which when executed sequentially, generate a valid serial flash memory transaction. Each sequence can have a maximum

of 10 instruction-operand pairs. The LUT can hold a maximum of sequences. The figure below shows the basic structure of the sequence in the LUT.

At reset, the index 0 of the LUT[0..4] is programmed with a basic read sequence as described in Read sequence. After reset, the complete LUT may be reprogrammed according to the chip connected on board. To protect its contents, during a code runover, the LUT might be locked, after which a write to the LUT will not be successful until it has been unlocked again. The key for locking or unlocking the LUT is 5AF05AF0h, and the associated processes are as follows:

Locking the LUT

1. Write the key 5AF05AF0h into the LUT Key Register (LUTKEY).
2. Write 0b01 to the LUT Lock Configuration Register (LCKCR). Note that this IPS transaction should immediately follow the above IPS transaction (no other IPS transaction can be issued). A successful write to this register locks the LUT.

Unlocking the LUT

1. Write the key 5AF05AF0h into the LUT Key Register (LUTKEY).
2. Write 0b10 to the LUT Lock Configuration Register (LCKCR). Note that this IPS transaction should immediately follow the above IPS transaction (no other IPS transaction can be issued in between). A successful write to this register unlocks the LUT.

The lock status of the LUT can be read from the LCKCR[UNLOCK] and LCKCR[LOCK] fields.

Some example sequences are defined in Example sequences. After reset the instruction sequence 0 is populated with the default read sequence shown in the table below.

Table 700. Read sequence

Instruction	Pad	Operand	Comment
CMD	0h	3h	Read data byte command on one pad
ADDR	0h	18h	24 address bits to be sent on one pad
READ	0h	8h	Read 64 bits
JMP_ON_CS	0h	0h	Jump to instruction 0 (CMD)

NOTE

If DLL is disabled then JMP_ON_CS or STOP instruction can be used else only STOP instruction can be used.

78.4.2.6 Issuing SFM commands

Each access to the external device follows this sequence:

1. You must pre-populate the LUT with the serial flash memory command sequences that are required for the flash memory device being used.
2. The module executes the instructions in this sequence one by one. The transaction starts and the module configures the value of SR[BUSY].
3. Communication with the external serial flash memory device starts and the transaction executes.
4. After the transaction is complete (all transmit and receive operations with the external serial flash memory device are complete), the module resets SR[BUSY]. In case of an IP command, FR[TFF] is asserted.

For details, see Flash memory programming and Flash memory read.

You can trigger the processing of SFM commands in the QuadSPI module in one of the following ways:

Using IP commands

For IP commands, the required components need to be written into the following registers and in this sequence:

1. Write the serial flash memory address to be used as provided in the Serial Flash Address Register (SFAR). For IP commands not related to specific addresses, the base address of the related flash memory needs to be programmed. For example, for an instruction which does not require an address (that is, write enable instruction), the SFAR should be programmed with the base address of the memory the command is to be sent to.
2. Write the sequence ID and data size details in the IP Configuration Register (IPCR).
3. Note that writing a value to IPCR[SEQID] must be the last step of the sequence. It is possible to combine all the fields of the IPCR into one single write. See IP Configuration Register (IPCR) for details.

Using AHB commands

Any AHB memory-mapped access is routed to one of the buffers depending on the master ID of the request. If the access is a "miss," a new serial flash memory transaction is initiated. The transaction is based on the sequence pointed to by BFGENCR[SEQID] as described in Flexible read xAHB buffers.

An AHB access is considered memory mapped when the access is to the memory-mapped serial flash memories, as described in Memory Mapped Serial Flash Data - Individual Flash mode on Flash memory A.

78.4.2.7 Flash memory programming

In all NOR Flash devices memory sector to be written needs to be erased first. The programming sequence is then initiated in the following way:

1. Check that SR[BUSY] is de-asserted or the value of the BUSY field is 0, also, check that the TX buffer is empty. If you need to discard the data present in the TX buffer (SR[TXNE]) then the TX buffer must be cleared by writing 1 to MCR[CLR_TXF].
2. Program the address related to the command in SFAR.
3. Provide initial data for the program command into the circular buffer through the TBDR. At least one words of data must be written into the TX buffer up to a maximum of 32 entries.
4. Program the IPCR to trigger the command. IPCR[SEQID] should point to an index of the LUT that has the flash memory program sequence pre-programmed. Write an appropriate value to IPCR[IDATSZ] to denote the size of the write in bytes.
5. Repeat step 3, depending on the amount of data required, until all of the required data is written to the TBDR. SR[TXFULL] can be used to check if the buffer is ready to receive more data. At any time, TBSR[TRCTR] can be read to check how many words have been written into the TX buffer.

After writing to IPCR[SEQID] (see step 4), the module starts executing the programmed sequence. The software ensures that the correct sequence is programmed into the LUT in accordance with the flash memory connected to the module. The data is fetched from the TX buffer. It consists of 32 entries of 32-bit sizes and is organized as a circular FIFO, the read pointer for which is incremented after each fetch. When all the data is transmitted, the QuadSPI module returns from the busy state to the idle state. However, this is not true for the external device because the internal programming is still ongoing. You may monitor the relevant status information available from the serial flash memory device and ensure that the programming is done appropriately.

78.4.2.8 Flash memory read

Host access to the data stored in the external serial flash memory device is performed in two steps. First, the data must be read into the internal buffers and in the second step, these internal buffers can be read by the host.

Reading serial flash memory data into the QuadSPImodule internal buffers

A read access to the external serial flash memory device can be triggered in two different ways:

- IP command read: For reading flash memory data into the RX buffer, you must provide the correct sequence ID in IPCR[SEQID]. The sequence ID points to a sequence in the LUT. The software needs to ensure that a correct read sequence is programmed in the LUT in accordance with the serial flash memory device connected on board. You must program the SFAR , and IPCRs. All available read commands supported by the external serial flash memory are possible.

  Optionally, it is possible to clear the RX buffer pointer prior to triggering the IP command by writing a 1 to MCR[CLR_RXF]. This will invalidate the data currently present in the RX buffer and any new read data will overwrite the old one.

Using these inputs, the complete transaction is built when IPCR[SEQID] is written to. The transaction related to the read access starts and the requested number of bytes is fetched from the external serial flash memory device into the RX buffer. As the read access is triggered by an IP command, the value of both SR[IP_ACC] and SR[BUSY] is set to 1.. A count of the number of entries currently in the Rx buffer can be obtained from RBSR[RDBFL].

Communication with the external serial flash memory stops if the specified number of bytes are read (on successful completion of the transaction).

- AHB command read: For reading flash memory data into the AHB buffer, you must:
  - Set up a read access by a master to the address range in the system memory map, which the external serial flash memory devices are mapped to.
  -
  - Program the buffer registers corresponding to the AHB master initiating the request.
  - Provide the correct sequence ID in the BFGENCR. The software ensures that a correct read sequence is programmed in the LUT in accordance with the serial flash memory device connected on board. Flash memory device selection and access mode are determined by the address accessed in the AHB address space associated with the QuadSPI module (see Memory-mapped serial flash memory data—individual flash memory mode on flash memory A

  On each AHB read access to the memory mapped area, the valid data in the AHB buffer is checked against the address requested in the actual read. When the AHB read request cannot be served from the content of the AHB buffer, the buffer is flushed and the controller executes the sequence pointed to by the sequence ID. The requested number of buffer entries defined in BUFxCR[ADATSZ] is then fetched from the external serial flash memory device into the internal AHB buffer. As the read access is triggered through the AHB bus, the value of SR[AHB_ACC] is set, driving SR[BUSY] in turn, until the transaction is complete. Communication with the external serial flash memory stops when the specified number of entries is filled.

Data transfer from the QuadSPI module internal buffers

The data read out from the external serial flash memory device by the QuadSPI module is stored in the internal buffers. The means of accessing the data from the buffer differs depending on which buffer the data is loaded to. See Block diagram for details on the two available buffers, the RX buffer and the AHB buffer, in this module. The buffer appears transparent to you and is non-memory mapped. See the "Flexible AHB Buffer" section for details.

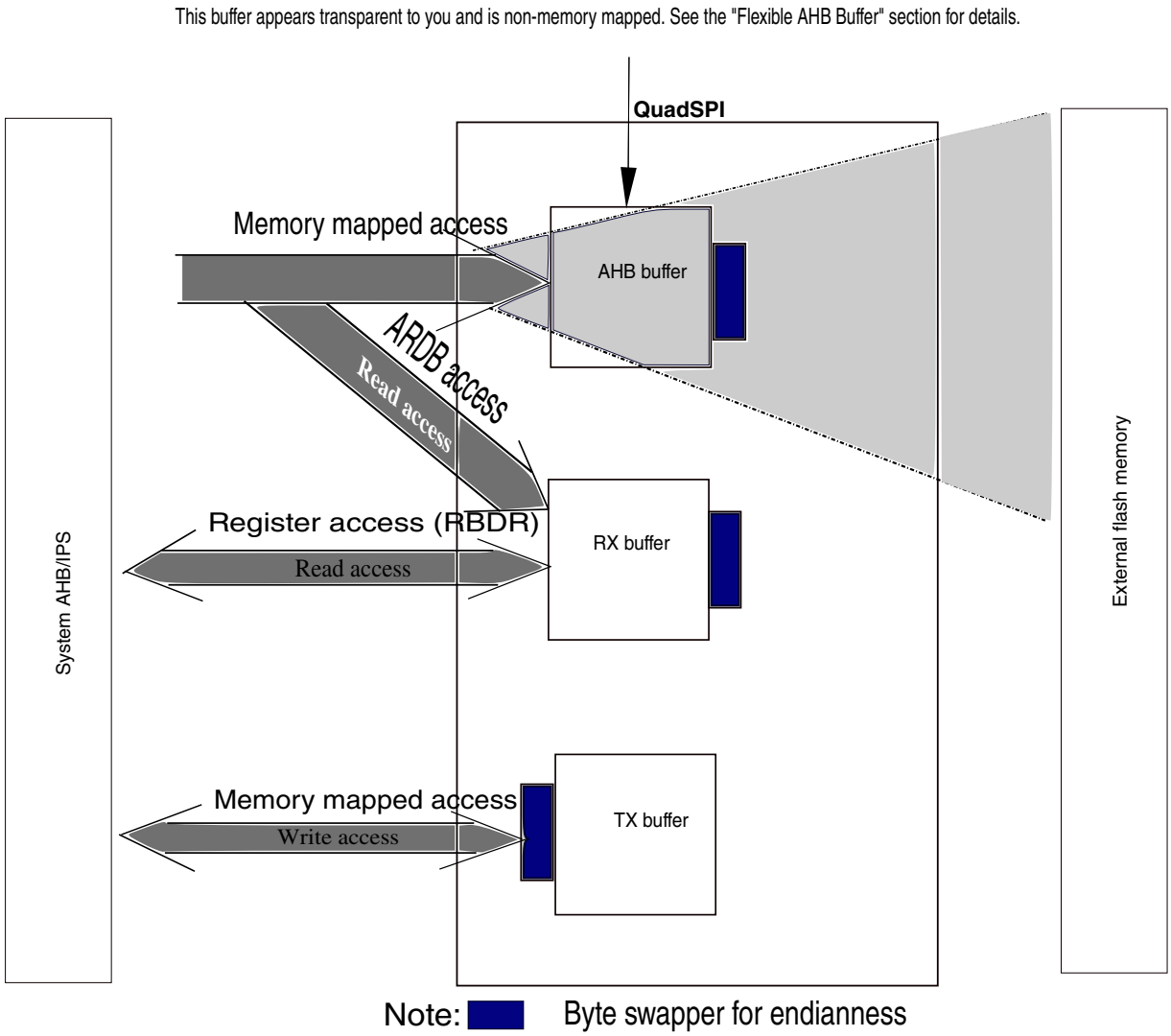


Figure 501. QuadSPI memory map

The RX buffer is implemented as FIFO of depth entries of 4 bytes. Its content is accessible in two different address areas, both referring to identical data and the same physical memory:

- In the IPS address space in the area associated with RX Buffer Data Register (RBDR0 - RBDR63).
- In the AHB address space in the area associated with AHB RX Data Buffer Register (ARDB0 - ARDB127).Two successive entries are accessed with one single 64-bit AHB read operation.

The RX buffer operation can be summarized as follows:

- RBCT[WMRK] determines at which fill level SR[RXWE] is asserted and how many entries are removed from the RX buffer on each buffer POP operation.
- SR[RXWE] indicates that the configured number of data entries is available in the RX buffer and RBSR[RDBFL] indicates how many valid entries are available in total.
- The first entry (RBDR0 or ARDB0) always corresponds to the first valid entry in the RX buffer.

For details, see RX Buffer Data Register (RBDR0 - RBDR63) and AHB RX Data Buffer Register (ARDB0 - ARDB127).

- Flag-based data read of the RX buffer is performed by polling SR[RXWE]. When it is asserted, the valid entries can be read either via the IPS address space (RBDRn) or the AHB address space (ARDBn). A buffer POP operation must be

triggered by the application by writing a 1 to FR[RBDF]. This automatically updates the FIFO to point to the next entry as defined by RBCT[WMRK]. For example, if WMRK is set to 3, then the buffer discards 16 bytes of data.

- DMA-controlled data read of the RX buffer is performed by using the DMA module. The application must ensure that the DMA controller of the related chip is programmed appropriately, as described in DMA usage.
- DMA-controlled read out is triggered fully automatically by the assertion of SR[RXWE]. The related buffer POP operation is also handled completely inside the QuadSPI module. As in the case explained here, accessing the RX buffer content either on RBDRn or ARDBn related addresses is equivalent.
- AHB buffer data read via memory-mapped access: This kind of access is performed by reading one of the addresses assigned to the external serial flash memory device(s) within the range specified in Table 722. If this is not the case, a memory-mapped AHB command read is triggered as described above. If the requested data is already available in the AHB buffer, it is provided directly to the host.

When an AHB access is made to the flash memory mapped address, the data is fetched and returned to the AHB interface. The AHB interface is stalled until the data is fetched. As soon as the data from the requested address is read by the QuadSPI module, the AHB read access is served. Therefore, it is possible to run sequential reads from the AHB buffer at arbitrary speed without the need to monitor any information about the availability of the data. Nevertheless, this access scheme stalls the AHB bus for the time required to read the data from the serial flash memory device. If you know that the access is sequential, a better way is to have a prefetch enabled by programming the value of BUFxCR[ADATSZ] so that the data is fetched into the buffer before the next sequential AHB access.

As long as the host restricts its accesses to the data present in the buffer and to the data currently fetched from the serial flash memory, it is possible to run the host read from the AHB buffer simultaneously with the serial flash memory read into the AHB buffer.

78.4.2.9 Byte ordering of serial flash memory read data

The basic scheme is that the first byte read out of the serial flash memory device, which is addressed by SFAR[SFADR], corresponds to RBDR0[31:24] for IP command read. Similarly, to send a single byte it should be positioned in TBDR[0:7]. In contrast to that for AHB command read, the bytes are always positioned according to the byte ordering of the AHB bus.

- Byte ordering in individual flash memory mode

  The following table provides the byte ordering scheme of how the byte oriented data space of the serial flash memory device is mapped into one single 32-bit entry of the RX buffer or the AHB buffer. The table is valid within the following context:

  — Flash memory A in individual flash memory mode

  — All AHB data read commands with 32-bit access size

Table 701. Byte ordering in individual flash memory mode

Serial flash memory byte numbering	3	2	1	0
Buffer entry bit position [31:0] (32-bit data width)	[31:24]	[23:16]	[15:8]	[7:0]

NOTE

For IP commands, the read size can be specified as number of bytes. If this number is not a multiple of four, then the last buffer entry is not completely filled with the missing higher numbered bytes at undefined values.

  For AHB commands and reads, starting from an address not aligned to 32-bit boundaries, the requested bytes are given at the appropriate positions according to the AMBA AHB specification.

- Buffer entry ordering for 64-bit read access

  For read access via the AHB interface, a 64-bit access is possible. Each 64-bit access reads two 32-bit entries, simultaneously. The ordering of these 32-bit entries within the 64-bit word is provided in the following table.

Table 702. 64-bit read access buffer entry ordering

AHB read data bit position [63:0]	[63:32]	[31:0]
Buffer entry #	Odd (1, 3, 5, ...)	Even (0, 2, 4, ...)

78.4.2.10 Normal mode interrupt and DMA requests

The QuadSPI module has different flags that can only generate interrupt requests and one flag that can generate an interrupt as well as DMA requests. The following table lists the eight conditions. Note that the flags mentioned in the table are associated with the Flag Register (FR).

Table 703. Interrupt and DMA request conditions

Condition	Flag (FR)	DMA
TX buffer fill	TBFF	-
TX buffer underrun	TBUF	-
Illegal instruction error	ILLINE	-
RX buffer drain	RBDF	X
RX buffer overflow	RBOF	-
AHB buffer overflow	ABOF	-
AHB sequence error	ABSEF	-
AHB illegal transaction error	AITEF	-
AHB illegal burst size error	AIBSEF	-
IP command trigger during AHB access error	IPAEF	-
IP command trigger could not be executed error	IPIEF	-
IP command related transaction finished	TFF	-

Each condition has a corresponding field in Flag Register (FR) and a request enable field in DMA Request Select and Enable Register (RSER). FR[RBDF] has separate enable fields for generating IRQ and DMA requests. Note that not all the fields have an individual IRQ line. See the chip's Interrupt Vector table for details.

- Transmit buffer fill interrupt request

  This indicates that the TX buffer can accept new data. The buffer is asserted if FR[TBFF] is asserted and if the value of the corresponding enable field, RSER[TBFIE], is 1. See TX buffer Operation for details on the assertion of FR[TBFF].

  Apart from IRQ, it is possible to handle the TX buffer fill by using the DMA. If the value of RSER[TBFDE] is 1, a DMA request is triggered when the number of available space in the TX buffer is more than the TBCT[WMRK] valid entries and value of SR[TXWA] is set. The application must configure the environment appropriately (for example, the DMA controller) for the DMA transfer.

- Receive buffer drain interrupt or DMA request

  This is derived from FR[RBDF], indicating that the RX buffer of the QuadSPI module has data available from the serial flash memory device to be read by the host. It remains set as long as RBSR[RXWE] is configured. Also, RSER[RBDIE] enables the related IRQ.

Apart from the IRQ, it is possible to handle the RX buffer drain by using the DMA. If the value of RSER[RBDDE] is 1, a DMA request is triggered when the RX buffer contains more than RBCT[WMRK] valid entries. The application must set the environment appropriately (for example, the DMA controller) for the DMA transfers.

- Buffer overflow/underrun interrupt request

  This is a combination of the following fields (all located in the Flag Register (FR) with the related enable bits in the DMA Request Select and Enable Register (RSER)):

  — TBUF - TX buffer underrun, enabled by TBUIE

  — RBOF - RX buffer overflow, enabled by RBOIE

  — ABOF - AHB buffer overflow, enabled by ABOIE

  — The transmit buffer underrun indicates that an underrun condition in the TX buffer has occurred. It is generated when a write instruction is triggered whilst the TX buffer is empty and the the value of RSER[TBUIE] is 1.

  — The receive buffer overflow indicates that an overflow condition in the RX buffer has occurred. It is generated when the RX buffer is full, an additional read transfer attempts to write into the RX buffer, and the value of RSER[RBOIE] is 1.

  — The AHB buffer overflow indicates that an overflow condition in the AHB buffer has occurred. It is generated when the AHB buffer is full, an additional read transfer attempts to write into the AHB buffer and the value of RSER[ABOIE] is 1.

  — The data from the transfers that generated the individual overflow conditions is ignored.

- Serial flash memory command error interrupt request

  If the IPAEF, IPIEF fields in the FR are set, and the related interrupt enable bits in the RSER are also set, then an interrupt is requested.

- Transaction finished interrupt request

  The IP command transaction finished IRQ indicates the completion of the current IP command. It is triggered by FR[TFF] and is masked by RSER[TFIE].

78.4.2.11 TX buffer operation

The TX buffer provides the data used for page programming. For proper operation, it is required to provide at least one entry in the TX buffer prior to starting the execution of the page programming command. The application must ensure that the required number of data bytes is written into the TX buffer fast enough as long as the command is executed without a TX buffer overflow or underrun.

The QuadSPI module sets the FR[TBFF] field as long as the TX buffer is not full and can accept more data. At the end of write through TX buffer, you must clear FR[TBFF] to avoid unnecessary last TX buffer fill interrupt. However, there would always be a pending request asserted from QuadSPI controller at the end of any DMA transfer. When external DMA finishes transfer iteration, this request from QuadSPI is kept asserted for the next iteration loop.

NOTE

Even if the generation of DMA requests for filling the TX buffer is disabled by using RSER[TBFDE], the TX buffer still accepts a DMA transfer because of the last asserted pending request.

Disabling of DMA transfer should be controlled by an external DMA master.

When the QuadSPI module tries to pull data out of an empty TX buffer, FR[TBUB] signals the TX buffer underrun. The TX buffer underrun flag is also asserted when the TX buffer contains less than 32-bits of data and the QuadSPI module tries to pull out data from it. The current IP command leading to the underrun condition is continued until the specified number of bytes is sent to the serial flash memory device. Also, the data that is transferred is in the Fs format. This means, after the underrun flag is set under this condition, it returns Fs until the required number of bytes are not sent. This has been done to ensure that the software does not erase the whole sector after underrun and just reprogramming from failure point serves the purpose. When this sequence command is complete, FR[TBFF] is asserted, indicating that the TX buffer is ready to be written again.

The TX buffer overflow is not signaled explicitly, but TBSR[TRBFL] can monitor the TX buffer fill level.

For more information, see TX Buffer Status Register (TBSR) and Flag Register (FR).

78.4.2.12 Address scheme

Earlier, serial flash memories supported only a 24-bit address space, restricting the maximum memory size of the serial flash memory to 16 MB. The new memory specification supports two types of 32-bit addressing mode in addition to the legacy 24-bit address mode.

Extended address mode

In this mode, the legacy 24-bit commands are converted to accept 32-bit address commands. The flash memory needs to be configured for the 32-bit address mode. Also, while programming the LUT sequence in QuadSPI for 32-bit mode, the ADDR commands should be programmed with 32d as the operand value . By default, QuadSPI is in 24-bit legacy address mode. Each of the memory vendors have a different way of enabling this mode (see the memory specification from memory vendors). For example, the command B7h sent to the Macronix flash memory enables it for the 32-bit address mode.

Extended address register

In this mode, the upper 8 bits of the 32-bit register are provided by the Extended address register in the memory, which provides a specific register that is updated according to the address to be accessed. This effectively converts the legacy 24-bit address command into 32-bit address commands. The memories greater in size than 16 MB consist of banks of 16 MB each. The 8 bits occupied in the extended address register effectively enable a bank. For example, in Spansion memory, when the extended address register is updated with a value of 1h, with the help of the 17h command, it opens Bank1 of the memory. The consequent 24-bit address commands lead to Bank1. The extended address register needs to be updated with the respective value for access to other banks. This effectively converts the legacy 24-bit address command into 32-bit address commands.

78.4.3 Module Disable mode

Module Disable mode is a block-specific mode that the QuadSPI can enter to disable serial flash memory clock and AHB command. This mode can be entered by:

- The host software: by writing a '1' to MCR[MDIS]

Below are the condition that must be fulfilled to enter the Module Disable mode:

- SR[BUSY] = 0
- SR[AHBTRN] = 0
- RBSR[RDBFL] = 0
- SR[RXDMA] = 0
- SR[TXDMA] = 0
- None of the flags in FR are enabled as interrupts is set

The conditions mentioned above ensures the following:

- There is no SFM command currently being executed.
- All the data read into the RX buffer from the serial flash memory have been fetched by the application.
- There is no current AHB access.
- There is no active DMA request.
- There is no enabled interrupt that is pending.

Certain read or write operations have a different effect when the QuadSPI is in the Module Disable mode. In the Module Disable mode, not all of the status and flag bits of the QuadSPI module are updated, and writing to them has no effect. Interrupt and DMA request signals cannot be cleared while in the Module Disable mode.

NOTE

It is illegal to issue a new SFM command starting two clock cycles prior to raising the request of entering the Module Disable mode until the QuadSPI stays in this mode.

78.4.4 Leaving Module Disable mode

In the Module Disable mode, the serial flash memory clock and AHB command to the QuadSPI module are switched off.

After the QuadSPI has left this mode and has returned to Normal mode, the execution of the first SFM command is deferred until the clock to drive that part of the module related to the serial flash memory device is available. Depending upon the point in time when the first SFM command is triggered, the actual execution of the command starts with a delay, respective with the re-enabling of the flash memory clock signal.

78.5 Initialization/application information

This section provides the initialization and application information of the QuadSPI module.

78.5.1 Power up and reset

The serial flash memory devices connected to the QuadSPI module might require special voltage characteristics of their inputs during power up or reset. The application must ensure this.

CAUTION

Erase or program commands should be completed before issuing a reset or power cycle to avoid corrupted flash pages. The application shall ensure there is a backup of critical data stored at a different location to enable recovery from corrupted flash pages.

Example: Flash reset sequence

Use the following sequence to reset flash A:

1. Make sure that the flash supports a reset for the condition CS#=high and IOF[3]=low.
2. Set MCR[SWRSTSD] and MCR[SWRSTHD] fields and then clear them.
3. Set MCR[MDIS] field.
4. Reset MCR[ISD3FA] field for flash A.
5. Clear MCR[MDIS] field.
6. Set MCR[MDIS] field.
7. Set MCR[ISD3FA] field for flash A.
8. Clear MCR[MDIS] field.

78.5.2 Available status/flag information

This section provides an overview of the different flags and statuses available, and their interdependencies for different use cases. The SR and FR are the related registers.

78.5.2.1 IP commands

See IP Configuration Register (IPCR) for additional details not explicitly covered in this paragraph.

- IP commands—normal operation

  Writing to IPCR[SEQID] triggers the execution of a new IP command. Given that this is a legal command, SR[IPACC] and SR[BUSY] are asserted simultaneously, immediately after the execution starts.

  After the instruction on the serial flash memory device is complete, these field deassert and FR[TFF] is configured.

- IP commands—error situations

  See Overview_of_Error_Flags for details.

78.5.2.2 AHB commands

See the "Reading serial flash memory data into the QuadSPI module internal buffers" topic in theFlash Memory Read section for details.

- AHB commands—normal operation

  Memory-mapped read access to a serial flash memory address not contained in the AHB buffer triggers the execution of an AHB command. Given that this is a legal command, SR[AHB_ACC] and SR[BUSY] are asserted simultaneously, immediately after the execution starts. After the instruction on the serial flash memory device is complete, these fields are deasserted.

- IP commands—error situations

  See Overview of FR error flags for details.

78.5.2.3 SFM commands

An SFM command consists of an instruction code and all other parameters (for example, size or mode bytes) needed for that specific instruction code. Triggering a command either initiates a transaction on the external serial flash memory or results in an error. See Table 704 for details on errors.

78.5.2.4 Overview of error flags

The following table provides an overview of the different error flags in the FR and additional error-related details.

Table 704. Overview of FR error flags

Error category	Error flag in FR	Command execution on serial flash memory device TFF behavior (in case of IP commands only)	Description
AHB error flag	ABOF	Flash memory transaction continues until it finishes	Set when the module tries to push data into the AHB buffer that exceeds the size of the AHB buffer. Occurs only because of the wrong programming of BUFxCR[ADATSZ].
AHB error flag	AIBSEF	Flash memory transaction is aborted	Total burst size of the AHB transaction is greater than prefetch data size.
AHB error flag	AITEF	Flash memory transaction is aborted	No response is generated from QuadSPI to AHB bus in case of illegal transaction. Also, the watchdog timer expires.
Miscellaneous error flag	ILLINE	Flash memory transaction aborted	Illegal instruction error flag - set when an illegal instruction is encountered by the controller in any of the sequences.
Command arbitration error	IPIEF	TFF not asserted in conjunction with that command	IP command error - caused when IP access is currently in progress (IP_ACC is set) and during: • Write attempt to IPCR register

Table continues on the next page...

Table 704. Overview of FR error flags (continued)

Error category	Error flag in FR	Command execution on serial flash memory device TFF behavior (in case of IP commands only)	Description
			• Write attempt to SFAR register • Write attempt to RBCT register
Command arbitration error	IPAEF	TFF not asserted in conjunction with that command	• AHB command already running, another IP command could not be executed • AHB command already running, write attempt to IPCR[SEQID]
Buffer-related error	RBOF	TFF is asserted on completion	• RX buffer overrun
Buffer-related error	TBUF	TFF is asserted on completion	• TX buffer underrun

Note that only the buffer-related errors are associated with a transaction on the external serial flash memory. All the other errors do not trigger an actual transaction.

78.5.2.5 IP bus and AHB access command collisions

There are following flags related to this topic: FR[IPAEF] and FR[IPIEF]. See the "Reading serial flash memory data into the QuadSPI module internal buffers" topic of the Flash Memory read section for a description of the flags.

78.5.3 Flash memory device selection

Regardless of the SFM command (IP or AHB), the access mode is selected by specifying the 32-bit address value for the following SFM command.

For IP commands, the access mode is selected with the address programmed into the SFAR register. See Serial Flash Address Register (SFAR) for details.

For AHB commands, the access mode is determined by the memory-mapped address. See AMBA Bus Register Memory Map for details.

78.5.4 DMA usage

For a complete description of the DMA module, see the related DMA Controller chapter. This section only provides QuadSPI-specific DMA usage details.

78.5.4.1 DMA usage in normal mode

78.5.4.1.1 Bandwidth considerations

Careful consideration of the throughput rate of the entire chain (serial flash memory -> AHB bus / IP bus -> DMA controller) involved in the read/write data process is essential for a proper operation. Such analysis must take into account not only the data rate provided by the serial flash memory but also the data rate of the AHB bus and the performance of the DMA controller in reading/writing data from/to the RX/TX buffer.

Two figures must match for a proper operation, which means that the data rate provided by the serial flash memory device must not exceed the average RX buffer readout data rate. Otherwise, the longer this state persists, it results into an RX buffer overflow.

Similarly, the data consumed by the serial flash memory device must not exceed the average TX buffer fill rate. If this persists, it leads to an underrun.

AHB bus side (data read)

The total number of bus cycles for each DMA minor loop completion is added from the following components:

The following table provides certain examples for typical use cases:

Case 1: DMA needs to read 4 bytes from SRAM and provide to QuadSPI. It costs total four bus clock cycles. Then, DMA handshake adds additional six bus clock cycles, resulting in a total of [6 + 4 * (32/4) = 38] bus clock cycles.

Table 705. Access duration examples for bus clock side

TBCT[WMRK]	Number of bytes per DMA loop	Number of bus clock cycles for DMA minor loop	Time duration of DMA minor loop for 60 MHz bus clock frequency
3	16	6+(16/4)*4 = 22	~366ns
7	32	6+(32/4)*4 = 38	~633ns
11	48	6+(48/4)*4 = 54	~900ns
15	64	6+(64/4)*4 = 70	~1166ns

Case 2: DMA needs to read 32 bytes from SRAM and provide to QuadSPI. DMA handshake takes an additional six bus cycles, with 32 bytes DMA read from SRAM costs (8 + 3) core clock cycles. DMA writes 32 bytes to QuadSPI, takes 2 * (32/4) = 16 bus cycles with one additional CPU access to QuadSPI, costing two bus clock cycles. This results in a total 6 + (8+3)/2 + 2 * (32/4) +2 = 30 bus clock cycles.

Table 706. Access duration examples for bus clock side

TBCT[WMRK]	Number of bytes per DMA loop >	Number of bus clock cycles for DMA minor loop	Time duration of DMA minor loop for 80 MHz bus clock frequency
3	16	6 + (4+3) /2 + (16/4)*2 + 2 = 20	~333ns
7	32	6 + (8+3) /2 + (32/4)*2 + 2 = 30	~500ns
11	48	6 + (4+3)/2*3 + (48/4)*2 + 2 = 44	~733ns
15	64	6 + (8+3) + (64/4)*2 + 2 = 51	~810ns

NOTE

This table figure represents an ideal scenario; actual performance depends on how the chip integrates DMA and QuadSPI modules.

Serial flash memory device side (data read)

The number of serial flash memory cycles can be determined in the following way:

- Number of serial flash memory clock cycles is required to read 4 bytes, corresponding to one RX buffer entry (setup of command and address not considered): , eight cycles for quad mode (SDR) instructions in individual flash memory mode, and so on.
- Overhead because of clock domain crossing: one cycle

The following table lists the number of clock cycles required to read the data from the serial flash memory corresponding to the different settings of RBCT[WMRK]:

Table 707. Access duration examples for serial flash memory side

RBCT[WMRK] setting	Num bytes per DMA loop [1]	Num SCKFx for 80 MHz SCKFx		Time duration of flash memory data readout for 80 MHz SCKFx (~12.5ns period)	
		IFM [2] quad	IFM quad DDR	IFM quad	IFM quad DDR
0	4	8	4	~100ns	~50ns
1	8	16	8	~200ns	~100ns
3	16	32	16	~400ns	~200ns
7	32	64	32	~800ns	~400ns

1. DMA loop refers to one minor loop completion that is equivalent to one major loop iteration.
2. Individual flash memory mode

NOTE

The table figure represents an ideal scenario; actual performance depends on how the chip integrates with DMA and QuadSPI modules.

A complementary example is when the watermark is set to be too high. In such a case, the time taken by the DMA to read out the RX buffer entries should be lesser than the time taken by the controller to push in the remaining entries in the buffer.

IPS bus side (data write)

The total number of bus cycles for each DMA minor loop completion are added from the following components:

- Overhead for each minor loop, given by DMA controller: assume 10 cycles
- Overhead because to clock domain crossing: assume two cycles
- Number of bus clock cycles required for 16 bytes (128-bit write size): assume four cycles (read/write sequence of DMA controller)

Note that the size of the minor loop is determined by the size of TBCT[WMRK]; therefore, the overhead specified above distributes among (TBCT[WMRK]+1) write accesses of 32-bit each.

The following table provides some examples for typical use cases:

Table 708. Access duration examples for bus clock side

TBCT[WMRK]	Number of bytes per DMA loop [1]	Number of bus clock cycles for DMA minor loop	Time duration of DMA minor loop for 80 MHz bus clock frequency
3	16	12+4 = 16	~200ns
7	32	12+8 = 20	~250ns
11	48	12+12 = 24	~300ns
15	64	12+16 = 28	~350ns
19	80	12+20 = 32	~400ns

1. DMA loop refers to one minor loop completion that is equivalent to one.

NOTE

The table figure represents an ideal scenario; actual performance depends on how the chip integrates with DMA and QuadSPI modules.

Serial flash memory device side (data write)

The number of serial flash memory cycles can be determined in the following way:

- Number of serial flash memory clock cycles required to write 16 bytes, corresponding to four TX buffer entry (setup of command and address not considered): 32 cycles for quad SDR writes in individual flash memory mode.
- Overhead due to clock domain crossing: one cycle

The following table lists the number of clock cycles required to read the data from the serial flash memory corresponding to the different settings of TBCT[WMRK]:

Table 709. Access duration examples for serial flash memory side

TBCT[WMRK] setting	Num bytes per DMA loop [1]	Num SCKFx		Time duration for consuming data at flash memory interface 100 MHz SCKFx (10 ns period)[2]		Time for FIFO to get empty[3]	
		IFM [4] quad	[5] single IO SDR mode	IFM quad	PFM single IO SDR	IFM quad	PFM single IO SDR
3	16	32	64	320ns	640ns	2240ns	4480ns
7	32	64	128	640ns	1280ns	1920ns	3840ns
15	64	128	256	1280ns	2560ns	1280ns	2560ns
23	96	192	384	1920ns	3840ns	640ns	1280ns

1. DMA loop refers to one minor loop completion that is equivalent to one major loop iteration.
2. Not all flash memory devices support writes at 100 MHz. See the flash memory data sheet for the actual page program frequency supported.
3. The assumption for these timings is that the TX Fifo is full when the transaction is initiated
4. Individual flash memory mode
5. Parallel flash memory mode

NOTE

The tables mentioned above are only examples which must be correlated with the DMA in the system.

Considering the examples provided in the two tables above for TX FIFO, it is evident that depending on the relationship between the bus clock and serial flash memory clock frequencies, there are settings possible where the serial flash memory consumes data faster than the IPS bus can write data in TX buffer. In these cases, a TX buffer underrun situation occurs. To avoid TX buffer underrun, the data transaction size should be large enough.

78.5.5 Flash memory devices address mapping

QuadSPI is configured in Single mode for the supported flash memory port A

The sizes of the flash memory devices are mapped with the system memory space based on the configurations of the following registers:

- SFA1AD
- SFA2AD
- SFB1AD
- SFB2AD

The total memory region for the flash memory devices is mapped between QuadSPI_AMBA_BASE and TOP_ADDR_MEMB2 such that the corresponding CS is asserted based on SFA1AD, SFA2AD and SFB1AD register configurations.

78.5.5.1 Single mode

For single mode configuration, you must write the same value to SFB1AD and SFB2AD registers that you write to the SFA2AD register.

For dual-die flash memories, the values you write to SFB1AD and SFB2AD registers corresponds to the mapped top addresses of each die.

For single-die flash memories, you must write the same value to SFA2AD register that you write to the SFA1AD register.

Following is a programming example for single mode single-die flash memory:

- QuadSPI_AMBA_BASE - 1000_0000h
- SFA1AD[TPADA1] - 2000_0000h
- SFA2AD[TPADA2] - 2000_0000h
- SFB1AD[TPADB1] - 2000_0000h
- SFB2AD[TPADB2] - 2000_0000h

The following figure illustrates the memory mapping for single mode QuadSPI configuration.

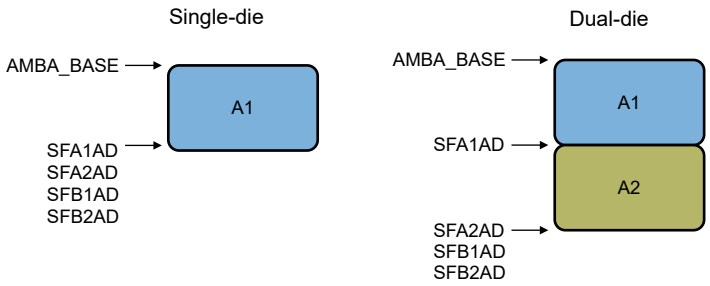


Figure 502. Memory map for Single mode

78.6 Byte ordering – endianness

The following topics show the byte ordering in 64-bit LE configuration for AHB buffer and 32-bit LE for TX/RX buffer.

78.6.1 Programming flash memory data

CPU writes instructions to the TBDR register, such as:

- Write TBDR: 4_03_02_01h
- Write TBDR: 8_07_06_05h

The following table shows the content against each TX buffer entry.

Table 710. Example of QuadSPI TX buffer

TX buffer entry	Content
0	4_03_02_01h
1	8_07_06_05h

Programming the TX buffer into the external serial flash memory device results in the following byte order to be sent to the serial flash memory:

- 01...02...03...04...05...06...07...08

78.6.2 Reading flash memory data into the RX buffer

Reading the content from the same address provides the following sequence of bytes, identical to the write case:

- 01...02...03...04...05...06...07...08

The following table shows the content against each TX buffer entry.

Table 711. Resulting RX buffer content

RX buffer entry	Content
0	4_03_02_01h
1	8_07_06_05h

78.6.2.1 Readout of the RX buffer through RBDRn

The RX buffer content appears at CPU read access through the peripheral bus interface in the following order:

- Read RBDR0: 4_03_02_01h
- Read RBDR1: 8_07_06_05h

78.6.2.2 Readout of the RX buffer through ARDBn

The RX buffer content appears at read access on the AMBA AHB interface at the QuadSPI module boundary:

- 32-bit access: Read ARDB0: 4_03_02_01h
- 32-bit access: Read ARDB1: 8_07_06_05h
- 64-bit access: Read ARDB0: 8_07_06_05_04_03_02_01h

78.6.3 Reading flash memory data into the AHB buffer

Reading the content from the same address as it was written to provides the following sequence of bytes, identical to the write case:

- 01...02...03...04...05...06...07...08

The following table shows the content against each TX buffer entry.

Table 712. Resulting AHB buffer content

AHB buffer entry	Content
0	8_07_06_05_04_03_02_01h

78.6.3.1 Readout of the AHB buffer through memory-mapped read

The AHB buffer content appears at read access on the AMBA AHB interface at the QuadSPI module boundary:

- 32-bit read access: 4_03_02_01h
- 32-bit read access: 8_07_06_05h
- 64-bit read access: 8_07_06_05_04_03_02_01h

78.7 Driving flash memory control signals in single and dual modes

In single and dual modes, the serial flash memory devices that can connect to the QuadSPI module expect additional control signals on the inputs, which are connected to IOFA[3], IOFA[2] in the quad mode. For easy interfacing, the outputs IOFA[3:2] for flash memory A are driven to the logic state given by the configuration fields MCR[ISD3FA], MCR[ISD2FA].

These outputs are driven all the time to the logic level programmed in the MCR except the time when quad commands of the serial flash memory are executed. See the specifications of the related serial flash memory device for details about the inactive level.

78.8 Serial flash memory devices

Several different vendors make flash memory devices with a QuadSPI interface. At present, there is no set standard for the QuadSPI instruction set. The most common commands currently have the same instruction code for all vendors; however, some commands are unique to specific vendors. Some of the example sequences are provided in the following sections.

78.8.1 Example sequences

This section provides the example sequences of the QuadSPI module.

Table 713. Exit 4 x I/O read enhance performance mode (XIP) (Macronix) and read status

Instruction	Pad	Operand	Description
CMD	0h	EBh	4xIO read command
ADDR	2h	18h	24-bit address to be sent on four pads
MODE	2h	0h	2 mode cycles (exit XIP)
DUMMY	0h	4h	4 dummy cycles
READ	2h	8h	Read 64 bits
CMD	0h	5h	Read Status register
READ	0h	1h	Status register data
STOP	0h	0h	Stop, instruction over

78.8.1.1 Fast read sequence (Macronix/Numonyx/Spansion/Winbond)

The following table shows the fast read sequence for Macronix/Numonyx/Spansion/Winbond flash memories.

Table 714. Fast read sequence

Instruction	Pad	Operand	Description
CMD	0h	Bh	Fast read command = 0Bh
ADDR	0h	18h	24 address bits to be sent on one pad
DUMMY	0h	8h	Eight dummy cycles
READ	0h	4h	Read 32 bits on one pad
JMP_ON_CS	0h	0h	Jump to instruction 0 (CMD)

NOTE

If DLL is disabled then JMP_ON_CS or STOP instruction can be used else only STOP instruction can be used.

78.8.1.2 Fast read quad output (Winbond)

The following table shows the fast read quad output sequence for Winbond memories

Table 715. Fast read quad output sequence

Instruction	Pad	Operand	Description
CMD	0h	6Bh	Fast read quad output command = 6Bh
ADDR	0h	18h	24 address bits to be sent on one pad
DUMMY	0h	8h	Eight dummy cycles
READ	2h	4h	Read 32 bits on four pads
JMP_ON_CS	0h	0h	Jump to instruction 0 (CMD)

NOTE

If DLL is disabled then JMP_ON_CS or STOP instruction can be used else only STOP instruction can be used.

78.8.1.3 4 x I/O read enhance performance mode (XIP) (Macronix)

The following table shows the 4 x I/O read enhance performance mode for Macronix flash memories. The enhanced performance mode is also known as XIP mode.

Table 716. Fast read quad output sequence

Instruction	Pad	Operand	Description
CMD	0h	EBh	4xI/O read command = EBh
ADDR	2h	18h	24 address bits to be sent on four pads
MODE	2h	A5h	Two mode cycles
DUMMY	0h	4h	Four dummy cycles
READ	2h	4h	Read 32 bits on four pads
JMP_ON_CS	0h	1h	Jump to instruction 1 (ADDR)

When in XIP mode, the software must ensure that all the flash memories connected to the controller are in this mode. As a part of initializing the controller, all the flash memories might be enabled with XIP by carrying out dummy reads.

78.8.1.4 Dual command page program (Numonyx)

The following table shows the dual command page program sequence for Numonyx flash memories.

Table 717. Dual command page program sequence

Instruction	Pad	Operand	Description
CMD	1h	2h	Dual command page program = 02h on 2 pads
ADDR	1h	18h	24 address bits to be sent on two pads
WRITE	1h	20h	Write 32 bytes on two pads

Table continues on the next page...

Table 717. Dual command page program sequence (continued)

Instruction	Pad	Operand	Description
STOP	0h	0h	Stop, instruction over

78.8.1.5 Sector erase (Macronix/Numonyx/Spansion)

The following table shows the Sector erase sequence for the Macronix/Numonyx/Spansion flash memories.

Table 718. Sector erase sequence

Instruction	Pad	Operand	Description
CMD	0h	20h	Sector erase command = 20h
ADDR	0h	18h	24 address bits to be sent on one pad
STOP	0h	0h	Stop, instruction over

78.8.1.6 Read status register (Macronix/Numonyx/Spansion/Winbond)

The following table shows the read status register sequence for Macronix/Numonyx/Spansion/Winbond flash memories.

Table 719. Read status register sequence

Instruction	Pad	Operand	Description
CMD	0h	0h5	Read status register command = 05h
READ	0h	0h1	Read status register data
STOP	0h	0h	Stop, instruction over

78.9 Sampling of serial flash memory input data

78.9.1 Basic description

QuadSPI is used to read data from the serial flash memory device. Depending on the actual implementation, there is a delay between the internal clocking in the QuadSPI module and the external serial flash memory device. See the following figure for an overview of this scheme.

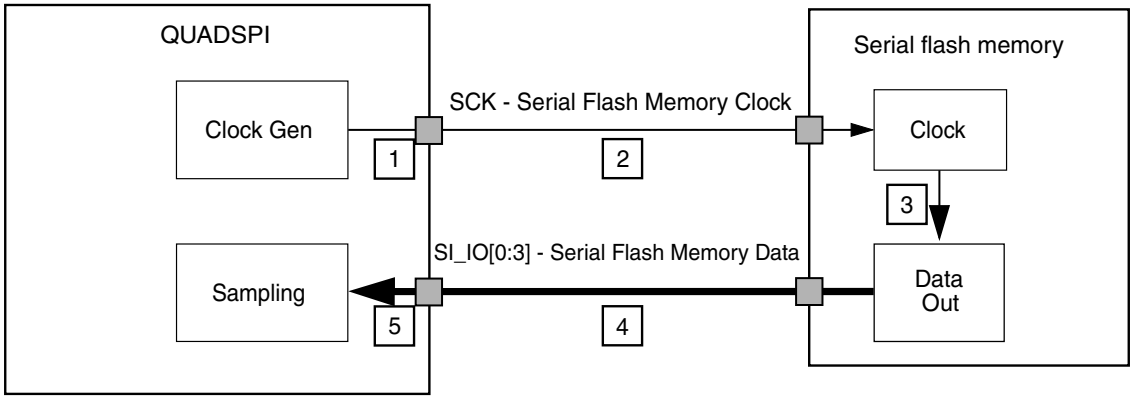


Figure 503. Serial flash memory sampling clock overview

The rising edge of the internal reference clock is taken as timing reference for the data output of the serial flash memory. After a time of t_{total_delay} the data arrives at the internal sampling stage of the QuadSPI module. Considering the figure provided here, the following parts of the delay chain contribute to t_{total_delay}:

- Output delay of the serial flash memory clock output of the device containing the QuadSPI module
- Wire delay of application/PCB from the device containing the QuadSPI module to the external serial flash memory device
- Clock to data out delay of the external serial flash memory device, including input and output delays
- Wire delay of application/PCB from the external serial flash memory device to the device containing the QuadSPI module
- Device delay corresponding to the input data

NOTE

The t_{total_delay} is specific to the characteristics of the actual implementation. Also, the serial flash memory device clock (SCK) is inverted with respect to the QuadSPI internal reference clock.

78.9.2 DQS sampling method

78.9.2.1 Basic description

In the DQS mode, the data strobe signal (DQS/RWDS) is used to sample the read data. Here, both DQS and the data sent by the flash memory move in the same direction; therefore, it is relatively easier to achieve at higher frequencies.

When using DQS for SDR reads, QuadSPI internally samples the incoming data on the rising edge of the strobe signal.

The next figure shows the sampling read data in the SDR mode using the DQS.

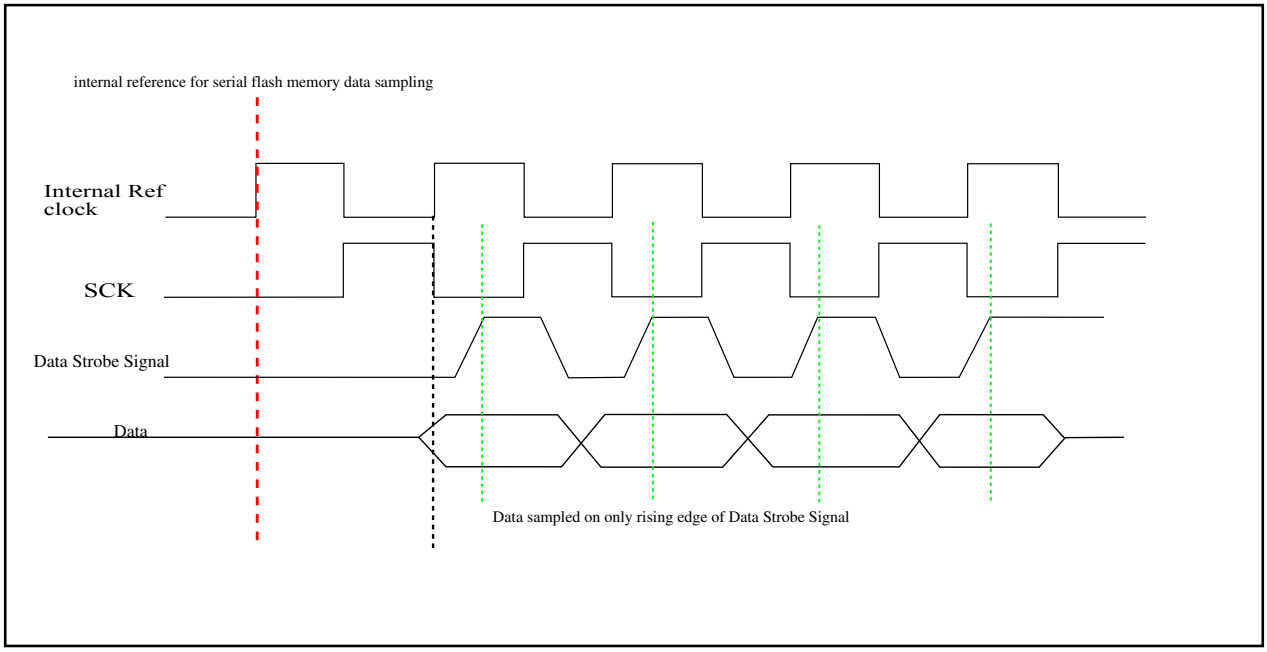


Figure 504. Data strobe functionality in SDR mode for read operation.

NOTE

Consider "Data Strobe Signal" as "Data Strobe Signal driven by memory" and "Data" as "Data from memory".

NOTE

Refer to the Datasheet for specific timing waveforms of QuadSPI

NOTE

For Specific details - Refer to the Data Sheet specification of QuadSPI module.

78.9.2.2 Dummy Pad loopback

The internal clock is loop-backed from the dummy internal pad to compensate data pad delays. This can be enabled by configuring the value of MCR[DQS_FA_SEL] as "01" for flash memory A. This mode can be used with the following configuration:

- High/low frequency delay chain manual programming in bypass mode using DLLCRA[SLV_DLY_COARSE] and DLLCR[FREQEN].

NOTE

Refer to Auto-DataLearning (4x Sampling method) section with DLL for further details

NOTE

This mode may not be available on the chip. See the "Supported read modes" section in the chip-specific QuadSPI information for the read modes that this chip supports.

78.10 Delay chain usage

Slave delay chain programming sequence—

Following is the programming sequence for DLL bypass mode.

1. Program DLLCRA[SLV_EN]=1, DLLCRA[SLV_DLL_BYPASS]=1, and DLLCRA[SLAVE_AUTO_UPDT]=0.

2. Program the following fields to provide the desired DQS delay for sampling: DLLCRA[SLV_FINE_OFFSET], DLLCRA[SLV_DLY_COARSE], and DLLCR[FREQEN]. See the chip-specific QuadSPI information for the supported programming settings.
3. Program DLLCRA[SLV_UPD]=1 to load these values in the slave delay chain.

78.11 Memory map and register definition

This section provides the memory map and register definitions for the QuadSPI module.

78.11.1 Register write access

Following are the write access restriction terms that apply to all the registers:

- Register write access restriction

  For each register field, the write access conditions are specified in the detailed register description.

  The following table provides a description of the write access conditions. If, for a specific register bit or field, none of the given write access conditions is fulfilled, any write attempt to this register bit or field is ignored without any notification. The values of the bits or fields are not changed.

  The condition term [A or B] indicates that the register or field can be written to if at least one of the conditions is fulfilled.

Table 720. Register write access restrictions

Condition	Description
Anytime	No write access restriction
Disabled mode	Write access only if MCR[MDIS] = 1
Normal mode	Write access only if the module is in the normal mode

- Register write access requirements

  You can access all registers using 8-bit, 16-bit, and 32-bit wide operations. For some of the registers, at least a 16-bit or 32-bit wide write access is required to ensure correct operation. This write access requirement is stated in the detailed register description for each affected register.

78.11.2 QuadSPI register descriptions

This section provides the memory map and register definitions for the QuadSPI module.

Access to the following addresses does not result in a transfer error:

- 4h
- 50h
- 64h
- 104h
- 120h
- 138h
- 168h
- 188h
- 18Ch

78.11.2.1 QuadSPI memory map

QuadSPI_34x base address: 404C_C000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Module Configuration Register (MCR)	32	RW	000F_404Ch
8h	IP Configuration Register (IPCR)	32	RW	0000_0000h
Ch	Flash Memory Configuration Register (FLSHCR)	32	RW	0000_0303h
10h	Buffer 0 Configuration Register (BUF0CR)	32	RW	0000_0003h
14h	Buffer 1 Configuration Register (BUF1CR)	32	RW	0000_0002h
18h	Buffer 2 Configuration Register (BUF2CR)	32	RW	0000_0001h
1Ch	Buffer 3 Configuration Register (BUF3CR)	32	RW	8000_0000h
20h	Buffer Generic Configuration Register (BFGENCR)	32	RW	0000_0000h
24h	SOC Configuration Register (SOCCR)	32	RW	0000_0000h
30h	Buffer 0 Top Index Register (BUF0IND)	32	RW	0000_0000h
34h	Buffer 1 Top Index Register (BUF1IND)	32	RW	0000_0000h
38h	Buffer 2 Top Index Register (BUF2IND)	32	RW	0000_0000h
60h	DLL Flash Memory A Configuration Register (DLLCRA)	32	RW	0120_0000h
100h	Serial Flash Memory Address Register (SFAR)	32	RW	0000_0000h
108h	Sampling Register (SMPR)	32	RW	FF00_0000h
10Ch	RX Buffer Status Register (RBSR)	32	R	0000_0000h
110h	RX Buffer Control Register (RBCT)	32	RW	0000_0000h
134h	Data Learning Status Flash Memory A Register (DLSR_FA)	32	R	0000_0000h
150h	TX Buffer Status Register (TBSR)	32	R	0000_0000h
154h	TX Buffer Data Register (TBDR)	32	RW	0000_0000h
158h	TX Buffer Control Register (TBCT)	32	RW	0000_0000h
15Ch	Status Register (SR)	32	R	0200_3800h
160h	Flag Register (FR)	32	RW	0800_0000h
164h	Interrupt and DMA Request Select and Enable Register (RSER)	32	RW	0000_0000h
16Ch	Sequence Pointer Clear Register (SPTRCLR)	32	RW	0000_0000h
180h	Serial Flash Memory A1 Top Address Register (SFA1AD)	32	RW	7000_0000h
184h	Serial Flash Memory A2 Top Address Register (SFA2AD)	32	RW	7000_0000h
188h	Serial Flash Memory B1 Top Address Register (SFB1AD)	32	RW	7000_0000h
18Ch	Serial Flash Memory B2 Top Address Register (SFB2AD)	32	RW	7000_0000h
200h - 2FCh	RX Buffer Data Register (RBDR0 - RBDR63)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
300h	LUT Key Register (LUTKEY)	32	RW	5AF0_5AF0h
304h	LUT Lock Configuration Register (LCKCR)	32	RW	0000_0002h
310h	LUT Register (LUT0)	32	RW	0818_0403h
314h	LUT Register (LUT1)	32	RW	2400_1C08h
318h - 35Ch	LUT Register (LUT2 - LUT19)	32	RW	0000_0000h

78.11.2.2 Module Configuration Register (MCR)

Offset

Register	Offset
MCR	0h

Function

This register holds configuration data associated with the QuadSPI operation.

Special write-access is permitted in different modes:

- DQS_FA_SEL: Disabled mode
- ISD3FA, ISD2FA: Disabled mode
- All other fields: Anytime

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved	Reserved	Reserved		Reserved	Reserved	DQS_FA_SEL		Reserved	Reserved	0		Reserved	Reserved	ISD3FA	ISD2FA
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	MDIS	Reserved	Reserved	0	0	0	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved		SWRSTHD	SWRSTSD
W					CLR_TXF	CLR_RXF										
Reset	0	1	0	0	0	0	0	0	0	1	0	0	1	1	0	0

Fields

Field	Function
31 —	Reserved
30 —	Reserved
29-28 —	Reserved
27 —	Reserved
26 —	Reserved
25-24 DQS_FA_SEL	DQS clock for sampling read data at flash memory A Selects DQS clock for sampling read data at flash memory A QuadSPI port 00b - Reserved 01b - Pad loopback 10b - Reserved 11b - Reserved
23 —	Reserved
22 —	Reserved
21-20 —	Reserved
19 —	Reserved
18 —	Reserved
17 ISD3FA	Idle signal drive IOFA[3] flash memory A Determines the logic level that the IOFA[3] output of the QuadSPI module is driven to in the inactive state. See Driving flash memory control signals in single and dual modes for details. 0b - IOFA[3] is driven to logic L

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - IOFA[3] is driven to logic H
16 ISD2FA	Idle signal drive IOFA[2] flash memory A Determines the logic level that the IOFA[2] output of the QuadSPI module is driven to in the inactive state. See Driving flash memory control signals in single and dual modes for details. 0b - IOFA[2] is driven to logic L. 1b - IOFA[2] is driven to logic H.
15 —	Reserved
14 MDIS	Module disable Allows the clock to the non-memory mapped logic in the QuadSPI to be stopped. 0b - Enable QuadSPI clocks 1b - Allow external logic to disable QuadSPI clocks
13 —	Reserved
12 —	Reserved
11 CLR_TXF	Clear TX FIFO/buffer This is a self-clearing field that invalidates the TX buffer content. **NOTE** Software must wait for at least five system cycles and three flash cycles after writing '1' to this field. 0b - No action 1b - Read and write pointers of the TX buffer are reset to 0 and TBSR[TRCTR] is reset to 0.
10 CLR_RXF	Clear RX FIFO This is a self-clearing field that invalidates the RX buffer content. 0b - No action 1b - Read and write pointers of the RX buffer are reset to 0 and RBSR[RDBFL] is reset to 0.
9 —	Reserved
8 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
7 —	Reserved
6 —	Reserved
5 —	Reserved
4 —	Reserved
3-2 —	Reserved
1 SWRSTHD	Software reset for AHB domain 0b - De-assert Software reset 1b - AHB domain flops are reset. This field does not reset configuration registers. It is advisable to reset both the serial flash memory domain and AHB domain at the same time. Resetting only one domain might lead to side effects. **NOTE** The software resets need the clock to be running to propagate to the design. The value of MCR[MDIS] should be 0 when the software reset bits are asserted. Also, before they can be deasserted again (by setting MCR[SWRSTHD] to 0), it is recommended to set the value of MCR[MDIS] to 1. After the software resets have been deasserted, the normal operation can be started by setting MCR[MDIS] to 0. **NOTE** Software must wait for at least three system cycles and three flash cycles after changing the value of this field.
0 SWRSTSD	Software reset for serial flash memory domain 0b - De-assert Software reset 1b - Serial flash memory domain flops are reset. This field does not reset configuration registers. It is advisable to reset both the serial flash memory domain and AHB domain at the same time. Resetting only one domain might lead to side effects.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE The software resets need the clock to be running to propagate to the design. The value of MCR[MDIS] should therefore be 0 when the software reset bits are asserted. Also, before they can be deasserted again (by specifying 0 as the value for MCR[SWRSTSD]), it is recommended to specify 1 as the value for MCR[MDIS]. After the software resets are deasserted, the normal operation can be started by specifying 0 as the value for MCR[MDIS]. **NOTE** Software must wait for at least three system cycles and three flash cycles after changing the value of this field.

78.11.2.3 IP Configuration Register (IPCR)

Offset

Register	Offset
IPCR	8h

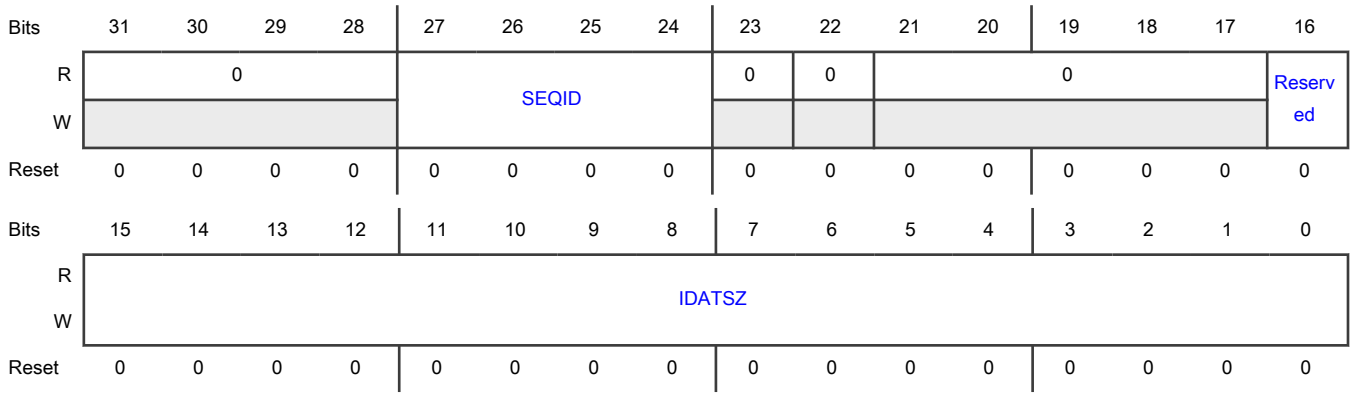
Function

This register provides all the configuration required for an IP-initiated command, which can be triggered by writing in the SEQID field of this register. If the SEQID field is written successfully, a new command to the external serial flash memory is initiated per the sequence pointed to by this field. See Normal mode for details on command triggering and command execution.

Special write-access is permitted if:

- SR[IP_ACC]=0

Diagram



Fields

Field	Function
31-28 —	Reserved
27-24 SEQID	Points to a sequence in the LUT This field contains the sequence index of the LUT. See LUT for details. Each sequence index can accommodate up to 10 instructions (2 instructions per register). A write to this field triggers a transaction on the serial flash memory interface.
23 —	Reserved
22 —	Reserved
21-17 —	Reserved
16 —	Reserved
15-0 IDATSZ	IP data transfer size This field defines the data transfer size, in bytes, of the IP command.

78.11.2.4 Flash Memory Configuration Register (FLSHCR)

Offset

Register	Offset
FLSHCR	Ch

Function

This register contains the timings that are specific to the flash memory device. The QuadSPI controller must meet these timings for the device to function correctly.

Special write-access is permitted if:

- SR[AHB_ACC] = 0
- SR[IP_ACC] = 0

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0														Reserved	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				TCSH				0				TCSS			
W																
Reset	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1	1

Fields

Field	Function
31-18 —	Reserved
17-16 —	Reserved
15-12 —	Reserved
11-8 TCSH	Serial flash memory CS hold time This hold time is in terms of serial flash memory clock cycles, and it must be greater than or equal to five flash memory clock cycles. Refer the chip datasheet for the exact value.
7-4 —	Reserved
3-0 TCSS	Serial flash memory CS setup time This setup time is in terms of serial flash memory clock cycles, and it must be greater than or equal to two flash memory clock cycles. Refer the chip Datasheet for the exact value.

78.11.2.5 Buffer 0 Configuration Register (BUF0CR)

Offset

Register	Offset
BUF0CR	10h

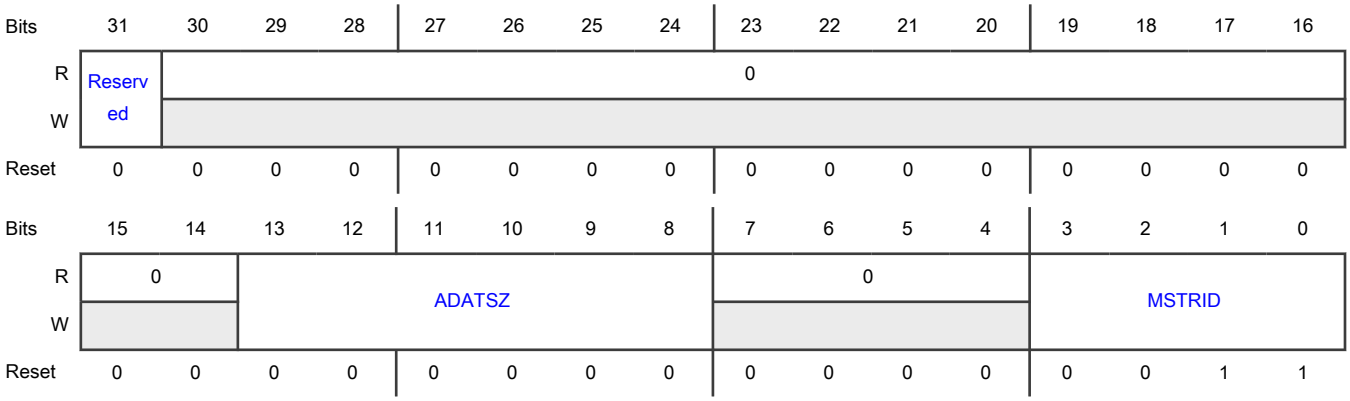
Function

This register provides the configuration for any read access routed to buffer0, which happens when the master ID of the incoming AHB request matches BUF0CR[MSTRID]. Any buffer "miss" leads to a serial flash memory transaction being triggered per the sequence pointed to by BFGENCR[SEQID].

Special write-access is permitted if:

- SR[AHB_ACC] = 0

Diagram



Fields

Field	Function
31 —	Reserved
30-14 —	Reserved
13-8 ADATSZ	AHB data transfer size Defines the read data transfer size in 8 bytes of an AHB triggered read access to serial flash memory. For example, a value of 0x2 sets transfer size to 16 bytes. When ADATSZ = 0, the data size mentioned in the sequence pointed to by the SEQID field overrides this value. The software should ensure that this transfer size is not greater than the size of the buffer.
7-4 —	Reserved
3-0 MSTRID	Master ID ID of the AHB master associated with BUFFER 0 Any AHB read access with this master ID is routed to this buffer. You must ensure that the master IDs associated with all buffers are different. **NOTE** See the chip-specific QuadSPI information for details about master IDs and their corresponding components.

78.11.2.6 Buffer 1 Configuration Register (BUF1CR)

Offset

Register	Offset
BUF1CR	14h

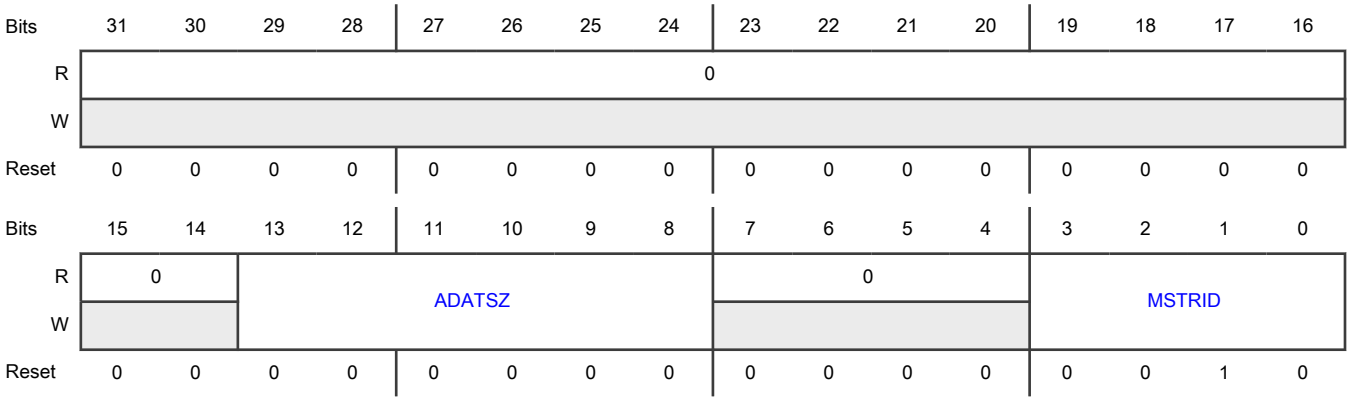
Function

This register provides the configuration for any access routed to buffer 1, which happens when the master ID of the incoming AHB request matches the MSTRID field of this register. Any buffer "miss" leads to the buffer being flushed and a serial flash memory transaction being triggered per the sequence pointed to by BFGENCR[SEQID].

Special write-access is permitted if:

- SR[AHB_ACC] = 0

Diagram



Fields

Field	Function
31-14 —	Reserved
13-8 ADATSZ	AHB data transfer size This field defines the read data transfer size in 8 bytes of an AHB triggered read access to serial flash memory. For example, a value of 0x2 sets the transfer size to 16 bytes. When ADATSZ = 0, the data size mentioned in the sequence pointed to by the SEQID field overrides this value. Software should ensure that this transfer size is not greater than the size of this buffer.
7-4 —	Reserved
3-0 MSTRID	Master ID ID of the AHB master associated with BUFFER 1

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Any AHB read access with this master ID is routed to this buffer. You must ensure that the master IDs associated with all buffers are different. **NOTE** See the chip-specific QuadSPI information for details about master IDs and their corresponding components.

78.11.2.7 Buffer 2 Configuration Register (BUF2CR)

Offset

Register	Offset
BUF2CR	18h

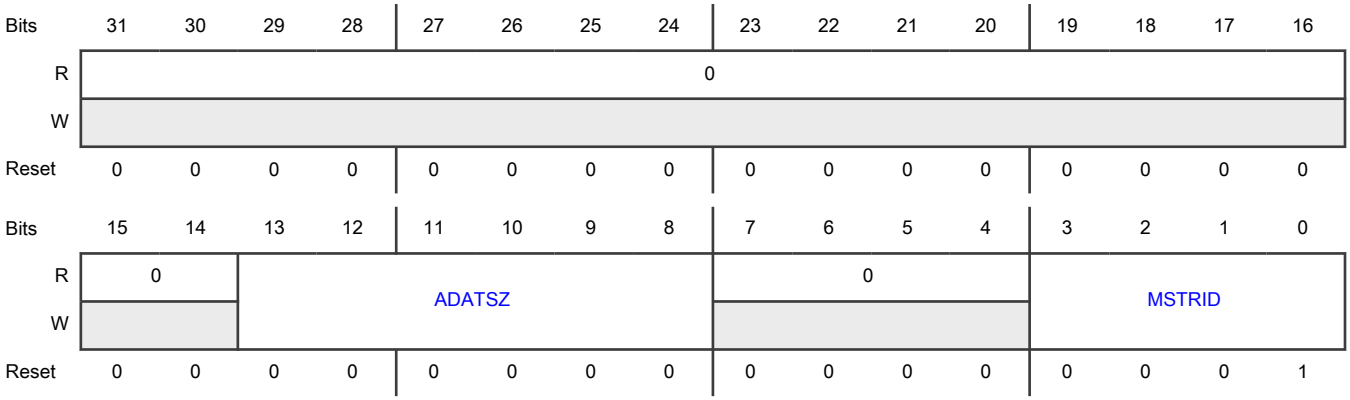
Function

This register provides the configuration for any access routed to buffer 2, which happens when the master ID of the incoming AHB request matches the MSTRID field of this register. Any buffer "miss" leads to the buffer being flushed and a serial flash memory transaction being triggered per the sequence pointed to by BFGENCR[SEQID].

Special write-access is permitted if:

- SR[AHB_ACC] = 0

Diagram



Fields

Field	Function
31-14 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
13-8 ADATSZ	AHB data transfer size This field defines the read data transfer size in 8 bytes of an AHB triggered read access to the serial flash memory. For example, a value of 0x2 sets transfer size to 16 bytes. When ADATSZ = 0, the data size mentioned in the sequence pointed to by the SEQID field overrides this value. The software should ensure that this transfer size is not greater than the size of this buffer.
7-4 —	Reserved
3-0 MSTRID	Master ID The ID of the AHB master associated with BUFFER2. Any AHB read access with this master ID is routed to this buffer. It must be ensured that the master IDs associated with all buffers are different. **NOTE** See the chip-specific QuadSPI information for details about master IDs and their corresponding components.

78.11.2.8 Buffer 3 Configuration Register (BUF3CR)

Offset

Register	Offset
BUF3CR	1Ch

Function

This register provides the configuration for any access to buffer 3.

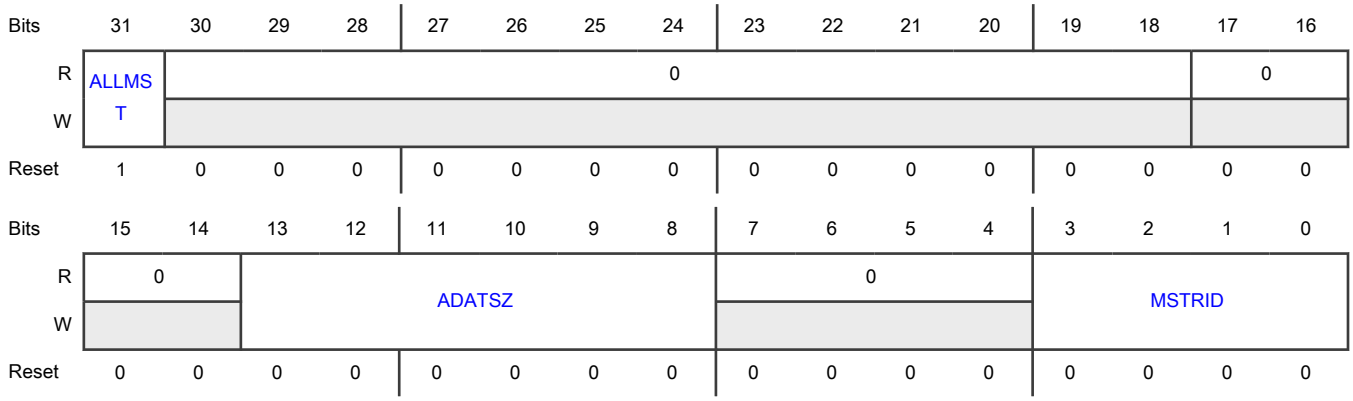
An access is routed to buffer 3 when the master ID of the incoming AHB request matches the MSTRID field of BUF3CR. Any buffer "miss" leads to the buffer being flushed and a serial flash memory transaction being triggered per the sequence pointed to by BFGENCR[SEQID].

In case the value of the ALLMST field is not 1, any such transaction (where master ID does not match any of the MSTRID fields) is returned with an ERROR response.

Special write-access is permitted if:

- SR[AHB_ACC] = 0

Diagram



Fields

Field	Function
31 ALLMST	All master enable When set, buffer3 acts as an all-master buffer. Any AHB access with a master ID not matching with the master ID of buffer0, buffer1, or buffer2 is routed to buffer3. When set, the MSTRID field of this register is ignored.
30-18 —	Reserved
17-14 —	Reserved
13-8 ADATSZ	AHB data transfer size Defines the read data transfer size in 8 bytes of an AHB triggered read access to serial flash memory. When ADATSZ = 0, the data size mentioned in the sequence pointed to by the SEQID field overrides this value.
7-4 —	Reserved
3-0 MSTRID	Master ID ID of the AHB master associated with BUFFER 3. Any AHB read access with this master ID is routed to this buffer. You must ensure that the master IDs associated with all buffers are different. **NOTE** See the chip-specific QuadSPI information for details about master IDs and their corresponding components.

78.11.2.9 Buffer Generic Configuration Register (BFGENCR)

Offset

Register	Offset
BFGENCR	20h

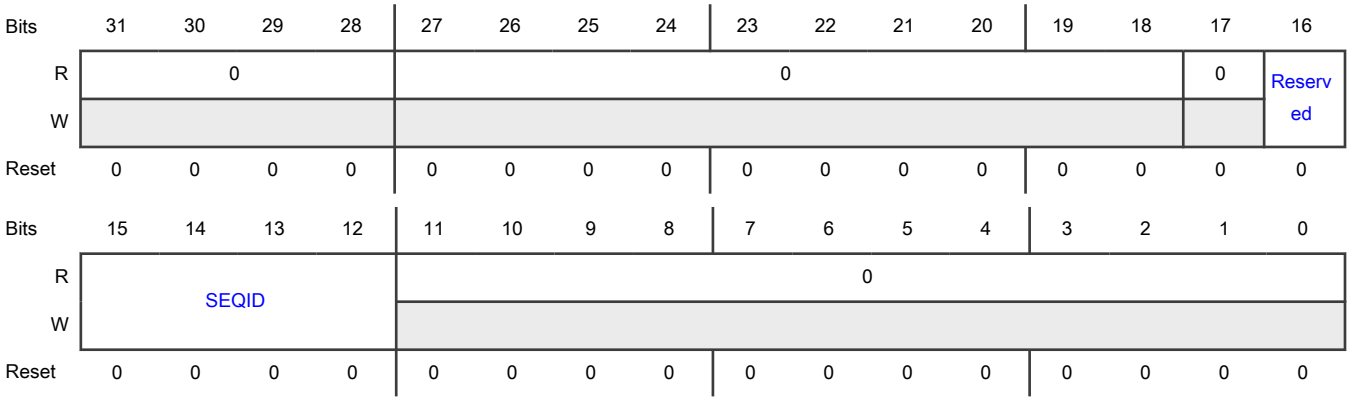
Function

This register provides generic configuration to any of the buffer accesses. Any buffer "miss" leads to the buffer being flushed and a serial flash memory transaction being triggered per the sequence pointed to by the SEQID field.

Special write-access is permitted if:

- SR[AHB_ACC] = 0

Diagram



Fields

Field	Function
31-28 —	Reserved
27-18 —	Reserved
17 —	Reserved
16 —	Reserved
15-12 SEQID	Points to a sequence in the LUT. This field contains the sequence index of the LUT... See LUT.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE If the sequence pointer differs in the new and the previous sequences, you should reset it. See sequence pointer clear register for more information.
11-0 —	Reserved

78.11.2.10 SOC Configuration Register (SOCCR)

Offset

Register	Offset
SOCCR	24h

Function

This register is programmed at the chip level for QuadSPI configuration. For details, see chip-specific QuadSPI information.

Special write-access is permitted if:

- SR[AHB_ACC] = 0

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SOCCFG															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SOCCFG															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 SOCCFG	SOC configuration This field configuration is specific to chip. For details, see chip-specific QuadSPI information.

78.11.2.11 Buffer 0 Top Index Register (BUF0IND)

Offset

Register	Offset
BUF0IND	30h

Function

This register specifies the top index for buffer 0, which defines its size. Note that the three LSBs of this register are set to 0. This ensures that the buffer is 64-bit aligned because each buffer entry is 64-bits long.

The register value should be set to the desired number of bytes. For example, setting BUF0IND[31:3] to 0 gives 0 bytes, setting the value to 1 gives 8 bytes, and so on.

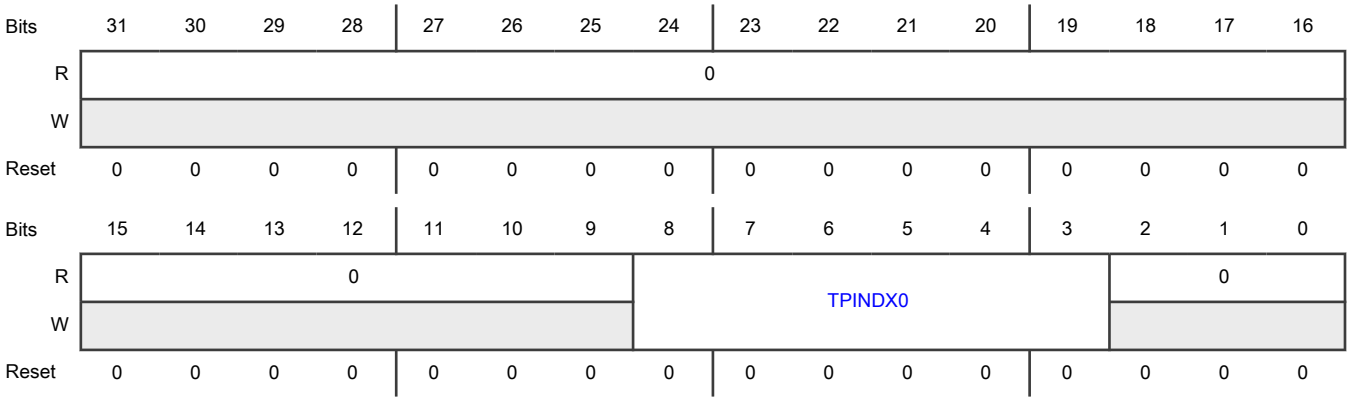
The size of buffer 0 is the difference between BUF0IND and 0.

The software must ensure that the value of TPINDX0 is not greater than the size of buffer 0.

Special write-access is permitted if:

- SR[AHB_ACC] = 0

Diagram



Fields

Field	Function
31-9 —	Reserved
8-3 TPINDX0	Top index of buffer 0
2-0 —	Reserved

78.11.2.12 Buffer 1 Top Index Register (BUF1IND)

Offset

Register	Offset
BUF1IND	34h

Function

This register specifies the top index of buffer 1, which defines its size. Note that the three LSBs of this register are set to 0. This ensures that the buffer is 64-bit aligned because each buffer entry is 64-bits long.

The size of buffer 1 is the difference between BUF1IND and BUF0IND. The register value should be entered in bytes. For example, if BUF0IND = 0x100, then setting BUF1IND = 0x130 sets the size of buffer 1 to 0x30 bytes.

The software must ensure that the value of TPINDX1 is not greater than the size of buffer 1.

Special write-access is permitted if:

- SR[AHB_ACC] = 0

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				TPINDX1								0			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-9 —	Reserved
8-3 TPINDX1	Top index of buffer 1
2-0 —	Reserved

78.11.2.13 Buffer 2 Top Index Register (BUF2IND)

Offset

Register	Offset
BUF2IND	38h

Function

This register specifies the top index of buffer 2, which defines its size. Note that the three LSBs of this register are set to 0. This ensures that the buffer is 64-bit aligned because each buffer entry is 64-bits long.

The size of buffer 2 is the difference between BUF2IND and BUF1IND. The register value should be entered in bytes. For example, if BUF1IND = 0x130 then setting BUF2IND = 0x180 sets the size of buffer 2 to 0x50 bytes.

The software must ensure that the value of TPINDX2 is not greater than the size of buffer 2.

Special write-access is permitted if:

- SR[AHB_ACC] = 0

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				TPINDX2								0			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-9 —	Reserved
8-3 TPINDX2	Top index of buffer 2
2-0 —	Reserved

78.11.2.14 DLL Flash Memory A Configuration Register (DLLCRA)

Offset

Register	Offset
DLLCRA	60h

Function

This register configures slave delay chain for flash memory A.

See Delay chain usage for the programming sequence.

NOTE

See the chip data sheet for information on programming register fields.

NOTE

Please see the chip specific section of QuadSPI DLLCRA register for delay elements data

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved	FREQEN	0		Reserved				Reserved				SLV_FINE_OFFSET			
W																
Reset	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	SLV_DLY_OFFSET			SLV_DLY_COARSE				Reserved			0	Reserved	SLV_EN	SLV_DLL...	SLV_UPD
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30 FREQEN	Frequency enable 0b - Selects delay chain for low frequency of operation 1b - Selects delay chain for high frequency of operation
29-28 —	Reserved
27-24	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
23-20 —	Reserved
19-16 SLV_FINE_OFFSET	Fine offset delay elements in incoming DQS This field sets the number of fine offset delay elements up to 16 in incoming DQS, and the default must be 1 element.
15 —	Reserved
14-12 SLV_DLY_OFFSET	T/16 offset delay elements in incoming DQS This field sets the number of T/16 offset delay elements in incoming DQS; default is 0.
11-8 SLV_DLY_COARSE	Delay elements in each delay tap This field sets the number of delay elements in each delay tap. The field is used to overwrite DLL-generated delay values and works when the value of SLV_DLL_BYPASS is 1. Note : Please refer to the QuadSPI datasheet for more details.
7-5 —	Reserved
4 —	Reserved
3 —	Reserved
2 SLV_EN	Slave enable 0b - DLL slave logic remains in reset, and its value should be 0 for at least three flash memory clock cycles for reset. 1b - Enables DQS slave delay chain, and should be 1 before any slave configuration settings take place.
1 SLV_DLL_BYPASS	Slave DLL bypass This field enables selection of the number of delays in each slave delay tap. 0b - Disables manual selection of coarse delays in the slave delay chain. 1b - Enables selection of number of delays in each slave delay tap, based on DLLCRA[SLV_DLY_COARSE].
0	Slave update

Table continues on the next page...

Table continued from the previous page...

Field	Function
SLV_UPD	You must program this field only after slave delay chain configuration takes place. 0b - Disables any further update on DQS slave delay chain. 1b - Updates the DQS slave delay chain with either ref-delay or bypass slave delay value, and should be set in the absence of the DQS clock.

78.11.2.15 Serial Flash Memory Address Register (SFAR)

Offset

Register	Offset
SFAR	100h

Function

The module automatically translates this address on the memory map to the address on the flash memory. When operating in a 24-bit mode, only bits 23-0 are sent to the flash memory. In the 32-bit mode, bits 27-0 are used with bits 31-28 driven to 0 . See Table 721 for the mapping between the access mode and the SFAR content and Normal mode for details on command triggering and command execution. The software must ensure that the serial flash memory address provided in the SFAR register lies in the valid flash memory address range, as defined in Table 721.

Special write-access is permitted if:

- SR[IP_ACC] = 0

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SFADR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SFADR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 SFADR	Serial flash memory address

78.11.2.16 Sampling Register (SMPR)

Offset

Register	Offset
SMPR	108h

Function

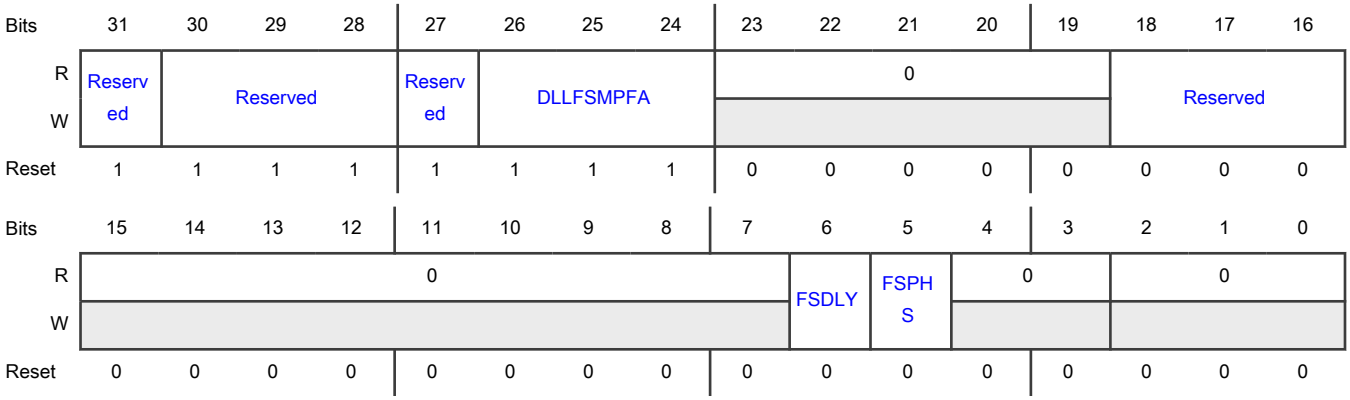
This register allows configuration of how the incoming data from the external serial flash memory devices is sampled in the QuadSPI module.

NOTE

See the chip data sheet for programming the register fields.

Special write-access is permitted in the disabled mode.

Diagram



Fields

Field	Function
31 —	Reserved
30-28 —	Reserved
27 —	Reserved
26-24 DLLFSMPFA	Selects the nth tap provided by slave delay chain for flash memory A The value of *n* can vary from 0 to 7, with each tap delay based on the DLLCRA register.
23-19	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
18-16 —	Reserved
15-7 —	Reserved
6 FSDLY	Full speed delay selection for SDR instructions Select the delay with respect to the reference edge for the sample point valid for full speed commands. 0b - One clock cycle delay 1b - Two clock cycles delay
5 FSPHS	Full-speed phase selection for SDR instructions This field selects the edge of the sampling clock valid for full-speed commands. 0b - Select sampling at non-inverted clock 1b - Select sampling at inverted clock
4-3 —	Reserved
2-0 —	Reserved

78.11.2.17 RX Buffer Status Register (RBSR)

Offset

Register	Offset
RBSR	10Ch

Function

This register contains information related to the receive data buffer.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RDCTR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								RDBFL							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 RDCTR	Read counter Indicates the number of 4-byte entries removed from the RX buffer. For example, a value of 0x2 indicates that 8 bytes have been removed. It is incremented by the number (RBCT[WMRK] + 1) on RX buffer POP event. The RX buffer can be popped using DMA or FR[RBDF]. The RSER[RBDDE] defines which pop should be pursued. For details, see AHB RX Data Buffer Register (ARDB0 - ARDB127) and Data Transfer from the QuadSPI Module Internal Buffers.
15-8 —	Reserved
7-0 RDBFL	RX buffer fill level Indicates the number of 4-byte entries available in the RX buffer. For example, a value of 0x2 indicates 8 bytes are available.

78.11.2.18 RX Buffer Control Register (RBCT)

Offset

Register	Offset
RBCT	110h

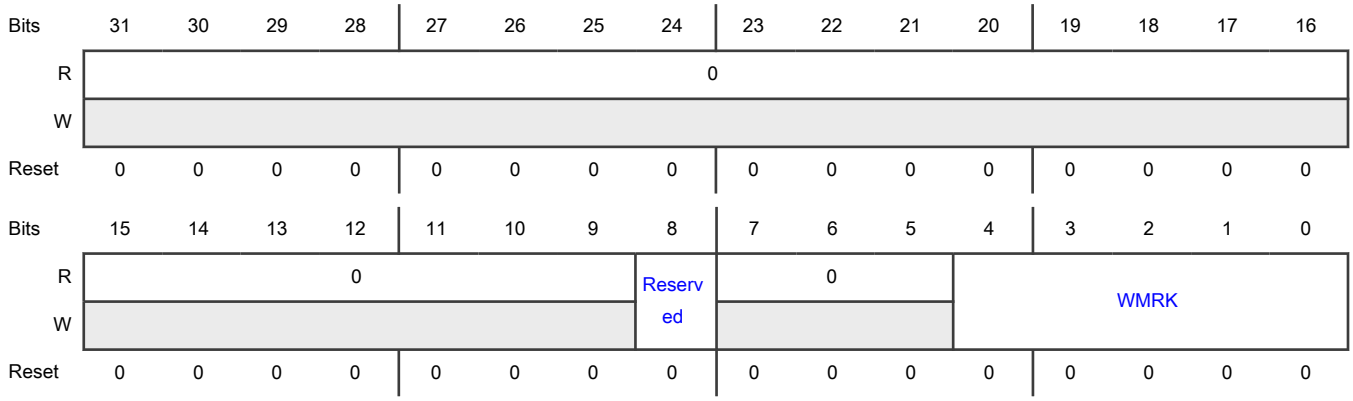
Function

This register contains control data related to the receive data buffer.

Special write-access is permitted if:

- SR[IP_ACC] = 0

Diagram



Fields

Field	Function
31-9 —	Reserved
8 —	Reserved
7-5 —	Reserved
4-0 WMRK	RX buffer watermark This field determines when the readout action of the RX buffer is triggered. When the number of valid entries in the RX buffer is equal to or greater than the number provided by (WMRK+1), the SR[RXWE] flag is asserted. The value should be entered as the number of 4-byte entries minus 1. For example, a value of 0x0 sets the watermark to 4 bytes, 1 to 8bytes, 2 to 12 bytes, and so on. For details, see DMA usage. **NOTE** This field should never be programmed above 63 because there are only 64 memory mapped RBDR registers. If watermark is programmed above 63, data above 64 words will be lost.

78.11.2.19 Data Learning Status Flash Memory A Register (DLSR_FA)

Offset

Register	Offset
DLSR_FA	134h

Function

This register shows sampling point selected by data learning algorithm when the value of DLSR_FA[DLPFFA] is 0. Otherwise, it shows the pattern matching outline.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DLPFFA	0														
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	POS_EDGE								NEG_EDGE							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 DLPFFA	Data learning pattern fail This field asserts when data learning fails at flash memory A.
30-16 —	Reserved
15-8 POS_EDGE	DLP positive edge match signature for flash memory A
7-0 NEG_EDGE	DLP negative edge match signature for flash memory A

78.11.2.20 TX Buffer Status Register (TBSR)

Offset

Register	Offset
TBSR	150h

Function

This register contains information related to the transmit data buffer.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TRCTR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								0				TRBFL			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 TRCTR	Transmit counter This field indicates how many entries of 4 bytes have been written into the TX buffer by host accesses. It is reset to 0 when a 1 is written to MCR[CLR_TXF]. It is incremented on each write access to the TBDR register when another word has been pushed onto the TX buffer. When it is not cleared, the TRCTR field wraps around to 0. See TX Buffer Data Register (TBDR) for details.
15-9 —	Reserved
8-6 —	Reserved
5-0 TRBFL	TX buffer fill level This field contains the number of entries of 4 bytes each available in the TX buffer for the QuadSPI module to transmit to the serial flash memory device. The value of this field can reach maximum up to the total TX buffer size.

78.11.2.21 TX Buffer Data Register (TBDR)

Offset

Register	Offset
TBDR	154h

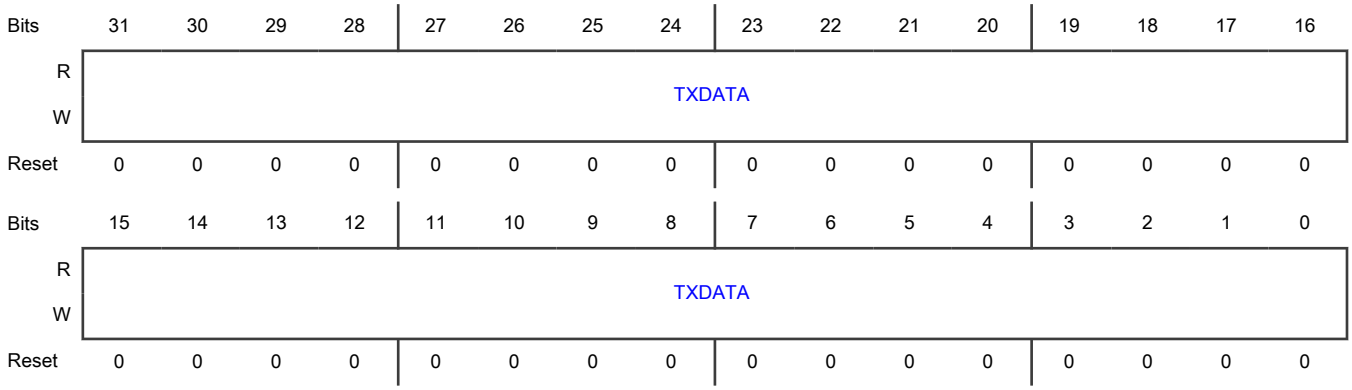
Function

This register provides access to the circular TX buffer of depth 32 , so the total size is 32 * 4 bytes. This buffer provides the data written into it as write data for the page programming commands to the serial flash memory device. See Table 701 for the byte ordering scheme. A write transaction on the flash memory with data size of less than 32 bits leads to the removal of one data entry from the TX buffer. The valid bits are used and the rest of the bits are discarded.

Special write-access is permitted if:

- SR[TXFULL] = 0

Diagram



Fields

Field	Function
31-0 TXDATA	TX data On write access, the data is written to the next available entry of the TX buffer and TBSR[TRBFL] is updated accordingly. On a read access, the last data written to the register is returned.

78.11.2.22 TX Buffer Control Register (TBCT)

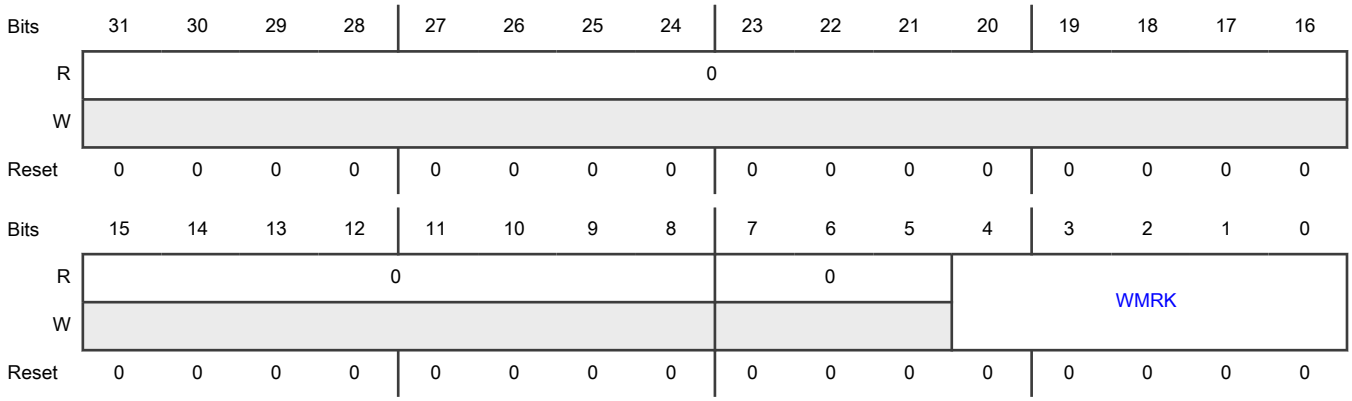
Offset

Register	Offset
TBCT	158h

Function

This register contains control information for transmit data buffer.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-5 —	Reserved
4-0 WMRK	Watermark for TX buffer Determines the watermark for the TX buffer When the number of available space in the TX buffer is greater than or equal to the number provided by WMRK (number of 4-byte entries), SR[TXWA] is asserted. For example, a value of 0x1 sets the watermark to 4 bytes, 0x2 sets it to 8 bytes, 0x3 sets it to 12 bytes, and so on. For details, see DMA usage. WMRK = 0 is invalid.

78.11.2.23 Status Register (SR)

Offset

Register	Offset
SR	15Ch

Function

This register provides all the available status information about SFM command execution and arbitration, the RX buffer, TX buffer, and the AHB buffer.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0			0	TXFULL	TXDMA	TXWA	TXNE	RXDMA	0			RXFULL	0		RXWE
W																
Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	AHB3FUL	AHB2FUL	AHB1FUL	AHB0FUL	AHB3NE	AHB2NE	AHB1NE	AHB0NE	AHBTRN	Reserved	0	Reserved	AHB_ACC	IP_ACC	BUSY
W																
Reset	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	Reserved
28 —	Reserved
27 TXFULL	TX buffer full Asserted when the FIFO level reaches 39 (that is, TX buffer size of 32 + async FIFO size of 7)
26 TXDMA	TX DMA Asserted when the TXFIFO fill via DMA is active and DMA is requested or running
25 TXWA	TX buffer watermark available Asserted when the number of available spaces in the TX buffer is greater than or equal to the value provided by TBCT[WMRK] Example: When TBCT[WMRK]=1, SR[TXWA] is de-asserted when TX FIFO has 32+7(size of async FIFO) entries
24 TXNE	TX buffer not empty Asserted when TX buffer contains data
23 RXDMA	RX buffer DMA Asserted when RX buffer read out via DMA is active; that is, when DMA is requested or running
22-20 —	Reserved
19 RXFULL	RX buffer full Asserted when the RX buffer is full; that is, when RBSR[RDBFL] is equal to 128
18-17 —	Reserved
16 RXWE	RX buffer watermark exceeded Asserted when the number of valid entries in the RX buffer exceeds the number provided in RBCT[WMRK]
15 —	Reserved
14 AHB3FUL	AHB 3 buffer full Asserted when AHB 3 buffer is full
13	AHB 2 buffer full

Table continues on the next page...

Table continued from the previous page...

Field	Function
AHB2FUL	Asserted when AHB 2 buffer is full
12 AHB1FUL	AHB 1 buffer full Asserted when the AHB 1 buffer is full
11 AHB0FUL	AHB 0 buffer full Asserted when the AHB 0 buffer is full
10 AHB3NE	AHB 3 buffer not empty Asserted when the AHB 3 buffer contains data
9 AHB2NE	AHB 2 buffer not empty Asserted when the AHB 2 buffer contains data
8 AHB1NE	AHB 1 buffer not empty Asserted when the AHB 1 buffer contains data
7 AHB0NE	AHB 0 buffer not empty Asserted when the AHB 0 buffer contains data
6 AHBTRN	AHB access transaction pending Asserted when there is a pending request on the AHB interface. See Flash memory mapped AMBA bus.
5 —	Reserved
4 —	Reserved
3 —	Reserved
2 AHB_ACC	AHB read access Asserted when the currently executed transaction is initiated by the AHB bus
1 IP_ACC	IP access Asserted when transaction currently executed is initiated by the IP bus
0 BUSY	Module busy Asserted when module is currently busy handling a transaction to an external flash memory device

78.11.2.24 Flag Register (FR)

Offset

Register	Offset
FR	160h

Function

This register provides all available flags about SFM command execution and arbitration, which may serve as the source for the generation of interrupt service requests. Note that the error flags in this register do not relate directly to the execution of the transaction in the serial flash memory device itself but only to the behavior and conditions visible in the QuadSPI module.

Special write-access is permitted in the enabled mode.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved	0	Reserved	0	TBFF	TBUF	0	0	ILLINE	0		0	0		RBOF	RBDF
W					W1C	W1C			W1C						W1C	W1C
Reset	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	AITEF	AIBSEF	ABOF	Reserved	0	0	0	IPAEF	IPIEF	0	0	0			TFF
W		W1C	W1C	W1C	W1C				W1C	W1C						W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30 —	Reserved
29 —	Reserved
28 —	Reserved
27 TBFF	TX buffer fill flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Before writing to the TX buffer, this field should be cleared. Then, it should be read back. If it is set, the TX buffer can include more data. If the field remains cleared, the TX buffer can be considered as full. See TX buffer operation for details.
26 TBUF	TX buffer underrun flag This field is set if the module tries to pull data when the TX buffer is empty. The IP command leading to the TX buffer underrun is continued (data sent to the serial flash memory device is undefined). Here, a valid underrun means that it should have occurred during the transaction so that few bytes (that is, less than 4 bytes) are left in FIFO and the remaining are filled with "FFFFh". The software should initiate a TX transaction only when 128 bits are written in the TX buffer. This field does not set if transfer is less than 128 bits. The application must clear the TX buffer in response to this event by writing a 1 to MCR[CLR_TXF]. The application must clear the TX buffer in response to this event by writing a 1 to MCR[CLR_TXF].
25 —	Reserved
24 —	Reserved
23 ILLINE	Illegal instruction error flag This field is set when an illegal instruction is encountered by the controller in any of the sequences. As soon as the field is set, you must assert MCR[SWRSTSD] and MCR[SWRSTHD]. That is, reset the flash memory and AHB domain after reconfiguring the correct sequence instruction. See Table 699 for a list of legal instructions.
22-21 —	Reserved
20 —	Reserved
19-18 —	Reserved
17 RBOF	RX buffer overflow flag This field is set when no more data can be pushed into the RX buffer from the serial flash memory device. The IP command leading to this condition is continued until the number of bytes in IPCR[IDATSZ] are read from the serial flash memory device. The content of the RX buffer remains unchanged.
16 RBDF	RX buffer drain flag This field is set if SR[RXWE] is asserted. Writing 1 to this field triggers one of the following actions:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• If the RX buffer has up to RBCT[WMRK] valid entries, then the flag is cleared. • If the RX buffer has more than RBCT[WMRK] valid entries and the RSER[RBDDE] field is not set (flag driven mode), an RX buffer POP event is triggered. The flag remains set if the RX buffer contains more than RBCT[WMRK] valid entries after the RX buffer POP event is complete. The flag is cleared if the RX buffer contains less than or equal to RBCT[WMRK] valid entries after the RX buffer POP event is complete. See the "Receive Buffer Drain Interrupt or DMA Request" section in Normal mode interrupt and DMA requests for details.
15 —	Reserved
14 AITEF	AHB illegal transaction error flag This is set whenever there is no response generated from QuadSPI to AHB bus in case of an illegal transaction and the watchdog timer expires. The timer value is considered as a parameter.
13 AIBSEF	AHB illegal burst size error flag This is set whenever the total burst size (size x beat) of an AHB transaction is greater than the prefetch data size, which is defined by BUFxCR[ADATSZ] or data size mentioned in the sequence pointed to by the SEQID field in case ADATSZ = 0. See HBURST support with AHB read details on HBURST feature.
12 ABOF	AHB buffer overflow flag This is set when the size of the AHB access exceeds the size of the AHB buffer. This condition can occur only if BUFxCR[ADATSZ] is programmed incorrectly. The AHB command leading to this condition is continued until the number of entries according to BUFxCR[ADATSZ] have been read from the serial flash memory device. The content of the AHB buffer is not changed.
11 —	Reserved
10 —	Reserved
9 —	Reserved
8 —	Reserved
7 IPAEF	IP command trigger during AHB access error flag This is set when the following condition occurs:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• A write access occurs to IPCR[SEQID] and the SR[AHB_ACC] field is set. Any command leading to the assertion of the IPAEF field is ignored.
6 IPIEF	IP command trigger could not be executed error flag This is set when the SR[IP_ACC] and SR[AWRACC] fields are set (that is, an IP triggered command is currently executing) and any of the following conditions occurs: • Write access to the IPCR. Any command leading to the assertion of the IPIEF flag is ignored. • Write access to the SFAR • Write access to the RBCT
5 —	Reserved
4 —	Reserved
3-1 —	Reserved
0 TFF	IP command transaction finished flag This field is set after the QuadSPI module completes a running IP command. If an error occurs, and the related error flags are valid in the same clock cycle, the TFF flag is asserted.

78.11.2.25 Interrupt and DMA Request Select and Enable Register (RSER)

Offset

Register	Offset
RSER	164h

Function

This register provides enables and selectors for the interrupts in the QuadSPI module.

NOTE

Each field of the FR enabled as source for an interrupt prevents the QuadSPI module from entering the Stop mode or Module Disable mode when this flag is set.

Special write-access is permitted in the "Anytime" mode.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved	Reserved		0	TBFIE	TBUIE	TBFDE	Reserved	ILLINIE	Reserved	RBDDE	Reserved	0		RBOIE	RBDIE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	AITIE	AIBSIE	ABOIE	Reserved	Reserved	Reserved	Reserved	IPAEIE	IPIEIE	0	Reserved	0			TFIE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30-29 —	Reserved
28 —	Reserved
27 TBFIE	TX buffer fill interrupt enable flag This field indicates the TX buffer fill interrupt enable flag. 0b - No TBFF interrupt is generated. 1b - TBFF interrupt is generated.
26 TBUIE	TX buffer underrun interrupt enable flag This field indicates the TX buffer underrun interrupt enable flag. 0b - No TBUF interrupt is generated 1b - TBUF interrupt is generated
25 TBFDE	TX buffer fill DMA enable Enables generation of DMA requests for TX buffer fill. When the value of this field is 1, DMA requests are generated as long as number of available spaces in the TX buffer is greater than or equal to the value provided by TBCT[WMRK]. **NOTE** After you write 1 to this field (to enable DMA transfers), writing 0 does not disable DMA transfers. You must perform a software reset for the AHB domain by using MCR[SWRSTHD] to disable DMA transfers.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No DMA request is generated 1b - DMA request is generated
24 —	Reserved
23 ILLINIE	Illegal instruction error interrupt enable Triggered by the ILLINE flag in FR 0b - No ILLINE interrupt is generated. 1b - ILLINE interrupt is generated.
22 —	Reserved
21 RBDDE	RX buffer drain DMA enable This field enables generation of DMA requests for RX buffer drain. When the value of this field is 1, the DMA requests are generated as long as SR[RXWE] is set. **NOTE** After you write 1 to this field (to enable DMA transfers), writing 0 does not disable DMA transfers. You must perform a software reset for the AHB domain by using MCR[SWRSTHD] to disable DMA transfers. 0b - No DMA request is generated. 1b - DMA request is generated.
20 —	Reserved
19-18 —	Reserved
17 RBOIE	RX buffer overflow interrupt enable This field indicates the RX buffer overflow interrupt enable flag. 0b - No RBOF interrupt is generated. 1b - RBOF interrupt is generated.
16 RBDIE	RX buffer drain interrupt enable This field enables generation of IRQ requests for RX buffer drain. When the value of this field is 1, the interrupt is asserted as long as SR[RBDF] is set. 0b - No RBDF interrupt is generated. 1b - RBDF Interrupt is generated.

Table continues on the next page...

Table continued from the previous page...

Field	Function
15 —	Reserved
14 AITIE	AHB illegal transaction interrupt enable flag This field indicates the AHB illegal transaction interrupt enable flag. 0b - No AITEF interrupt is generated. 1b - AITEF interrupt is generated.
13 AIBSIE	AHB illegal burst size interrupt enable flag This field indicates the AHB illegal burst size interrupt enable flag. 0b - No AIBSEF interrupt is generated. 1b - AIBSEF interrupt is generated.
12 ABOIE	AHB buffer overflow interrupt enable flag This field indicates the AHB buffer overflow interrupt enable flag. 0b - No ABOF interrupt is generated. 1b - ABOF interrupt is generated.
11 —	Reserved
10 —	Reserved
9 —	Reserved
8 —	Reserved
7 IPAEIE	IP command trigger during AHB read access error interrupt enable flag This field indicates IP command trigger during AHB read access error interrupt enable flag. 0b - No IPAEF interrupt is generated 1b - IPAEF interrupt is generated
6 IPIEIE	IP command trigger during IP access error interrupt enable flag This field indicates IP command trigger during IP access error interrupt enable flag. 0b - No IPIEF interrupt is generated 1b - IPIEF interrupt is generated

Table continues on the next page...

Table continued from the previous page...

Field	Function
5 —	Reserved
4 —	Reserved
3-1 —	Reserved
0 TFIE	Transaction finished interrupt enable flag This field indicates the transaction finished interrupt enable flag. 0b - No TFF interrupt is generated. 1b - TFF interrupt is generated.

78.11.2.26 Sequence Pointer Clear Register (SPTRCLR)

Offset

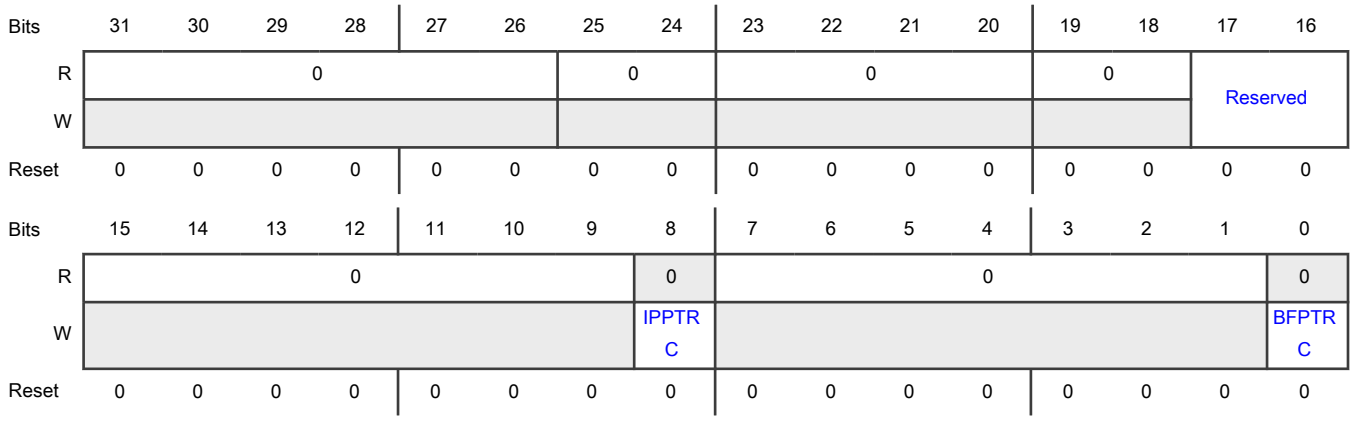
Register	Offset
SPTRCLR	16Ch

Function

This register provides fields to reset the IP and buffer sequence pointers. The sequence pointer contains the index of instructions within the LUT entry that is to be executed next. For example, if the LUT entry ends on a JMP_ON_CS value of 2, the index is stored as 2.

The software should reset the sequence pointers defined by JMP_ON_CS operand whenever the sequence ID is required to be changed by updating the SEQID field in the IPCR or BFGENCR.

Diagram



Fields

Field	Function
31-26 —	Reserved
25-24 —	Reserved
23-20 —	Reserved
19-18 —	Reserved
17-16 —	Reserved
15-9 —	Reserved
8 IPPTRC	IP pointer clear This is a self-clearing field. 1b - Clears the sequence pointer for IP accesses as defined in IPCR.
7-1 —	Reserved
0 BFPTRC	Buffer pointer clear This is a self-clearing field. 1b - Clears the sequence pointer for AHB read accesses as defined in BFGENCR.

78.11.2.27 Serial Flash Memory A1 Top Address Register (SFA1AD)

Offset

Register	Offset
SFA1AD	180h

Function

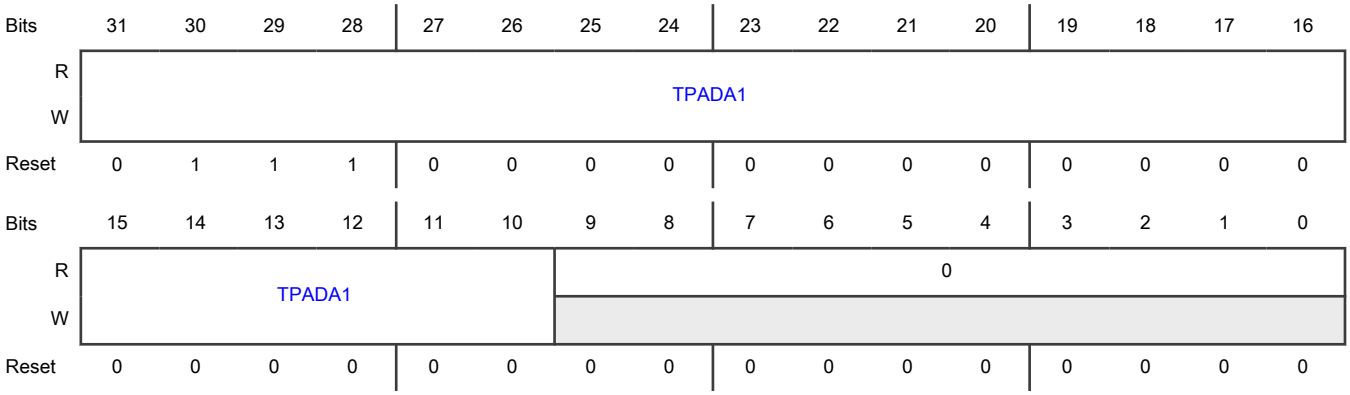
This register provides the address mapping for serial flash memory A1. The difference between SFA1AD[TPADA1] and AMBA_BASE defines the size of the memory map for serial flash memory A1.

Special write-access is permitted if:

- SR[IP_ACC] = 0

- SR[AHB_ACC] = 0

Diagram



Fields

Field	Function
31-10 TPADA1	Top address for serial flash memory A1 In effect, TPADxx is the first location of the next memory.
9-0 —	Reserved

78.11.2.28 Serial Flash Memory A2 Top Address Register (SFA2AD)

Offset

Register	Offset
SFA2AD	184h

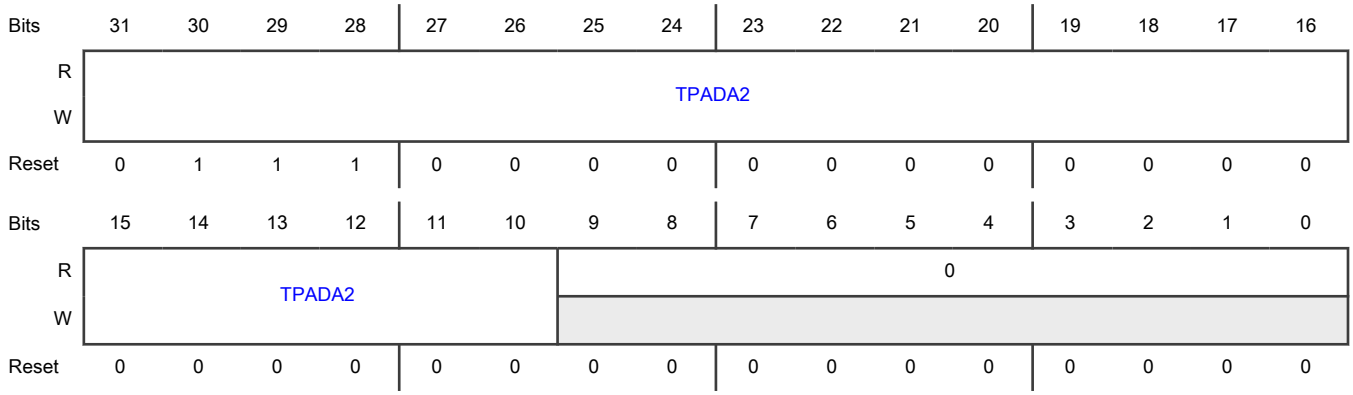
Function

This register provides the address mapping for serial flash memory A2. The difference between SFA2AD[TPADA2] and SFA1AD[TPADA1] defines the size of the memory map for serial flash memory A2.

Special write-access is permitted if:

- SR[IP_ACC] = 0
- SR[AHB_ACC] = 0

Diagram



Fields

Field	Function
31-10 TPADA2	Top address for serial flash memory A2 In effect, TPxxAD is the first location of the next memory.
9-0 —	Reserved

78.11.2.29 Serial Flash Memory B1 Top Address Register (SFB1AD)

Offset

Register	Offset
SFB1AD	188h

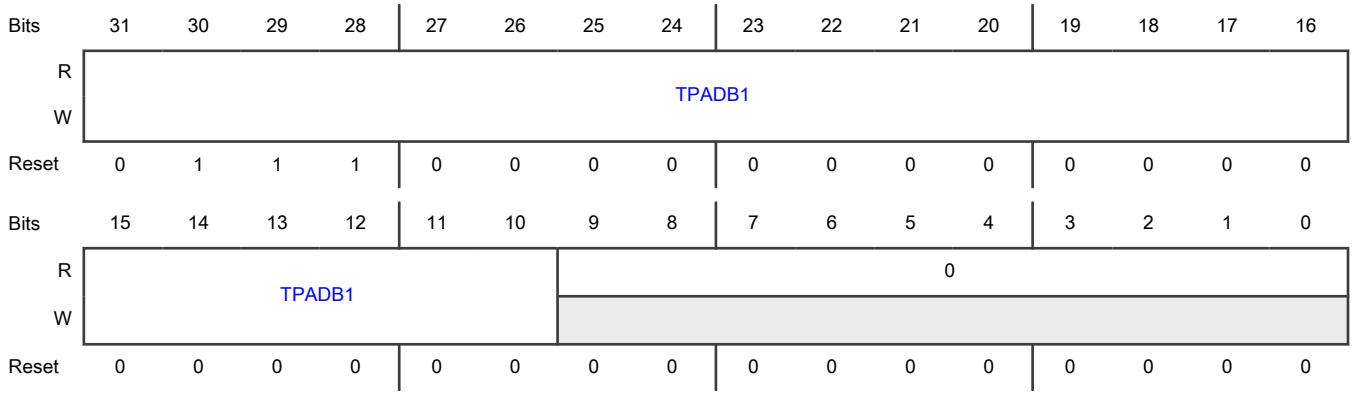
Function

This register provides the address mapping for serial flash memory B1. The difference between SFB1AD[TPADB1] and SFA2AD[TPADA2] defines the size of the memory map for serial flash memory B1.

Special write-access is permitted if:

- SR[IP_ACC] = 0
- SR[AHB_ACC] = 0

Diagram



Fields

Field	Function
31-10 TPADB1	Top address for serial flash memory B1. In effect, TPxxAD is the first location of the next memory.
9-0 —	Reserved

78.11.2.30 Serial Flash Memory B2 Top Address Register (SFB2AD)

Offset

Register	Offset
SFB2AD	18Ch

Function

This register provides the address mapping for serial flash memory B2. The difference between SFB2AD[TPADB2] and SFB1AD[TPADB1] defines the size of the memory map for serial flash memory B2.

Special write-access is permitted if:

- SR[IP_ACC] = 0
- SR[AHB_ACC] = 0

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TPADB2															
W																
Reset	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TPADB2						0									
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-10 TPADB2	Top address for serial flash memory B2. In effect, TPxxAD is the first location of the next memory.
9-0 —	Reserved

78.11.2.31 RX Buffer Data Register (RBDR0 - RBDR63)

Offset

For a = 0 to 63:

Register	Offset
RBDRa	200h + (a × 4h)

Function

These registers provide access to individual entries in the RX buffer. See Table 701 for the byte ordering scheme.

RBDR0 corresponds to the actual position of the read pointer within the RX buffer. The number of valid entries available depends on the number of RX buffer entries implemented and on the number of valid buffer entries available in the RX buffer.

Example 1 - RX buffer filled completely with 128 words: In this case, the address range for valid read access extends from RBDR0 to RBDR63.

Example 2 - RX buffer filled with five valid words: RX buffer fill level of RBSR[RDBFL] is 5. In this case, access to RBDR4 provides the last valid entry.

Any access beyond the range of valid RX buffer entries provides undefined results.

NOTE

To access data beyond RBDR[63], you must pop the data by using FR[RBDF]. See here for details.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RXDATA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXDATA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 RXDATA	RX data This field contains the data associated with the related RX buffer entry. For data format and byte ordering, see Byte ordering of serial flash memory read data.

78.11.2.32 LUT Key Register (LUTKEY)

Offset

Register	Offset
LUTKEY	300h

Function

This register contains the key to lock and unlock the LUT. See LUT for details.

Special write-access is permitted in the "Anytime" mode.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	KEY															
W																
Reset	0	1	0	1	1	0	1	0	1	1	1	1	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	KEY															
W																
Reset	0	1	0	1	1	0	1	0	1	1	1	1	0	0	0	0

Fields

Field	Function
31-0 KEY	Key to lock or unlock the LUT The key is 0x5AF05AF0 and the read value is always 0x5AF05AF0.

78.11.2.33 LUT Lock Configuration Register (LCKCR)

Offset

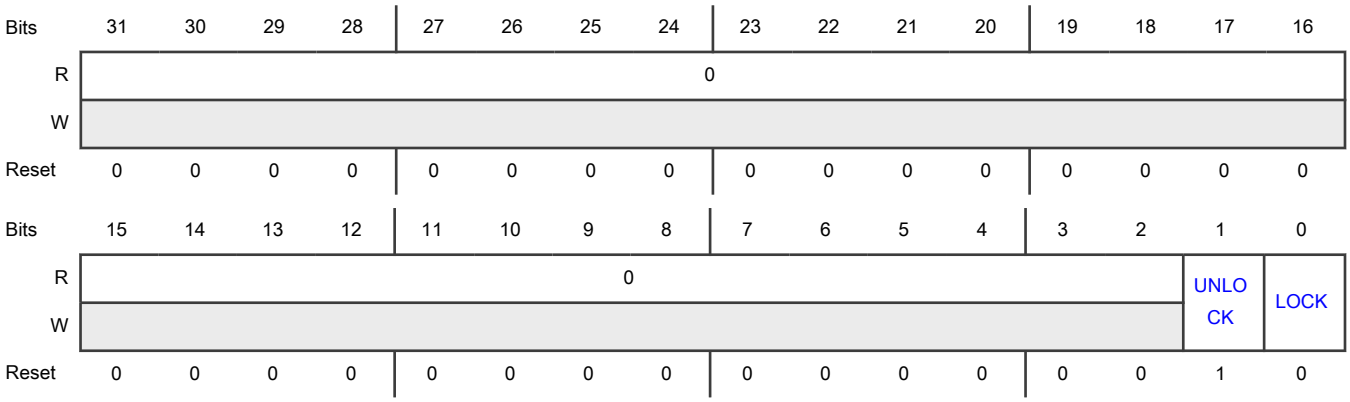
Register	Offset
LCKCR	304h

Function

This register is used along with the LUTKEY register to lock or unlock the LUT. This register should be written immediately after the LUTKEY register for the lock or unlock operation to be successful. See LUT for details. Setting both the LOCK and UNLOCK bits as "00" or "11" is not allowed.

Special write access is permitted after writing the LUT key register.

Diagram



Fields

Field	Function
31-2 —	Reserved
1 UNLOCK	Unlock LUT Unlocks the LUT when the following two conditions are met: • This register is written just after the LUT Key Register (LUTKEY). • The LUT key register was written with the 0x5AF05AF0 key.

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 LOCK	Lock LUT Locks the LUT when the following conditions are met: • This register is written just after the LUT Key Register (LUTKEY). • The LUT key register is written with the 0x5AF05AF0 key.

78.11.2.34 LUT Register (LUT0)

Offset

Register	Offset
LUT0	310h

Function

A sequence of instruction-operand pairs may be pre-populated in the LUT according to the device connected on board. Each instruction-operand pair is of 16 bits (2 bytes) each. Every sequence preprogrammed by Program Sequence Engine in the LUT is referred to by its index. The LUT registers act as lookup tables for sequences of instructions. The programmable sequence engine executes the instructions in these sequences to generate a valid serial flash memory transaction. There are a total of 20 LUT registers. These 20 registers are divided into groups of 5 registers that make a valid sequence. Therefore, LUT[0], LUT[5], LUT[10] ..... LUT[15] are the starting registers of a valid sequence. Each of these sets of 5 registers can have a maximum of 10 instructions. Reset value of the register shown below is only applicable to LUT2 to LUT19. A maximum of 4 sequences can be defined at one time. See LUT that describes the LUT registers in detail.

Special write-access is permitted if the LUT is unlocked.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	INSTR1						PAD1		OPRND1							
Reset	0	0	0	0	1	0	0	0	0	0	0	1	1	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	INSTR0						PAD0		OPRND0							
Reset	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	1

Fields

Field	Function
31-26	Instruction 1

Table continues on the next page...

Table continued from the previous page...

Field	Function
INSTR1	
25-24 PAD1	Pad information for INSTR1 00b - 1 Pad 01b - 2 Pads 10b - 4 Pads 11b - NA
23-16 OPRND1	Operand for INSTR1
15-10 INSTR0	Instruction 0
9-8 PAD0	Pad information for INSTR0 00b - 1 Pad 01b - 2 Pads 10b - 4 Pads 11b - NA
7-0 OPRND0	Operand for INSTR0

78.11.2.35 LUT Register (LUT1)

Offset

Register	Offset
LUT1	314h

Function

A sequence of instruction-operand pairs may be pre-populated in the LUT according to the device connected on board. Each instruction-operand pair is of 16 bits (2 bytes) each. Every sequence preprogrammed by Program Sequence Engine in the LUT is referred to by its index. The LUT registers act as lookup tables for sequences of instructions. The programmable sequence engine executes the instructions in these sequences to generate a valid serial flash memory transaction. There are a total of 20 LUT registers. These 20 registers are divided into groups of 5 registers that make a valid sequence. Therefore, LUT[0], LUT[5], LUT[10] ..... LUT[15] are the starting registers of a valid sequence. Each of these sets of 5 registers can have a maximum of 10 instructions. Reset value of the register shown below is only applicable to LUT2 to LUT19. A maximum of 4 sequences can be defined at one time. See LUT that describes the LUT registers in detail.

Special write-access is permitted if the LUT is unlocked.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	INSTR1						PAD1		OPRND1							
Reset	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	INSTR0						PAD0		OPRND0							
Reset	0	0	0	1	1	1	0	0	0	0	0	0	1	0	0	0

Fields

Field	Function
31-26 INSTR1	Instruction 1
25-24 PAD1	Pad information for INSTR1 00b - 1 Pad 01b - 2 Pads 10b - 4 Pads 11b - NA
23-16 OPRND1	Operand for INSTR1
15-10 INSTR0	Instruction 0
9-8 PAD0	Pad information for INSTR0 00b - 1 Pad 01b - 2 Pads 10b - 4 Pads 11b - NA
7-0 OPRND0	Operand for INSTR0

78.11.2.36 LUT Register (LUT2 - LUT19)

Offset

For a = 2 to 19:

Register	Offset
LUTa	310h + (a × 4h)

Function

A sequence of instruction-operand pairs may be pre-populated in the LUT according to the device connected on board. Each instruction-operand pair is of 16 bits (2 bytes) each. Every sequence preprogrammed by Program Sequence Engine in the LUT is referred to by its index. The LUT registers act as lookup tables for sequences of instructions. The programmable sequence engine executes the instructions in these sequences to generate a valid serial flash memory transaction. There are a total of 20 LUT registers. These 20 registers are divided into groups of 5 registers that make a valid sequence. Therefore, LUT[0], LUT[5], LUT[10] ..... LUT[15] are the starting registers of a valid sequence. Each of these sets of 5 registers can have a maximum of 10 instructions. Reset value of the register shown below is only applicable to LUT2 to LUT19. A maximum of 4 sequences can be defined at one time. See LUT that describes the LUT registers in detail.

Special write-access is permitted if the LUT is unlocked.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	INSTR1						PAD1		OPRND1							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	INSTR0						PAD0		OPRND0							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-26 INSTR1	Instruction 1
25-24 PAD1	Pad information for INSTR1 00b - 1 Pad 01b - 2 Pads 10b - 4 Pads 11b - NA
23-16 OPRND1	Operand for INSTR1
15-10	Instruction 0

Table continues on the next page...

Table continued from the previous page...

Field	Function
INSTR0	
9-8 PAD0	Pad information for INSTR0 00b - 1 Pad 01b - 2 Pads 10b - 4 Pads 11b - NA
7-0 OPRND0	Operand for INSTR0

78.11.3 Serial flash memory address assignment

The serial flash memory address assignment can be modified by writing into Serial Flash Memory A1 Top Address Register (SFA1AD) and Serial Flash Memory A2 Top Address Register (SFA2AD) for device A

Table 721. Serial flash memory address assignment

Parameter	Function	Access mode
QuadSPI_AMBA_BASE ((31:10) - 22 bits)	QuadSPI AHB base address First address of the serial flash memory device as presented to the QuadSPI controller. This might be the base of the serial flash memory in the system address map or it may be a remapping, for instance to 0h, performed by the system. (See the system address map file attached to this document)	
QuadSPI_ARDB_BASE	First address of the QuadSPI Rx buffer on system memory map	
TOP_ADDR_MEMA1(TPADA1)	Top address for the external flash memory A1 (the first of the two independent flash memories sharing the IOFA)	Any access to the address space between TOP_ADDR_MEMA1 and QuadSPI_AMBA_BASE is routed to serial flash memory A1.
TOP_ADDR_MEMA2(TPADA2)	Top address for the external flash memory A2 (the second of the two independent flash memories sharing the IOFA)	Any access to the address space between TOP_ADDR_MEMA2 and TOP_ADDR_MEMA1 is routed to serial flash memory A2.

78.12 Flash memory mapped AMBA bus

QuadSPI_AMBA_BASE defines the address to be used as the start address of the serial flash memory device, as defined by the system memory map. Note that this may be a remapping of the physical address of the serial flash memory in the system. See the system address map for details.

Table 722. QuadSPI AMBA bus memory map

Address	Register name
QuadSPI_AMBA_BASE to (TOP_ADDR_MEMA2 - 1h)	• See Memory-mapped serial flash memory data—individual flash memory mode on flash memory A. • For information about byte ordering, see Table 701 and Table 702.
QuadSPI_ARDB_BASE to... (32 * 4 Byte) QuadSPI_ARDB_BASE + 1FFh	• See AHB RX Data Buffer Register (ARDB0 - ARDB127). • For information about the byte ordering, see Table 701.

NOTE

Any read access to non-implemented addresses provides undefined results.

In individual flash memory modes, the 3/4 address bytes (as programmed in the instruction/operand in the sequence) available for the flash memory address are determined by SFADR[23:0] or SFADR[31:0] as provided in the table shown above.

78.12.1 AHB bus access read considerations

Note that all logic in the QuadSPI module implementing the AHB bus access is designed to read the content of an external serial flash memory device. Therefore, the following restrictions apply to the QuadSPI module with respect to accesses to the AHB bus:

- Any AHB command resulting in the assertion of the FR[ABSEF] flag is answered with the ERROR condition according to the AMBA_AHB specification. The resulting AHB command is ignored.
- AHB bus read access types—NONSEQ and BUSY—are fully supported.
- AHB read access type—SEQ—is treated in the same way as NONSEQ. See Flash memory mapped AMBA bus for details.
- Early burst termination is not supported for AHB transactions.

78.12.2 Memory-mapped serial flash memory data—individual flash memory mode on flash memory A

Starting with address QuadSPI_AMBA_BASE, the content of the first external serial flash memory device is mapped into the address space of the device containing the QuadSPI module. Serial flash memory byte address 0h corresponds to bus address, QuadSPI_AMBA_BASE, in an increasing order. . See the following table for the address mapping. The byte ordering for 32-bit access is provided in Table 701 and for 64-bit read access, the byte ordering is provided in Table 702.

Table 723. Memory-mapped individual flash memory mode—flash memory A address scheme

Memory-mapped address 32-bit access	Memory-mapped address 64-bit access	Serial flash memory byte address	Flash memory device
QuadSPI_AMBA_BASE + 0h	QuadSPI_AMBA_BASE + 0h	0h to 3h	A1
QuadSPI_AMBA_BASE + 4h		4h to 7h	

Table continues on the next page...

Table 723. Memory-mapped individual flash memory mode—flash memory A address scheme (continued)

Memory-mapped address 32-bit access	Memory-mapped address 64-bit access	Serial flash memory byte address	Flash memory device
...	...	...	
TOP_ADDR_MEMA1 - 8h	TOP_ADDR_MEMA1 - 8h	(TOP_ADDR_MEMA1- 8h) to (TOP_ADDR_MEMA1 - 0h4 -0h1)	
TOP_ADDR_MEMA1 - 4h		(TOP_ADDR_MEMA1 - 4h) to (TOP_ADDR_MEMA1 - 0h1)	
TOP_ADDR_MEMA1 + 4h	TOP_ADDR_MEMA1 + 0h	0h to 3h	A2
TOP_ADDR_MEMA1 + 0h4		4h to 7h	
...	...	...	
TOP_ADDR_MEMA2 - 8h	TOP_ADDR_MEMA2 - 8h	(TOP_ADDR_MEMA2 - 8h) to (TOP_ADDR_MEMA2 - 4h - 1h)	
TOP_ADDR_MEMA2 - 4h		(TOP_ADDR_MEMA2 - 4h) to (TOP_ADDR_MEMA2 - 1h)	

The available address range depends on the size of the external serial flash memory device. Any access beyond the size of the external serial flash memory provides undefined results.

For details concerning the read process, see Flash memory read.

78.12.3 ARDB register descriptions

NOTE

See the system memory map in this document for the base address of the QuadSPI AHB RX data buffer.

78.12.3.1 ARDB memory map

QuadSPI_34x_ARDB base address: 6800_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h - 1FCh	AHB RX Data Buffer Register (ARDB0 - ARDB127)	32	R	0000_0000h

78.12.3.2 AHB RX Data Buffer Register (ARDB0 - ARDB127)

Offset

For a = 0 to 127:

Register	Offset
ARDBa	0h + (a × 4h)

Function

This register is used to read the buffer content of the RX buffer from successive addresses. ARDB0 corresponds to the RX buffer register entry corresponding to the current value of the read pointer in an increasing order.

The increment of the read pointer depends on the access scheme (DMA or flag-driven). See Flash memory read, RX buffer, data read through register interface, and AHB read for the description of successive accesses to the RX buffer content. See Byte Ordering of Serial Flash Memory Read Data for the byte ordering scheme.

Valid address range accessible in the ARDB*n* range depends on the number of RX buffer entries implemented and on the number of valid buffer entries available in the RX buffer.

- Example 1 - RX buffer filled completely with 128 words: In this case, the address range for valid read access extends from ARDB0 to ARDB127.
- Example 2 - RX buffer filled with five valid words; RX buffer fill level RBSR[RDBFL] is 5. In this case, an access to ARDB4 provides the last valid entry.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ARXD															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ARXD															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 ARXD	ARDB provided RX buffer data Byte order (endianness) is identical to the RX buffer data registers.

78.13 Glossary

AHB	Advanced high-performance bus, a version of AMBA
AMBA	Advanced microcontroller bus architecture
APB	Advanced peripheral bus
BE	Big endian byte ordering
CRS	Center aligned read strobe
CS	Chip select

FRAD	Flash region access descriptor
I/O	Input output, I/O lines are also referred to as pads in this chapter
IFM	Individual flash memory mode
LE	Little endian
MDAD	Master domain access descriptors
MGID	Master-Group identifier
PCS	Peripheral chip select
SCK	Serial communications clock
SFM	Serial flash memory

Chapter 79
Quad Serial Peripheral Interface (QuadSPI) for S32K358, S32K356, S32K348, S32K338, and S32K328

79.1 Chip-specific QuadSPI information

79.1.1 QuadSPI configuration

Table 724. QuadSPI instances

Instance	S32K358/S32K356/S32K348/S32K338/S32K328
QuadSPI	Yes

Table 725. QuadSPI configuration details

Configuration	S32K358/S32K356/S32K348/ S32K338/S32K328
QuadSPI Tx FIFO size	256 words
QuadSPI Rx FIFO size	32 words
Look Up Table Size	16 words
Rx Buffer	1024 Bytes

For supported data rates, see device Datasheet.

NOTE

Boot from QuadSPI is not supported but execution from external memory is supported.

Table 726. Features supported

Feature	S32K358/S32K356/S32K348/S32K338/S32K328
AHB Write	Yes
Data learning feature	Yes
DLL	Yes
OTFAD (On-the-fly-AES-decryption engine)	No
DDR mode	Yes
HyperRAM	Yes
HyperFlash	Yes
External DQS (Data Strobe)	Yes
Boot from QuadSPI interface	No

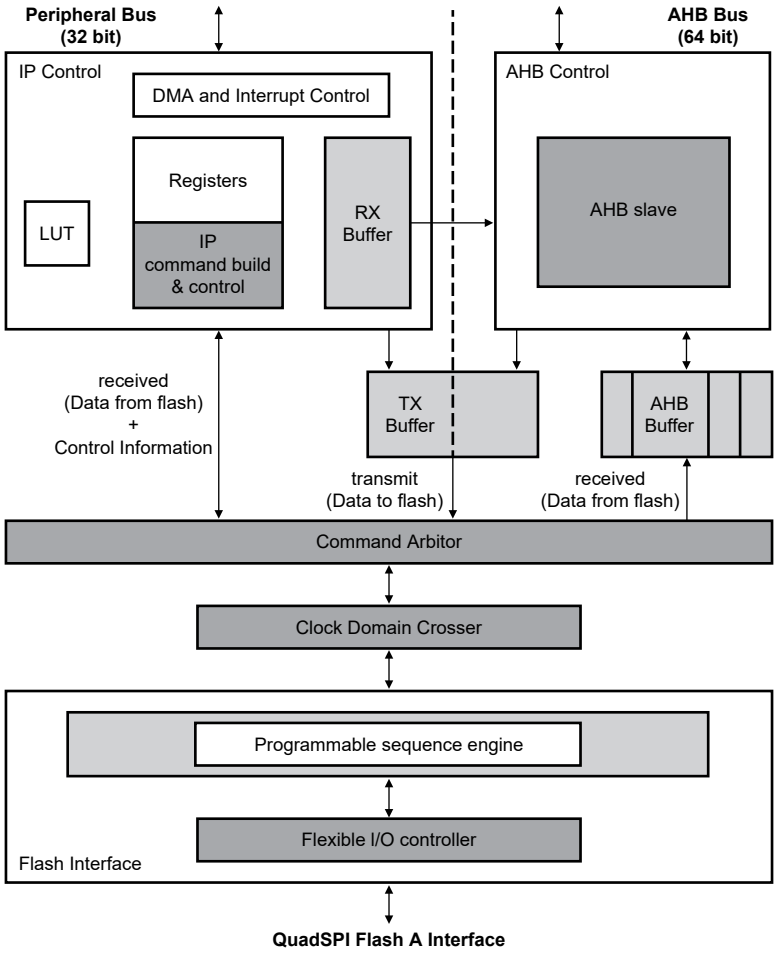


Figure 505. Block diagram

79.1.2 Supported read modes

The table below provides an overview of the QuadSPI read modes.

Table 727. QuadSPI read modes

Read modes		SDR support (QuadSPI_MCR [DDR_EN]=0)	QuadSPI_MCR [DQS_EN]	QuadSPI_MCR [DQS_FA_SEL]	DLL/Data learning support
DQS sampling method	Pad loopback	Yes	1	01	Yes

79.1.3 QuadSPI initialization sequence

Following initialization sequence should be followed for proper QuadSPI operation:

- Enable QuadSPI module by MC_ME peripheral clock enable (register PRTNx_COFBy_CLKEN present within MC_ME). Refer MC_ME chapter for peripheral mapping.
- Configure the SIUL2 registers MSCR[OBE] as 1 and MSCR[SSS] as 0 for QuadSPI_SCKFA pin.
- Initialize QuadSPI SCKFA by writing a sequence of 1010 to the SIUL2 register GPDO[PDO_a] for QuadSPI_SCKFA pin.
- Configure the SIUL2 register MSCR[OBE] back as 0 for the pins.

- Initiate a dummy flash read to reset all DQS flops by itself and any crossing from DQS domain to IPS/AHB is taken care by CDC logic.
- Initiate a peripheral software reset to QuadSPI controller by writing to QuadSPI controller's MCR.
- Post this initialization sequence, the QuadSPI will work in intended deterministic manner.

NOTE

QuadSPI initialization is to be done before using QuadSPI after each functional reset.

79.1.4 Pad clock loopback

This chip supports pad clock loopback. The QuadSPI can be configured to use clock loopback to sample input data. SCK is delayed by the SCK pin output delay, plus the SCK pin input delay using pad loopback, and is configured by setting QuadSPI config registers SOCCR[SOCCFG] and MCR[DQS_FA_SEL]. Enabling the loopback version of SCK can improve the setup time of the input data from the Flash.

For details of these register, see QuadSPI register descriptions.

79.1.5 QuadSPI SOC Configuration register SOCCR[SOCCFG] implementation

The QuadSPI SOC Configuration register QuadSPI_SOCCR[SOCCFG] register is used to control dummy loopback pads and obe_pull_timing_relax_b. Below is the description of it's bits:

Table 728. SOCCR[SOCCFG] implementation

Bit	Description
Bit[0]	obe_pull_timing_relax_b : enables the timing relaxation by pulling obe for pad 1 for half cycle, if 0 then enabled else disabled.
Bit[1]	ibe of QSPIA_SCK_DUMMY pad. 0: Disable input receiver 1: Enable input receiver.
Bit[2]	obe of QSPIA_SCK_DUMMY pad. 0: Disable output driver 1: Enable output driver.
Bit[3]	dse of QSPIA_SCK_DUMMY pad. 0: Disable drive strength 1: Enable drive strength.
Bit[4]	pue of QSPIA_SCK_DUMMY pad. 0: Disable internal pullup or pulldown resistor 1: Enable internal pullup or pulldown resistor.
Bit[5]	pus of QSPIA_SCK_DUMMYpad. 0: Enable internal pulldown resistor if pue is set 1: Enable internal pullup resistor if pue is set.
Bit[6]	sre of QSPIA_SCK_DUMMY pad. 0: Disable slew rate control 1: Enable slew rate control.
Bit[31:7]	Reserved

To Enable the Quadspi dummy PAD loopback use following settings

For Flash-A: MCR[DQS_FB_SEL] = 0x2

SOCCR[SOCCFG] = 0x0000000E (ibe=1, obe=1, dse=1 and sre=0)

NOTE

This SoC does not support dual die flashes. Hence, the signal PCSFA2 from QuadSPI is not used.

79.2 Introduction

The QuadSPI module acts as an interface to a single serial flash memory device, with up to eight bidirectional data lines.

79.2.1 Features

QuadSPI supports the following features:

- Flexible sequence engine to support various flash memory vendor devices. As there is no specific standard, the module supports various kinds of flash memories from different vendors. See Serial flash memory devices for example sequences.
- Single, dual, and quad modes of operation supported for Quad flash memories
- Octal and single IO modes of operation supported for Octal flash memories
- Double Data Rate (DDR)/Double Transfer Rate (DTR) mode in which the data is generated on every edge of the serial flash memory clock
- Flash memory data strobe signal to support data sampling in DDR and Single Data Rate (SDR) mode
- Support for HyperFlash memory
- Support for HyperRAM
- Support for Macronix Octal Data integrity features such as ECC and R/W parity (CRC1)
- AHB master to read RX buffer data through AMBA AHB (64-bit width interface) or IPS registers space (32-bit access) and fill TX buffer via IPS registers space (32-bit access) or using AHB (64-bit width interface)
  - AHB master can be a DMA with a configurable inner loop size
- Multi-master accesses are allowed
  - Flexible and configurable buffer for each master—total available buffer size is 1024 bytes.
- All AHB accesses to flash/RAM memory devices are directly memory mapped to the chip system memory
- Programmable sequence engine to cater to future command/protocol changes and ability to support all existing vendor commands and operations. The software needs to select the corresponding sequence according to the connected flash memory device.
  - Support for all types of addressing

79.2.2 RX buffer push event

To add the valid entries into the RX buffer

By default, each buffer push event adds two entries to the RX buffer because the interface to the serial clock domain is 64 bits in width. Depending on the number of bytes read from the serial flash memory device, it is possible for the very last buffer push event that only one entry is added.

RBSR[RDBFL] is incremented by the number of entries added to the RX buffer.

79.2.3 RX buffer POP event

To remove valid entries from the RX buffer

Each buffer POP event removes (RBCT[WMRK] + 1) valid entries from the buffer. BSR[RDBFL] is decremented by the same number and RBSR[RDCTR] is incremented accordingly.

79.2.4 Block diagram

The following figure shows a block diagram of the QuadSPI module.

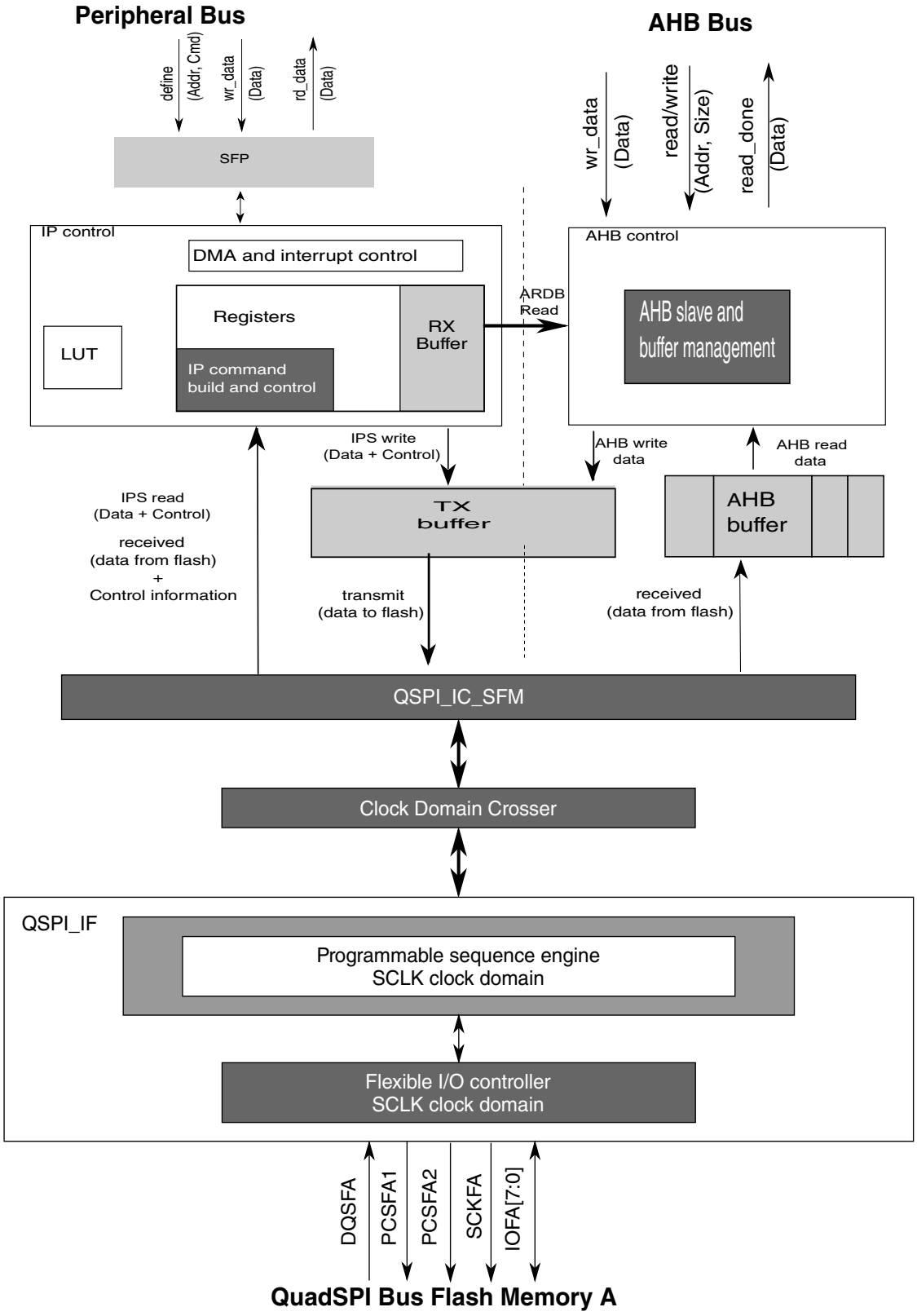


Figure 506. QuadSPI block diagram

79.2.5 QuadSPI modes of operation

QuadSPI supports the following modes of operation:

- Normal mode: You can use this mode for write or read accesses to an external serial flash memory device. See Normal mode for details.
  - Serial flash memory write: You can program data into the flash memory through the IP interface only. See Flash memory programming for details.
  - Serial flash memory read: Read the contents of the serial flash memory device. Two separate read channels are available through the RX buffer and AHB buffer. See Flash memory read for details.
- Module Disable mode: You can use this mode for disabling serial flash memory clock and AHB command. The clock to the non-memory mapped logic in QuadSPI can be stopped in the Module Disable mode. The module enters the mode by setting MCR[MDIS].

79.3 External signal description

This section provides the external signal information for the QuadSPI module.

The following table lists the external signals belonging to the module in conjunction with the different modes of operation.

Table 729. Signal properties

Signal name	Function	Direction	Description
PCSFA1	Peripheral Chip Select Flash Memory A1	O	This signal is the chip select for the serial flash memory device A1 that represents the first of the two flash memory devices that share IOFA.
PCSFA2	Peripheral Chip Select Flash Memory A2	O	This signal is the chip select for the serial flash memory device A2 that represents the second of the two flash memory devices that share IOFA.
SCKFA	Serial Clock Flash Memory A	O	This signal is the serial clock output to the serial flash memory device A.
IOFA[7:0]	Serial I/O Flash memory A	I/O	These signals are the data I/O lines to/from the serial flash memory device A. See Driving external signals for details about the signal drive and timing behavior. Note that the signal pins of the serial flash memory device may change their function according to the SFM Command executed, leaving them as control inputs when single and dual instructions are executed. The module supports driving these inputs to dedicated values. In single I/O mode, QuadSPI drives data on IOFA[0] and expects data on IOFA[1].
DQSFA	Data Strobe signal Flash Memory A	I/O	This is the data strobe signal for port A. Some flash memory vendors provide the Data Read Strobe (DQS) signal to which the read data is aligned in DDR mode. It is also provided as an output signal during write data phase.
INTA	ECC error signal for Flash Memory A	I	Flash Memory A drives this signal to active low value in case of an ECC error.

NOTE

Please refer to chip specific information to check the configuration of QuadSPI block.

79.3.1 Driving external signals

Single/dual/quad/octal instructions

Depending on the serial flash memory device connected to the QuadSPI module, there are instructions using a different number of data lines:

- Single pad: Single line I/O with one data out and one data in line to/from the serial flash memory device
- Dual pad: Dual line I/O with two bidirectional I/O lines, driven alternatively by the serial flash memory device or the QuadSPI module
- Quad pad: Quad line I/O with four bidirectional I/O lines, driven alternatively by the serial flash memory device or the QuadSPImodule
- Octal pad: Octal line I/O with eight bidirectional I/O lines, driven alternatively by the serial flash memory device or the QuadSPI module

The different phases of the serial flash memory access scheme are shown in the following figure.

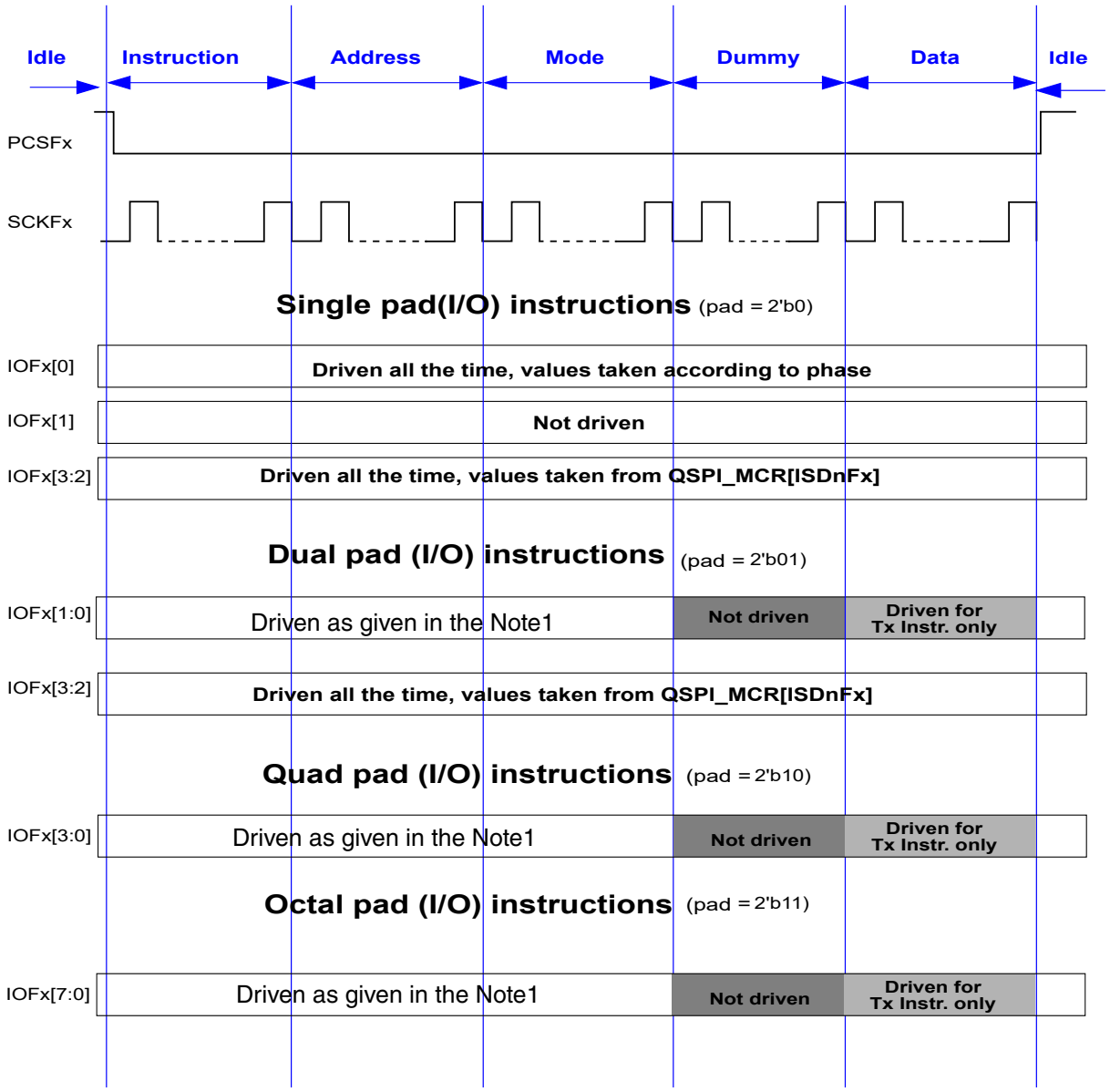


Figure 507. Serial flash memory access scheme

Note1:The IOs are driven from QuadSPI as per the number of pads configured for ongoing phase.

Note:The lines status will change based on command mode in case of instruction, address and mode phases. It can be either 1,2 or 4 lines

Following are the different phases and the I/O driving characteristics of the QuadSPI module:

- Idle: Serial flash memory device not selected – no interaction with the serial flash memory device and the IOFx signals are driven.
- Instruction: Serial flash memory device selected – the instruction is sent to the serial device and all the IOFx signals are driven.
- Address: Serial flash memory address is sent to the device – all the IOFx signals are driven and this phase is not applicable for all SFM commands.
- Mode: Mode bytes are sent to the serial flash memory device – all the IOFx signals are driven and this phase is not applicable for all SFM commands.
- Dummy: Dummy clocks are provided to the serial flash memory device. See Figure 507 for the IOFx signals driven. The actual data lines required for the SFM command executed are not driven for data read commands.

NOTE

— This phase is not applicable for all the SFM commands.

— All read commands in Dual pad, Quad pad /or Octal pad modes must use a Dummy phase immediately before the Data phase. The Dummy phase pad configuration in the LUT must use the same number of pads as the subsequent Data phase. Note that this restriction is not applicable to Single-pad mode.

- Data: Serial flash memory data are sent to or received from the serial flash memory device. See the preceding figure for the IOFx signals driven. The actual data lines required for the SFM command executed are not driven for data read commands.

NOTE

This phase is not applicable for all the SFM commands.

The PCSFx and SCKFx signals are driven permanently throughout all the phases. In the individual flash memory mode, this applies to the selected flash memory device.

Access to a single, individual serial flash memory device

See Serial flash memory access schemes for details.

Read access to two serial flash memory devices attached to the QuadSPI module in parallel. See Serial flash memory access schemes for details.

79.4 Functional description

This section provides a functional description of the QuadSPI module.

79.4.1 Serial flash memory access schemes

In the individual flash memory mode, all supported commands are available.

79.4.2 Normal mode

This mode allows communication with an external serial flash memory device. Compared to the standard SPI protocol, this communication method uses up to eight bidirectional data lines operating at high-data rates. The communication to the external serial flash memory device consists of an instruction code and optional address, mode, dummy, and data transfers. The flexible programmable core engine described below is immune to a wide variety of command or protocol differences in the serial flash memory devices provided by various flash memory vendors.

79.4.2.1 Programmable sequence engine

The core of the QuadSPI module is a programmable sequence engine that works on "instruction-operand" pairs. The core controller executes each programmed instruction sequentially. The complete list of instructions and the corresponding operands are provided in the following table.

Table 730. Instruction set

Instruction	Instruction encoding	Pins	Operand	Action on serial flash memories
CMD	1d	N={0,1,2,3}d 0d - One pad 1d - Two pads 2d - Four pads 3d- Eight pads	8-bit command value	Provides the serial flash memory with the SFM command operand (Encoded) on the number of pads specified in STR mode.
ADDR	2d		Number of address bits to be sent (for example, 24d => 24 address bits required)	Provide the serial flash memory with address cycles according to the operand on the number of pads specified The actual address to be provided is derived from the incoming address in case of AHB-initiated transactions and the value of SFAR in case of IPS-initiated transactions, if the value of SFACR[CAS] is set to 0. Otherwise, the actual address takes CAS into consideration.
DUMMY	3d		Number of dummy clock cycles (should be <= 64 and > 2 cycles)	Provide the serial flash memory with dummy cycles according the operand The PAD information defines the number of pads in input mode. For example, one pad implies that pad 1 is not driven, rest all are driven. **NOTE** If DLL is enabled and N dummy cycles are needed, you must program two back-to-back DUMMY instructions: DUMMY: N-2 followed by DUMMY:2.
MODE	4d		8-bit mode value	Provide the serial flash memory with 8-bit operand on the number of pads specified
MODE2	5d	N={0,1}d	2-bit mode value	Provide the serial flash memory with 2-bit operand on the number of pads[1] specified
MODE4	6d	N={0,1,2}d	4-bit mode value	Provide the serial flash memory with 4-bit operand on the number of pads[2] specified
READ	7d	N={0,1,2,3}d 0d - One pad 1d - Two pads 2d - Four pads 3d- Eight pads	Read data size in bytes (for AHB transactions, your application should ensure that data size is a multiple of 8 bytes)	Read data from flash memory on the number of pads specified The data size can be overwritten by writing to the ADATSZ field of the BUFxCR registers for AHB-initiated transactions and to the IDATSZ field of the IP Configuration Register (IPCR) for IPS-initiated transactions.
WRITE	8		Write data size in bytes	Write data on the number of pads specified The data size can be overwritten by writing to the IDATSZ field of IP Configuration Register (IPCR).

Table continues on the next page...

Table 730. Instruction set (continued)

Instruction	Instruction encoding	Pins	Operand	Action on serial flash memories
JMP_ON_CS [3]	9d	NA	Instruction number	Every time the CS is deasserted, jump to the instruction pointed to by the operand. This instruction allows the programmer to specify the behavior of the controller when a new read transaction is initiated following a CS deassertion. **NOTE** This instruction is not supported if DLL is enabled.
ADDR_DDR	10d	N={0,1,2,3}d 0d - One pad 1d - Two pads 2d - Four pads 3d- Eight pads	Number of address bits to be sent (for example, 24d => 24 address bits required)	Provide the serial flash memory with address cycles according to the operand on the number of pads specified at each clock edge of serial flash memory clock. The actual address to be provided is derived from the incoming address in case of AHB-initiated transactions and the value of SFAR in case of IPS-initiated transactions if SFACR[CAS] is set to 0. Otherwise, the actual address takes CAS into consideration.
MODE_DDR	11d		8-bit mode value	Provide the serial flash memory with 8-bit operand on the number of pads specified at each clock edge of serial flash memory
MODE2_DDR	12d	N={0}d	2-bit mode value	Provide the serial flash memory with 2-bit operand on the number of pads specified at each clock edge of serial flash memory [4]
MODE4_DDR	13d	N={0,1}d	4-bit mode value	Provide the serial flash memory with 4-bit operand on the number of pads specified at each clock edge of serial flash memory [5].
READ_DDR	14d	N={0,1,2,3}d 0d - One pad 1d - Two pads 2d - Four pads 3d - Eight pads	Read data size in bytes (for AHB transactions, the application should ensure that data size is in multiple of 8 bytes)	Read data from flash memory on the number of pads specified at each clock edge of serial flash memory. The data size might be overwritten by writing to the ADATSZ field of the BUFxCR registers for AHB-initiated transactions and IDATSZ field of IP Configuration Register (IPCR) for IP initiated transactions.
WRITE_DDR	15d		Write data size in bytes	Write data on the number of pads specified at each clock edge of serial flash memory The data size can be overwritten by writing to the IDATSZ field of IP Configuration Register (IPCR).
DATA_LEARN	16d	N={0,1,2,3}d 0d - One pad	Operand[7]-	Find the correct sampling point with the data learning pattern.

Table continues on the next page...

Table 730. Instruction set (continued)

Instruction	Instruction encoding	Pins	Operand	Action on serial flash memories
		1d - Two pads 2d - Four pads 3d- Eight pads	• 1 - DDR mode • 0 - SDR mode Operand[6:0]- • Number of bits to be used for data learning. For example, if operand is 16 then a 16-bit data learning pattern is read and compared.	When this instruction is encountered, the value of Sampling Register (SMPR) is ignored and the controller finds the correct sampling point on its own by sampling the data learning pattern.[6]
CMD_DDR	17d	N={0,1,2,3}d 0d - One pad 1d - Two pads 2d - Four pads 3d- Eight pads	8-bit command value	Provides the serial flash memory with the SFM command operand (Encoded) on the number of pads specified in DTR mode.
CADDR	18d		Number of column address bits to be sent (8d means 8 bits of column address is to be sent)	Provide the serial flash memory with column address cycles according to the operand on the number of pads specified. The actual address to be provided to flash memory depends on the value of SFACR[CAS]. For example, if the value of SFACR[CAS] is 3, then the address to flash memory will be [2:0] of incoming address in case of AHB and the value of SFAR in case of IP. This is appended with 0 if SFACR[CAS] is less than number of pads for a flash memory.
CADDR_DDR	19d		Number of column address bits to be sent(8d means 8 bits of column address is to be sent)	Provide the serial flash memory with column address cycles according to the operand on the number of pads specified at each clock edge of the serial flash memory. The actual address to be provided to flash memory depends on the value of SFACR[CAS]. For example, if CAS is 3, then the address to flash memory will be [2:0] of incoming address in case of AHB and the value of SFAR in case of IP. This is appended with 0 if CAS is less than the number of pads for a flash memory.
JMP_TO_SEQ[3]	20d	NA	Sequence number	Every time CS is deasserted, jump to the LUT sequence is pointed to by the operand. This instruction allows the programmer to join two sequences and initiates the second

Table continues on the next page...

Table 730. Instruction set (continued)

Instruction	Instruction encoding	Pins	Operand	Action on serial flash memories
				flash memory transaction automatically. It can be used to join write-enables with flash memory write or data learn with read. **NOTE** This instruction is not supported if DLL is enabled.
STOP[3]	0d	NA	NA	Stop execution; deassert CS

1. For a one-pad instruction, MODE2 takes two serial flash memory clock cycles on the flash memory interface.
2. For a one-pad instruction, MODE4 takes four serial flash memory clock cycles on the flash memory interface. For a four-pad instruction, MODE4 takes one serial flash memory clock cycle on the flash memory interface.
3. Sequence ending with this instruction must have all remaining bits as 0s after it.
4. For a one-pad instruction, MODE2_DDR takes one serial flash memory clock cycle on the flash memory interface.
5. For a one-pad instruction, MODE4_DDR takes two serial flash memory clock cycles on the flash memory interface. For a four-pad instruction MODE4_DDR takes half a cycle on the serial flash memory interface.
6. It is not recommended to have 0h or FFh as the data learning pattern.

The programmable sequence engine allows you to configure the QuadSPI module according to the serial flash memory connected on board. This flexible structure is compatible with new command or protocol changes from different vendors.

79.4.2.2 Flexible read xAHB buffers

To reduce the latency of the reads for AHB masters, the data read from serial flash memory is buffered in flexible AHB buffers. There are four such flexible buffers. The size of each of these buffers is configurable with the minimum size being 0 bytes and maximum size being the size of the complete buffer instantiated (1024 bytes). The size of buffer 0 ranges from 0 to BUF0IND. The size of buffer 1 ranges from BUF0IND to BUF1IND, buffer2 from BUF1IND to BUF2IND and, buffer 3 ranges from BUF2IND to the size of the complete buffer (1024 bytes).

Each flexible AHB buffer is associated with the following:

- An AHB master: Optionally, buffer3 may be configured as an "all master" buffer by writing 1 to BUF3CR[ALLMST]. When buffer3 is configured in such a way, any access from a master not associated with any other buffer is routed to buffer3.
- A datasize field representing the amount of data to be fetched from the flash memory on every missed access.

The master ID of every incoming request is checked and the data is returned or fetched into the corresponding associated buffer. See the chip-specific QuadSPI information for details about master IDs and their corresponding components. Every missed access to the buffer causes the controller to clear the buffer and fetch the BUFxCR[ADATSZ] amount of data from the serial flash memory. As such, you need not configure the buffer size to be greater than ADATSZ because the locations greater than ADATSZ are never used. For any AHB access, the sequence pointed to by BFGENCR[SEQID] is used for the initiated flash memory transaction. The data is returned to the master as soon as the requested amount is read from the serial flash memory. The controller; however, continues to prefetch the rest of the data in anticipation of a next consecutive request. See Figure 508 that shows flexible AHB buffers.

BFGENCR[SEQID] points to an index of the LUT. See LUT for details.

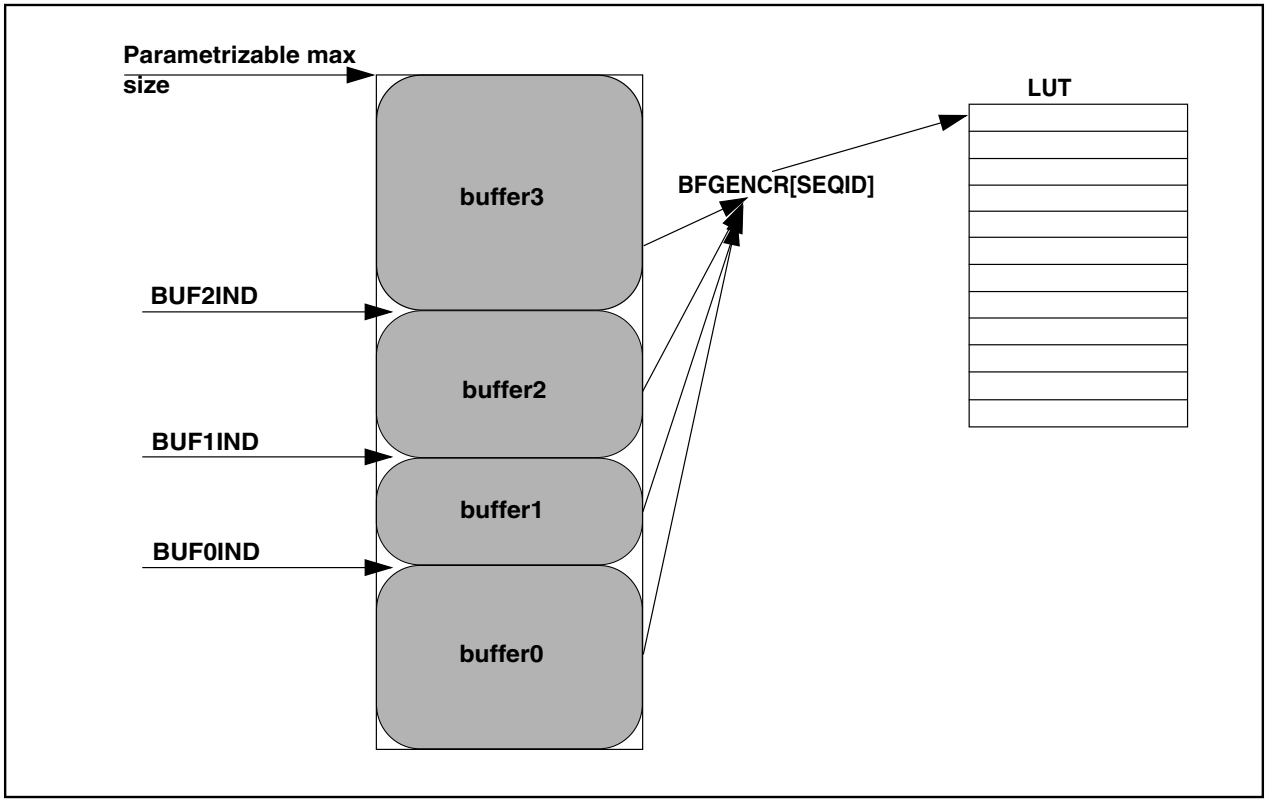


Figure 508. Flexible read AHB buffers

79.4.2.3 Abort mechanism during AHB read

Any ongoing read transaction is aborted if a request from the same master arrives for a location other than the location at which the transaction is going on. The abort can happen at any point of time.

79.4.2.4 HBURST support with AHB read

QuadSPI controller supports HBURST and HSIZE on the AHB read interface. HBURST indicates if the transfer forms part of a burst. Four, eight, and sixteen beat bursts are supported and the bursts might either be incrementing or wrapping. HSIZE indicates the size of the transfer, and supports 8-, 16-, 32-, and 64-bit data sizes. In case of WRAP accesses, QuadSPI generates aligned accesses to serial flash memory if there is no buffer hit for any incoming, non-sequential AHB read access. In case there is a buffer hit, the incoming address in the haddr line is latched as it is. If the total burst size is more than the data prefetch size, an error response is generated and the value of FR[AIBSEF] is configured as 1. The data prefetch size can be defined by BUFxCR[ADATSZ] or data size mentioned in the sequence pointed to by the SEQID field when ADATSZ is programmed as 0. In case of wrap burst, data prefetch size must be greater than or equal to the wrap burst size + 32 bytes. A few examples are shown in the figure below:

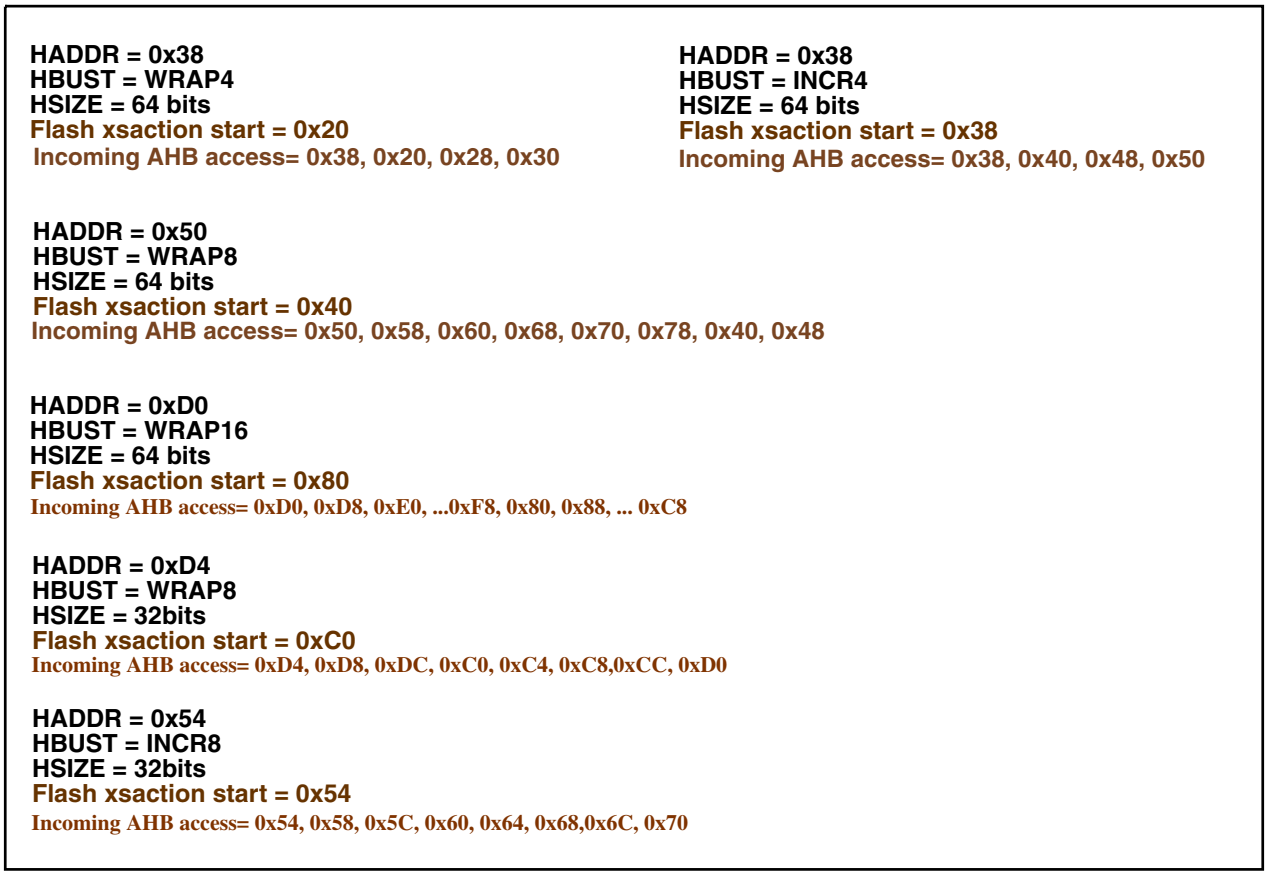


Figure 509. QuadSPI HBURST support

NOTE

The software must take care that the prefetch size should never be set less than the minimum data needed by any external interface to start processing.

NOTE

Whenever a core accesses QuadSPI memory with cache enabled, the prefetch size must be configured as equal or more than the cache line size; otherwise, FR[AIBSEF] error appears.

79.4.2.5 LUT

The LUT consists of a number of pre-programmed sequences. Each sequence is basically a sequence of instruction-operand pairs, which when executed sequentially, generate a valid serial flash memory transaction. Each sequence can have a maximum of 10 instruction-operand pairs. The LUT can hold a maximum of 16 sequences. The figure below shows the basic structure of the sequence in the LUT.

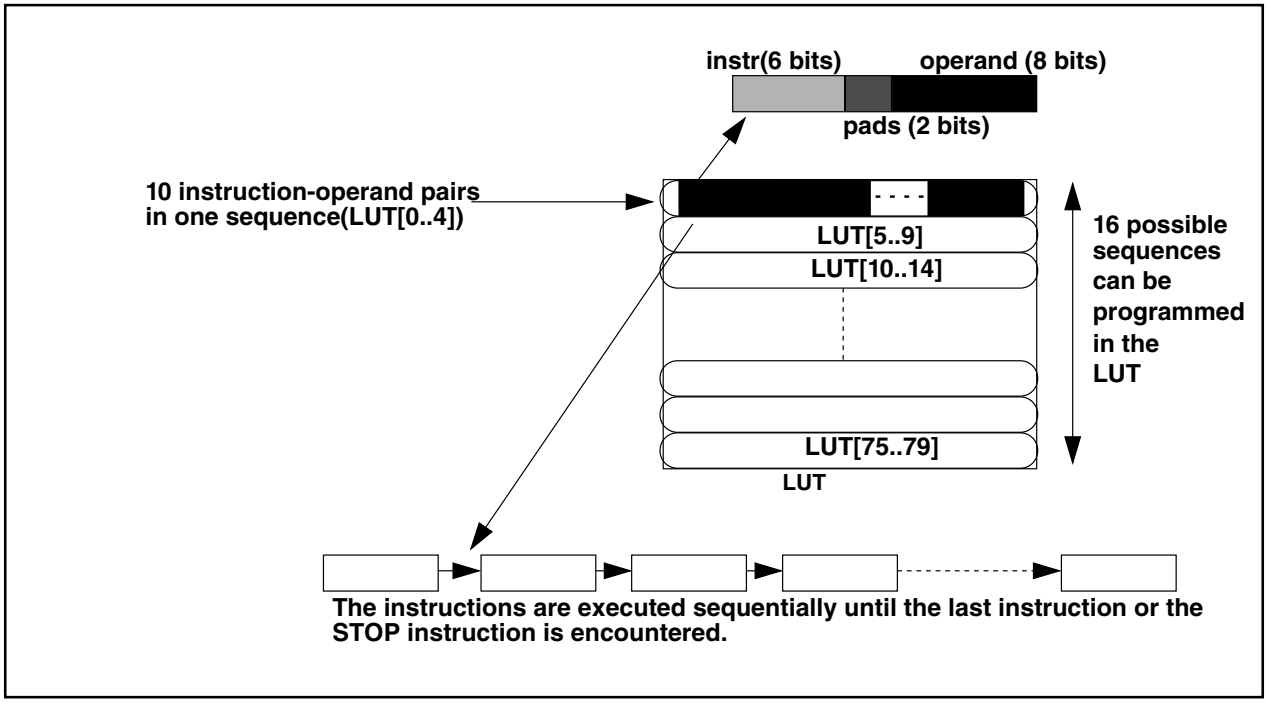


Figure 510. LUT and sequence structure

At reset, the index 0 of the LUT[0..4] is programmed with a basic read sequence as described in Read sequence. After reset, the complete LUT may be reprogrammed according to the chip connected on board. To protect its contents, during a code runover, the LUT might be locked, after which a write to the LUT will not be successful until it has been unlocked again. The key for locking or unlocking the LUT is 5AF05AF0h, and the associated processes are as follows:

Locking the LUT

1. Write the key 5AF05AF0h into the LUT Key Register (LUTKEY).
2. Write 0b01 to the LUT Lock Configuration Register (LCKCR). Note that this IPS transaction should immediately follow the above IPS transaction (no other IPS transaction can be issued). A successful write to this register locks the LUT.

Unlocking the LUT

1. Write the key 5AF05AF0h into the LUT Key Register (LUTKEY).
2. Write 0b10 to the LUT Lock Configuration Register (LCKCR). Note that this IPS transaction should immediately follow the above IPS transaction (no other IPS transaction can be issued in between). A successful write to this register unlocks the LUT.

The lock status of the LUT can be read from the LCKCR[UNLOCK] and LCKCR[LOCK] fields.

Some example sequences are defined in Example sequences. After reset the instruction sequence 0 is populated with the default read sequence shown in the table below.

Table 731. Read sequence

Instruction	Pad	Operand	Comment
CMD	0h	3h	Read data byte command on one pad
ADDR	0h	18h	24 address bits to be sent on one pad

Table continues on the next page...

Table 731. Read sequence (continued)

Instruction	Pad	Operand	Comment
READ	0h	8h	Read 64 bits
JMP_ON_CS	0h	0h	Jump to instruction 0 (CMD)

NOTE

If DLL is disabled then JMP_ON_CS or STOP instruction can be used else only STOP instruction can be used.

79.4.2.6 Issuing SFM commands

Each access to the external device follows this sequence:

1. You must pre-populate the LUT with the serial flash memory command sequences that are required for the flash memory device being used.
2. The module executes the instructions in this sequence one by one. The transaction starts and the module configures the value of SR[BUSY].
3. Communication with the external serial flash memory device starts and the transaction executes.
4. After the transaction is complete (all transmit and receive operations with the external serial flash memory device are complete), the module resets SR[BUSY]. In case of an IP command, FR[TFF] is asserted.

For details, see Flash memory programming and Flash memory read.

You can trigger the processing of SFM commands in the QuadSPI module in one of the following ways:

Using IP commands

For IP commands, the required components need to be written into the following registers and in this sequence:

1. Write the serial flash memory address to be used as provided in the Serial Flash Address Register (SFAR). For IP commands not related to specific addresses, the base address of the related flash memory needs to be programmed. For example, for an instruction which does not require an address (that is, write enable instruction), the SFAR should be programmed with the base address of the memory the command is to be sent to.
2. Write the sequence ID and data size details in the IP Configuration Register (IPCR).
3. Note that writing a value to IPCR[SEQID] must be the last step of the sequence. It is possible to combine all the fields of the IPCR into one single write. See IP Configuration Register (IPCR) for details.

Using AHB commands

Any AHB memory-mapped access is routed to one of the buffers depending on the master ID of the request. If the access is a "miss," a new serial flash memory transaction is initiated. The transaction is based on the sequence pointed to by BFGENCR[SEQID] as described in Flexible read xAHB buffers.

An AHB access is considered memory mapped when the access is to the memory-mapped serial flash memories, as described in Memory Mapped Serial Flash Data - Individual Flash mode on Flash memory A.

79.4.2.7 Flash memory programming

In all NOR Flash devices memory sector to be written needs to be erased first. The programming sequence is then initiated in the following way:

1. Program the FRAD and MDAD registers.
2. Program the address related to the command in SFAR (with access attributes programmed in MDAD). If required, write the desired value to SFACR[CAS], otherwise write 0 to it. Also, write 1 to SFACR[WA] if the serial flash memory is a word addressable flash memory, or write 0 in case serial flash memory is byte addressable

3. Program the IPCR register (with access attributes programmed in MDAD). IPCR[SEQID] should point to an index of the LUT that has the flash memory program sequence pre-programmed. Write an appropriate value to IPCR[IDATSZ] to denote the size of the write in bytes.
4. Check FSM Status (FSMSTAT) register to know status of current transaction. Valid[31] bit should be '1' and domain ID [11:8] should match the ID with which SFAR and IPCR were written. Program the TX watermark in TBCT[WMRK] field and wait for STAT[1:0] bits to turn '01'.
5. Check that the TX buffer is empty. If you need to discard the data present in the TX buffer (SR[TXEDA]) field is set, then the TX buffer must be cleared by writing 1 to MCR[CLR_TXF].
6. Provide data for the program command into the circular buffer through the TBDR. Once the TX buffer is written till watermark level and SR[TXWA] flag is de-asserted this IP command will be triggered. FSMSTAT[STAT] will get set to 10.
7. Repeat step 3, depending on the amount of data required, until all of the required data is written to the TBDR. SR[TXFULL] can be used to check if the buffer is ready to receive more data. At any time, TBSR[TRCTR] can be read to check how many words have been written into the TX buffer.
8. Once the transaction completes FSMSTAT[VLD] bit will set to 0, SR[BUSY] is reset and FR[TFF] is asserted. Please refer to section Secure flash programming for more details.

After writing to IPCR[SEQID] , the module starts executing the programmed sequence when QuadSPI SFM is IDLE. The software ensures that the correct sequence is programmed into the LUT in accordance with the flash memory connected to the module. The data is fetched from the TX buffer. It consists of 256 entries of 32-bit sizes and is organized as a circular FIFO, the read pointer for which is incremented after each fetch. When all the data is transmitted, the QuadSPI module returns from the busy state to the idle state. However, this is not true for the external device because the internal programming is still ongoing. You may monitor the relevant status information available from the serial flash memory device and ensure that the programming is done appropriately.

79.4.2.8 Flash memory read

Host access to the data stored in the external serial flash memory device is performed in two steps. First, the data must be read into the internal buffers and in the second step, these internal buffers can be read by the host.

Reading serial flash memory data into the QuadSPImodule internal buffers

A read access to the external serial flash memory device can be triggered in two different ways:

- IP command read: For reading flash memory data into the RX buffer, you must provide the correct sequence ID in IPCR[SEQID]. The sequence ID points to a sequence in the LUT. The software needs to ensure that a correct read sequence is programmed in the LUT in accordance with the serial flash memory device connected on board. You must program the SFAR , SFACR[CAS], and IPCRs. All available read commands supported by the external serial flash memory are possible.

  Optionally, it is possible to clear the RX buffer pointer prior to triggering the IP command by writing a 1 to MCR[CLR_RXF]. This will invalidate the data currently present in the RX buffer and any new read data will overwrite the old one.

  Using these inputs, the complete transaction is built when IPCR[SEQID] is written to and if MDAD checks are passing (based on the access attributes while writing SFAR and IPCR). The transaction related to the read access starts when QuadSPI is IDLE and FSMSTAT[VLD] bit is 1. The data is fetched for the domain ID that is shown in FSMSTAT[11:8] bits when FSMSTAT[STAT] is set to 11 and the requested number of bytes is fetched from the external serial flash memory device into the RX buffer. As the read access is triggered by an IP command, the value of both SR[IP_ACC] and SR[BUSY] is set to 1.. A count of the number of entries currently in the Rx buffer can be obtained from RBSR[RDBFL].

  Communication with the external serial flash memory stops if the specified number of bytes are read (on successful completion of the transaction).

  NOTE

  In case of external DQS strobe based sampling, read data size (IDATSZ or LUT size) must be in multiples of 8 bytes.

- AHB command read: For reading flash memory data into the AHB buffer, you must:

- Set up a read access by a master to the address range in the system memory map, which the external serial flash memory devices are mapped to.
- Write a desired value to SFACR[CAS], if required, or write 0 to the field.
- Program the buffer registers corresponding to the AHB master initiating the request.
- Provide the correct sequence ID in the BFGENCR. The software ensures that a correct read sequence is programmed in the LUT in accordance with the serial flash memory device connected on board. Flash memory device selection and access mode are determined by the address accessed in the AHB address space associated with the QuadSPI module (see Memory-mapped serial flash memory data—individual flash memory mode on flash memory A

On each AHB read access to the memory mapped area, the valid data in the AHB buffer is checked against the address requested in the actual read. When the AHB read request cannot be served from the content of the AHB buffer, the buffer is flushed and the controller executes the sequence pointed to by the sequence ID. The requested number of buffer entries defined in BUFxCR[ADATSZ] is then fetched from the external serial flash memory device into the internal AHB buffer. As the read access is triggered through the AHB bus, the value of SR[AHB_ACC] is set, driving SR[BUSY] in turn, until the transaction is complete. Communication with the external serial flash memory stops when the specified number of entries is filled.

Data transfer from the QuadSPI module internal buffers

The data read out from the external serial flash memory device by the QuadSPI module is stored in the internal buffers. The means of accessing the data from the buffer differs depending on which buffer the data is loaded to. See Block diagram for details on the two available buffers, the RX buffer and the AHB buffer, in this module. The buffer appears transparent to you and is non-memory mapped. See the "Flexible AHB Buffer" section for details.

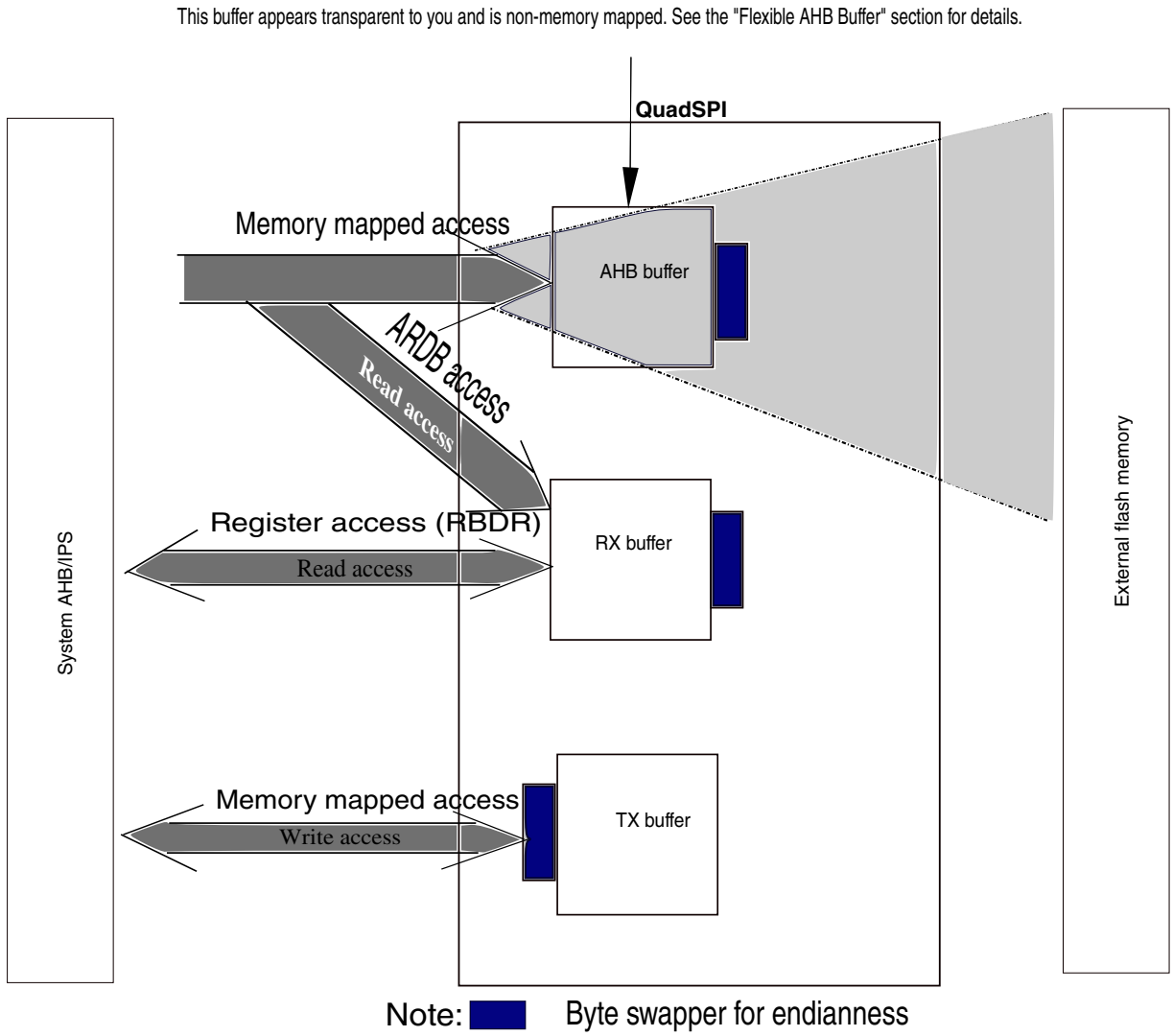


Figure 511. QuadSPI memory map

The RX buffer is implemented as FIFO of depth 32 entries of 4 bytes. Its content is accessible in two different address areas, both referring to identical data and the same physical memory:

- In the IPS address space in the area associated with RX Buffer Data Register (RBDR0 - RBDR31).
- In the AHB address space in the area associated with AHB RX Data Buffer Register (ARDB0 - ARDB31).Two successive entries are accessed with one single 64-bit AHB read operation.

The RX buffer operation can be summarized as follows:

- RBCT[WMRK] determines at which fill level SR[RXWE] is asserted and how many entries are removed from the RX buffer on each buffer POP operation.
- SR[RXWE] indicates that the configured number of data entries is available in the RX buffer and RBSR[RDBFL] indicates how many valid entries are available in total.
- The first entry (RBDR0 or ARDB0) always corresponds to the first valid entry in the RX buffer.

For details, see RX Buffer Data Register (RBDR0 - RBDR31) and AHB RX Data Buffer Register (ARDB0 - ARDB31).

- Flag-based data read of the RX buffer is performed by polling SR[RXWE]. When it is asserted, the valid entries can be read either via the IPS address space (RBDRn) or the AHB address space (ARDBn). A buffer POP operation must be

triggered by the application by writing a 1 to FR[RBDF]. This automatically updates the FIFO to point to the next entry as defined by RBCT[WMRK]. For example, if WMRK is set to 3, then the buffer discards 16 bytes of data.

- DMA-controlled data read of the RX buffer is performed by using the DMA module. The application must ensure that the DMA controller of the related chip is programmed appropriately, as described in DMA usage.
- DMA-controlled read out is triggered fully automatically by the assertion of SR[RXWE]. The related buffer POP operation is also handled completely inside the QuadSPI module. As in the case explained here, accessing the RX buffer content either on RBDRn or ARDBn related addresses is equivalent.
- AHB buffer data read via memory-mapped access: This kind of access is performed by reading one of the addresses assigned to the external serial flash memory device(s) within the range specified in Table 769. If this is not the case, a memory-mapped AHB command read is triggered as described above. If the requested data is already available in the AHB buffer, it is provided directly to the host.

When an AHB access is made to the flash memory mapped address, the data is fetched and returned to the AHB interface. The AHB interface is stalled until the data is fetched. As soon as the data from the requested address is read by the QuadSPI module, the AHB read access is served. Therefore, it is possible to run sequential reads from the AHB buffer at arbitrary speed without the need to monitor any information about the availability of the data. Nevertheless, this access scheme stalls the AHB bus for the time required to read the data from the serial flash memory device. If you know that the access is sequential, a better way is to have a prefetch enabled by programming the value of BUFxCR[ADATSZ] so that the data is fetched into the buffer before the next sequential AHB access.

As long as the host restricts its accesses to the data present in the buffer and to the data currently fetched from the serial flash memory, it is possible to run the host read from the AHB buffer simultaneously with the serial flash memory read into the AHB buffer.

79.4.2.9 AHB write

This module supports a single master AHB write with these burst types — INCR4, INCR8, INCR16, and INCR unspecified 4n bytes and single 4n bytes — and these burst types are supported with all Hsizes. An AHB write transaction size can be expressed in multiples of 4 bytes, and a single transaction must have a minimum of 4 bytes. In case of HyperRAM ,this limitation of minimum 4 bytes is not present. Also unspecified INCR and single transactions are supported without the limitation of 4n bytes in case of HyperRAM.

The following restrictions apply to the QuadSPI module with respect to AHB write transactions:

- AHB transactions of WRAP* types are not supported. For these unsupported access types, the controller returns an error response and does not initiate any flash memory transaction.
- Back to back, fixed address AHB write is not supported.
- Early burst termination is not supported for AHB transactions.

The QuadSPI controller queues back-to-back AHB writes with continuous address and sends them in a single flash memory transaction. However, in case the AHB address of a burst is not in continuous addressing sequence to previous write burst, a new flash memory transaction is initiated.

Following is the programming sequence for an AHB write:

1. Check that SR[BUSY] is deasserted or is 0 and check that the TX buffer is empty. If there is any residue data present in the TX buffer(SR[TXNE]), the buffer must be cleared by writing '1'1 to MCR[CLR_TXF].
2. Based on the type of flash memory device that is used for page programming, AWRCR[PPW_WR_DIS] and AWRCR[PPW_RD_DIS] need to be programmed.

   The default reset value for both the fields is 0 and allows writing to flash memory device assuming a very low T_{pp} (page program wait time) period. This period is very high for most of the flash memory devices. After any write to a flash memory device, the module waits for the write to finish before proceeding further.

   This is controlled by the AWRCR[PPW_WR_DIS] and AWRCR[PPW_RD_DIS] field configurations.
3. Program the BFGENCR[SEQID] to point to the LUT where flash memory write sequence is stored.

4. AHB write master can now enable the transactions to be written to the flash memory. The AHB master must ensure data throughout to avoid underflow.
5. When a flash memory write completes and flash memory CS is deasserted, the software must poll SR[BUSY] for the deasserted value, which is 0. This indicates that the AHB write is complete. In case an underflow has occurred, flash memory CS is deasserted with (SR[TXNE])to know that an AHB write is pending with some residue data in the TX buffer.
6. Whenever flash memory CS is deasserted after an AHB write sequence, FR[PPWF] is set if any of AWRCR[PPW_RD/WR_DIS] is enabled. FR[PPWF] disables any further flash memory read/write based on AWRCR[PPW_RD]/AWRCR[WR_DIS]. After FR[PPWF] is set, it returns every incoming AHB transaction with an error response.
7. The software must clear FR[PPWF] to resume AHB transactions after the expiry of T_{pp} period of flash memory. It can also send IPS read command to the flash memory for checking if T_{pp} is over.
8. The software must ensure that the correct read and write LUT sequences are selected corresponding to the AHB read and write transactions. QuadSPI would malfunction if flash memory read sequence is selected during AHB write or vice versa.

NOTE

An AHB error response occurs in case of AHB write if flash is not able to consume data at the rate master is writing in TX FIFO and TX FIFO eventually becomes full

79.4.2.10 Byte ordering of serial flash memory read data

The basic scheme is that the first byte read out of the serial flash memory device, which is addressed by SFAR[SFADR], corresponds to RBDR0[31:24] for IP command read. Similarly, to send a single byte it should be positioned in TBDR[0:7]. In contrast to that for AHB command read, the bytes are always positioned according to the byte ordering of the AHB bus.

- Byte ordering in individual flash memory mode

  The following table provides the byte ordering scheme of how the byte oriented data space of the serial flash memory device is mapped into one single 32-bit entry of the RX buffer or the AHB buffer. The table is valid within the following context:

  — Flash memory A in individual flash memory mode

  — All AHB data read commands with 32-bit access size

Table 732. Byte ordering in individual flash memory mode

Serial flash memory byte numbering	3	2	1	0
Buffer entry bit position [31:0] (32-bit data width)	[31:24]	[23:16]	[15:8]	[7:0]

NOTE

For IP commands, the read size can be specified as number of bytes. If this number is not a multiple of four, then the last buffer entry is not completely filled with the missing higher numbered bytes at undefined values.

  For AHB commands and reads, starting from an address not aligned to 32-bit boundaries, the requested bytes are given at the appropriate positions according to the AMBA AHB specification.

- Buffer entry ordering for 64-bit read access

  For read access via the AHB interface, a 64-bit access is possible. Each 64-bit access reads two 32-bit entries, simultaneously. The ordering of these 32-bit entries within the 64-bit word is provided in the following table.

Table 733. 64-bit read access buffer entry ordering

AHB read data bit position [63:0]	[63:32]	[31:0]
Buffer entry #	Odd (1, 3, 5, ...)	Even (0, 2, 4, ...)

79.4.2.11 Normal mode interrupt and DMA requests

The QuadSPI module has different flags that can only generate interrupt requests and one flag that can generate an interrupt as well as DMA requests. The following table lists the eight conditions. Note that the flags mentioned in the table are associated with the Flag Register (FR).

Table 734. Interrupt and DMA request conditions

Condition	Flag (FR)	DMA
Data learn pattern failure	DLPFF	-
TX buffer fill	TBFF	-
TX buffer underrun	TBUF	-
Illegal instruction error	ILLINE	-
RX buffer drain	RBDF	X
RX buffer overflow	RBOF	-
AHB buffer overflow	ABOF	-
AHB sequence error	ABSEF	-
AHB illegal transaction error	AITEF	-
AHB illegal burst size error	AIBSEF	-
IP command trigger could not be executed error	IPIEF	-
IP command related transaction finished	TFF	-

Each condition has a corresponding field in Flag Register (FR) and a request enable field in DMA Request Select and Enable Register (RSER). FR[RBDF] has separate enable fields for generating IRQ and DMA requests. Note that not all the fields have an individual IRQ line. See the chip's Interrupt Vector table for details.

- Transmit buffer fill interrupt request

  This indicates that the TX buffer can accept new data. The buffer is asserted if FR[TBFF] is asserted and if the value of the corresponding enable field, RSER[TBFIE], is 1. See TX buffer Operation for details on the assertion of FR[TBFF].

  Apart from IRQ, it is possible to handle the TX buffer fill by using the DMA. If the value of RSER[TBFDE] is 1, a DMA request is triggered when the number of available space in the TX buffer is more than the TBCT[WMRK] valid entries and value of SR[TXWA] is set. The application must configure the environment appropriately (for example, the DMA controller) for the DMA transfer.

  In SFP Enabled configurations, program RSER[TBFDE] bit to 1 only when arbitration is won and TBDR access is unlocked after checking STATE field of FSMSTAT register. After TBDR programming is completed by DMA, reset this bit otherwise, QuadSPI SFM state will remain busy and no further IPS accesses can be served by that Target queue.

- Receive buffer drain interrupt or DMA request

  This is derived from FR[RBDF], indicating that the RX buffer of the QuadSPI module has data available from the serial flash memory device to be read by the host. It remains set as long as RBSR[RXWE] is configured. Also, RSER[RBDIE] enables the related IRQ.

  Apart from the IRQ, it is possible to handle the RX buffer drain by using the DMA. If the value of RSER[RBDDE] is 1, a DMA request is triggered when the RX buffer contains more than RBCT[WMRK] valid entries. The application must set the environment appropriately (for example, the DMA controller) for the DMA transfers.

- Buffer overflow/underrun interrupt request

This is a combination of the following fields (all located in the Flag Register (FR) with the related enable bits in the DMA Request Select and Enable Register (RSER)):

- TBUF - TX buffer underrun, enabled by TBUIE
- RBOF - RX buffer overflow, enabled by RBOIE
- ABOF - AHB buffer overflow, enabled by ABOIE
- The transmit buffer underrun indicates that an underrun condition in the TX buffer has occurred. It is generated when a write instruction is triggered whilst the TX buffer is empty and the value of RSER[TBUIE] is 1.
- The receive buffer overflow indicates that an overflow condition in the RX buffer has occurred. It is generated when the RX buffer is full, an additional read transfer attempts to write into the RX buffer, and the value of RSER[RBOIE] is 1.
- The AHB buffer overflow indicates that an overflow condition in the AHB buffer has occurred. It is generated when the AHB buffer is full, an additional read transfer attempts to write into the AHB buffer and the value of RSER[ABOIE] is 1.
- The data from the transfers that generated the individual overflow conditions is ignored.

- Serial flash memory command error interrupt request
- Transaction finished interrupt request

  The IP command transaction finished IRQ indicates the completion of the current IP command. It is triggered by FR[TFF] and is masked by RSER[TFIE].

79.4.2.12 TX buffer operation

The TX buffer provides the data used for page programming. For proper operation, it is required to provide at least one entry in the TX buffer prior to starting the execution of the page programming command. The application must ensure that the required number of data bytes is written into the TX buffer fast enough as long as the command is executed without a TX buffer overflow or underrun.

The QuadSPI module sets the FR[TBFF] field as long as the TX buffer is not full and can accept more data. At the end of write through TX buffer, you must clear FR[TBFF] to avoid unnecessary last TX buffer fill interrupt. However, there would always be a pending request asserted from QuadSPI controller at the end of any DMA transfer. When external DMA finishes transfer iteration, this request from QuadSPI is kept asserted for the next iteration loop.

NOTE

Even if the generation of DMA requests for filling the TX buffer is disabled by using RSER[TBFDE], the TX buffer still accepts a DMA transfer because of the last asserted pending request.

Disabling of DMA transfer should be controlled by an external DMA master.

When the QuadSPI module tries to pull data out of an empty TX buffer, FR[TBUB] signals the TX buffer underrun. The TX buffer underrun flag is also asserted when the TX buffer contains less than 32-bits of data and the QuadSPI module tries to pull out data from it. The current IP command leading to the underrun condition is continued until the specified number of bytes is sent to the serial flash memory device. Also, the data that is transferred is in the Fs format. This means, after the underrun flag is set under this condition, it returns Fs until the required number of bytes are not sent. This has been done to ensure that the software does not erase the whole sector after underrun and just reprogramming from failure point serves the purpose. When this sequence command is complete, FR[TBFF] is asserted, indicating that the TX buffer is ready to be written again.

The TX buffer overflow is not signaled explicitly, but TBSR[TRBFL] can monitor the TX buffer fill level.

For more information, see TX Buffer Status Register (TBSR) and Flag Register (FR).

79.4.2.13 Address scheme

Earlier, serial flash memories supported only a 24-bit address space, restricting the maximum memory size of the serial flash memory to 16 MB. The new memory specification supports two types of 32-bit addressing mode in addition to the legacy 24-bit address mode. It also supports segregation of address programmed into Row address and Column address of the flash memory, as per the requirement.

Extended address mode

In this mode, the legacy 24-bit commands are converted to accept 32-bit address commands. The flash memory needs to be configured for the 32-bit address mode. Also, while programming the LUT sequence in QuadSPI for 32-bit mode, the ADDR and ADDR_DDR commands should be programmed with 32d as the operand value with SFACR[CAS] programmed to 0. If a flash memory needs some bits of the address as its column address, then it must be considered that a total of 12 bits are required by the flash memory; however, the number of bits should not exceed 32 because the maximum address supported by QuadSPI is 32 bits.. Each of the memory vendors have a different way of enabling this mode (see the memory specification from memory vendors). For example, the command B7h sent to the Macronix flash memory enables it for the 32-bit address mode.

Extended address register

In this mode, the upper 8 bits of the 32-bit register are provided by the Extended address register in the memory, which provides a specific register that is updated according to the address to be accessed. This effectively converts the legacy 24-bit address command into 32-bit address commands. The memories greater in size than 16 MB consist of banks of 16 MB each. The 8 bits occupied in the extended address register effectively enable a bank. For example, in Spansion memory, when the extended address register is updated with a value of 1h, with the help of the 17h command, it opens Bank1 of the memory. The consequent 24-bit address commands lead to Bank1. The extended address register needs to be updated with the respective value for access to other banks. This effectively converts the legacy 24-bit address command into 32-bit address commands.

Separation of address into rows and columns

This mode has been introduced for flash memories that need addresses segregated into rows and columns. The value in SFACR[CAS] defines the width of the column address required by a flash memory. The actual address to be provided is derived from the incoming address in case of AHB-initiated transactions and the value of SFAR in case of IPS-initiated transactions, if the value of SFACR[CAS] is 0. Otherwise, the actual address takes CAS into consideration. If the value of SFACR[CAS] is 3, then bits 26-3 of the programmed address are sent to the flash memory as its page address. This is in case the flash memory is operating in a 24-bit mode and bits 2-0 are sent as its column address. If a flash memory requirement for column address is less than the number of pads in which address has to be sent, then QuadSPI appends the remaining bits by 0. You must program the operand value in CADDR and CADDR_DDR command accordingly. It must be ensured that the total number of address bits requested by flash memory as its page and column address must not be more than 32 bits.

Word addressable mode for flash memory

This mode has been introduced for flash memories, which have a word-addressable memory. This means, each address of the flash memory contains one word (two bytes) of data. The value of SFACR [WA] is configured to 1 to enter this mode. QuadSPI internally divides the incoming address in the AHB bus or the address in the SFAR to map it to a valid flash memory location. For example, if the incoming address is 2004h, the controller re-maps this address to access the flash memory location 1002h. If not in this mode, the incoming address 2004h is mapped to flash memory location 2004h.

79.4.3 Module Disable mode

Module Disable mode is a block-specific mode that the QuadSPI can enter to disable serial flash memory clock and AHB command. This mode can be entered by:

- The host software: by writing a '1' to MCR[MDIS]

Below are the condition that must be fulfilled to enter the Module Disable mode:

- SR[BUSY] = 0
- SR[AHBTRN] = 0
- RBSR[RDBFL] = 0
- SR[RXDMA] = 0
- SR[TXDMA] = 0
- None of the flags in FR are enabled as interrupts is set

The conditions mentioned above ensures the following:

- There is no SFM command currently being executed.
- All the data read into the RX buffer from the serial flash memory have been fetched by the application.

- There is no current AHB access.
- There is no active DMA request.
- There is no enabled interrupt that is pending.

Certain read or write operations have a different effect when the QuadSPI is in the Module Disable mode. In the Module Disable mode, not all of the status and flag bits of the QuadSPI module are updated, and writing to them has no effect. Interrupt and DMA request signals cannot be cleared while in the Module Disable mode.

NOTE

It is illegal to issue a new SFM command starting two clock cycles prior to raising the request of entering the Module Disable mode until the QuadSPI stays in this mode.

79.4.4 Leaving Module Disable mode

In the Module Disable mode, the serial flash memory clock and AHB command to the QuadSPI module are switched off.

After the QuadSPI has left this mode and has returned to Normal mode, the execution of the first SFM command is deferred until the clock to drive that part of the module related to the serial flash memory device is available. Depending upon the point in time when the first SFM command is triggered, the actual execution of the command starts with a delay, respective with the re-enabling of the flash memory clock signal.

79.4.5 HyperRAM support

QuadSPI supports HyperRAM memories, and by virtue of this protocol, QuadSPI supports the following functionalities.

- Bidirectional data strobe/read write data strobe (RWDS)
  - If QuadSPI is configured to use the HyperRAM mode, the RWDS pad should be pulled down.
- Variable refresh latency
  - If the value of MCR[VAR_LAT_EN] is 1, based on the status of RWDS from HyperRAM during the command/address phase, QuadSPI includes an additional initial access latency. If RWDS is high, QuadSPI includes twice + 1 the latency mentioned in the dummy phase. If RWDS is low, latency mentioned in the dummy phase is not included.
  - If the value of MCR[VAR_LAT_EN] is not set, fixed latency mentioned in the dummy phase is included.
- Read data strobe to latch data from HyperRAM
- Programming considerations for HyperRAM support:
  - Data masking during the AHB write is enabled when MCR[DQS_OUT_EN] bit is programmed as ’1’.
  - It is suggested that while using Hyper RAM program the SPTRCLR[PREFETCH_DIS] bit as ‘1’ so that every time a new AHB read transaction starts it reads new data from external memory instead of reading the data already stored in AHB buffers. This will ensure that the data read back after an AHB write is not a stale data.
  - AHB write doesn’t supports wrap transactions (WRAP4, WRAP8, WRAP16) even with HyperRAM.
  - If there is multiple switching between AHB read and write transactions then software should program AWRCR[AWTRGLVL] cautiously. It should ensure that during AHB write the data written to memory is more than the AWRCR[AWTRGLVL] programmed before switching over to AHB read transactions. This will ensure that IP is not left waiting for sufficient write data to be collected.

79.5 Initialization/application information

This section provides the initialization and application information of the QuadSPI module.

79.5.1 Power up and reset

The serial flash memory devices connected to the QuadSPI module might require special voltage characteristics of their inputs during power up or reset. The application must ensure this.

CAUTION

Erase or program commands should be completed before issuing a reset or power cycle to avoid corrupted flash pages. The application shall ensure there is a backup of critical data stored at a different location to enable recovery from corrupted flash pages.

Example: Flash reset sequence

Use the following sequence to reset flash A:

1. Make sure that the flash supports a reset for the condition CS#=high and IOF[3]=low.
2. Set MCR[SWRSTSD] and MCR[SWRSTHD] fields and then clear them.
3. Set MCR[MDIS] field.
4. Reset MCR[ISD3FA] field for flash A.
5. Clear MCR[MDIS] field.
6. Set MCR[MDIS] field.
7. Set MCR[ISD3FA] field for flash A.
8. Clear MCR[MDIS] field.

79.5.2 Available status/flag information

This section provides an overview of the different flags and statuses available, and their interdependencies for different use cases. The SR and FR are the related registers.

79.5.2.1 IP commands

See IP Configuration Register (IPCR) for additional details not explicitly covered in this paragraph.

- IP commands—normal operation

  Writing to IPCR[SEQID] triggers the execution of a new IP command. Given that this is a legal command, SR[IPACC] and SR[BUSY] are asserted simultaneously, immediately after the execution starts.

  After the instruction on the serial flash memory device is complete, these field deassert and FR[TFF] is configured.

- IP commands—error situations

  See Overview_of_Error_Flags for details.

79.5.2.2 AHB commands

See the "Reading serial flash memory data into the QuadSPI module internal buffers" topic in theFlash Memory Read section for details.

- AHB commands—normal operation

  Memory-mapped read access to a serial flash memory address not contained in the AHB buffer triggers the execution of an AHB command. Given that this is a legal command, SR[AHB_ACC] and SR[BUSY] are asserted simultaneously, immediately after the execution starts. After the instruction on the serial flash memory device is complete, these fields are deasserted.

- IP commands—error situations

  See Overview of FR error flags for details.

79.5.2.3 SFM commands

An SFM command consists of an instruction code and all other parameters (for example, size or mode bytes) needed for that specific instruction code. Triggering a command either initiates a transaction on the external serial flash memory or results in an error. See Table 735 for details on errors.

79.5.2.4 Overview of error flags

The following table provides an overview of the different error flags in the FR and additional error-related details.

Table 735. Overview of FR error flags

Error category	Error flag in FR	Command execution on serial flash memory device TFF behavior (in case of IP commands only)	Description
AHB error flag	ABOF	Flash memory transaction continues until it finishes	Set when the module tries to push data into the AHB buffer that exceeds the size of the AHB buffer. Occurs only because of the wrong programming of BUFxCR[ADATSZ].
AHB error flag	AIBSEF	Flash memory transaction is aborted	Total burst size of the AHB transaction is greater than prefetch data size.
AHB error flag	AITEF	Flash memory transaction is aborted	No response is generated from QuadSPI to AHB bus in case of illegal transaction. Also, the watchdog timer expires.
Miscellaneous error flag	DLPFF	Flash memory transaction continues until it finishes	Set when the DATA_LEARN instruction is encountered in a sequence but no sampling point is found for the data learning pattern.
Miscellaneous error flag	ILLINE	Flash memory transaction aborted	Illegal instruction error flag - set when an illegal instruction is encountered by the controller in any of the sequences.
Command arbitration error	IPIEF	TFF not asserted in conjunction with that command	IP command error - caused when IP access is currently in progress (IP_ACC is set) and during: • Write attempt to RBCT register
Buffer-related error	RBOF	TFF is asserted on completion	• RX buffer overrun
Buffer-related error	TBUF	TFF is asserted on completion	• TX buffer underrun

Note that only the buffer-related errors are associated with a transaction on the external serial flash memory. All the other errors do not trigger an actual transaction.

79.5.2.5 IP bus and AHB access command collisions

There are following flags related to this topic: FR[IPIEF]. See the "Reading serial flash memory data into the QuadSPI module internal buffers" topic of the Flash Memory read section for a description of the flags.

79.5.3 Flash memory device selection

Regardless of the SFM command (IP or AHB), the access mode is selected by specifying the 32-bit address value for the following SFM command.

For IP commands, the access mode is selected with the address programmed into the SFAR register. See Serial Flash Address Register (SFAR) for details.

For AHB commands, the access mode is determined by the memory-mapped address. See AMBA Bus Register Memory Map for details.

79.5.4 DMA usage

For a complete description of the DMA module, see the related DMA Controller chapter. This section only provides QuadSPI-specific DMA usage details.

79.5.4.1 DMA usage in normal mode

79.5.4.1.1 Bandwidth considerations

Careful consideration of the throughput rate of the entire chain (serial flash memory -> AHB bus / IP bus -> DMA controller) involved in the read/write data process is essential for a proper operation. Such analysis must take into account not only the data rate provided by the serial flash memory but also the data rate of the AHB bus and the performance of the DMA controller in reading/writing data from/to the RX/TX buffer.

Two figures must match for a proper operation, which means that the data rate provided by the serial flash memory device must not exceed the average RX buffer readout data rate. Otherwise, the longer this state persists, it results into an RX buffer overflow. Similarly, the data consumed by the serial flash memory device must not exceed the average TX buffer fill rate. If this persists, it leads to an underrun.

AHB bus side (data read)

The total number of bus cycles for each DMA minor loop completion is added from the following components:

The following table provides certain examples for typical use cases:

Case 1: DMA needs to read 4 bytes from SRAM and provide to QuadSPI. It costs total four bus clock cycles. Then, DMA handshake adds additional six bus clock cycles, resulting in a total of [6 + 4 * (32/4) = 38] bus clock cycles.

Table 736. Access duration examples for bus clock side

TBCT[WMRK]	Number of bytes per DMA loop	Number of bus clock cycles for DMA minor loop	Time duration of DMA minor loop for 60 MHz bus clock frequency
3	16	6+(16/4)*4 = 22	~366ns
7	32	6+(32/4)*4 = 38	~633ns
11	48	6+(48/4)*4 = 54	~900ns
15	64	6+(64/4)*4 = 70	~1166ns

Case 2: DMA needs to read 32 bytes from SRAM and provide to QuadSPI. DMA handshake takes an additional six bus cycles, with 32 bytes DMA read from SRAM costs (8 + 3) core clock cycles. DMA writes 32 bytes to QuadSPI, takes 2 * (32/4) = 16 bus cycles with one additional CPU access to QuadSPI, costing two bus clock cycles. This results in a total 6 + (8+3)/2 + 2 * (32/4) +2 = 30 bus clock cycles.

Table 737. Access duration examples for bus clock side

TBCT[WMRK]	Number of bytes per DMA loop >	Number of bus clock cycles for DMA minor loop	Time duration of DMA minor loop for 80 MHz bus clock frequency
3	16	6 + (4+3) /2 + (16/4)*2 + 2 = 20	~333ns
7	32	6 + (8+3) /2 + (32/4)*2 + 2 = 30	~500ns
11	48	6 + (4+3)/2*3 + (48/4)*2 + 2 = 44	~733ns
15	64	6 + (8+3) + (64/4)*2 + 2 = 51	~810ns

NOTE

This table figure represents an ideal scenario; actual performance depends on how the chip integrates DMA and QuadSPI modules.

Serial flash memory device side (data read)

The number of serial flash memory cycles can be determined in the following way:

- Number of serial flash memory clock cycles is required to read 4 bytes, corresponding to one RX buffer entry (setup of command and address not considered): , two cycles for the Octal DDR mode in the individual flash memory mode, eight cycles for quad mode (SDR) instructions in individual flash memory mode, and so on.
- Overhead because of clock domain crossing: one cycle

The following table lists the number of clock cycles required to read the data from the serial flash memory corresponding to the different settings of RBCT[WMRK]:

Table 738. Access duration examples for serial flash memory side

RBCT[WMRK] setting	Num bytes per DMA loop [1]	Num SCKFx for 80 MHz SCKFx		Time duration of flash memory data readout for 80 MHz SCKFx (~12.5ns period)	
		IFM [2] quad	IFM quad DDR	IFM quad	IFM quad DDR
0	4	8	4	~100ns	~50ns
1	8	16	8	~200ns	~100ns
3	16	32	16	~400ns	~200ns
7	32	64	32	~800ns	~400ns
11	48	96	48	~1200ns	~600ns

1. DMA loop refers to one minor loop completion that is equivalent to one major loop iteration.
2. Individual flash memory mode

NOTE

The table figure represents an ideal scenario; actual performance depends on how the chip integrates with DMA and QuadSPI modules.

A complementary example is when the watermark is set to be too high. In such a case, the time taken by the DMA to read out the RX buffer entries should be lesser than the time taken by the controller to push in the remaining entries in the buffer.

IPS bus side (data write)

The total number of bus cycles for each DMA minor loop completion are added from the following components:

- Overhead for each minor loop, given by DMA controller: assume 10 cycles
- Overhead because to clock domain crossing: assume two cycles
- Number of bus clock cycles required for 16 bytes (128-bit write size): assume four cycles (read/write sequence of DMA controller)

Note that the size of the minor loop is determined by the size of TBCT[WMRK]; therefore, the overhead specified above distributes among (TBCT[WMRK]+1) write accesses of 32-bit each.

The following table provides some examples for typical use cases:

Table 739. Access duration examples for bus clock side

TBCT[WMRK]	Number of bytes per DMA loop [1]	Number of bus clock cycles for DMA minor loop	Time duration of DMA minor loop for 80 MHz bus clock frequency
3	16	12+4 = 16	~200ns
7	32	12+8 = 20	~250ns

1. DMA loop refers to one minor loop completion that is equivalent to one.

NOTE

The table figure represents an ideal scenario; actual performance depends on how the chip integrates with DMA and QuadSPI modules.

Serial flash memory device side (data write)

The number of serial flash memory cycles can be determined in the following way:

- Number of serial flash memory clock cycles required to write 16 bytes, corresponding to four TX buffer entry (setup of command and address not considered): Eight cycles for Octal DDR mode instructions in individual flash memory mode, 32 cycles for quad SDR writes in individual flash memory mode.
- Overhead due to clock domain crossing: one cycle

The following table lists the number of clock cycles required to read the data from the serial flash memory corresponding to the different settings of TBCT[WMRK]:

Table 740. Access duration examples for serial flash memory side

TBCT[WMRK] setting	Num bytes per DMA loop [1]	Num SCKFx		Time duration for consuming data at flash memory interface 100 MHz SCKFx (10 ns period)[2]		Time for FIFO to get empty[3]	
		IFM [4] quad	IFM octal DDR	IFM quad	IFM octal DDR	IFM quad	IFM octal DDR
3	16	32	8	320ns	80ns	2240ns	560ns
7	32	64	16	640ns	160ns	1920ns	480ns

1. DMA loop refers to one minor loop completion that is equivalent to one major loop iteration.
2. Not all flash memory devices support writes at 100 MH. See the flash memory data sheet for the actual page program frequency supported.
3. The assumption for these timings is that the TX Fifo is full when the transaction is initiated
4. Individual flash memory mode

NOTE

The tables mentioned above are only examples which must be correlated with the DMA in the system.

Considering the examples provided in the two tables above for TX FIFO, it is evident that depending on the relationship between the bus clock and serial flash memory clock frequencies, there are settings possible where the serial flash memory consumes data faster than the IPS bus can write data in TX buffer. In these cases, a TX buffer underrun situation occurs. To avoid TX buffer underrun, the data transaction size should be large enough.

79.5.5 Flash memory devices address mapping

QuadSPI is configured in Single mode for the supported flash memory port A

The sizes of the flash memory devices are mapped with the system memory space based on the configurations of the following registers:

- SFA1AD
- SFA2AD

The total memory region for the flash memory devices is mapped between QuadSPI_AMBA_BASE and TOP_ADDR_MEMA2 such that the corresponding CS is asserted based on SFA1AD and SFA2AD register configurations.

79.5.5.1 Single mode

For single-die flash memories, you must write the same value to SFA2AD register that you write to the SFA1AD register.

Following is a programming example for single mode single-die flash memory:

- QuadSPI_AMBA_BASE - 1000_0000h
- SFA1AD[TPADA1] - 2000_0000h
- SFA2AD[TPADA2] - 2000_0000h

The following figure illustrates the memory mapping for single mode QuadSPI configuration.

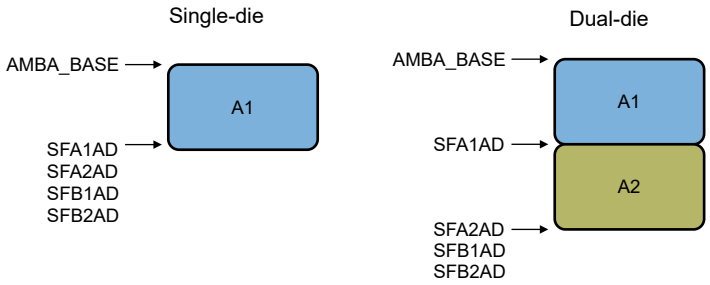


Figure 512. Memory map for Single mode

79.6 Byte ordering – endianness

The following topics show the byte ordering in 64-bit LE configuration for AHB buffer and 32-bit LE for TX/RX buffer.

79.6.1 Programming flash memory data

CPU writes instructions to the TBDR register, such as:

- Write TBDR: 4_03_02_01h
- Write TBDR: 8_07_06_05h

The following table shows the content against each TX buffer entry.

Table 741. Example of QuadSPI TX buffer

TX buffer entry	Content
0	4_03_02_01h
1	8_07_06_05h

Programming the TX buffer into the external serial flash memory device results in the following byte order to be sent to the serial flash memory:

- 01...02...03...04...05...06...07...08

79.6.2 Reading flash memory data into the RX buffer

Reading the content from the same address provides the following sequence of bytes, identical to the write case:

- 01...02...03...04...05...06...07...08

The following table shows the content against each TX buffer entry.

Table 742. Resulting RX buffer content

RX buffer entry	Content
0	4_03_02_01h
1	8_07_06_05h

79.6.2.1 Readout of the RX buffer through RBDRn

The RX buffer content appears at CPU read access through the peripheral bus interface in the following order:

- Read RBDR0: 4_03_02_01h
- Read RBDR1: 8_07_06_05h

79.6.2.2 Readout of the RX buffer through ARDBn

The RX buffer content appears at read access on the AMBA AHB interface at the QuadSPI module boundary:

- 32-bit access: Read ARDB0: 4_03_02_01h
- 32-bit access: Read ARDB1: 8_07_06_05h
- 64-bit access: Read ARDB0: 8_07_06_05_04_03_02_01h

79.6.3 Reading flash memory data into the AHB buffer

Reading the content from the same address as it was written to provides the following sequence of bytes, identical to the write case:

- 01...02...03...04...05...06...07...08

The following table shows the content against each TX buffer entry.

Table 743. Resulting AHB buffer content

AHB buffer entry	Content
0	8_07_06_05_04_03_02_01h

79.6.3.1 Readout of the AHB buffer through memory-mapped read

The AHB buffer content appears at read access on the AMBA AHB interface at the QuadSPI module boundary:

- 32-bit read access: 4_03_02_01h
- 32-bit read access: 8_07_06_05h
- 64-bit read access: 8_07_06_05_04_03_02_01h

79.7 Driving flash memory control signals in single and dual modes

In single and dual modes, the serial flash memory devices that can connect to the QuadSPI module expect additional control signals on the inputs, which are connected to IOFA[3], IOFA[2] in the quad mode. For easy interfacing, the outputs IOFA[3:2] for flash memory A are driven to the logic state given by the configuration fields MCR[ISD3FA], MCR[ISD2FA].

These outputs are driven all the time to the logic level programmed in the MCR except the time when quad commands of the serial flash memory are executed. See the specifications of the related serial flash memory device for details about the inactive level.

79.8 Serial flash memory devices

Several different vendors make flash memory devices with a QuadSPI interface. At present, there is no set standard for the QuadSPI instruction set. The most common commands currently have the same instruction code for all vendors; however, some commands are unique to specific vendors. Some of the example sequences are provided in the following sections.

79.8.1 Example sequences

This section provides the example sequences of the QuadSPI module.

Table 744. Exit 4 x I/O read enhance performance mode (XIP) (Macronix) and read status

Instruction	Pad	Operand	Description
CMD	0h	EBh	4xIO read command
ADDR	2h	18h	24-bit address to be sent on four pads
MODE	2h	0h	2 mode cycles (exit XIP)
DUMMY	2h	4h	4 dummy cycles
READ	2h	8h	Read 64 bits
CMD	0h	5h	Read Status register
READ	0h	1h	Status register data
STOP	0h	0h	Stop, instruction over

79.8.1.1 Read command (hyperflash memory/HyperRAM)

This section provides the read command sequences (hyperflash memory/HyperRAM) of the QuadSPI module.

Table 745. Read command (hyperflash memory/HyperRAM)

Instruction	Pad	Operand	Comment
CMD_DDR	3h	A0h	Read command with continuous burst type
ADDR_DDR	3h	18h	24-bit row address

Table continues on the next page...

Table 745. Read command (hyperflash memory/HyperRAM) (continued)

CADDR_DDR	3h	10h	16-bit column address with lower 3 bits valid and rest 0
Dummy	3h	0hF	15 dummy cycles
READ_DDR	3h	4h	32-bit data read on 8 pads
STOP	3h	0h	Stop, instruction over

Hyperflash memory/HyperRAM is a word addressable flash memory. This means that each address accesses a word-wide (two-byte) data value. The software should ensure that when Hyperflash memory/HyperRAM is connected to the controller, the value of SFACR[WA] must be 1. With this value, the controller remaps a byte addressable access to a word addressable access.

79.8.1.2 Read status register (hyperflash memory/HyperRAM)

This section provides details related to the read status register of the QuadSPI module.

Table 746. Read status register (hyperflash memory/HyperRAM)

Instruction	Instruction sequence	Pad	Operand	Description
CMD_DDR	Read Pre Command	3h	0h	Write command with wrapped burst type
ADDR_DDR		3h	18h	24-bit row address (0000AAh)
CADDR_DDR		3h	10h	16-bit column address with lower 3 bits valid and rest 0(0005h) treated as command
CMD_DDR		3h	0h	Write command with wrapped burst type
CMD_DDR		3h	70h	Write data to be sent to flash memory as pre-command
CMD_DDR	Command phase (fourth/final chip select phase)	3h	A0h	Read command with continuous burst type
ADDR_DDR		3h	18h	24-bit row address
CADDR_DDR		3h	10h	16-bit column address with lower 3 bits as valid and rest 0
DUMMY		3h	0hF	15 dummy cycles
READ_DDR		3h	4h	32-bit data read on 8 pads
STOP		3h	0h	Stop, instruction over

79.8.1.3 Word program (hyperflash memory/HyperRAM)

This section provides the word program (hyperflash memory/HyperRAM) of the QuadSPI module.

Table 747. Word program (hyperflash memory/HyperRAM)

Instruction	Instruction sequence	Pad	Operand	Description
CMD_DDR	Unlock sequence 1 (first chip select phase)	3h	0h	Write command with wrapped burst type
CMD_DDR		3h	0h	8-bit address 00h treated as command
CMD_DDR		3h	0h	8-bit address 00h treated as command
CMD_DDR		3h	AAh	8-bit address AAh treated as command
CADDR_DDR		3h	10h	16-bit column address with lower 3 bits valid and rest 0(0005h) treated as command
CMD_DDR		3h	0h	Write command with wrapped burst type
CMD_DDR		3h	AAh	Write data to be sent to flash memory as pre-command
CMD_DDR	Unlock sequence 2 (second chip select phase)	3h	0h	Write command with wrapped burst type
CMD_DDR		3h	0h	8-bit address 00h treated as command
CMD_DDR		3h	0h	8-bit address 00h treated as command
CMD_DDR		3h	55h	8-bit address 55h treated as command
CADDR_DDR		3h	10h	16-bit column address with lower 3-bits valid and rest 0(0002h) treated as command
CMD_DDR		3h	0h	Write command with wrapped burst type
CMD_DDR		3h	55h	Write data to be sent to flash memory as pre-command
CMD_DDR	Program setup phase (third chip select phase)	3h	0h	Write command with wrapped burst type
CMD_DDR		3h	0h	8-bit address 00h treated as command
CMD_DDR		3h	0h	8-bit address 00h treated as command

Table continues on the next page...

Table 747. Word program (hyperflash memory/HyperRAM) (continued)

CMD_DDR		3h	AAh	8-bit address AAh treated as command
CADDR_DDR		3h	10h	16-bit column address with lower 3 bits valid and rest 0(0005h) treated as command
CMD_DDR		3h	0h	Write command with wrapped burst type
CMD_DDR		3h	A0h	Write data to be sent to flash memory as pre-command
CMD_DDR	Command phase (fourth/final chip select phase)	3h	0h	Write command with wrapped burst type
ADDR_DDR		3h	18h	24-bit row address
CADDR_DDR		3h	10h	16-bit column address with lower 3 bits as valid and rest as 0
WRITE_DDR		3h	2h	2-byte data written on 8 pads (D1D2)
STOP		3h	0h	Stop, instruction over

79.8.1.4 Fast read sequence (Macronix/Numonyx/Spansion/Winbond)

The following table shows the fast read sequence for Macronix/Numonyx/Spansion/Winbond flash memories.

Table 748. Fast read sequence

Instruction	Pad	Operand	Description
CMD	0h	Bh	Fast read command = 0Bh
ADDR	0h	18h	24 address bits to be sent on one pad
DUMMY	0h	8h	Eight dummy cycles
READ	0h	4h	Read 32 bits on one pad
JMP_ON_CS	0h	0h	Jump to instruction 0 (CMD)

NOTE

If DLL is disabled then JMP_ON_CS or STOP instruction can be used else only STOP instruction can be used.

79.8.1.5 Fast dual I/O DT read sequence (Macronix)

The following table shows the fast dual I/O DT read sequence for Macronix flash memories.

Table 749. Fast dual I/O DT read sequence

Instruction	Pad	Operand	Description
CMD	0h	BDh	Fast dual I/O DT read command = BDh
ADDR_DDR	1h	18h	24 address bits to be sent on two pads in the DDR mode
MODE4_DDR	1h	0h	P2=P0 or P3=P1 is necessary. See the Macronix data sheet for details. One clock cycle for mode.
DUMMY	1h	6h	Six dummy cycles
READ_DDR	1h	4h	Read 32 bits on two pads in the DDR mode
JMP_ON_CS	0h	0h	Jump to instruction 0 (CMD)

NOTE

If DLL is disabled then JMP_ON_CS or STOP instruction can be used else only STOP instruction can be used.

79.8.1.6 Fast read quad output (Winbond)

The following table shows the fast read quad output sequence for Winbond memories

Table 750. Fast read quad output sequence

Instruction	Pad	Operand	Description
CMD	0h	6Bh	Fast read quad output command = 6Bh
ADDR	0h	18h	24 address bits to be sent on one pad
DUMMY	2h	8h	Eight dummy cycles
READ	2h	4h	Read 32 bits on four pads
JMP_ON_CS	0h	0h	Jump to instruction 0 (CMD)

NOTE

If DLL is disabled then JMP_ON_CS or STOP instruction can be used else only STOP instruction can be used.

79.8.1.7 4 x I/O read enhance performance mode (XIP) (Macronix)

The following table shows the 4 x I/O read enhance performance mode for Macronix flash memories. The enhanced performance mode is also known as XIP mode.

Table 751. Fast read quad output sequence

Instruction	Pad	Operand	Description
CMD	0h	EBh	4xI/O read command = EBh

Table continues on the next page...

Table 751. Fast read quad output sequence (continued)

Instruction	Pad	Operand	Description
ADDR	2h	18h	24 address bits to be sent on four pads
MODE	2h	A5h	Two mode cycles
DUMMY	2h	4h	Four dummy cycles
READ	2h	4h	Read 32 bits on four pads
JMP_ON_CS	0h	1h	Jump to instruction 1 (ADDR)

When in XIP mode, the software must ensure that all the flash memories connected to the controller are in this mode. As a part of initializing the controller, all the flash memories might be enabled with XIP by carrying out dummy reads.

79.8.1.8 Dual command page program (Numonyx)

The following table shows the dual command page program sequence for Numonyx flash memories.

Table 752. Dual command page program sequence

Instruction	Pad	Operand	Description
CMD	1h	2h	Dual command page program = 02h on 2 pads
ADDR	1h	18h	24 address bits to be sent on two pads
WRITE	1h	20h	Write 32 bytes on two pads
STOP	0h	0h	Stop, instruction over

79.8.1.9 Sector erase (Macronix/Numonyx/Spansion)

The following table shows the Sector erase sequence for the Macronix/Numonyx/Spansion flash memories.

Table 753. Sector erase sequence

Instruction	Pad	Operand	Description
CMD	0h	20h	Sector erase command = 20h
ADDR	0h	18h	24 address bits to be sent on one pad
STOP	0h	0h	Stop, instruction over

79.8.1.10 Read status register (Macronix/Numonyx/Spansion/Winbond)

The following table shows the read status register sequence for Macronix/Numonyx/Spansion/Winbond flash memories.

Table 754. Read status register sequence

Instruction	Pad	Operand	Description
CMD	0h	0h5	Read status register command = 05h
READ	0h	0h1	Read status register data
STOP	0h	0h	Stop, instruction over

79.8.1.11 Data learn instruction sequence

The following table shows the data learn sequence for 4 I/O flash memory.

Table 755. Data learn instruction sequence (4 I/O)

Instruction	Pad	Operand	Description
CMD	0h	6Bh	Fast read quad command = 6Bh
ADDR_DDR	2h	18h	24 address bits in the DDR mode to be sent on four pads
DUMMY	2h	8h	8-cycle dummy
DATA_LEARN	2h	1h	1-byte data learn
READ_DDR	2h	6h	Read 6 bytes in 4 pads[1]
JMP_ON_CS	0h	0h	Jump to Inst 0 (CMD)

1. Data learn instruction needs to be followed by a read DDR instruction of minimum 6 bytes in case of 4 I/O mode.

NOTE

If DLL is disabled then JMP_ON_CS or STOP instruction can be used else only STOP instruction can be used.

The following table shows the data learn sequence for 8 I/O (hyperflash memory) flash memory.

Table 756. Data learn instruction sequence (8 I/O)

Instruction	Pad	Operand	Description
CMD_DDR	3h	AOh	Read command for hyperflash memory
ADDR_DDR	3h	18h	24-bit row address[1]
CADDR_DDR	3h	10h	16-bit column address
DUMMY	3h	Fh	15-cycle dummy
DATA_LEARN	3h	1h	1-byte data learn
READ_DDR	3h	Ch	Read 12 bytes in 8 pads[2]
STOP	3h	0h	Stop, instruction over

1. The address needs to be aligned, which means, no latency should be there between the RWDS edges.
2. The data learn instruction needs to be followed by a read DDR instruction of minimum 12 bytes in case of 8 I/O mode.

The following table shows the data learn sequence for 4 I/O flash memory.

Table 757. Data learn instruction sequence (4 I/O)

Instruction	Pad	Operand	Description
CMD	0h	6Bh	Fast read quad command = 6Bh
ADDR_DDR	2h	18h	24 address bits in DDR mode to be sent on four pads
DUMMY	2h	8h	8 dummy cycles
DATA_LEARN	2h	1h	1-byte data learn
READ_DDR	2h	6h	Read 6 bytes in 4 pads
JMP_ON_CS	0h	0h	Jump to Inst 0 (CMD)

NOTE

If DLL is disabled then JMP_ON_CS or STOP instruction can be used else only STOP instruction can be used.

The following table shows the data learn sequence for 8 I/O (Hyperflash) flash memory.

Table 758. Data learn instruction sequence (8 I/O)

Instruction	Pad	Operand	Description
CMD_DDR	3h	AOh	Read command for hyperflash memory
ADDR_DDR	3h	18h	24-bit row address[1]
CADDR_DDR	3h	10h	16-bit column address
DUMMY	3h	Fh	15-cycle dummy
DATA_LEARN	3h	1h	1-byte data learn
READ_DDR	3h	8h	Read 8 bytes in 8 pads
STOP	3h	0h	Stop, instruction over

1. The address needs to be aligned, which means, no latency should be there between the RWDS edges.

NOTE

If DLL is disabled then JMP_ON_CS or STOP instruction can be used else only STOP instruction can be used.

79.9 Sampling of serial flash memory input data

79.9.1 Basic description

QuadSPI is used to read data from the serial flash memory device. Depending on the actual implementation, there is a delay between the internal clocking in the QuadSPI module and the external serial flash memory device. See the following figure for an overview of this scheme.

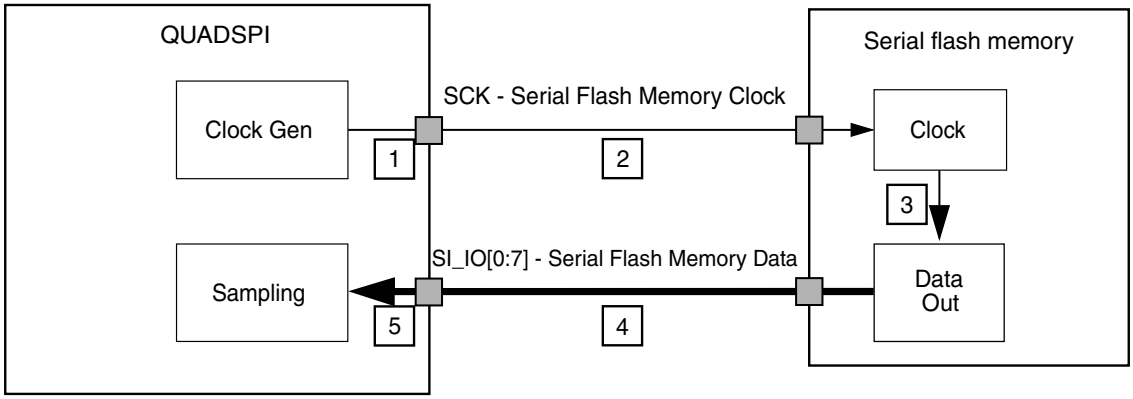


Figure 513. Serial flash memory sampling clock overview

The rising edge of the internal reference clock is taken as timing reference for the data output of the serial flash memory. After a time of t_{total_delay} the data arrives at the internal sampling stage of the QuadSPI module. Considering the figure provided here, the following parts of the delay chain contribute to t_{total_delay}:

- Output delay of the serial flash memory clock output of the device containing the QuadSPI module
- Wire delay of application/PCB from the device containing the QuadSPI module to the external serial flash memory device
- Clock to data out delay of the external serial flash memory device, including input and output delays
- Wire delay of application/PCB from the external serial flash memory device to the device containing the QuadSPI module
- Device delay corresponding to the input data

NOTE

The t_{total_delay} is specific to the characteristics of the actual implementation.

79.9.2 DQS sampling method

79.9.2.1 Basic description

In the DQS mode, the data strobe signal (DQS/RWDS) is used to sample the read data. Here, both DQS and the data sent by the flash memory move in the same direction; therefore, it is relatively easier to achieve at higher frequencies.

When using DQS for SDR reads, QuadSPI internally samples the incoming data on the rising edge of the strobe signal.

The next figure shows the sampling read data in the SDR mode using the DQS.

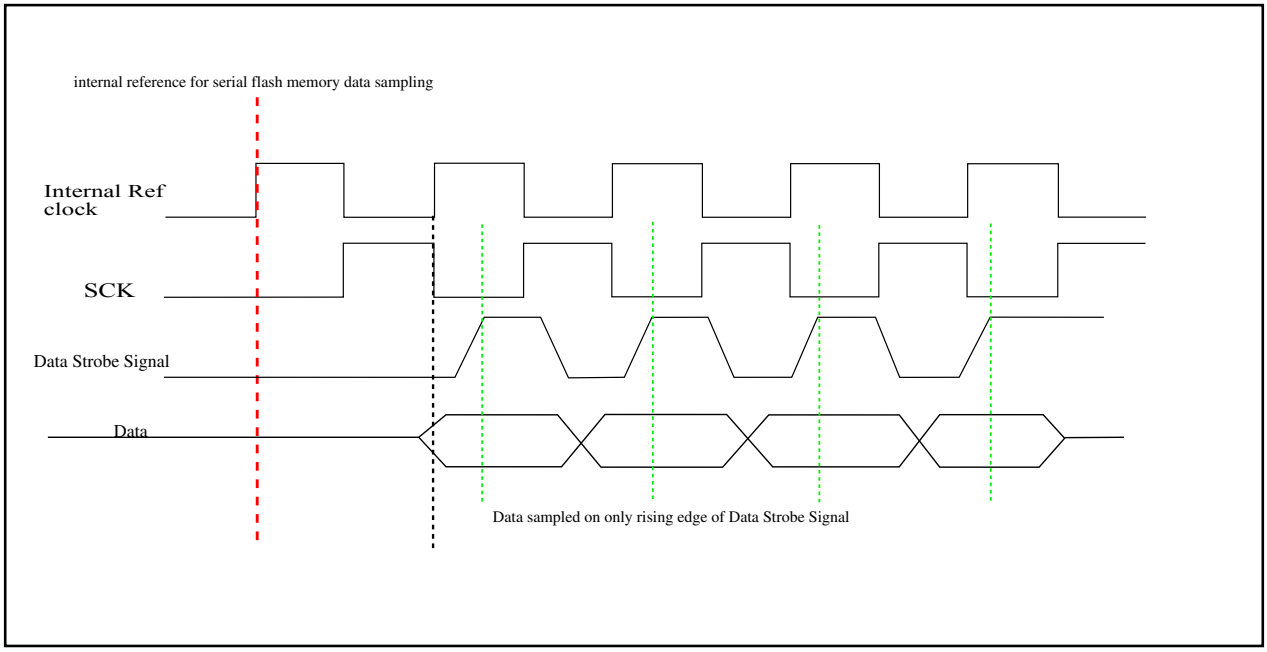


Figure 514. Data strobe functionality in SDR mode for read operation.

NOTE

Consider "Data Strobe Signal" as "Data Strobe Signal driven by memory" and "Data" as "Data from memory".

NOTE

Refer to the Datasheet for specific timing waveforms of QuadSPI

When using the DQS for DDR reads, QuadSPI internally samples the incoming data on both the edges of the strobe signal. See the next figure for details.

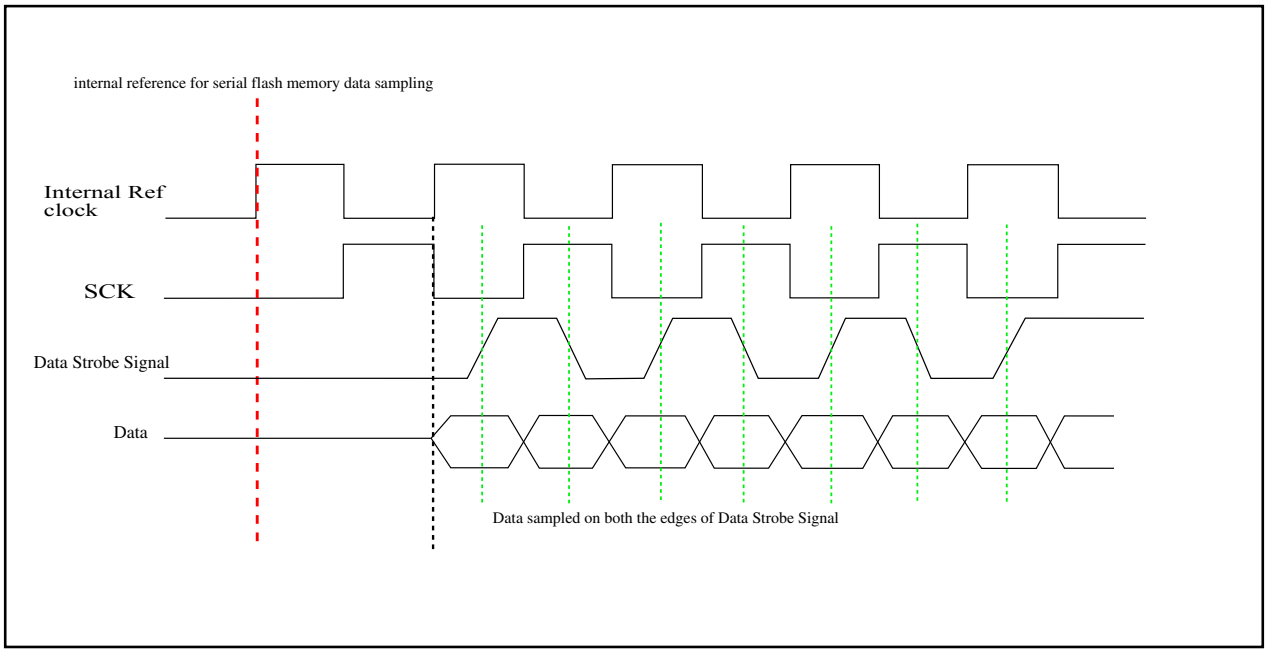


Figure 515. Data strobe functionality in DDR mode for read operation.

NOTE

Consider "Data Strobe Signal" as "Data Strobe Signal driven by memory" and "Data" as "Data from memory".

NOTE

For Specific details - Refer to the Data Sheet specification of QuadSPI module.

79.9.2.2 External DQS

In serial flash memories supporting DQS, the data strobe signal is an output from the flash memory device that indicates when data is being transferred from the flash memory to the host controller. The data is then captured by the controller on:

- Only one edge (either rising or falling edge) of DQS signal in the SDR mode
- Both rising/falling edges of the DQS signal in the DDR mode

This can be enabled by programming MCR[DQS_FA_SEL] = "11" for flash memory A. This mode supports the following configuration:

- Both high/low frequency delay chain manual programming using DLLCRA[SLV_DLY_COARSE] and DLLCRA[SLV_FLINE_OFFSET]
- DLL-assisted sampling only using and DLLCRB[DLLEN]
- DLL-assisted auto data learning using and DLLCRB[DLLEN] along with MCR{DLPEN]
- should be disabled as DLL is used

NOTE

This mode may not be available on the chip. See the "Supported read modes" section in the chip-specific QuadSPI information for the read modes that this chip supports.

79.9.2.3 Dummy Pad loopback

The internal clock is loop-backed from the dummy internal pad to compensate data pad delays. This can be enabled by configuring the value of MCR[DQS_FA_SEL] as "01" for flash memory A. This mode can be used with the following configuration:

- High/low frequency delay chain manual programming in bypass mode using DLLCRA[SLV_DLY_COARSE] and DLLCR[FREQEN].
- DLL-assisted sampling only using DLLCRA[DLLEN]

NOTE

Refer to DS or "Chip-specific section" of QSPI for Fixed sampling Tap specific settings/details with out without DLL (in bypass mode)

- DLL-assisted auto data learning using DLLCRA[DLLEN] and along with MCR[DLPEN]

NOTE

Refer to Auto-DataLearning (4x Sampling method) section with DLL for further details

NOTE

This mode may not be available on the chip. See the "Supported read modes" section in the chip-specific QuadSPI information for the read modes that this chip supports.

79.10 Data learning

79.10.1 Basic description

Data learning is used to manage varying data valid windows from the flash memory as well as any variations in the chip based on PVT conditions in the DDR mode. QuadSPI provides this feature via the DATA_LEARN instruction for all flash memories, irrespective of whether the flash memory supports it. The DATA_LEARN instruction accepts an operand that defines the number of bits of the known pattern for which the data learning has to be done. The known pattern is provided in the DLPR register.

QuadSPI supports the following data learning:

- QuadSPI supports data learning with DLL enabled only

The following sections explain data learning in detail.

79.10.2 DQS sampling method

The semi-automatic data learning is supported in the DQS sampling method. To start data learning:

1. Configure a known pattern in the Data Learn register inside flash memories that support data learning. The pattern should be selected to have multiple low and high transitions in the data bus.

   NOTE

   Certain flash memories provide data learning as a feature for DDR data reads.

2. Set up a QuadSPI DDR data read sequence, inserting a DATA_LEARN instruction before the READ_DDR instruction. The flash memory returns the data learning pattern in between dummy cycles in every read command. Each I/O gives the same DLP value for every clock edge.

   NOTE

   For flash memories that *do not* support data learning, configure a known location in flash memory with a known pattern. The pattern should not have 0h or FFh included. The pattern should be selected to have multiple low and high transitions in the data bus. Set up a QuadSPI DDR data read sequence, inserting a DATA_LEARN instruction. The address in the SFAR register should point to the known location within the pattern. Configure the known pattern in DLPR[DLPV]. See Programming the data learning pattern in flash memory for details on how the data has to be ordered in memory for correct operation.

3. Select a sampling point. See chip-specific QuadSPI information for selection of the sampling point.
4. Initiate the read via a peripheral transaction.
5. QuadSPI reads the data from the flash memory. It then encounters the DATA_LEARN instruction and samples the incoming data on both edges (rising and falling) of the DQS.

   NOTE

   If the data from the flash memory does not match the data learning pattern, the FR[DLPFF] flag is set. The QuadSPI module reports no sampling point automatically.

6. Repeat steps 2-7 with varying VT conditions for a particular process until FR[DLPFF] is set. The sampling points where no FR[DLPFF] is set signify a valid setting.

In the case of multiple valid settings, the software should select the middle point. The sampling point should be fixed within the above selected point for the next READ transactions.

NOTE

Ensure that QuadSPI is not accessed during calibration.

This feature might be used to auto-calibrate the sampling point for DDR reads, and auto-calibration might be triggered at fixed intervals, or depending on a change in PVT conditions.In case the sampling point needs to be changed based on initiating options (such as, DLL lock status, time period, or software forced interrupt), data learn instruction has to be re-initiated.

The following figure shows data learning in DQS mode.

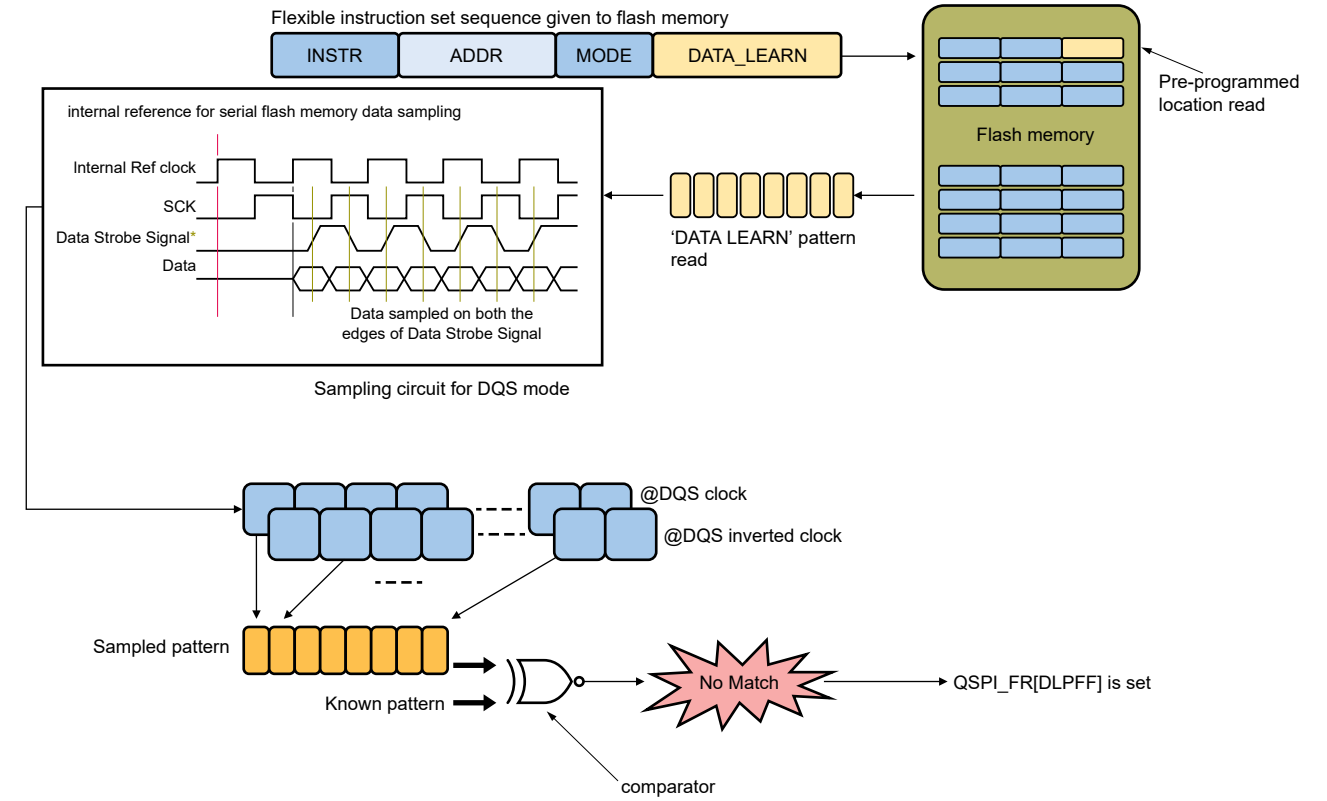


Figure 516. Data learning with DQS sampling method

NOTE

This mode might not be available on the chip. See the chip-specific QuadSPI information for the read modes that this chip supports.

79.10.3 Programming the data learning pattern in flash memory

Example scenario: If the value of DLPR[DLPV] is 43h, QuadSPI controller tries to match the sequence of bits sent on selected data lines with the pattern in DLPR[DLPV]. The following table shows the data programmed in flash memory with endianness taken into consideration.

Table 759. Programming the data learning pattern in flash memory

	Single Pad	Double Pad	Quad Pad	Octal Pad
DLPR IO1 pattern	0x349A	0x349A	0x3514	0x349A
DLPR IO3 pattern	NA	NA	0x349A	0x3514
Flash Pattern	0x2C59	0xF04CC226	0x0DF02D0D 0x00FFF020	0xF708f700 0xF70000FF 0x00FF0008 0x0000FFFF

79.11 DLL and delay chain usage

The DLL is a general-purpose, dynamically adaptive clock delay module. It provides the ability to select a quantized delay (in fractions of the clock period) regardless of on-chip variations such as process, voltage, and temperature (PVT). The DLL is suitable for applications where accurate delay adjustment is required, such as in case of DDR interfaces.

DLL working concept

The DLL control loop consists of a counter, reference delay line, and phase detector which operate on the ref_clock reference clock input. The reference clock (clk_ref) is fed into the reference delay line. After reset, a single delay tap is selected. A phase detector is used to detect the condition where a half shift has occurred. In addition to this, signals are generated to either increment or decrement the counter, which controls the delay line (if the half-phase detect condition is not met). Any changes in the delay of the individual elements of the delay chain (because of PVT) automatically cause the phase detector logic to determine if a change in the counter value is required. After the half-phase shift is detected, an internal lock signal is generated, and after a lock event occurs, the slave delay chain is ready to be updated based on a triggering event.

Slave delay chain programming sequence—

Following is the programming sequence for DLL bypass mode.

1. Program DLLCRA[SLV_EN]=1, DLLCRA[SLV_DLL_BYPASS]=1, and DLLCRA[SLAVE_AUTO_UPDT]=0.
2. Program the following fields to provide the desired DQS delay for sampling: DLLCRA[SLV_FINE_OFFSET], DLLCRA[SLV_DLY_COARSE], and DLLCR[FREQEN]. See the chip-specific QuadSPI information for the supported programming settings.
3. Program DLLCRA[SLV_UPD]=1 to load these values in the slave delay chain.
4. Check the slave delay chain update status by polling DLLSR[SLVA_LOCK]=1 and clear DLLCRA[SLV_UPD] after confirming the update state.

Following is the programming sequence for DLL auto update mode.

1. Program DLLCRA[SLV_EN]=1, DLLCRA[SLV_DLL_BYPASS]=0, and DLLCRA[SLAVE_AUTO_UPDT]=1.
2. Program the DLL configuration by using DLLCRA[DLL_REFCNTR] and DLLCRA[DLLRES]. See the chip-specific QuadSPI information for the supported DLL configuration settings.
3. Program the slave settings to delay DQS by using these fields: DLLCRA[SLV_FINE_OFFSET], DLLCRA[SLV_DLY_OFFSET], and DLLCR[FREQEN]. See the chip-specific QuadSPI information for the supported settings.
4. If offset delay needs to be updated on the slave chain, program DLLCRA[SLV_UPD]=1.
5. Enable DLL by programming DLLCRA[DLLEN]=1 and reset DLLCRA[SLV_UPD]=0. Slave delay chain is updated automatically and can be checked by polling DLLSR[SLVA_LOCK]==1

79.12 Secure flash protection

This secure flash module enforces access control policies based on the MGID, privilege and secure attributes of the IPS transactions. All accesses throughout the flash device need to be monitored to determine the validity of all accesses. If transaction from a given master has appropriate access rights, it is forwarded to flash, else the access is denied, and an error is generated.

There are two level of checks present in this module, based on MDAD and FRAD descriptors programmed. First level checks the input transactions based on secure attribute and MGID associated with that transaction which are checked according to MDAD. Second level check selects the appropriate FRAD based on address range of transaction. And each FRAD allows the transaction to pass through only when secure, privilege and other MGID related attributes are matching. Else the transaction is not forwarded and IPS bus error/interrupt is generated. Flash regions can be programmed to cover the entire address space to provide a default set of access permissions. For flash read transactions only first level of check is enabled.

Software can bypass both or anyone of these checks by writing to MGC[GVLDFRAD], MGC[GVLDMDAD] and MGC[GVLD] fields.

The IPS masters should write on QuadSPI register (SFAR, IPCR and TBDR) addresses for generating transfers to flash. SFP block monitors these IPS transfers and checks if the write is being done on SFAR or IPCR registers and forwards them for MDAD and FRAD checks. If the transaction passes both checks then it is passed through to QuadSPI register else bus error or interrupt is

generated. FRAD check is only done for flash write transactions not for read. Actual write on QuadSPI SFAR and IPCR register is done by SFP module only after the transaction passes MDAD and FRAD checks. There is only a single IPS slave interface which can be used for QuadSPI IPS register access and SFP IPS register access.

An IPS master transaction is secure if IPS_NONSECURE_ACCESS attribute is set to 0 for that transaction. An IPS master transaction is privileged if IPS_SUPERVISOR_ACCESS attribute is set to 1 for that transaction.

QuadSPI SFP related register sets are present here.

79.12.1 MDAD

These descriptors form the first level of checks on the input IPS transaction. There are two target group queues (TG0 and TG1) present in SFP block and each of these queues can be programmed with different set of MGAD descriptors checks in TG*n*MDAD registers. These registers are under access control and can only be programmed by privilege masters. These descriptors (one or both) should be programmed before start of any flash transaction else any IPS write on SFAR, IPCR or TBDR registers will generate an IPS bus error. After programming the target group descriptor the TG*n*MDAD[VLD] should be set to 1 for that register.

There are two target group queues (TG0 and TG1) so two transactions by different master ID can be stored inside SFP block while one of them is being processed by QuadSPI.

79.12.1.1 MDAD checks

- Secure check – This is decided by the TG*n*MDAD[SA] field of the MDAD descriptor. If the fields are set to 01 then queue is reserved for only non-secure transactions. If set to 10 then queue is reserved for only secure transactions. If the bits are set to 11 then this check is bypassed.
- MasterID check – This check compares themasterID of the input transaction with the MID match value set inTG*n*TG*n*MDAD[MIDMATCH]fields and allows only matching transactions inside the queue. There are 6 mask fields (TG*n*TG*n*MDAD[MASK]) inside the descriptor. If the mask type (TG*n*TG*n*MDAD[MASKTYPE]) is set to 0 then the input transactions master ID is ANDed with the mask bits before comparing with the MID match value. If the mask type is set to 1 then the input master ID is ORed with the masks bits and then compared with the MID match bits. Multiple master IDs can be allowed inside the queue using this mask.

When an IPS write is done to SFAR or IPCR registers then SFP block checks the attributes with MDAD descriptors of both the queues. If the attributes match with descriptor of any queue, then the value is written inside that queue and master ID is assigned to that queue. The queue is considered locked once the SEQID bits of IPCR register are written. So software should ensure that IPCR SEQID bits are written only after SFAR and other IPCR bits like DATSZ are written in that queue. The queue will be unlocked when flash transfer is completed for that transaction.

Once a master ID is assigned to a queue then transactions with only that master ID are allowed inside the queue. Any write of SFAR/IPCR is mapped to this queue even if other queue is free. So same master ID cannot have transaction on both queues at a time. The data in the queue is cleared once the flash transaction is finally completed by the QuadSPI.

The data path for input IPS writes to SFAR or IPCR registers are shown in the following figure.

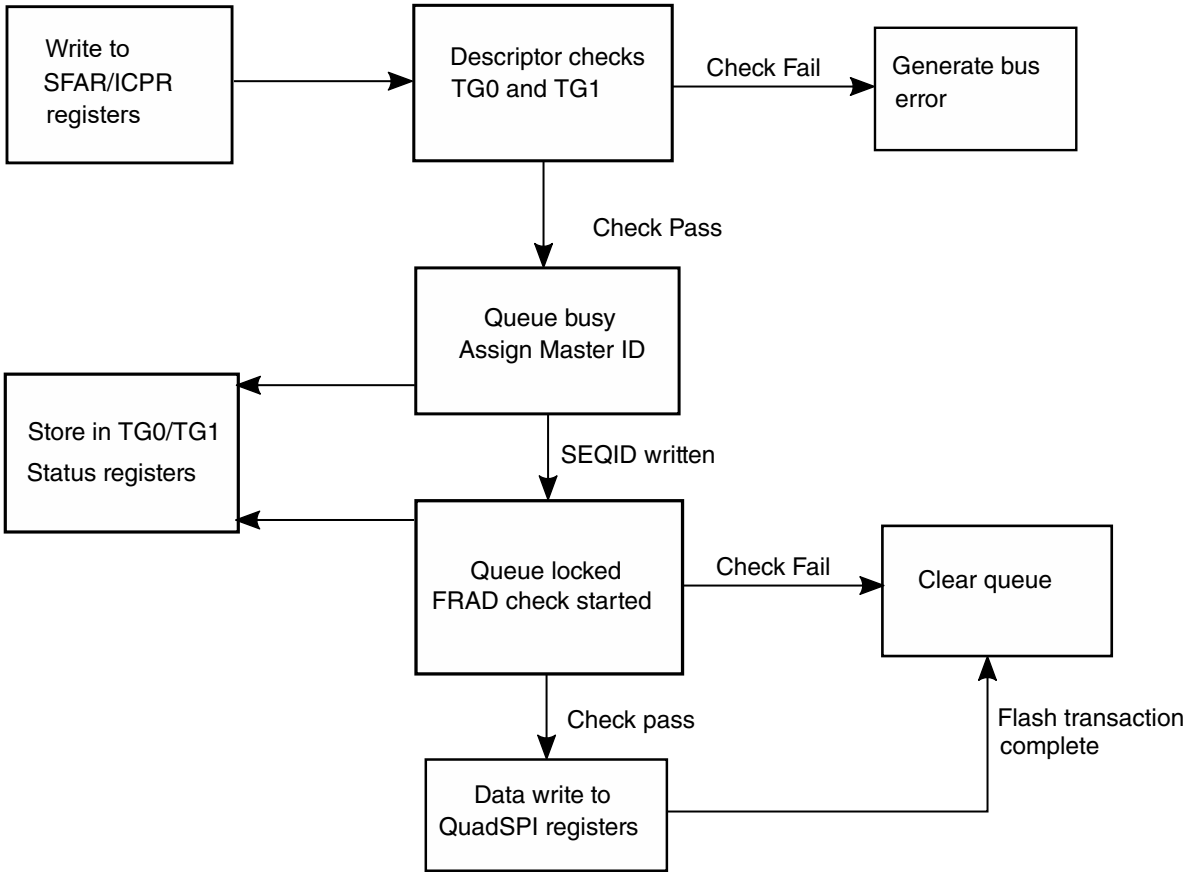


Figure 517. Data path for input IPS write to QuadSPI SFAR or IPCR registers

If the SFAR write passes first level of MDAD checks then the value is written in SFP register TG*n*SFAR and valid bit is set in TG*n*SFARS register and queue is considered as busy. Similarly when ICPR write passes the checks, its value is written into TG*n*IPRCS register and TG*n*IPRCS[VLD] field is set to 1 and queue is considered as locked. It is mandatory that IPCR is written only after SFAR has been written for a queue and queue is busy. If IPCR is written before SFAR then it will result in IPS transfer error. Master should ensure that it completes the programming sequence for flash access request (write SFAR and IPCR with SEQID) so that target queue is not left after writing SFAR or IPCR without the SEQID. Both queues TG0 and TG1 are processed in round robin arbitration. The queue receiving the data first is processed first and next queue is processed after the first transfer completes.

NOTE

TBDR register write is not allowed until the SFAR and IPCR entries of that master has passed MDAD and FRAD checks and the transaction is queued for writing to QuadSPI. If TBDR registers are written before arbitration win, transfer error will be generated. IPS master can read FSMSTAT register to check the status of their transaction. TBDR access is allowed when FSMSTAT[VLD] field is set and for the master whose master ID matches with fields [11:8] of this register, and field [1:0] are set to 01 or 10. IPCR and SFAR registers are forwarded to QuadSPI only after the TX buffer is filled till watermark level in case of write transactions. IPCR should be written only after SFAR has been written successfully to a queue.

79.12.1.2 Error conditions

The error conditions for first level of MDAD checks and the response is given in table below. These error conditions can occur when SFAR, IPCR or TBDR registers are being written.

NOTE

Queue is considered busy if it has been written successfully once with any of the values SFAR, IDATSZ or PAR.
Queue is considered locked when SEQID has been written successfully by passing MDAD checks.

Table 760. MDAD check error conditions

Error condition	Response	Common/Queue error
None of the queue descriptors are programmed	IPS transfer error is generated	Common error
Both queues are locked	IPS transfer error is generated	Common error
Both queues are busy and another master writes whose master ID is not matching with either queue.	IPS transfer error is generated.	Common error
Queue busy and same master writes SFAR or IPCR registers with wrong security attribute	IPS transfer error is generated and queue is cleared (SFAR and IPCR needs to be written again for transfer to start).	Queue error
Queue is locked and same master writes	IPS transfer error is generated	Common error
If master ID of transaction doesn't qualify MID check of any of the queues	IPS transfer error is generated	Common error
If both the queues are empty and security attribute of transaction doesn't match with any of the queues	IPS transfer error is generated	Common error
TBDR register is written before SFAR and IPCR SEQID qualifies MDAD checks	IPS transfer error is generated	-
If none of FRAD descriptors are programmed	IPS transfer error is generated	Common error
IPCR is written before SFAR	IPS transfer error is generated	Common Error - If the transaction doesn't matches the security attribute check or MID check of any MDAD. No common or queue error - If it matches security and MID attribute of any queue.

When a common error is generated its details are latched in IPSERROR register and corresponding error field is set to 1. This register will tell if the transaction failed because of queue locked, MID mismatch or security failure along with the master ID of that transaction. This register will keep the last error transaction latched unless cleared by writing 1 to CLR bit of this register. Then from next cycle onwards this register can latch new errors. Common error will also be generated if none of the FRAD descriptors are programmed. In case a queue is busy and same master writes IPCR or SFAR with wrong security attributes the queue is cleared, the Target Group n SFAR Status (TG0SFARS - TG1SFARS) valid bit goes low. In this case the master should write the SFAR an IPCR again for proper transaction to start.

In case of queue error the details are latched in Target Group n IPCR Status (TG0IPCRS - TG1IPCRS) or Target Group n SFAR Status (TG0SFARS - TG1SFARS) registers. These registers will also keep the error status latched until cleared by writing 1 to CLR bit of respective registers.

79.12.2 FRAD

These descriptors form the second level check for IPS flash write and flash erase transactions. IPS read transaction are not processed by this level of check and are passed through after MDAD checks. There are 8 FRADs which can be programmed by privilege masters. So the access checks can be programmed for up to 8 unique address ranges. Each of the address range

is equipped with different access attribute checks and based on this write access can be controlled for the flash region. These descriptors must be programmed before starting transactions on flash.

After programming the FRAD*n*_WORD0, FRAD*n*_WORD1, FRAD*n*_WORD2 and FRAD*n*_WORD3 registers, FRAD*n*_WORD3[VLD] field should be written 1. SFP block doesn't consider the FRAD descriptors which are not valid. If none of the FRAD are programmed, then error interrupt will be generated if enabled, and transaction is not forwarded to QuadSPI.

There is a descriptor lock feature to prevent spurious changes to the FRAD. FRAD*n*_WORD3[LOCK] can be programmed to make all FRAD registers as read only or to allow write access to all or some masters with specific master ID.

79.12.2.1 FRAD checks

- Address range checks – Start and end address for FRAD region are programmed in FRAD*n*_WORD0 and FRAD*n*_WORD1 registers. Start address of IPS flash transaction written in SFAR register and data size programmed in IPCR register, is used to check if the transaction lies within any of the FRAD address ranges. This address range match will be used to select the corresponding FRAD from which other access checks will be done. The start and end address are programmed around 64 KB boundary so 16 MSB are only used for range check..

  The data size should not be programmed as 0 in IPCR register for normal data transfers to flash and actual size should be programmed. In the cases where only flash instruction has to be send and no data has to be written in TBDR register, this DATSZ can be programmed as 0. In case the DATSZ is programmed as 0 in IPCR, SFP module blocks the write access to TBDR register and if any write is done then that generates IPS transfer error.

- Exclusive access lock – Exclusive master access lock can be enabled over any flash address defined under a FRAD. The master ID of the master which enables/owns the exclusive access is captured in FRAD*n*_WORD2[EALO] fields. Once enabled, the exclusive lock can be released only by the same master which enabled the lock by writing on FRAD*n*_WORD3[EAL] fields. The exclusive write access permissions of a flash region are decided as given in table below:

Table 761. Exclusive access lock

FRAD*n*_WORD3[EAL]	Write permissions
00	No lock. The write permissions are decided as set in FRAD*n*_WORD2[MDnACP] fields for respective master domain.
10	Write permissions are revoked for all masters. Any write transaction coming to this flash address region will not be forwarded through and will result in an error.
11	Write permissions are revoked for all masters except the master ID matching the ID specified by FRAD*n*_WORD2[EALO] fields.

- Flash region privilege checks – If the FRAD*n*_WORD3[EAL] fields are set to 00 then flash write access permissions for any IPS master is decided based on the settings done in FRAD*n*_WORD2[MDnACP] fields as given in table below. MD0ACP permission checks are done for transaction coming through target group 0 queue (TG0) and MD1ACP permission checks are for transactions coming through target group 1 queue (TG1).

Table 762. Flash region privilege checks

MDxACP value	Privilege access	Secure access	Write access
111	No	No	Not allowed
	No	Yes	Not allowed
	Yes	No	Allowed
	Yes	Yes	Allowed
110	No	No	Allowed
	No	Yes	Allowed

Table continues on the next page...

Table 762. Flash region privilege checks (continued)

MDxACP value	Privilege access	Secure access	Write access
	Yes	No	Allowed
	Yes	Yes	Allowed
101	No	No	Not allowed
	No	Yes	Not allowed
	Yes	No	Not allowed
	Yes	Yes	Allowed
100	No	No	Not allowed
	No	Yes	Allowed
	Yes	No	Not allowed
	Yes	Yes	Allowed
0xx	-	-	Not allowed

If any transaction fails FRAD check then error interrupt is generated if enabled.

FRAD status registers keep the details of last transaction which lie within the address range of that respective FRAD and will be updated only when a new transaction comes which falls within the same address range.

Flash Region Compare Address Status (FRAD0_WORD4 - FRAD7_WORD4) register stores the flash start address that was compared and Flash Region Compare Status Data (FRAD0_WORD5 - FRAD7_WORD5) contains the master ID, privilege and secure attributes of the transaction. The transaction details are stored in FRAD register for which the address compare check passed. These status registers are not updated in case of read transfers or when FRAD check is disabled. For example, if the transaction address lies in the range set in FRAD2_WORD0 and FRAD2_WORD1 then the transaction details will be stored in FRAD2_WORD4 and FRAD2_WORD5 registers and valid field [30] will be set. If transaction fails, the security or privilege attribute check for the corresponding FRAD then error field [29] is set in FRADx_WORD5 status register. This bit and other transaction details will remain latched unless cleared by writing '1' to W1C bit of corresponding FRAD in ERRSTAT register.

If transaction address doesn't match any of the FRAD address range or if matches with multiple FRAD address ranges, then also error interrupt is generated. But for no FRAD match, the details will not be stored in any of the FRAD status registers.

FRAD checks are done only for flash write transactions, flash read transactions are passed through to QuadSPI if MDAD checks are passed. At least one FRAD region should be made valid (if the FRAD check is enabled from MGC register) for flash read instructions to go through even when FRAD check is not done for read instructions. SFP module checks if a transaction is read or write from the SEQID written in IPCR register and LUT table programmed inside QuadSPI. Master Read Command (MRC) register shows the read command code which SFP block identifies. There are two command codes which are predefined READ_DDR and READ commands. Software can program two other command code as per its convenience by writing into fields [21:16] or [29:24] and then writing respective valid bit as 1.

79.12.2.2 Error conditions

Any transaction passing the MDAD checks will be checked for FRAD checks and can lead to following error scenarios.

Table 763. Error conditions

Error condition	Response
None of the FRAD are programmed	IPS transfer error will be generated and status will be latched in IPS_ERROR register

Table continues on the next page...

Table 763. Error conditions (continued)

Error condition	Response
Transaction address doesn’t fall within any of the FRAD address range	Error interrupt will be generated if enabled
Transaction match with multiple FRAD address range	Error interrupt will be generated if enabled. Error bit will be set, and status will be stored in FRADx_WORD5 status registers for whom the address check passed
Transaction passes FRAD address check but fails privilege or security checks	Error interrupt will be generated if enabled. Error bit will be set, and status will be stored in FRADx_WORD5 status registers for whom the address check passed
Transaction passes FRAD address check but range was locked by EAL bits	Error interrupt will be generated if enabled. Error bit will be set, and status will be stored in FRADx_WORD5 status registers for whom the address check passed
Read transaction	No FRAD check will be done and no error will be generated for FRAD failure. But if FRAD check is enabled in MGC then at least one FRAD should be valid else it will give error during SFAR/IPCR write

79.12.2.3 Atomic commands considerations

For security reasons, the atomic commands (where flash's internal configuration region is accessed) like flash erase etc. should always be programmed in last 6 SEQID locations of LUT. FRAD0 has access to all the LUT locations including these last 6 SEQID. Remaining FRAD regions will not allow write access if the IPCR SEQID is set for any of the last 6 locations of LUT. It will result in FRAD access error and it will set in FRAD status registers. Also the software must set the FRAD0 MDACP to 101 so that write access is allowed only for secure privilege masters. This is done to ensure that no other master can use the atomic commands like flash erase to change the contents of flash. So when any master wants to use these atomic command programmed in last 6 locations of LUT it will have to write FRAD0 start address in SFAR and keep IPCR DATSZ such that it falls under FRAD0 address region and MDACP programmed in FRAD0 are checked.

For flash configuration in case of dual-die the software needs to change the address range of FRAD0 accordingly so that address goes to second die. Similar consideration to be taken in case of dual flash where Port A and B are being used.

79.12.3 TBDR register write lock

SFP module keeps the TBDR write locked for IPS transactions by default. When a IPS master writes SFAR and IPCR transactions and this transaction passes MDAD and FRAD checks then the accesses are unlocked for that master ID. Lock and unlock procedure is shown with an example in figure below:

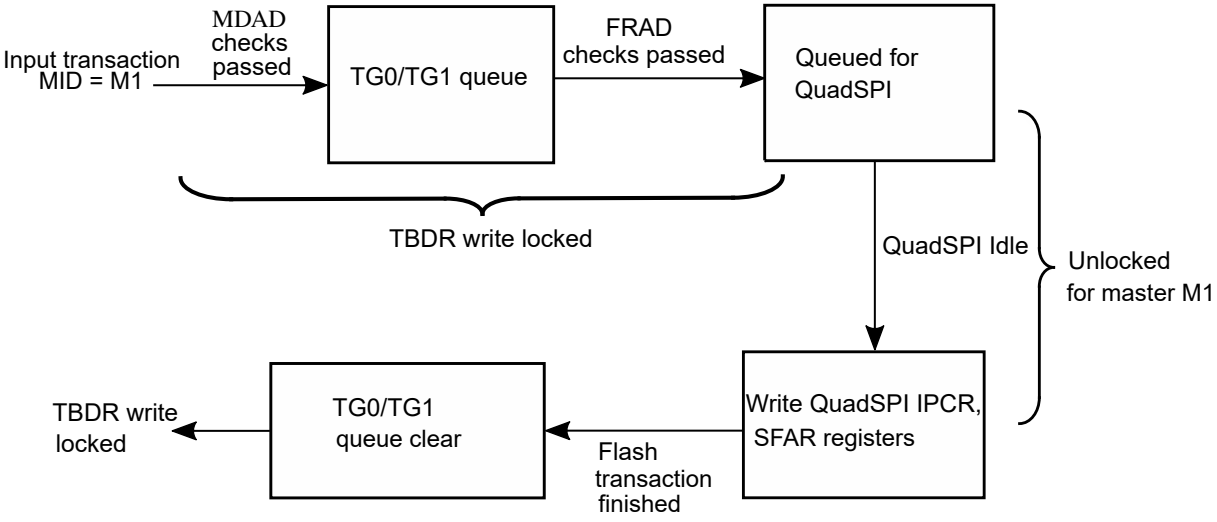


Figure 518. TBDR lock and unlock procedure

If any master writes on TBDR register when it is not unlocked for that master ID it will result in IPS transfer error. IPS master should ensure that it writes the TBDR register once QuadSPI registers have been written by the SFP module. This can be checked by reading FSM Status (FSMSTAT). Bits [11:8] of this register tells which master ID is currently active, and if the value of bits [1:0] is 01 or 10 and bit 31 is set as 1, then the TBDR lock is open for write transactions. It is advisable that for write transactions where IDATSZ is non zero, master clears the TBDR buffer from any residue data and program the TBCT[WMRK] watermark once its request is granted in arbitration (FSMSTAT bit [1:0] are 01). In case of transfers having IDATSZ programmed as 0 the TBDR register remains locked and data can't be written to this register. In case DMA is being used to fill the TBDR buffer, program RSER[TBFDE] field to 1 only when write access is granted and TBDR access is unlocked after checking FSMSTAT[STATE] field. Once DMA completes the TBDR write reset, this field is set back to 0. After this, the SFP will write SFAR and IPCR to QuadSPI triggering the flash transfer. It should be made sure that the DMA should have same master ID as the master which won the arbitration.

79.12.4 Transaction status registers

SFP module has two registers (FSM Status (FSMSTAT) and FlashSeq Request (FLSEQREQ)) which shows the current and last transaction status. These registers are updated only after MDAD and FRAD check are passed.

- FSM Status (FSMSTAT) register shows the current transaction which is queued for QuadSPI when valid field [31] is set. Field [16] of this register if set to 1 then it is a non-read instruction sequence and it is set to 0 then it means that this is a read sequence. Fields [11:8] show the master ID from which this transaction is generated. Fields [1:0] show the current state of active transaction when valid field [31] is set.

  The figure below shows the IPS transaction flow inside SFP block along with various values of bits [1:0]:

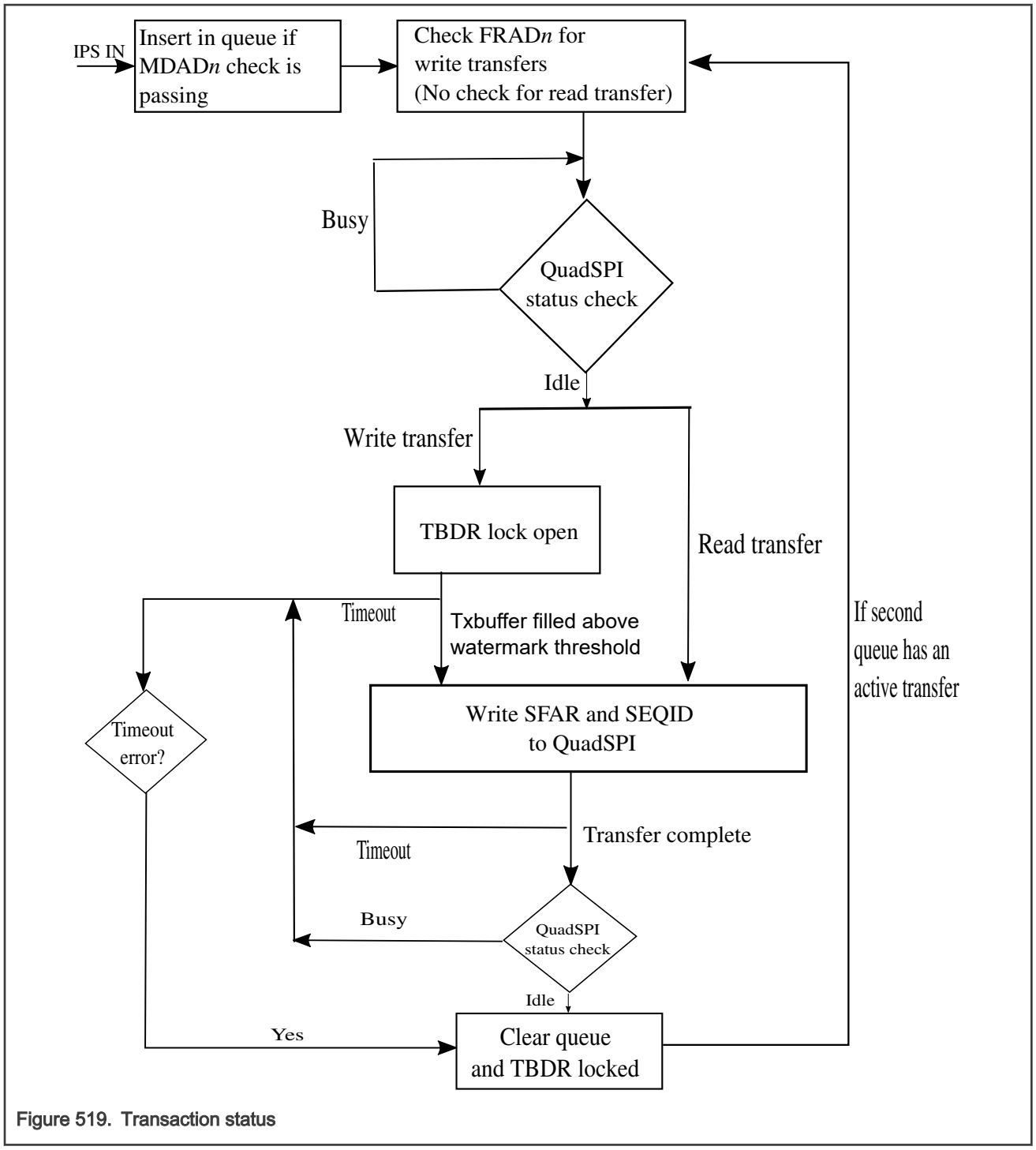


Figure 519. Transaction status

NOTE

If the DATSZ programmed in TG0/TG1 IPCR register is more than the TX buffer watermark threshold value (256-Watermark programmed in TBCT) then it waits for threshold to cross before writing the SFAR and IPCR registers to QuadSPI. Else SFP block waits for one entry to be written to TBDR register and after that it writes the SFAR and IPCR to QuadSPI thus triggering the transfer. In case of read transfers, SFAR and IPCR are written as soon as QSPI is IDLE.

NOTE

QuadSPI is considered IDLE if transaction is finished at flash, No DMA and interrupt pending and RX data is completely read. In case RX data is pending it can be cleared by writing to MCR[CLR_RXF]. Refer to section Timeout error for more details.

FSM state values in FSM Status (FSMSTAT) for different states are in the following table:

Table 764. FSMSTAT register field details

FSMSTAT register fields	
VLD	0 - No IPS transfer is queued for launch to QuadSPI 1 - IPS transfer is queued or running in QuadSPI
STAT	00 - QuadSPI is busy with AHB transfer, any DMA transfer is pending, RDBFL has residue data or any interrupt is pending. 01 - TBDR lock is open. QuadSPI considers IPS transfer. AHB transfer should not start. 10 - TX buffer filled above threshold. Write transfer is triggered. SEQID is written. 11 - Read transfer is triggered. SEQID is written.

For flash write transfers which passes MDAD and FRAD checks and QuadSPI is not busy with any previous transfer, TBDR lock is opened and bits [1:0] shows value of 01. Once the IPS master fills the TX buffer till the level where free entries in TX buffer are less than the watermark threshold set in TX Buffer Control Register (TBCT) and SR[TXWA] flag is de-asserted, SFP block writes the SFAR and IPCR entries to QuadSPI registers and transaction is triggered. To avoid underrun, program the watermark accordingly for size of the data to be transferred. This programming can be done when TBDR access is allowed after arbitration is granted. TBCT Watermark can be programmed as 256-(IDATSZ/4-1) in most of the scenarios. If the DATSZ entry written in IPCR is less such that TX buffer can never be filled till watermark threshold, in such cases SFP block does not wait for the TBCT[WMRK] to trigger and writes the SFAR and ICPR registers as soon as one data entry is written to TBDR register. When the SFAR and IPCR register are written, bits [1:0] are set to 10. In case of flash memory read transfers SFAR and IPCR entries are written to QuadSPI as soon as QuadSPI is IDLE and bits [1:0] are set to 11. QuadSPI is considered IDLE only if no AHB/IPS transfer is ongoing, any DMA transfer is not pending, RDBFL doesn't have residue data and any interrupt is not pending (FR[TFF], FR[TBFF], FR[TBUF], FR[RBDF], FR[RBOF], FR[ABOF], FR[AIBSEF], FR[AITEF], , , FR[IPIEF], , FR[ILLINE], ,). Valid bit [31] goes to 0 once the IPS transfer is completed. IPS master should clear the TX buffer before writing to TBDR register to ensure that no data from previous transaction is left in TX buffer.

If the IPS transaction has passed the MDAD and FRAD checks but QuadSPI is busy with some AHB transfer, then the bits [1:0] are set to 00. Other than ongoing AHB transfer QuadSPI can be busy because of other reasons also like if any TXDMA or RXDMA pending, if read buffer has data pending or if any QuadSPI interrupt is enabled and pending for servicing. So IPS master should ensure that no interrupt is pending or previous read transfer is completed. If the QuadSPI remains busy for a longer time it may result in timeout error.

Software must take care that no AHB transaction is launched when any IPS transaction is ongoing, TBDR lock is open or IPS transaction is getting executed on QuadSPI. It can be checked by reading this FSM status register. if the valid bit is 1 and bits [1:0] are set to 01, 10 or 11 then no AHB transfer should be generated towards QuadSPI.

For any IPS initiated RW transaction which needs two separate transactions at flash interface, like Write-Enable transaction followed by Write-data transaction (Data-Learn flash transaction followed by Read-data transaction) at flash interface must be joined together with JMP_2_SEQ instruction.

- FlashSeq Request (FLSEQREQ) register holds the details of transaction which was last send to QuadSPI. After a flash transaction is completed this register is updated and its valid field [31] is set to 1. This register holds the command type (read or write), transaction master ID (fields [3:0]), target group queue (TG0 or TG1) from which this transaction passed in field [4], privilege and secure attributes for the transaction, SEQID (fields [19:16]) and number of flash descriptor FRAD within whose address range the transaction lies (fields [14:12]).

This register also contains a timeout error field [27] which is set when a timeout error occurs, and transaction is aborted. When the error occurs this register keeps the status latched unless it is cleared by writing 1 to clear field [29].

79.12.5 Timeout error

MTO register can be programmed with a timeout value above which the transaction should be aborted. When the transaction passes MDAD and FRAD level checks and QuadSPI is idle (no AHB transaction is ongoing, no DMA transfer is pending, RDBFL doesn't have any residue data from last transfer and (FR[TFF], FR[TBFF], FR[TBUF], FR[RBDF], FR[RBOF], FR[ABOF], FR[AIBSEF], FR[AITEF], , , FR[IPIEF], , FR[ILLINE], ,) interrupt is not pending) this counter is started. If the transaction is not completed before this timeout counter reaches the value programmed in MTO register the transaction is considered aborted by SFP and an error interrupt is generated if enabled. The error bit is also set in FLSEQREQ register along with the transaction details. After the timeout error, next transaction is written to QuadSPI register which was queued in another target group queue once SR[BUSY] flag goes low.

79.12.6 Arbitration lock

The arbitration lock feature is used to lock the arbitration for a particular queue. This lock can be requested by a master by writing '1' into IPCR[ARB_LOCK] field. Once the transaction passes both the checks (MDAD & FRAD) and wins the arbitration, this lock will be granted. After this, the transaction from the other queue will not be granted unless the arbitration is unlocked by previous queue. The master which locked the arbitration can unlock it by writing '1' to IPCR[ARB_UNLOCK] field. Once the transaction with unlock bit 1 passes the MDAD and FRAD checks, it will unlock the arbitration. After this transaction is finished, the other queue will be granted the arbitration which was waiting.

This arbitration lock feature can be used by software while doing any transaction to flash which may result in a waiting period. For example, in case of NOR flash, after a write is issued to flash there is a waiting period (TPP) for which the flash is inaccessible. In this case, the software should ensure that while writing this SEQID to IPCR register it should lock the arbitration so that transaction coming to other queue is not passed to flash controller which may result in failure. Once the software ensures that waiting period at flash is over, it can read the flash configuration/status registers to confirm. And after getting confirmation, it can send another read command to same configuration register with IPCR[ARB_UNLOCK] field set to '1'. This will ensure that the transaction waiting in another queue are granted. If the flash being used does not have a waiting period attached with transactions, then this ARB_LOCK field can be always kept to 0.

In case software uses this LOCK functionality RWW support will not be available. Because in this case, all the transaction from other queue which was blocked from arbitration will be blocked irrespective of read or write access. So, RWW support (read can go through even when flash is in waiting state) will not be available. The master which has arbitration lock granted can still send read accesses to flash during the waiting period in case the flash supports it.

79.12.7 Soft reset consideration with SFP

If there is any pending transaction in MDAD queue, do not assert QuadSPI reset bits unless you observe unexpected behavior like no response from the module. In that case, assert QSPI reset twice.

In case the soft reset is being used for QSPI, software should take some considerations so that correct functionality is maintained. if the soft reset is asserted when a transaction is being executed by SFP (FSM_STATE [1:0] bits are 01, 10 or 11) then this transaction will be aborted. Different conditions that can occur in case of soft reset are listed in table below:

Table 765. Soft reset conditions

Condition at time of soft reset	FSM state register at time of soft reset	Result	Timeout scenario
Some AHB transaction was ongoing and SFP/IPS is in waiting state.	FSM_STATE[1:0] = 00	After the soft reset if de-asserted the SFP will schedule its transaction over QSPI. FSM_STATE[1:0] will change to other values than 00. Software can fill TBDR in case it is a write transfer or start reading from RBDR.	In this case software doesn’t writes to TBDR or read from RBDR it will result in timeout error.

Table continues on the next page...

Table 765. Soft reset conditions (continued)

Condition at time of soft reset	FSM state register at time of soft reset	Result	Timeout scenario
QSPI was IDLE and SFP transaction has been arbitrated. TBDR is being filled and SFAR, IPCR are not currently written to QSPI.	FSM_STATE[1:0] = 01	Currently TBDR was being filled after soft reset all the entries in TBDR are flushed. So it is expected from software that it will start filling TBDR from start after the reset has de-asserted. SFP will write SFAR and IPCR to QSPI once its normal conditions are met (when number of available spaces in the TX buffer is greater than or equal to the value provided by TBCT[WMRK] or single TBDR write in case watermark check is disabled)	Timeout will occur if TBDR is not filled again after the soft reset is lifted.
SFP has written just SFAR to QSPI IPCR is not written yet. (For a read transfer or write transfer with data size 0)	FSM_STATE[1:0] = 01	After the soft reset is lifted the SFP will write ICPR to QSPI module and FSM_STATE will change to 11 (read) or 10(write). Transaction will finish normally.	Timeout conditions will be same as that for normal transfer.
SFP has written just SFAR to QSPI IPCR is not written yet. (Write data size is non 0)	FSM_STATE[1:0] = 01	After soft reset is lifted the SFP will write IPCR to QSPI module once software fill TBDR (from start) and FSM_STATE will change to 10.	Timeout conditions will be same as that for normal transfer.
SFP has written SFAR and IPCR to QSPI.	FSM_STATE[1:0] = 10 or 11	After the soft reset is de-asserted the current transaction will be deemed aborted. FSM_STATE valid will go low as transfer has finished.	No timeout.

So to conclude in case the soft reset is asserted when FSM_STATE[31] bit is 1 and state bits are 10 or 11 current transaction will be aborted & the next transaction will proceed as normal. In case the state bits are 00 or 11 then it is advisable that software doesn't writes into TBDR after the reset is de-asserted and waits for timeout error to be generated to know when current ongoing SFP transaction is aborted. After clearing the timeout error a new transaction can be launched to SFP or if already a new transaction is present in another queue it will be arbitrated.

79.12.8 Error Interrupt Enable

INT_EN register allow software to disable or enable the interrupt signal generation in case of any of the listed error conditions. This interrupt is generated on 'ipi_int_ored' interface signal of QuadSPI.

- Time out error (INT_EN[TO_ERR])
- IPCR register write error (INT_EN[TG*n*IPCR])
- SFAR register write error (INT_EN[TG*n*SFAR])
- IPS common error (INT_EN[IPS_ERR])
- FRAD access error (INT_EN[FRAD*n*ACC])
- No FRAD address range match error (INT_EN[FRADMTCH])

If the respective field is set to 1 for any of the error conditions, then the interrupt signal is set to 1 when that error occurs and it remains set until the error is cleared by writing 1 to error clear bit as listed in register descriptions.

79.13 Memory map and register definition

This section provides the memory map and register definitions for the QuadSPI module.

79.13.1 Register write access

Following are the write access restriction terms that apply to all the registers:

- Register write access restriction

  For each register field, the write access conditions are specified in the detailed register description.

  The following table provides a description of the write access conditions. If, for a specific register bit or field, none of the given write access conditions is fulfilled, any write attempt to this register bit or field is ignored without any notification. The values of the bits or fields are not changed.

  The condition term [A or B] indicates that the register or field can be written to if at least one of the conditions is fulfilled.

Table 766. Register write access restrictions

Condition	Description
Anytime	No write access restriction
Disabled mode	Write access only if MCR[MDIS] = 1
Normal mode	Write access only if the module is in the normal mode

- Register write access requirements

  You can access all registers using 8-bit, 16-bit, and 32-bit wide operations. For some of the registers, at least a 16-bit or 32-bit wide write access is required to ensure correct operation. This write access requirement is stated in the detailed register description for each affected register.

79.13.2 QuadSPI register descriptions

This section provides the memory map and register definitions for the QuadSPI module.

Access to the following addresses does not result in a transfer error:

- 64h
- 138h
- 168h
- 188h
- 18Ch
- 198h

79.13.2.1 QuadSPI memory map

QuadSPI_S32K358 base address: 404C_C000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Module Configuration Register (MCR)	32	RW	000F_404Ch
8h	IP Configuration Register (IPCR)	32	RW	0000_0000h
Ch	Flash Memory Configuration Register (FLSHCR)	32	RW	0000_0303h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
10h	Buffer 0 Configuration Register (BUF0CR)	32	RW	0000_0003h
14h	Buffer 1 Configuration Register (BUF1CR)	32	RW	0000_0002h
18h	Buffer 2 Configuration Register (BUF2CR)	32	RW	0000_0001h
1Ch	Buffer 3 Configuration Register (BUF3CR)	32	RW	8000_0000h
20h	Buffer Generic Configuration Register (BFGENCR)	32	RW	0000_0000h
24h	SOC Configuration Register (SOCCR)	32	RW	0000_0000h
30h	Buffer 0 Top Index Register (BUF0IND)	32	RW	0000_0000h
34h	Buffer 1 Top Index Register (BUF1IND)	32	RW	0000_0000h
38h	Buffer 2 Top Index Register (BUF2IND)	32	RW	0000_0000h
50h	AHB Write Configuration Register (AWRCR)	32	RW	0000_0000h
60h	DLL Flash Memory A Configuration Register (DLLCRA)	32	RW	0120_0000h
6Ch	Parity Configuration Register (PARITYCR)	32	RW	0000_0000h
100h	Serial Flash Memory Address Register (SFAR)	32	RW	0000_0000h
104h	Serial Flash Memory Address Configuration Register (SFACR)	32	RW	0000_0800h
108h	Sampling Register (SMPR)	32	RW	FF00_0000h
10Ch	RX Buffer Status Register (RBSR)	32	R	0000_0000h
110h	RX Buffer Control Register (RBCT)	32	RW	0000_0000h
120h	AHB Write Status Register (AWRSR)	32	R	0000_0000h
12Ch	DLL Status Register (DLLSR)	32	R	8000_8000h
130h	Data Learning Configuration Register (DLCR)	32	RW	40FF_40FFh
134h	Data Learning Status Flash Memory A Register (DLSR_FA)	32	R	0000_0000h
150h	TX Buffer Status Register (TBSR)	32	R	0000_0000h
154h	TX Buffer Data Register (TBDR)	32	RW	0000_0000h
158h	TX Buffer Control Register (TBCT)	32	RW	0000_0000h
15Ch	Status Register (SR)	32	R	0200_3800h
160h	Flag Register (FR)	32	RW	0800_0000h
164h	Interrupt and DMA Request Select and Enable Register (RSER)	32	RW	0000_0000h
16Ch	Sequence Pointer Clear Register (SPTRCLR)	32	RW	0100_0000h
180h	Serial Flash Memory A1 Top Address Register (SFA1AD)	32	RW	7000_0000h
184h	Serial Flash Memory A2 Top Address Register (SFA2AD)	32	RW	7000_0000h
190h	Data Learn Pattern Register (DLPR)	32	RW	AA55_3443h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
194h	Flash Memory A Failing Address Status Register (FAILA_ADDR)	32	R	FFFF_FFFFh
200h - 27Ch	RX Buffer Data Register (RBDR0 - RBDR31)	32	R	0000_0000h
300h	LUT Key Register (LUTKEY)	32	RW	5AF0_5AF0h
304h	LUT Lock Configuration Register (LCKCR)	32	RW	0000_0002h
310h	LUT Register (LUT0)	32	RW	0818_0403h
314h	LUT Register (LUT1)	32	RW	2400_1C08h
318h - 44Ch	LUT Register (LUT2 - LUT79)	32	RW	0000_0000h
800h	Flash Region Start Address (FRAD0_WORD0)	32	RW	0000_0000h
804h	Flash Region End Address (FRAD0_WORD1)	32	RW	0000_FFFFh
808h	Flash Region Privileges (FRAD0_WORD2)	32	RW	0000_0000h
80Ch	Flash Region Lock Control (FRAD0_WORD3)	32	RW	0000_0000h
810h	Flash Region Compare Address Status (FRAD0_WORD4)	32	R	0000_0000h
814h	Flash Region Compare Status Data (FRAD0_WORD5)	32	R	0000_0000h
820h	Flash Region Start Address (FRAD1_WORD0)	32	RW	0000_0000h
824h	Flash Region End Address (FRAD1_WORD1)	32	RW	0000_FFFFh
828h	Flash Region Privileges (FRAD1_WORD2)	32	RW	0000_0000h
82Ch	Flash Region Lock Control (FRAD1_WORD3)	32	RW	0000_0000h
830h	Flash Region Compare Address Status (FRAD1_WORD4)	32	R	0000_0000h
834h	Flash Region Compare Status Data (FRAD1_WORD5)	32	R	0000_0000h
840h	Flash Region Start Address (FRAD2_WORD0)	32	RW	0000_0000h
844h	Flash Region End Address (FRAD2_WORD1)	32	RW	0000_FFFFh
848h	Flash Region Privileges (FRAD2_WORD2)	32	RW	0000_0000h
84Ch	Flash Region Lock Control (FRAD2_WORD3)	32	RW	0000_0000h
850h	Flash Region Compare Address Status (FRAD2_WORD4)	32	R	0000_0000h
854h	Flash Region Compare Status Data (FRAD2_WORD5)	32	R	0000_0000h
860h	Flash Region Start Address (FRAD3_WORD0)	32	RW	0000_0000h
864h	Flash Region End Address (FRAD3_WORD1)	32	RW	0000_FFFFh
868h	Flash Region Privileges (FRAD3_WORD2)	32	RW	0000_0000h
86Ch	Flash Region Lock Control (FRAD3_WORD3)	32	RW	0000_0000h
870h	Flash Region Compare Address Status (FRAD3_WORD4)	32	R	0000_0000h
874h	Flash Region Compare Status Data (FRAD3_WORD5)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
880h	Flash Region Start Address (FRAD4_WORD0)	32	RW	0000_0000h
884h	Flash Region End Address (FRAD4_WORD1)	32	RW	0000_FFFFh
888h	Flash Region Privileges (FRAD4_WORD2)	32	RW	0000_0000h
88Ch	Flash Region Lock Control (FRAD4_WORD3)	32	RW	0000_0000h
890h	Flash Region Compare Address Status (FRAD4_WORD4)	32	R	0000_0000h
894h	Flash Region Compare Status Data (FRAD4_WORD5)	32	R	0000_0000h
8A0h	Flash Region Start Address (FRAD5_WORD0)	32	RW	0000_0000h
8A4h	Flash Region End Address (FRAD5_WORD1)	32	RW	0000_FFFFh
8A8h	Flash Region Privileges (FRAD5_WORD2)	32	RW	0000_0000h
8ACh	Flash Region Lock Control (FRAD5_WORD3)	32	RW	0000_0000h
8B0h	Flash Region Compare Address Status (FRAD5_WORD4)	32	R	0000_0000h
8B4h	Flash Region Compare Status Data (FRAD5_WORD5)	32	R	0000_0000h
8C0h	Flash Region Start Address (FRAD6_WORD0)	32	RW	0000_0000h
8C4h	Flash Region End Address (FRAD6_WORD1)	32	RW	0000_FFFFh
8C8h	Flash Region Privileges (FRAD6_WORD2)	32	RW	0000_0000h
8CCh	Flash Region Lock Control (FRAD6_WORD3)	32	RW	0000_0000h
8D0h	Flash Region Compare Address Status (FRAD6_WORD4)	32	R	0000_0000h
8D4h	Flash Region Compare Status Data (FRAD6_WORD5)	32	R	0000_0000h
8E0h	Flash Region Start Address (FRAD7_WORD0)	32	RW	0000_0000h
8E4h	Flash Region End Address (FRAD7_WORD1)	32	RW	0000_FFFFh
8E8h	Flash Region Privileges (FRAD7_WORD2)	32	RW	0000_0000h
8ECh	Flash Region Lock Control (FRAD7_WORD3)	32	RW	0000_0000h
8F0h	Flash Region Compare Address Status (FRAD7_WORD4)	32	R	0000_0000h
8F4h	Flash Region Compare Status Data (FRAD7_WORD5)	32	R	0000_0000h
900h	Target Group n Master Domain Access Descriptor (TG0MDAD)	32	RW	0000_0000h
904h	Target Group n SFAR Address (TG0SFAR)	32	R	0000_0000h
908h	Target Group n SFAR Status (TG0SFARS)	32	RW	0000_0000h
90Ch	Target Group n IPCR Status (TG0IPCRS)	32	RW	0000_0000h
910h	Target Group n Master Domain Access Descriptor (TG1MDAD)	32	RW	0000_0000h
914h	Target Group n SFAR Address (TG1SFAR)	32	R	0000_0000h
918h	Target Group n SFAR Status (TG1SFARS)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
91Ch	Target Group n IPCR Status (TG1IPCRS)	32	RW	0000_0000h
920h	Master Global Configuration (MGC)	32	RW	A800_0000h
924h	Master Read Command (MRC)	32	RW	0050_0E07h
928h	Master Timeout (MTO)	32	RW	FFFF_FFFFh
92Ch	FlashSeq Request (FLSEQREQ)	32	RW	0000_0000h
930h	FSM Status (FSMSTAT)	32	R	0000_0000h
934h	IPS Error (IPSERROR)	32	RW	0000_0000h
938h	Error Status (ERRSTAT)	32	RW	0000_0000h
93Ch	Interrupt Enable (INT_EN)	32	RW	0000_0000h

79.13.2.2 Module Configuration Register (MCR)

Offset

Register	Offset
MCR	0h

Function

This register holds configuration data associated with the QuadSPI operation.

NOTE

When out of reset, after first initial MCR programming and exiting module disable mode (that is, when the value of MCR[MDIS] is 0), you must write 1 to MCR[SWRSTSD] and release it after six (three system and three flash memory) clock cycles.

Special write-access is permitted in different modes:

- DQS_FA_SEL: Disabled mode
- ISD3FA, ISD2FA: Disabled mode
- All other fields: Anytime

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved	Reserved	Reserved		Reserved	Reserved	DQS_FA_SEL		Reserved	Reserved	0		Reserved	Reserved	ISD3FA	ISD2FA
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	MDIS	Reserved	DLPEN	0	0	0	VAR_LAT...	DDR_EN	DQS_EN	Reserved	DQS_OUT...	Reserved		SWRSTHD	SWRSTSD
W					CLR_TXF	CLR_RXF										
Reset	0	1	0	0	0	0	0	0	0	1	0	0	1	1	0	0

Fields

Field	Function
31 —	Reserved
30 —	Reserved
29-28 —	Reserved
27 —	Reserved
26 —	Reserved
25-24 DQS_FA_SEL	DQS clock for sampling read data at flash memory A Selects DQS clock for sampling read data at flash memory A QuadSPI port 00b - Reserved 01b - Pad loopback 10b - Reserved 11b - External DQS **NOTE** In case of an padloopback selection to access port A, port B cannot be programmed for external DQS and vice-versa.
23 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
22 —	Reserved
21-20 —	Reserved
19 —	Reserved
18 —	Reserved
17 ISD3FA	Idle signal drive IOFA[3] flash memory A Determines the logic level that the IOFA[3] output of the QuadSPI module is driven to in the inactive state. See Driving flash memory control signals in single and dual modes for details. 0b - IOFA[3] is driven to logic L 1b - IOFA[3] is driven to logic H
16 ISD2FA	Idle signal drive IOFA[2] flash memory A Determines the logic level that the IOFA[2] output of the QuadSPI module is driven to in the inactive state. See Driving flash memory control signals in single and dual modes for details. 0b - IOFA[2] is driven to logic L. 1b - IOFA[2] is driven to logic H.
15 —	Reserved
14 MDIS	Module disable Allows the clock to the non-memory mapped logic in the QuadSPI to be stopped. 0b - Enable QuadSPI clocks 1b - Allow external logic to disable QuadSPI clocks
13 —	Reserved
12 DLPEN	Data learning pattern enable Write 1 to this field to enable data learning mechanism.
11 CLR_TXF	Clear TX FIFO/buffer This is a self-clearing field that invalidates the TX buffer content.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE Software must wait for at least five system cycles and three flash cycles after writing '1' to this field. 0b - No action 1b - Read and write pointers of the TX buffer are reset to 0 and TBSR[TRCTR] is reset to 0.
10 CLR_RXF	Clear RX FIFO This is a self-clearing field that invalidates the RX buffer content. 0b - No action 1b - Read and write pointers of the RX buffer are reset to 0 and RBSR[RDBFL] is reset to 0.
9 —	Reserved
8 VAR_LAT_EN	Variable latency Is used to enable the variable latency feature in the controller. This field is valid for HyperRAM where data strobe acts as an output from the memory during the command and address (CA) cycles of a read or write transaction. This is to indicate whether additional initial access latency is needed to perform a dynamic memory refresh operation. For details, see HyperRAM support. 0b - Fixed latency: Twice + 1 latency enable 1b - Variable latency: "Once" or "twice + 1" the initial latency based on data strobe during the CA phase. If enabled, you need to ensure that the value of FLSHCR[TCSS] is >= 2.
7 DDR_EN	DDR mode enable Enables the DDR mode 0b - 2x clock disabled for SDR instructions only 1b - 2x clock enabled for DDR instructions. Note: 2x clock - This is twice the SCKF clock used to shift the TX data by 90 degree.
6 DQS_EN	DQS enable Is valid for both the SDR and DDR modes. For details, see DQS sampling method. 0b - Reserved. Do not program 0 to this field. 1b - DQS enabled. The incoming data is sampled on both the edges of the DQS input when the value of MCR[DDR_EN] is 1; else, on only one edge when MCR[DDR_EN] is 0.
5 —	Reserved
4 DQS_OUT_EN	DQS as an output

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Is valid when data strobe is also used as an output from controller during the write data phase. This is valid for HyperRAM where the data strobe acts as a Read Write Data Strobe (RWDS). For details, see HyperRAM support. 0b - DQS as an output from controller is disabled. 1b - DQS as an output from controller is enabled.
3-2 —	Reserved
1 SWRSTHD	Software reset for AHB domain 0b - De-assert Software reset 1b - AHB domain flops are reset. This field does not reset configuration registers. It is advisable to reset both the serial flash memory domain and AHB domain at the same time. Resetting only one domain might lead to side effects. **NOTE** The software resets need the clock to be running to propagate to the design. The value of MCR[MDIS] should be 0 when the software reset bits are asserted. Also, before they can be deasserted again (by setting MCR[SWRSTHD] to 0), it is recommended to set the value of MCR[MDIS] to 1. After the software resets have been deasserted, the normal operation can be started by setting MCR[MDIS] to 0. **NOTE** Software must wait for at least three system cycles and three flash cycles after changing the value of this field.
0 SWRSTSD	Software reset for serial flash memory domain 0b - De-assert Software reset 1b - Serial flash memory domain flops are reset. This field does not reset configuration registers. It is advisable to reset both the serial flash memory domain and AHB domain at the same time. Resetting only one domain might lead to side effects. **NOTE** The software resets need the clock to be running to propagate to the design. The value of MCR[MDIS] should therefore be 0 when the software reset bits are asserted. Also, before they can be deasserted again (by specifying 0 as the value for MCR[SWRSTSD]), it is recommended to specify 1 as the value for MCR[MDIS]. After the software resets are deasserted, the normal operation can be started by specifying 0 as the value for MCR[MDIS]. **NOTE** Software must wait for at least three system cycles and three flash cycles after changing the value of this field.

79.13.2.3 IP Configuration Register (IPCR)

Offset

Register	Offset
IPCR	8h

Function

This register provides all the configuration required for an IP-initiated command, which can be triggered by writing in the SEQID field of this register. If the SEQID field is written successfully, a new command to the external serial flash memory is initiated per the sequence pointed to by this field. See Normal mode for details on command triggering and command execution.

Special write-access is permitted if:

- SR[IP_ACC]=0

NOTE

Only one of the fields should be written '1' at a time out of ARB_LOCK and ARB_UNLOCK. If both the fields are written '1' at same time then it will invert the lock status. If the arbitration was already locked, it will unlock it and vice-versa.

NOTE

If MDAD and FRAD checks are enabled in MGC register but none of the MDAD and FRAD descriptors are valid, then any write on this register will generate a bus transfer error

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				SEQID				0	0	0					Reserved
W									ARB_UNL...	ARB_LOCK						
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	IDATSZ															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved
27-24 SEQID	Points to a sequence in the LUT This field contains the sequence index of the LUT. See LUT for details.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Each sequence index can accommodate up to 10 instructions (2 instructions per register). In case of read command, a write to this field triggers a transaction on the serial flash memory interface if QSPI SFM is IDLE. In case of write command, a write to this fields unlocks the TBDR access if QSPI SFM is not busy. Refer to section Secure flash protection for more details
23 ARB_UNLOCK	Arbitration Unlock Writing 1 to this bit unlocks the arbitration. Writing 0 will have no effect. The arbitration unlock will be done only if the transaction passes both MDAD and FRAD checks. To lock the arbitration write 1 to ARB_LOCK bit. This bit is always read as 0.
22 ARB_LOCK	Arbitration Lock Writing 1 to this bit locks the arbitration for that specific target queue. Writing 0 will have no effect. The arbitration lock will be granted only if the transaction passes both MDAD and FRAD checks. To unlock the arbitration write 1 to ARB_UNLOCK bit. This bit is always read as 0.
21-17 —	Reserved
16 —	Reserved
15-0 IDATSZ	IP data transfer size This field defines the data transfer size, in bytes, of the IP command.

79.13.2.4 Flash Memory Configuration Register (FLSHCR)

Offset

Register	Offset
FLSHCR	Ch

Function

This register contains the timings that are specific to the flash memory device. The QuadSPI controller must meet these timings for the device to function correctly.

Special write-access is permitted if:

- SR[AHB_ACC] = 0
- SR[IP_ACC] = 0

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0														Reserved	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				TCSH				0				TCSS			
W																
Reset	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1	1

Fields

Field	Function
31-18 —	Reserved
17-16 —	Reserved
15-12 —	Reserved
11-8 TCSH	Serial flash memory CS hold time This hold time is in terms of serial flash memory clock cycles, and it must be greater than or equal to two flash memory clock cycles . Refer the chip datasheet for the exact value.
7-4 —	Reserved
3-0 TCSS	Serial flash memory CS setup time This setup time is in terms of serial flash memory clock cycles, and it must be greater than or equal to two flash memory clock cycles. Refer the chip Datasheet for the exact value.

79.13.2.5 Buffer 0 Configuration Register (BUF0CR)

Offset

Register	Offset
BUF0CR	10h

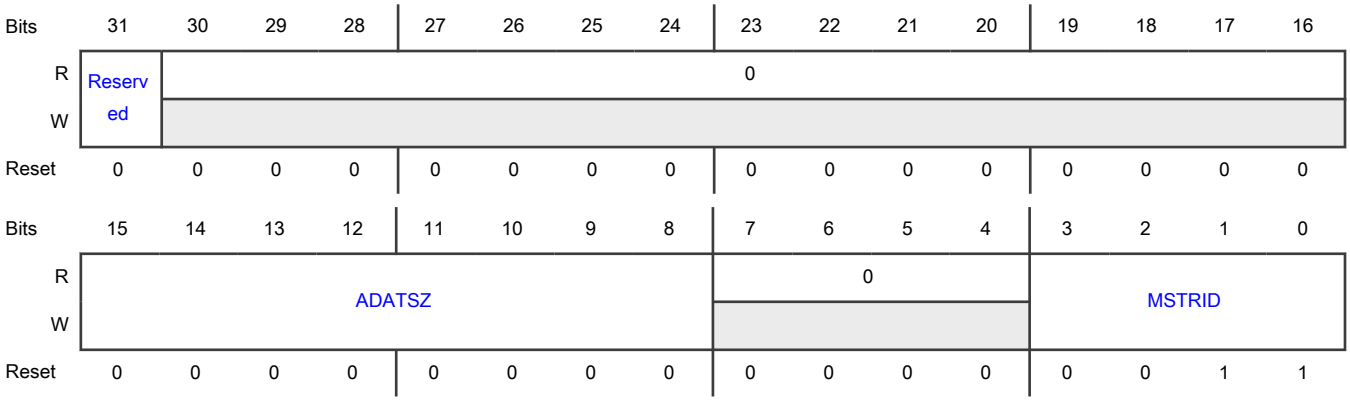
Function

This register provides the configuration for any read access routed to buffer0, which happens when the master ID of the incoming AHB request matches BUF0CR[MSTRID]. Any buffer "miss" leads to a serial flash memory transaction being triggered per the sequence pointed to by BFGENCR[SEQID].

Special write-access is permitted if:

- SR[AHB_ACC] = 0

Diagram



Fields

Field	Function
31 —	Reserved
30-16 —	Reserved
15-8 ADATSZ	AHB data transfer size Defines the read data transfer size in 8 bytes of an AHB triggered read access to serial flash memory. For example, a value of 0x2 sets transfer size to 16 bytes. When ADATSZ = 0, the data size mentioned in the sequence pointed to by the SEQID field overrides this value. The software should ensure that this transfer size is not greater than the size of the buffer.
7-4 —	Reserved
3-0 MSTRID	Master ID ID of the AHB master associated with BUFFER 0 Any AHB read access with this master ID is routed to this buffer. You must ensure that the master IDs associated with all buffers are different. **NOTE** See the chip-specific QuadSPI information for details about master IDs and their corresponding components.

79.13.2.6 Buffer 1 Configuration Register (BUF1CR)

Offset

Register	Offset
BUF1CR	14h

Function

This register provides the configuration for any access routed to buffer 1, which happens when the master ID of the incoming AHB request matches the MSTRID field of this register. Any buffer "miss" leads to the buffer being flushed and a serial flash memory transaction being triggered per the sequence pointed to by BFGENCR[SEQID].

Special write-access is permitted if:

- SR[AHB_ACC] = 0

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ADATSZ								0				MSTRID			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Fields

Field	Function
31-16 —	Reserved
15-8 ADATSZ	AHB data transfer size This field defines the read data transfer size in 8 bytes of an AHB triggered read access to serial flash memory. For example, a value of 0x2 sets the transfer size to 16 bytes. When ADATSZ = 0, the data size mentioned in the sequence pointed to by the SEQID field overrides this value. Software should ensure that this transfer size is not greater than the size of this buffer.
7-4 —	Reserved
3-0 MSTRID	Master ID ID of the AHB master associated with BUFFER 1

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Any AHB read access with this master ID is routed to this buffer. You must ensure that the master IDs associated with all buffers are different. **NOTE** See the chip-specific QuadSPI information for details about master IDs and their corresponding components.

79.13.2.7 Buffer 2 Configuration Register (BUF2CR)

Offset

Register	Offset
BUF2CR	18h

Function

This register provides the configuration for any access routed to buffer 2, which happens when the master ID of the incoming AHB request matches the MSTRID field of this register. Any buffer "miss" leads to the buffer being flushed and a serial flash memory transaction being triggered per the sequence pointed to by BFGENCR[SEQID].

Special write-access is permitted if:

- SR[AHB_ACC] = 0

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ADATSZ								0				MSTRID			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Fields

Field	Function
31-16 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-8 ADATSZ	AHB data transfer size This field defines the read data transfer size in 8 bytes of an AHB triggered read access to the serial flash memory. For example, a value of 0x2 sets transfer size to 16 bytes. When ADATSZ = 0, the data size mentioned in the sequence pointed to by the SEQID field overrides this value. The software should ensure that this transfer size is not greater than the size of this buffer.
7-4 —	Reserved
3-0 MSTRID	Master ID The ID of the AHB master associated with BUFFER2. Any AHB read access with this master ID is routed to this buffer. It must be ensured that the master IDs associated with all buffers are different. **NOTE** See the chip-specific QuadSPI information for details about master IDs and their corresponding components.

79.13.2.8 Buffer 3 Configuration Register (BUF3CR)

Offset

Register	Offset
BUF3CR	1Ch

Function

This register provides the configuration for any access to buffer 3.

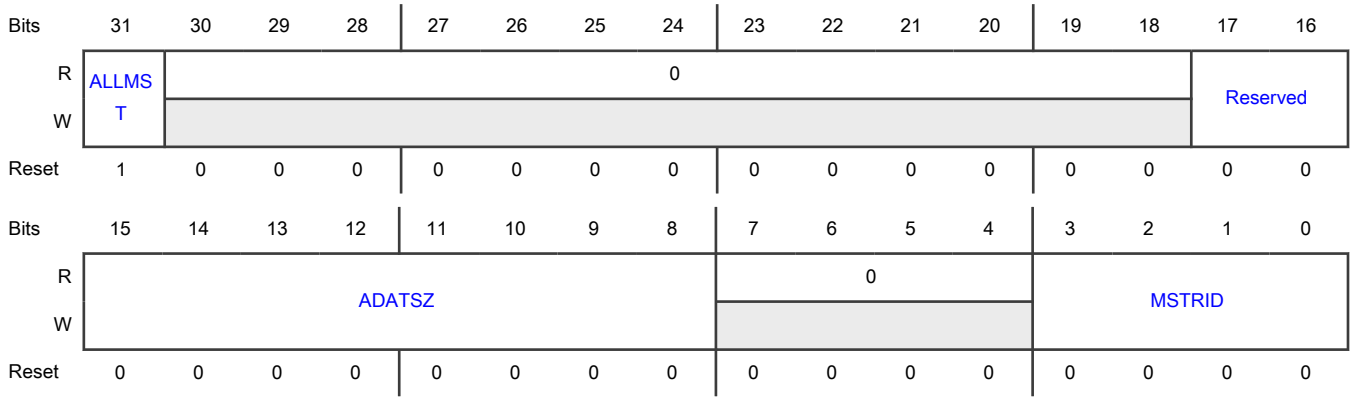
An access is routed to buffer 3 when the master ID of the incoming AHB request matches the MSTRID field of BUF3CR. Any buffer "miss" leads to the buffer being flushed and a serial flash memory transaction being triggered per the sequence pointed to by BFGENCR[SEQID].

In case the value of the ALLMST field is not 1, any such transaction (where master ID does not match any of the MSTRID fields) is returned with an ERROR response.

Special write-access is permitted if:

- SR[AHB_ACC] = 0

Diagram



Fields

Field	Function
31 ALLMST	All master enable When set, buffer3 acts as an all-master buffer. Any AHB access with a master ID not matching with the master ID of buffer0, buffer1, or buffer2 is routed to buffer3. When set, the MSTRID field of this register is ignored.
30-18 —	Reserved
17-16 —	Reserved
15-8 ADATSZ	AHB data transfer size Defines the read data transfer size in 8 bytes of an AHB triggered read access to serial flash memory. When ADATSZ = 0, the data size mentioned in the sequence pointed to by the SEQID field overrides this value.
7-4 —	Reserved
3-0 MSTRID	Master ID ID of the AHB master associated with BUFFER 3. Any AHB read access with this master ID is routed to this buffer. You must ensure that the master IDs associated with all buffers are different. **NOTE** See the chip-specific QuadSPI information for details about master IDs and their corresponding components.

79.13.2.9 Buffer Generic Configuration Register (BFGENCR)

Offset

Register	Offset
BFGENCR	20h

Function

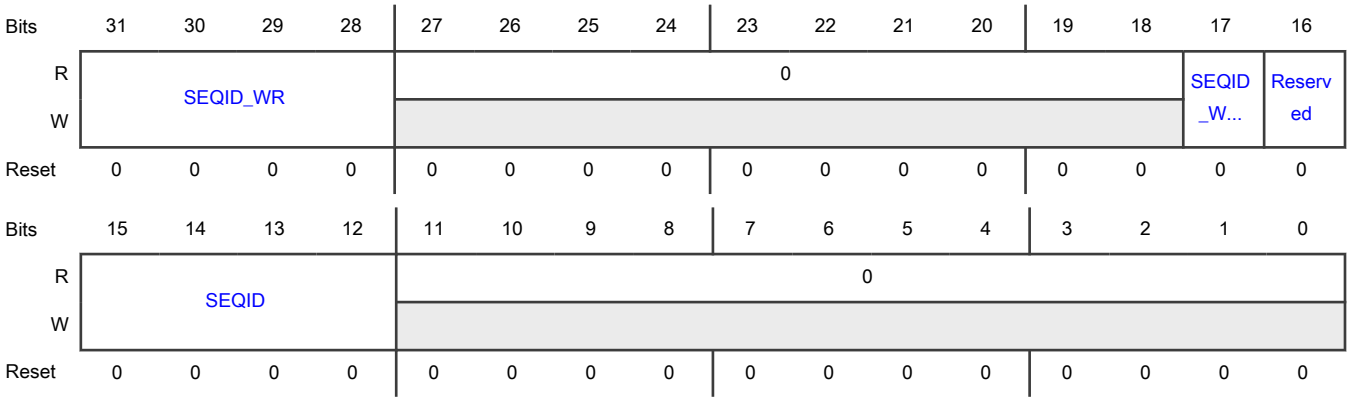
This register provides generic configuration to any of the buffer accesses. Any buffer "miss" leads to the buffer being flushed and a serial flash memory transaction being triggered per the sequence pointed to by the SEQID field.

Special write-access is permitted if:

- SR[AHB_ACC] = 0

This register is access controlled and can only be programmed by privilege masters.

Diagram



Fields

Field	Function
31-28 SEQID_WR	Write Sequence ID This field points to write seq-id of LUT and would be selected in case of AHB write transaction, if SEQID_WR_EN==1.
27-18 —	Reserved
17 SEQID_WR_EN	Enable Write Sequence ID
16 —	Reserved
15-12	Points to a sequence in the LUT.

Table continues on the next page...

Table continued from the previous page...

Field	Function
SEQID	This field contains the sequence index of the LUT. and would be selected in case of AHB read transaction (and AHB write transactions if SEQID_WR_EN==0).. See LUT. **NOTE** If the sequence pointer differs in the new and the previous sequences, you should reset it. See sequence pointer clear register for more information.
11-0 —	Reserved

79.13.2.10 SOC Configuration Register (SOCCR)

Offset

Register	Offset
SOCCR	24h

Function

This register is programmed at the chip level for QuadSPI configuration. For details, see chip-specific QuadSPI information.

Special write-access is permitted if:

- SR[AHB_ACC] = 0

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SOCCFG															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SOCCFG															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 SOCCFG	SOC configuration This field configuration is specific to chip. For details, see chip-specific QuadSPI information.

79.13.2.11 Buffer 0 Top Index Register (BUF0IND)

Offset

Register	Offset
BUF0IND	30h

Function

This register specifies the top index for buffer 0, which defines its size. Note that the three LSBs of this register are set to 0. This ensures that the buffer is 64-bit aligned because each buffer entry is 64-bits long.

The register value should be set to the desired number of bytes. For example, setting BUF0IND[31:3] to 0 gives 0 bytes, setting the value to 1 gives 8 bytes, and so on.

The size of buffer 0 is the difference between BUF0IND and 0.

The software must ensure that the value of TPINDX0 is not greater than the size of buffer 0.

Special write-access is permitted if:

- SR[AHB_ACC] = 0

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				TPINDX0									0		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-11 —	Reserved
10-3 TPINDX0	Top index of buffer 0
2-0 —	Reserved

79.13.2.12 Buffer 1 Top Index Register (BUF1IND)

Offset

Register	Offset
BUF1IND	34h

Function

This register specifies the top index of buffer 1, which defines its size. Note that the three LSBs of this register are set to 0. This ensures that the buffer is 64-bit aligned because each buffer entry is 64-bits long.

The size of buffer 1 is the difference between BUF1IND and BUF0IND. The register value should be entered in bytes. For example, if BUF0IND = 0x100, then setting BUF1IND = 0x130 sets the size of buffer 1 to 0x30 bytes.

The software must ensure that the value of TPINDX1 is not greater than the size of buffer 1.

Special write-access is permitted if:

- SR[AHB_ACC] = 0

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				TPINDX1										0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-11 —	Reserved
10-3 TPINDX1	Top index of buffer 1
2-0 —	Reserved

79.13.2.13 Buffer 2 Top Index Register (BUF2IND)

Offset

Register	Offset
BUF2IND	38h

Function

This register specifies the top index of buffer 2, which defines its size. Note that the three LSBs of this register are set to 0. This ensures that the buffer is 64-bit aligned because each buffer entry is 64-bits long.

The size of buffer 2 is the difference between BUF2IND and BUF1IND. The register value should be entered in bytes. For example, if BUF1IND = 0x130 then setting BUF2IND = 0x180 sets the size of buffer 2 to 0x50 bytes.

The software must ensure that the value of TPINDX2 is not greater than the size of buffer 2.

Special write-access is permitted if:

- SR[AHB_ACC] = 0

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				TPINDX2										0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-11 —	Reserved
10-3 TPINDX2	Top index of buffer 2
2-0 —	Reserved

79.13.2.14 AHB Write Configuration Register (AWRCR)

Offset

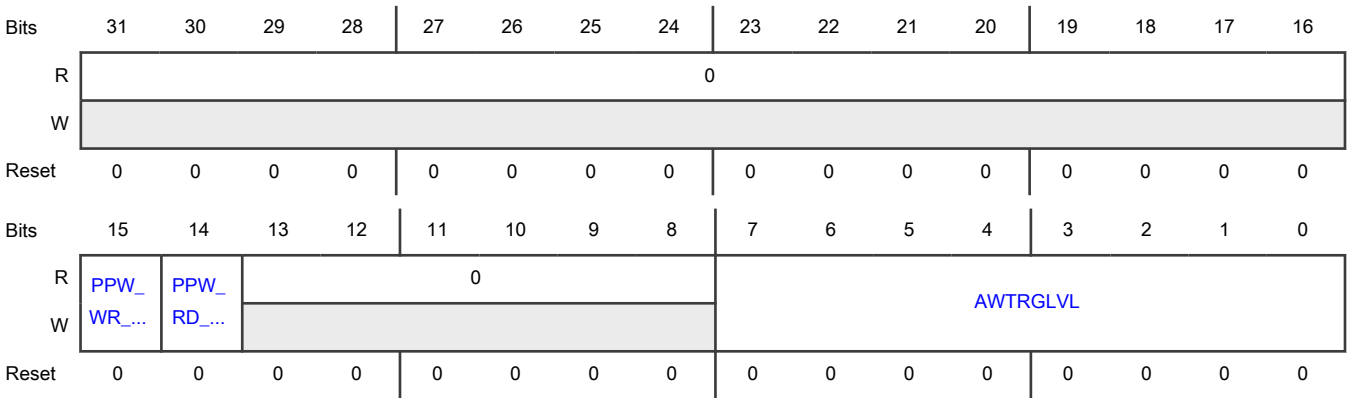
Register	Offset
AWRCR	50h

Function

Special write-access is permitted if:

- SR[AWRACC] = 0

Diagram



Fields

Field	Function
31-16 —	Reserved
15 PPW_WR_DIS	Page program wait write disabled 0b - Enables subsequent writes 1b - Disables subsequent writes to the flash memory. After the first write transaction, AHB write is returned with an error response.
14 PPW_RD_DIS	Page program wait read disabled 0b - Enables subsequent reads 1b - Disables subsequent reads to the flash memory. After the first write transaction, AHB read is returned with an error response.
13-8 —	Reserved
7-0	AHB write trigger level

Table continues on the next page...

Table continued from the previous page...

Field	Function
AWTRGLVL	Defines a trigger level of TX FIFO in terms of a 4-byte entry. AHB write to the flash memory is triggered when either of the following happens: • TX FIFO crosses the level defined by this field • AHB transaction completes This is done to prevent an under run in the flash memory. For HyperRAM, this field must be set to 0. For details, see HyperRAM support

79.13.2.15 DLL Flash Memory A Configuration Register (DLLCRA)

Offset

Register	Offset
DLLCRA	60h

Function

This register configures DLL and slave delay chain for flash memory A.

The value of the DLLEN field must be 1 after all reference (FREQEN, DLL_REFCNTR, DLLRES, SLAVE_AUTO_UPDT) delay chain configurations are programmed.

See DLL and delay chain usage for the programming sequence.

NOTE

See the chip data sheet for information on programming register fields.

NOTE

Please see the chip specific section of QuadSPI DLLCRA register for delay elements data

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DLLEN	FREQEN	0		DLL_REFCNTR				DLLRES				SLV_FINE_OFFSET			
W																
Reset	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	SLV_DLY_OFFSET			SLV_DLY_COARSE				Reserved			DLL_CDL8	SLAVE_A...	SLV_EN	SLV_DLL...	SLV_UPD
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 DLLEN	DLL enable 0b - DLL reference logic remains in reset and should be 0 for at least three flash memory clock cycles for reset. 1b - Enables DLL logic. Set it to 1 after all the configuration for DLLCR reference settings is complete.
30 FREQEN	Frequency enable 0b - Selects delay chain for low frequency of operation 1b - Selects delay chain for high frequency of operation
29-28 —	Reserved
27-24 DLL_REFCNTR	DLL reference counter Select the "n+1" interval of DLL phase detection and reference delay updating interval (minimum recommended value = 1).
23-20 DLLRES	DLL resolution Minimum resolution for DLL phase detector to remain locked/unlocked based on flash memory clock jitter. The minimum value is 2, and should be programmed to a more suitable value, such as 6.
19-16 SLV_FINE_OFFSET	Fine offset delay elements in incoming DQS This field sets the number of fine offset delay elements up to 16 in incoming DQS, and the default must be 1 element.
15 —	Reserved
14-12 SLV_DLY_OFFSET	T/16 offset delay elements in incoming DQS This field sets the number of T/16 offset delay elements in incoming DQS; default is 0.
11-8 SLV_DLY_COARSE	Delay elements in each delay tap This field sets the number of delay elements in each delay tap. The field is used to overwrite DLL-generated delay values and works when the value of SLV_DLL_BYPASS is 1. Note : Please refer to the QuadSPI datasheet for more details.
7-5 —	Reserved
4 DLL_CDL8	DLL CDL8 Enable 0b - DLL is implemented to support within 2x variation 1b - DLL is implemented to support within 3x variation (BCS -> WCS)

Table continues on the next page...

Table continued from the previous page...

Field	Function
3 SLAVE_AUTO_UPDT	Slave chain update This field automatically updates the slave chain as soon as DLL is locked. 0b - Auto-update feature is disabled. 1b - Auto-update feature is enabled.
2 SLV_EN	Slave enable 0b - DLL slave logic remains in reset, and its value should be 0 for at least three flash memory clock cycles for reset. 1b - Enables DQS slave delay chain, and should be 1 before any slave configuration settings take place.
1 SLV_DLL_BYPASS	Slave DLL bypass This field enables selection of the number of delays in each slave delay tap. 0b - Disables manual selection of coarse delays in the slave delay chain. 1b - Enables selection of number of delays in each slave delay tap, based on DLLCRA[SLV_DLY_COARSE].
0 SLV_UPD	Slave update You must program this field only after slave delay chain configuration takes place. 0b - Disables any further update on DQS slave delay chain. 1b - Updates the DQS slave delay chain with either ref-delay or bypass slave delay value, and should be set in the absence of the DQS clock.

79.13.2.16 Parity Configuration Register (PARITYCR)

Offset

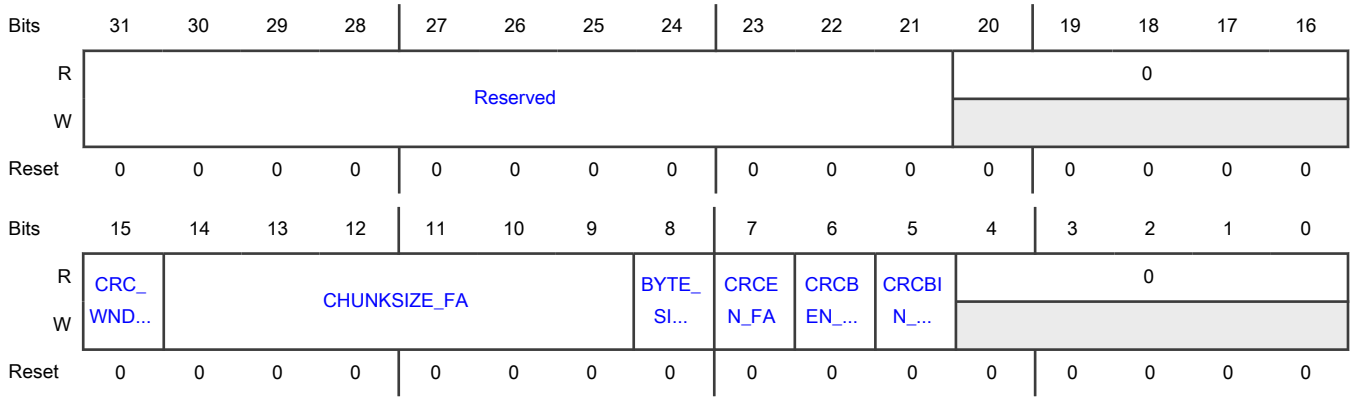
Register	Offset
PARITYCR	6Ch

Function

This register replicates the parity related configuration programming performed on the flash memory configuration registers.

You must write 1 to PARITYCR[CRCEN_FA] and PARITYCR[CRCEN_FB] after all parity configuration bits are programmed. However, you can program PARITYCR[BYTE_SIZE_FA] and PARITYCR[BYTE_SIZE_FB] at any time.

Diagram



Fields

Field	Function
31-21 —	Reserved
20-16 —	Reserved
15 CRC_WNDW_FA	CRC address window configuration 0b - Calculates parity (CRC) over fixed address window within the denoted flash memory A chunk size boundary. 1b - Calculates parity (CRC) over incremental window of flash memory A chunk size, irrespective of address.
14-9 CHUNKSIZE_FA	Chunk size for flash memory A Defines the chunk size (in terms of 4 bytes) after parity is inserted and compared by the QuadSPI controller for flash memory A. Values 0x1, 0x2, and so on indicate 4 bytes, 8 bytes, and so on.
8 BYTE_SIZE_FA	Byte size for flash memory A This field can be programmed any time. It enables single-byte, read/write parity with flash memory A configuration register. The field overrides the chunk size used for flash memory data array. You must write 1 to this field before any read/write operation on the flash memory internal registers. And you must write 0 to this field before reading flash memory data array.
7 CRCEN_FA	CRC parity checker logic 0b - Disables parity mechanism. In case of parity error, set it to 0 to clear parity error. Only supported for DDR Octal commands. 1b - CRC parity checker logic for flash memory A read paths. Configure this field after programming CHUNK_SIZEA, CRCBEN_FA and CRCBIN_FA.
6 CRCBEN_FA	Adds CRC bar parity from flash memory A output to QuadSPI controller

Table continues on the next page...

Table continued from the previous page...

Field	Function
5 CRCBIN_FA	Adds CRC bar parity to flash memory A input from QuadSPI controller
4-0 —	Reserved

79.13.2.17 Serial Flash Memory Address Register (SFAR)

Offset

Register	Offset
SFAR	100h

Function

The module automatically translates this address on the memory map to the address on the flash memory. When operating in a 24-bit mode, only bits 23-0 are sent to the flash memory. In the 32-bit mode, bits 27-0 are used with bits 31-28 driven to 0 when the value of SFACR[CAS] is 0. For example, if the value of SFACR[CAS] is 3, then bits 26-3 are sent to the flash memory as its page address in case flash memory is operating in a 24-bit mode. The total number of address bits requested by the flash memory, as its page and column address, must not be more than 32 bits. See Table 768 for the mapping between the access mode and the SFAR content and Normal mode for details on command triggering and command execution. The software must ensure that the serial flash memory address provided in the SFAR register lies in the valid flash memory address range, as defined in Table 768.

Special write-access is permitted if:

- SR[IP_ACC] = 0

NOTE

If MDAD and FRAD checks are enabled in MGC register but none of the MDAD and FRAD descriptors are valid, then any write on this register will generate a bus transfer error

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SFADR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SFADR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 SFADR	Serial flash memory address

79.13.2.18 Serial Flash Memory Address Configuration Register (SFACR)

Offset

Register	Offset
SFACR	104h

Function

This register contains the address requirements that are specific to serial flash memory. These requirements must be configured according to the connected flash memory, for the controller to function properly. The module automatically translates the address of SFAR on the memory map or the incoming address on the AHB bus to the column address on the flash memory. For example, if a flash memory needs 3 bits as its column address, then only the lower three bits of the SFAR/AHB address are sent to the flash memory as its column address. The software should ensure that the serial flash memory address provided in SFAR or the incoming AHB address lies in the valid flash memory address range.

Special write-access is permitted if:

- SR[IP_ACC] = 0
- SR[AHB_ACC] = 0

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0											CAS_INT...	0		Reserved	WA
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		Reserved	PPWB					0				CAS			
W																
Reset	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-21 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
20 CAS_INTRLVD	CAS Interleaving 0b - CAS interleaving is disabled 1b - CAS interleaving is enabled
19-18 —	Reserved
17 —	Reserved
16 WA	Word addressable Defines whether the serial flash memory is a byte addressable flash memory or a word addressable flash memory. According to the configuration of this field, the address is remapped to the flash memory interface. See Address scheme for details. 0b - Byte addressable serial flash memory mode 1b - Word (2-byte) addressable serial flash memory mode
15-14 —	Reserved
13 —	Reserved
12-8 PPWB	Page program boundary Flash memory-specific page program boundary size should be programmed in Log2 (size in bytes) format. The default is 8 = log2(256) for a 256-byte page program size.
7-4 —	Reserved
3-0 CAS	Column address space Defines the width of the column address. If the column address is, for example, [2:0] of SFAR/AHB address, then CAS must be 3. If there is no column address separation in any serial flash memory, the value of this field must be specified as 0.

79.13.2.19 Sampling Register (SMPR)

Offset

Register	Offset
SMPR	108h

Function

This register allows configuration of how the incoming data from the external serial flash memory devices is sampled in the QuadSPI module.

NOTE

See the chip data sheet for programming the register fields.

Special write-access is permitted in the disabled mode.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved	Reserved			Reserved	DLLFSMPFA			0				Reserved			
W																
Reset	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0									FSDLY	FSPHS	0		0		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30-28 —	Reserved
27 —	Reserved
26-24 DLLFSMPFA	Selects the nth tap provided by slave delay chain for flash memory A The value of *n* can vary from 0 to 7, with each tap delay based on the DLLCRA register.
23-19 —	Reserved
18-16 —	Reserved
15-7 —	Reserved
6	Full-speed delay selection for internal/pad loop back DQS sampling

Table continues on the next page...

Table continued from the previous page...

Field	Function
FSDLY	This field selects the delay in accordance with the reference edge for the valid sample point. 0b - Same DQS 1b - Half-cycle early DQS
5 FSPHS	Full-speed phase selection for SDR instructions This field selects the edge of the sampling clock valid for full-speed commands. 0b - Select sampling at non-inverted clock 1b - Select sampling at inverted clock
4-3 —	Reserved
2-0 —	Reserved

79.13.2.20 RX Buffer Status Register (RBSR)

Offset

Register	Offset
RBSR	10Ch

Function

This register contains information related to the receive data buffer.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RDCTR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0										RDBFL					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 RDCTR	Read counter Indicates the number of 4-byte entries removed from the RX buffer. For example, a value of 0x2 indicates that 8 bytes have been removed. It is incremented by the number (RBCT[WMRK] + 1) on RX buffer POP event. The RX buffer can be popped using DMA or FR[RBDF]. The RSER[RBDDE] defines which pop should be pursued. For details, see AHB RX Data Buffer Register (ARDB0 - ARDB31) and Data Transfer from the QuadSPI Module Internal Buffers.
15-6 —	Reserved
5-0 RDBFL	RX buffer fill level Indicates the number of 4-byte entries available in the RX buffer. For example, a value of 0x2 indicates 8 bytes are available.

79.13.2.21 RX Buffer Control Register (RBCT)

Offset

Register	Offset
RBCT	110h

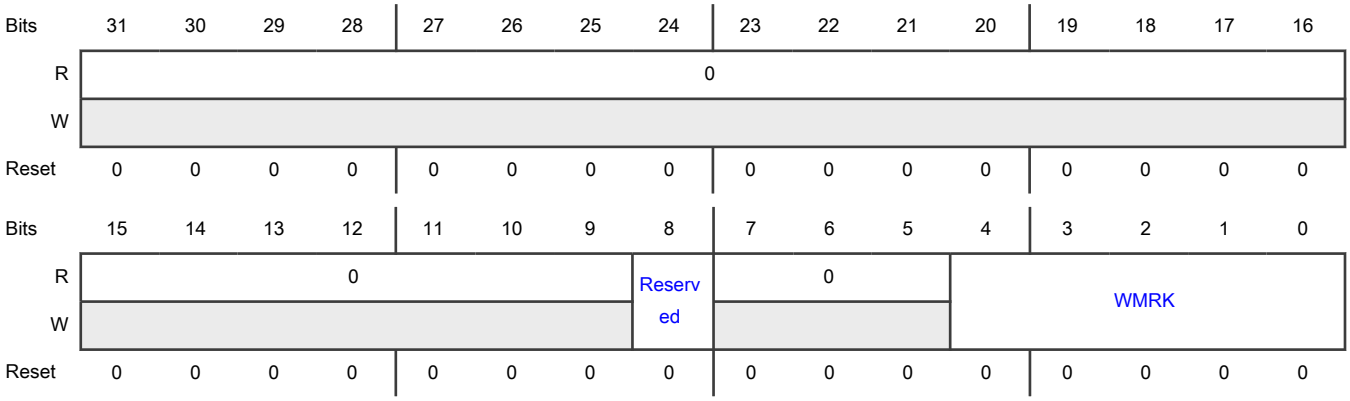
Function

This register contains control data related to the receive data buffer.

Special write-access is permitted if:

- SR[IP_ACC] = 0

Diagram



Fields

Field	Function
31-9 —	Reserved
8 —	Reserved
7-5 —	Reserved
4-0 WMRK	RX buffer watermark This field determines when the readout action of the RX buffer is triggered. When the number of valid entries in the RX buffer is equal to or greater than the number provided by (WMRK+1), the SR[RXWE] flag is asserted. The value should be entered as the number of 4-byte entries minus 1. For example, a value of 0x0 sets the watermark to 4 bytes, 1 to 8bytes, 2 to 12 bytes, and so on. For details, see DMA usage. **NOTE** This field should never be programmed above 31 because there are only 32 memory mapped RBDR registers. If watermark is programmed above 31, data above 32 words will be lost.

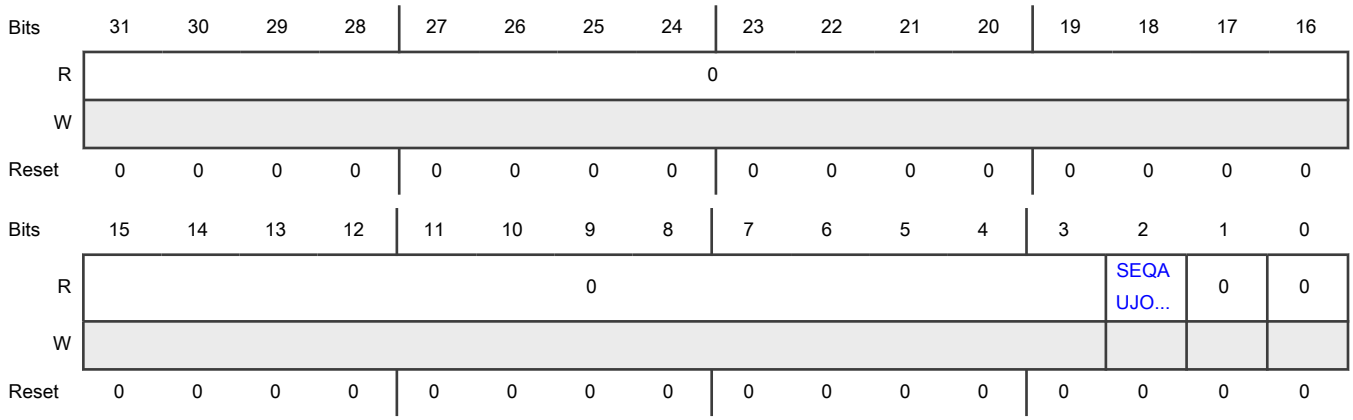
79.13.2.22 AHB Write Status Register (AWRSR)

Offset

Register	Offset
AWRSR	120h

Function

Diagram



Fields

Field	Function
31-3 —	Reserved
2 SEQAUJOIN	Sequence auto join Asserted when the LUT sequences are automatically joined using the JMP_TO_SEQ command. This remains asserted as long as you do not encounter the STOP/JMP_ON_CS command.
1 —	Reserved
0 —	Reserved

79.13.2.23 DLL Status Register (DLLSR)

Offset

Register	Offset
DLLSR	12Ch

Function

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved				0				Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DLLA_LO...	SLVA_LO...	DLLA_RA...	DLLA_FI...	0				DLLA_SLV_FINE_VAL				DLLA_SLV_COARSE_VAL			
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
27-24 —	Reserved
23-16 —	Reserved
15 DLLA_LOCK	DLL A lock status
14 SLVA_LOCK	Slave high lock status High-frequency slave delay chain locked. The field is configured when the slave decoder is updated with DLLCRA[SLV_UPD] and is reset after you configure SLV_UPD to 0.
13 DLLA_RANGE_ERR	DLL master delay chain The value 1 indicates that DLL master delay chain is working out of delay range because of incorrect DLL configuration.
12 DLLA_FINE_UNDERFLOW	Fine delay chain underflow The value 1 indicates that fine delay chain underflow has occurred.
11-8 —	Reserved
7-4 DLLA_SLV_FINE_VAL	Fine delay cells in slave delay chain This value indicates the total number of fine delay cells (1 delay unit) selected in the slave delay chain.
3-0 DLLA_SLV_COARSE_VAL	Coarse delay cells in slave delay chain This value indicates the total number of coarse delay cells (16 delay units) selected in the slave delay chain.

79.13.2.24 Data Learning Configuration Register (DLCR)

Offset

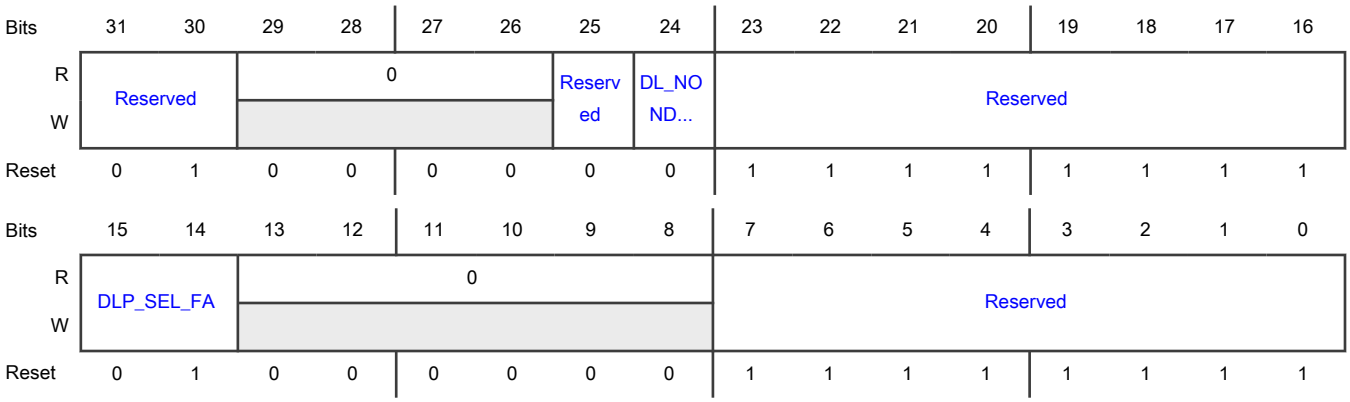
Register	Offset
DLCR	130h

Function

This register is used for programming the data learning settings.

- A flash memory device that supports data learning but cannot provide more than 10 bits of data learn pattern must be considered as non-DLP Flash. For Non-DLP flash devices, you must read at least 16 bits data learn pattern from a known location.
- You must program different learning pattern on the two data pins–IO1 and IO3. This is required to accommodate pessimistic data skew between the different IO lines.

Diagram



Fields

Field	Function
31-30 —	Reserved
29-26 —	Reserved
25 —	Reserved
24 DL_NONDLP_FLSH	Data learning enabled for non-DLP flash memory Configure the value of this field as 1 to enable data learning for flash memories that do not provide a data learning pattern. For a non-DLP flash memory, execute a one-time data learning through the IPS to select a tap before an AHB read operation.
23-16 —	Reserved
15-14 DLP_SEL_FA	Selects pattern matching IO pads 00b - Pattern matching is ignored. This is only for debugging purpose and should not be programmed. 01b - IO1 is used for matching

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10b - IO3 is used for matching. This is only for debugging purpose and should not be programmed. 11b - Both IO1 and IO3 are used for pattern matching
13-8 —	Reserved
7-0 —	Reserved

79.13.2.25 Data Learning Status Flash Memory A Register (DLSR_FA)

Offset

Register	Offset
DLSR_FA	134h

Function

This register shows sampling point selected by data learning algorithm when the value of DLSR_FA[DLPFFA] is 0. Otherwise, it shows the pattern matching outline.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DLPFFA	0														
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	POS_EDGE								NEG_EDGE							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 DLPFFA	Data learning pattern fail This field asserts when data learning fails at flash memory A.

Table continues on the next page...

Table continued from the previous page...

Field	Function
30-16 —	Reserved
15-8 POS_EDGE	DLP positive edge match signature for flash memory A
7-0 NEG_EDGE	DLP negative edge match signature for flash memory A

79.13.2.26 TX Buffer Status Register (TBSR)

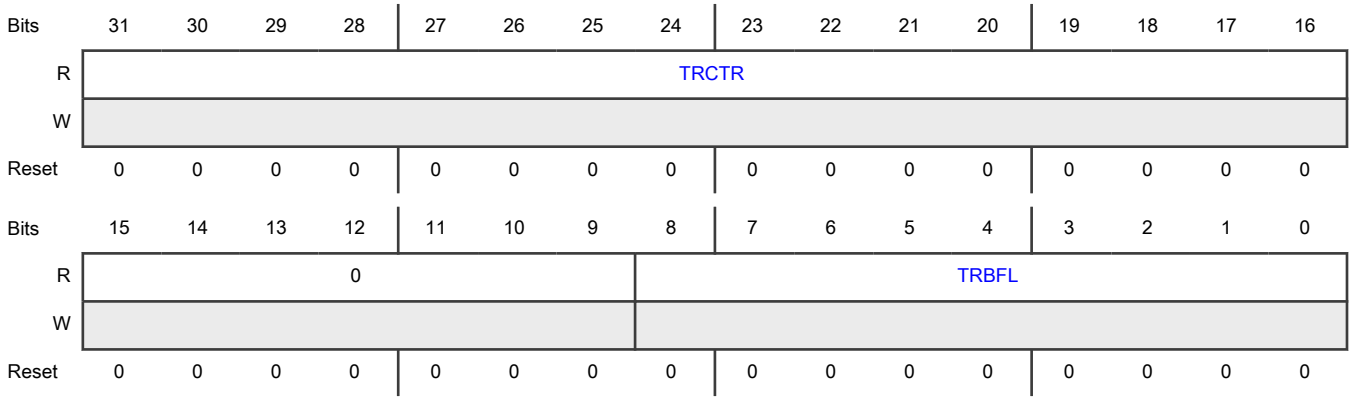
Offset

Register	Offset
TBSR	150h

Function

This register contains information related to the transmit data buffer.

Diagram



Fields

Field	Function
31-16 TRCTR	Transmit counter This field indicates how many entries of 4 bytes have been written into the TX buffer by host accesses. It is reset to 0 when a 1 is written to MCR[CLR_TXF]. It is incremented on each write access to the TBDR register when another word has been pushed onto the TX buffer. When it is not cleared, the TRCTR field wraps around to 0. See TX Buffer Data Register (TBDR) for details.

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-9 —	Reserved
8-0 TRBFL	TX buffer fill level This field contains the number of entries of 4 bytes each available in the TX buffer for the QuadSPI module to transmit to the serial flash memory device. The value of this field can reach maximum up to the total TX buffer size.

79.13.2.27 TX Buffer Data Register (TBDR)

Offset

Register	Offset
TBDR	154h

Function

This register provides access to the circular TX buffer of depth 256, so the total size is 256 * 4 bytes. This buffer provides the data written into it as write data for the page programming commands to the serial flash memory device. See Table 732 for the byte ordering scheme. A write transaction on the flash memory with data size of less than 32 bits leads to the removal of one data entry from the TX buffer. The valid bits are used and the rest of the bits are discarded.

Special write-access is permitted if:

- SR[TXFULL] = 0

NOTE

This register can only be written when write access is granted by SFP block else bus transfer error is generated on write. Please refer to section TBDR register write lock

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	TXDATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	TXDATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TXDATA	TX data On write access, the data is written to the next available entry of the TX buffer and TBSR[TRBFL] is updated accordingly. On a read access, the last data written to the register is returned.

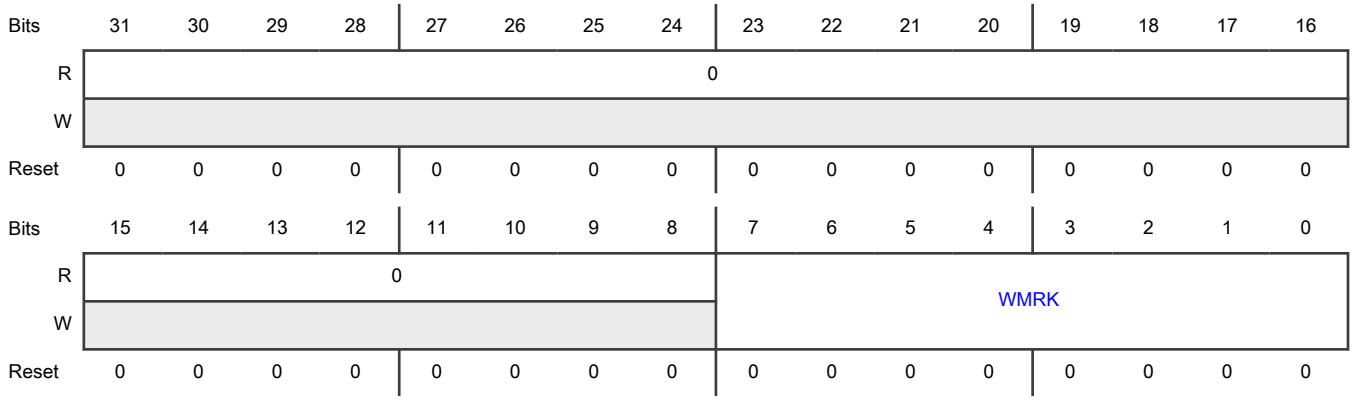
79.13.2.28 TX Buffer Control Register (TBCT)

Offset

Register	Offset
TBCT	158h

Function
This register contains control information for transmit data buffer.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 WMRK	Watermark for TX buffer Determines the watermark for the TX buffer When the number of available space in the TX buffer is greater than or equal to the number provided by WMRK (number of 4-byte entries), SR[TXWA] is asserted. For example, a value of 0x1 sets the watermark to 4 bytes, 0x2 sets it to 8 bytes, 0x3 sets it to 12 bytes, and so on. For details, see DMA usage.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	WMRK = 0 is invalid. **NOTE** For IPS write with SFP enabled, this field must be programmed as per formula to prevent TX underflow: If data size in words >1, watermark=(TX FIFO size in words - data size in words) +1, If data size in words =1, watermark=(TX FIFO size in words - data size in words)

79.13.2.29 Status Register (SR)

Offset

Register	Offset
SR	15Ch

Function

This register provides all the available status information about SFM command execution and arbitration, the RX buffer, TX buffer, and the AHB buffer.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0			0	TXFULL	TXDMA	TXWA	TXNE	RXDMA	0			RXFULL	0		RXWE
W																
Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	AHB3FUL	AHB2FUL	AHB1FUL	AHB0FUL	AHB3NE	AHB2NE	AHB1NE	AHB0NE	AHBTRN	Reserved	AWRACC	Reserved	AHB_ACC	IP_ACC	BUSY
W																
Reset	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	Reserved
28 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
27 TXFULL	TX buffer full Asserted when the FIFO level reaches TX buffer size
26 TXDMA	TX DMA Asserted when the TXFIFO fill via DMA is active and DMA is requested or running
25 TXWA	TX buffer watermark available Asserted when the number of available spaces in the TX buffer is greater than or equal to the value provided by TBCT[WMRK] Example: When TBCT[WMRK]=1, SR[TXWA] is de-asserted when TX FIFO has 256+7(size of async FIFO) entries
24 TXNE	TX buffer not empty Asserted when TX buffer contains data
23 RXDMA	RX buffer DMA Asserted when RX buffer read out via DMA is active; that is, when DMA is requested or running
22-20 —	Reserved
19 RXFULL	RX buffer full Asserted when the RX buffer is full; that is, when RBSR[RDBFL] is equal to 32
18-17 —	Reserved
16 RXWE	RX buffer watermark exceeded Asserted when the number of valid entries in the RX buffer exceeds the number provided in RBCT[WMRK]
15 —	Reserved
14 AHB3FUL	AHB 3 buffer full Asserted when AHB 3 buffer is full
13 AHB2FUL	AHB 2 buffer full Asserted when AHB 2 buffer is full
12 AHB1FUL	AHB 1 buffer full Asserted when the AHB 1 buffer is full
11	AHB 0 buffer full

Table continues on the next page...

Table continued from the previous page...

Field	Function
AHB0FUL	Asserted when the AHB 0 buffer is full
10 AHB3NE	AHB 3 buffer not empty Asserted when the AHB 3 buffer contains data
9 AHB2NE	AHB 2 buffer not empty Asserted when the AHB 2 buffer contains data
8 AHB1NE	AHB 1 buffer not empty Asserted when the AHB 1 buffer contains data
7 AHB0NE	AHB 0 buffer not empty Asserted when the AHB 0 buffer contains data
6 AHBTRN	AHB access transaction pending Asserted when there is a pending request on the AHB interface. See Flash memory mapped AMBA bus.
5 —	Reserved
4 AWRACC	AHB write access Asserted when AHB write access is enabled
3 —	Reserved
2 AHB_ACC	AHB read access Asserted when the currently executed transaction is initiated by the AHB bus
1 IP_ACC	IP access Asserted when transaction currently executed is initiated by the IP bus
0 BUSY	Module busy Asserted when module is currently busy handling a transaction to an external flash memory device

79.13.2.30 Flag Register (FR)

Offset

Register	Offset
FR	160h

Function

This register provides all available flags about SFM command execution and arbitration, which may serve as the source for the generation of interrupt service requests. Note that the error flags in this register do not relate directly to the execution of the transaction in the serial flash memory device itself but only to the behavior and conditions visible in the QuadSPI module.

Special write-access is permitted in the enabled mode.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DLPFF	0	Reserved	DLLABRT	TBFF	TBUF	0	DLLUNLCK	ILLINE	0			0		RBOF	RBDF
W	W1C			W1C	W1C	W1C		W1C	W1C						W1C	W1C
Reset	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	AAEF	AITEF	AIBSEF	ABOF	Reserved	CRCAEF	Reserved	PPWF	0	IPIEF	0	0	0			TFF
W	W1C	W1C	W1C	W1C	W1C	W1C		W1C		W1C						W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 DLPFF	Data learning pattern failure flag This field is set when the DATA_LEARN instruction is encountered in a sequence, but no sampling point is found for the data learning pattern.
30 —	Reserved
29 —	Reserved
28 DLLABRT	DLL abort 1b - This field is set whenever DLL is unlocked while reading data from the flash memory.
27 TBFF	TX buffer fill flag Before writing to the TX buffer, this field should be cleared. Then, it should be read back. If it is set, the TX buffer can include more data. If the field remains cleared, the TX buffer can be considered as full. See TX buffer operation for details.
26 TBUF	TX buffer underrun flag This field is set if the module tries to pull data when the TX buffer is empty.. The IP command leading to the TX buffer underrun is continued (data sent to the serial flash memory device is undefined). Here, a valid underrun means that it should have occurred during the transaction so that few bytes (that is, less than 4

Table continues on the next page...

Table continued from the previous page...

Field	Function
	bytes) are left in FIFO and the remaining are filled with "FFFFh". This field does not set if transfer is less than 128 bits. The application must clear the TX buffer in response to this event by writing a 1 to MCR[CLR_TXF]. The application must clear the TX buffer in response to this event by writing a 1 to MCR[CLR_TXF].
25 —	Reserved
24 DLLUNLCK	DLL unlock 1b - This field is set whenever DLL unlock event occurs, irrespective of flash memory access.
23 ILLINE	Illegal instruction error flag This field is set when an illegal instruction is encountered by the controller in any of the sequences. As soon as the field is set, you must assert MCR[SWRSTSD] and MCR[SWRSTHD]. That is, reset the flash memory and AHB domain after reconfiguring the correct sequence instruction. See Table 730 for a list of legal instructions.
22-21 —	Reserved
20 —	Reserved
19-18 —	Reserved
17 RBOF	RX buffer overflow flag This field is set when no more data can be pushed into the RX buffer from the serial flash memory device. The IP command leading to this condition is continued until the number of bytes in IPCR[IDATSZ] are read from the serial flash memory device. The content of the RX buffer remains unchanged.
16 RBDF	RX buffer drain flag This field is set if SR[RXWE] is asserted. Writing 1 to this field triggers one of the following actions: • If the RX buffer has up to RBCT[WMRK] valid entries, then the flag is cleared. • If the RX buffer has more than RBCT[WMRK] valid entries and the RSER[RBDDE] field is not set (flag driven mode), an RX buffer POP event is triggered. The flag remains set if the RX buffer contains more than RBCT[WMRK] valid entries after the RX buffer POP event is complete. The flag is cleared if the RX buffer contains less than or equal to RBCT[WMRK] valid entries after the RX buffer POP event is complete.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	See the "Receive Buffer Drain Interrupt or DMA Request" section in Normal mode interrupt and DMA requests for details.
15 AAEF	AHB abort error flag This flag can be set when AHB transaction in ongoing and any of the below conditions occur which may result in AHB error response (HRESP) generation: • When Software abort is asserted from SPTRCLT[ABRT_CLR] during AHB read. • During data learning pattern failure if any AHB transaction is ongoing. (FR[DLPFF]) • CRC or ECC error from flash memory during AHB transaction. (FR[CRCAEF]) • If AIBSEF flag is set during the AHB transfer. (FR[AIBSEF]) Software should clear this flag and other related flags in FR register (mentioned in points above) before initiating any new AHB transfer.
14 AITEF	AHB illegal transaction error flag This is set whenever there is no response generated from QuadSPI to AHB bus in case of an illegal transaction and the watchdog timer expires. The timer value is considered as a parameter.
13 AIBSEF	AHB illegal burst size error flag This is set whenever the total burst size (size x beat) of an AHB transaction is greater than the prefetch data size, which is defined by BUFxCR[ADATSZ] or data size mentioned in the sequence pointed to by the SEQID field in case ADATSZ = 0. See HBURST support with AHB read details on HBURST feature.
12 ABOF	AHB buffer overflow flag This is set when the size of the AHB access exceeds the size of the AHB buffer. This condition can occur only if BUFxCR[ADATSZ] is programmed incorrectly. The AHB command leading to this condition is continued until the number of entries according to BUFxCR[ADATSZ] have been read from the serial flash memory device. The content of the AHB buffer is not changed.
11 —	Reserved
10 CRCAEF	Sets when there is CRC or ECC error for flash memory A 0b - CRCEF interrupt is not generated. 1b - CRCEF interrupt is generated.
9 —	Reserved
8 PPWF	Page-program wait flag after flash memory write flag This field indicates page-program wait flag after flash memory write.
7 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
6 IPIEF	IP command trigger could not be executed error flag This is set when the SR[IP_ACC] and SR[AWRACC] fields are set (that is, an IP triggered command is currently executing) and any of the following conditions occurs: • Write access to the RBCT
5 —	Reserved
4 —	Reserved
3-1 —	Reserved
0 TFF	IP command transaction finished flag This field is set after the QuadSPI module completes a running IP command. If an error occurs, and the related error flags are valid in the same clock cycle, the TFF flag is asserted.

79.13.2.31 Interrupt and DMA Request Select and Enable Register (RSER)

Offset

Register	Offset
RSER	164h

Function

This register provides enables and selectors for the interrupts in the QuadSPI module.

NOTE

Each field of the FR enabled as source for an interrupt prevents the QuadSPI module from entering the Stop mode or Module Disable mode when this flag is set.

Special write-access is permitted in the "Anytime" mode.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DLPFIE	Reserved		0	TBFIE	TBUIE	TBFDE	DLLULIE	ILLINIE	Reserved	RBDDE	Reserved	0		RBOIE	RBDIE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	AAIE	AITIE	AIBSIE	ABOIE	Reserved	CRCAIE	Reserved	PPWIE	Reserved	IPIEIE	0	Reserved	0			TFIE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 DLPFIE	Data learning pattern failure interrupt enable Triggered by DLPFF flag in FR. 0b - No DLPFF interrupt is generated. 1b - DLPFF interrupt is generated.
30-29 —	Reserved
28 —	Reserved
27 TBFIE	TX buffer fill interrupt enable flag This field indicates the TX buffer fill interrupt enable flag. This interrupt should not be enabled when using SFP functionality because it causes QSPI BUSY SFM to go high and keeps SFP to generate any transaction over to QSPI. 0b - No TBFF interrupt is generated. 1b - TBFF interrupt is generated.
26 TBUIE	TX buffer underrun interrupt enable flag This field indicates the TX buffer underrun interrupt enable flag. 0b - No TBUF interrupt is generated 1b - TBUF interrupt is generated
25 TBFDE	TX buffer fill DMA enable Enables generation of DMA requests for TX buffer fill. When the value of this field is 1, DMA requests are generated as long as number of available spaces in the TX buffer is greater than or equal to the value provided by TBCT[WMRK].

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No DMA request is generated 1b - DMA request is generated
24 DLLULIE	DLL unlock interrupt enable 1b - Write 1 to this to enable generation of interrupt on DLL unlock event.
23 ILLINIE	Illegal instruction error interrupt enable Triggered by the ILLINE flag in FR 0b - No ILLINE interrupt is generated. 1b - ILLINE interrupt is generated.
22 —	Reserved
21 RBDDE	RX buffer drain DMA enable This field enables generation of DMA requests for RX buffer drain. When the value of this field is 1, the DMA requests are generated as long as SR[RXWE] is set. 0b - No DMA request is generated. 1b - DMA request is generated.
20 —	Reserved
19-18 —	Reserved
17 RBOIE	RX buffer overflow interrupt enable This field indicates the RX buffer overflow interrupt enable flag. 0b - No RBOF interrupt is generated. 1b - RBOF interrupt is generated.
16 RBDIE	RX buffer drain interrupt enable This field enables generation of IRQ requests for RX buffer drain. When the value of this field is 1, the interrupt is asserted as long as SR[RBDF] is set. 0b - No RBDF interrupt is generated. 1b - RBDF Interrupt is generated.
15 AAIE	AHB abort error interrupt enable Triggered by AAEF flags in FR.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No AAEF interrupt is generated 1b - AAEF interrupt is generated
14 AITIE	AHB illegal transaction interrupt enable flag This field indicates the AHB illegal transaction interrupt enable flag. 0b - No AITEF interrupt is generated. 1b - AITEF interrupt is generated.
13 AIBSIE	AHB illegal burst size interrupt enable flag This field indicates the AHB illegal burst size interrupt enable flag. 0b - No AIBSEF interrupt is generated. 1b - AIBSEF interrupt is generated.
12 ABOIE	AHB buffer overflow interrupt enable flag This field indicates the AHB buffer overflow interrupt enable flag. 0b - No ABOF interrupt is generated. 1b - ABOF interrupt is generated.
11 —	Reserved
10 CRCAIE	CRC and ECC interrupt enable for flash memory A 0b - CRCAEF interrupt is not generated. 1b - CRCAEF interrupt is generated.
9 —	Reserved
8 PPWIE	Page-program wait interrupt flag This field indicates page-program wait interrupt. 0b - No PPWIE interrupt is generated 1b - PPWIE interrupt is generated
7 —	Reserved
6 IPIEIE	IP command trigger during IP access error interrupt enable flag This field indicates IP command trigger during IP access error interrupt enable flag. 0b - No IPIEF interrupt is generated 1b - IPIEF interrupt is generated

Table continues on the next page...

Table continued from the previous page...

Field	Function
5 —	Reserved
4 —	Reserved
3-1 —	Reserved
0 TFIE	Transaction finished interrupt enable flag This field indicates the transaction finished interrupt enable flag. 0b - No TFF interrupt is generated. 1b - TFF interrupt is generated.

79.13.2.32 Sequence Pointer Clear Register (SPTRCLR)

Offset

Register	Offset
SPTRCLR	16Ch

Function

This register provides fields to reset the IP and buffer sequence pointers. The sequence pointer contains the index of instructions within the LUT entry that is to be executed next. For example, if the LUT entry ends on a JMP_ON_CS value of 2, the index is stored as 2.

The software should reset the sequence pointers defined by JMP_ON_CS operand whenever the sequence ID is required to be changed by updating the SEQID field in the IPCR or BFGENCR.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0						Reserved		0				Reserved	Reserved	PREFETC...	ABRT_CLR
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0							0	0							0
W								IPPTRC								BFPTRC
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-26 —	Reserved
25-24 —	Reserved
23-20 —	Reserved
19 —	Reserved
18 —	Reserved
17 PREFETCH_DIS	Prefetch disable This field should be configured to disable the prefetch mechanism of receiver. It is not based on dynamic programming. Therefore, it should be programmed initially at once. When this field is set, then during an ongoing flash memory read, any subsequent AHB read is checked for buffer hit. However, after the end of flash memory read, as soon as chip select is deasserted, any subsequent AHB read results in flushing of the current AHB buffer data and issues fresh flash memory transaction if the AHB buffer data is not updated with the flash memory.
16 ABRT_CLR	Flash memory Abort/AHB buffer clear This is a dynamic field. Writing a 1 to it, irrespective of the prefetch disable, clears the AHB buffer pointers and also aborts any ongoing flash memory transaction (if any) and rejects any ongoing AHB read with an error response (if any). QuadSPI sets this field to 0 after clearing the AHB buffer pointers.
15-9 —	Reserved
8 IPPTRC	IP pointer clear This is a self-clearing field. 1b - Clears the sequence pointer for IP accesses as defined in IPCR.
7-1 —	Reserved
0 BFPTRC	Buffer pointer clear This is a self-clearing field. 1b - Clears the sequence pointer for AHB read accesses as defined in BFGENCR.

79.13.2.33 Serial Flash Memory A1 Top Address Register (SFA1AD)

Offset

Register	Offset
SFA1AD	180h

Function

This register provides the address mapping for serial flash memory A1. The difference between SFA1AD[TPADA1] and AMBA_BASE defines the size of the memory map for serial flash memory A1.

Special write-access is permitted if:

- SR[IP_ACC] = 0
- SR[AHB_ACC] = 0

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	TPADA1															
Reset	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TPADA1								0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-10 TPADA1	Top address for serial flash memory A1 In effect, TPADxx is the first location of the next memory.
9-0 —	Reserved

79.13.2.34 Serial Flash Memory A2 Top Address Register (SFA2AD)

Offset

Register	Offset
SFA2AD	184h

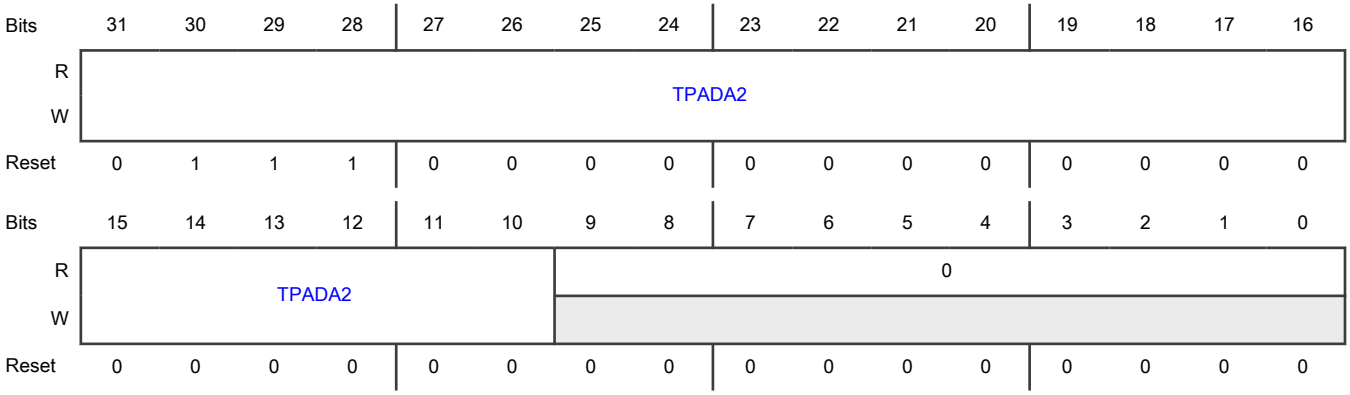
Function

This register provides the address mapping for serial flash memory A2. The difference between SFA2AD[TPADA2] and SFA1AD[TPADA1] defines the size of the memory map for serial flash memory A2.

Special write-access is permitted if:

- SR[IP_ACC] = 0
- SR[AHB_ACC] = 0

Diagram



Fields

Field	Function
31-10 TPADA2	Top address for serial flash memory A2 In effect, TPxxAD is the first location of the next memory.
9-0 —	Reserved

79.13.2.35 Data Learn Pattern Register (DLPR)

Offset

Register	Offset
DLPR	190h

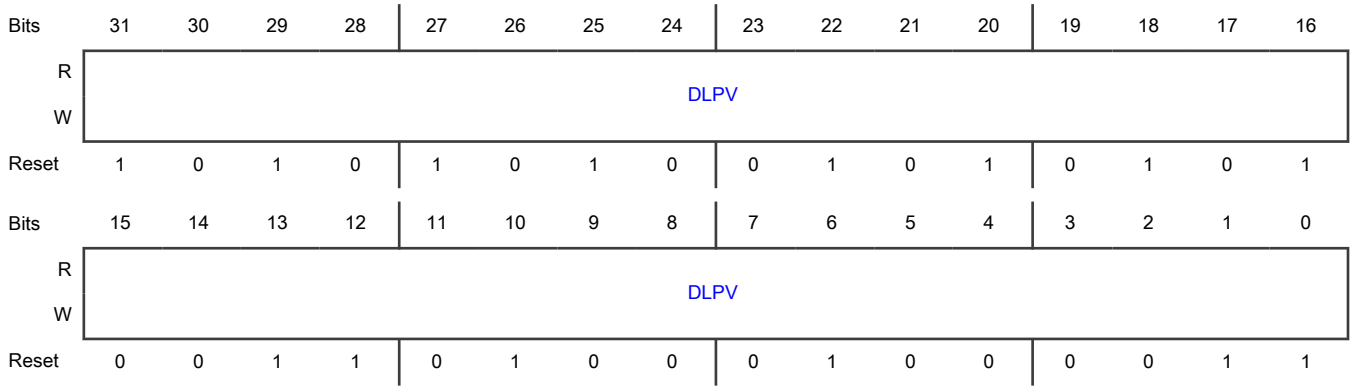
Function

This register contains the information of the data to be used for data learning.

Special write-access is permitted if:

- SR[IP_ACC] = 0
- SR[AHB_ACC] = 0

Diagram



Fields

Field	Function
31-0 DLPV	Data learning pattern value This value is used for data learning in the DDR and DQS modes. For example, if you request for more than 32 bits of data learning, the value is repeated in the register. For example, if 64-bit data learning is requested by any flash memory and the value of DLPR is aa55_3443, then the 64-bit value is aa55_3443_aa55_3443. This value is used for data learning in SDR/DDR modes along with DATA_LEARN instruction. Different patterns can be matched at selected IOs. That is, IO1 and IO3 using DLCR[DLP_SEL_FA], DLPR[31:16] for IO3, and DLPR[15:0] for IO1. DLP pattern for IO1 and IO3 should be stored, bit-wise, in the little endian format. For details, see Data learning.

79.13.2.36 Flash Memory A Failing Address Status Register (FAILA_ADDR)

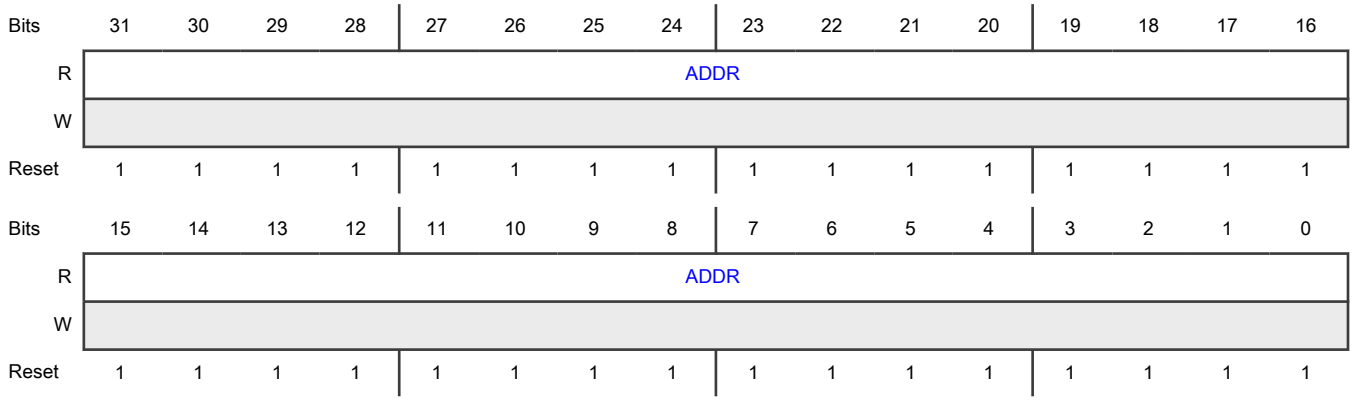
Offset

Register	Offset
FAILA_ADDR	194h

Function

This register provides the flash memory address where data learning or parity error has occurred.

Diagram



Fields

Field	Function
31-0 ADDR	Failing address for flash memory A

79.13.2.37 RX Buffer Data Register (RBDR0 - RBDR31)

Offset

For a = 0 to 31:

Register	Offset
RBDRa	200h + (a × 4h)

Function

These registers provide access to individual entries in the RX buffer. See Table 732 for the byte ordering scheme.

RBDR0 corresponds to the actual position of the read pointer within the RX buffer. The number of valid entries available depends on the number of RX buffer entries implemented and on the number of valid buffer entries available in the RX buffer.

Example 1 - RX buffer filled completely with 32 words: In this case, the address range for valid read access extends from RBDR0 to RBDR31 RBDR63.

Example 2 - RX buffer filled with five valid words: RX buffer fill level of RBSR[RDBFL] is 5. In this case, access to RBDR4 provides the last valid entry.

Any access beyond the range of valid RX buffer entries provides undefined results.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RXDATA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXDATA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 RXDATA	RX data This field contains the data associated with the related RX buffer entry. For data format and byte ordering, see Byte ordering of serial flash memory read data.

79.13.2.38 LUT Key Register (LUTKEY)

Offset

Register	Offset
LUTKEY	300h

Function

This register contains the key to lock and unlock the LUT. See LUT for details.

Special write-access is permitted in the "Anytime" mode.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	KEY															
W																
Reset	0	1	0	1	1	0	1	0	1	1	1	1	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	KEY															
W																
Reset	0	1	0	1	1	0	1	0	1	1	1	1	0	0	0	0

Fields

Field	Function
31-0 KEY	Key to lock or unlock the LUT The key is 0x5AF05AF0 and the read value is always 0x5AF05AF0.

79.13.2.39 LUT Lock Configuration Register (LCKCR)

Offset

Register	Offset
LCKCR	304h

Function

This register is used along with the LUTKEY register to lock or unlock the LUT. This register should be written immediately after the LUTKEY register for the lock or unlock operation to be successful. See LUT for details. Setting both the LOCK and UNLOCK bits as "00" or "11" is not allowed.

Special write access is permitted after writing the LUT key register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0														UNLOCK	LOCK
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Fields

Field	Function
31-2 —	Reserved
1 UNLOCK	Unlock LUT Unlocks the LUT when the following two conditions are met: • This register is written just after the LUT Key Register (LUTKEY). • The LUT key register was written with the 0x5AF05AF0 key.

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 LOCK	Lock LUT Locks the LUT when the following conditions are met: • This register is written just after the LUT Key Register (LUTKEY). • The LUT key register is written with the 0x5AF05AF0 key.

79.13.2.40 LUT Register (LUT0)

Offset

Register	Offset
LUT0	310h

Function

A sequence of instruction-operand pairs may be pre-populated in the LUT according to the device connected on board. Each instruction-operand pair is of 16 bits (2 bytes) each. Every sequence preprogrammed by Program Sequence Engine in the LUT is referred to by its index. The LUT registers act as lookup tables for sequences of instructions. The programmable sequence engine executes the instructions in these sequences to generate a valid serial flash memory transaction. There are a total of 80 LUT registers. These 80 registers are divided into groups of 5 registers that make a valid sequence. Therefore, LUT[0], LUT[5], LUT[10] ..... LUT[75] are the starting registers of a valid sequence. Each of these sets of 5 registers can have a maximum of 10 instructions. Reset value of the register shown below is only applicable to LUT2 to LUT79. A maximum of 16 sequences can be defined at one time. See LUT that describes the LUT registers in detail.

Special write-access is permitted if the LUT is unlocked.

This register is access controlled and can only be programmed by privilege masters. Last six LUT SEQID locations should be used for programming SEQID for atomic commands (where flash's internal config region is accessed) like flash erase etc. where data transfer is not required. These 6 SEQID can be send to QSPI using only FRAD0 and can't be qualified using any other FRAD. For details, see Atomic commands considerations.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	INSTR1						PAD1		OPRND1							
Reset	0	0	0	0	1	0	0	0	0	0	0	1	1	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	INSTR0						PAD0		OPRND0							
Reset	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	1

Fields

Field	Function
31-26 INSTR1	Instruction 1
25-24 PAD1	Pad information for INSTR1 00b - 1 Pad 01b - 2 Pads 10b - 4 Pads 11b - 8 Pads
23-16 OPRND1	Operand for INSTR1
15-10 INSTR0	Instruction 0
9-8 PAD0	Pad information for INSTR0 00b - 1 Pad 01b - 2 Pads 10b - 4 Pads 11b - 8 Pads
7-0 OPRND0	Operand for INSTR0

79.13.2.41 LUT Register (LUT1)

Offset

Register	Offset
LUT1	314h

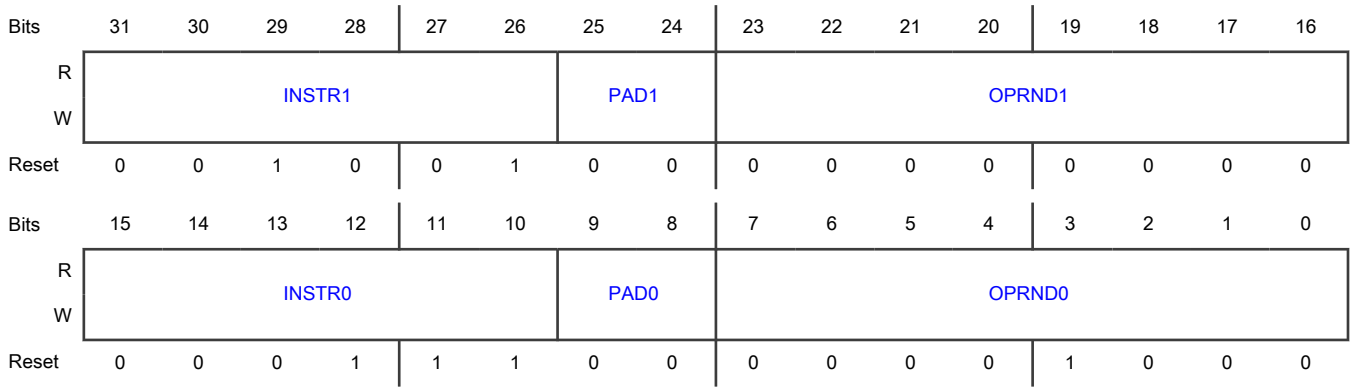
Function

A sequence of instruction-operand pairs may be pre-populated in the LUT according to the device connected on board. Each instruction-operand pair is of 16 bits (2 bytes) each. Every sequence preprogrammed by Program Sequence Engine in the LUT is referred to by its index. The LUT registers act as lookup tables for sequences of instructions. The programmable sequence engine executes the instructions in these sequences to generate a valid serial flash memory transaction. There are a total of 80 LUT registers. These 80 registers are divided into groups of 5 registers that make a valid sequence. Therefore, LUT[0], LUT[5], LUT[10] ..... LUT[75] are the starting registers of a valid sequence. Each of these sets of 5 registers can have a maximum of 10 instructions. Reset value of the register shown below is only applicable to LUT2 to LUT79. A maximum of 16 sequences can be defined at one time. See LUT that describes the LUT registers in detail.

Special write-access is permitted if the LUT is unlocked.

This register is access controlled and can only be programmed by privilege masters. Last six LUT SEQID locations should be used for programming SEQID for atomic commands (where flash's internal config region is accessed) like flash erase etc. where data transfer is not required. These 6 SEQID can be send to QSPI using only FRAD0 and can't be qualified using any other FRAD. For details, see Atomic commands considerations.

Diagram



Fields

Field	Function
31-26 INSTR1	Instruction 1
25-24 PAD1	Pad information for INSTR1 00b - 1 Pad 01b - 2 Pads 10b - 4 Pads 11b - 8 Pads
23-16 OPRND1	Operand for INSTR1
15-10 INSTR0	Instruction 0
9-8 PAD0	Pad information for INSTR0 00b - 1 Pad 01b - 2 Pads 10b - 4 Pads 11b - 8 Pads
7-0 OPRND0	Operand for INSTR0

79.13.2.42 LUT Register (LUT2 - LUT79)

Offset

For a = 2 to 79:

Register	Offset
LUTa	310h + (a × 4h)

Function

A sequence of instruction-operand pairs may be pre-populated in the LUT according to the device connected on board. Each instruction-operand pair is of 16 bits (2 bytes) each. Every sequence preprogrammed by Program Sequence Engine in the LUT is referred to by its index. The LUT registers act as lookup tables for sequences of instructions. The programmable sequence engine executes the instructions in these sequences to generate a valid serial flash memory transaction. There are a total of 80 LUT registers. These 80 registers are divided into groups of 5 registers that make a valid sequence. Therefore, LUT[0], LUT[5], LUT[10] ..... LUT[75] are the starting registers of a valid sequence. Each of these sets of 5 registers can have a maximum of 10 instructions. Reset value of the register shown below is only applicable to LUT2 to LUT79. A maximum of 16 sequences can be defined at one time. See LUT that describes the LUT registers in detail.

Special write-access is permitted if the LUT is unlocked.

This register is access controlled and can only be programmed by privilege masters. Last six LUT SEQID locations should be used for programming SEQID for atomic commands (where flash's internal config region is accessed) like flash erase etc. where data transfer is not required. These 6 SEQID can be send to QSPI using only FRAD0 and can't be qualified using any other FRAD. For details, see Atomic commands considerations.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	INSTR1						PAD1		OPRND1							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	INSTR0						PAD0		OPRND0							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-26 INSTR1	Instruction 1
25-24 PAD1	Pad information for INSTR1 00b - 1 Pad

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01b - 2 Pads 10b - 4 Pads 11b - 8 Pads
23-16 OPRND1	Operand for INSTR1
15-10 INSTR0	Instruction 0
9-8 PAD0	Pad information for INSTR0 00b - 1 Pad 01b - 2 Pads 10b - 4 Pads 11b - 8 Pads
7-0 OPRND0	Operand for INSTR0

79.13.2.43 Flash Region Start Address (FRAD0_WORD0 - FRAD7_WORD0)

Offset

For n = 0 to 7:

Register	Offset
FRADn_WORD0	800h + (n × 20h)

Function

This register is access controlled and can only be programmed by privilege masters.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	STARTADR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 STARTADR	Start Address Specifies the specific flash memory region starting address (around 64 KB boundary)
15-0 —	Reserved

79.13.2.44 Flash Region End Address (FRAD0_WORD1 - FRAD7_WORD1)

Offset

For n = 0 to 7:

Register	Offset
FRADn_WORD1	804h + (n × 20h)

Function

This register is access controlled and can only be programmed by privilege masters.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ENDADR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-16 ENDADR	End Address Specifies the specific flash memory region end address (around 64 KB boundary)
15-0 —	Reserved

79.13.2.45 Flash Region Privileges (FRAD0_WORD2 - FRAD7_WORD2)

Offset

For n = 0 to 7:

Register	Offset
FRADn_WORD2	808h + (n × 20h)

Function
This register is access controlled and can only be programmed by privilege masters.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		EALO						0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0										MD1ACP		MD0ACP			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29-24 EALO	Exclusive Access Lock Owner When FRAD*n*_WORD3[EAL] = 11, this field indicates the domain/master ID that owns the exclusive access lock.
23-6	Reserved

Table continues on the next page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>—</td><td></td></tr>
<tr><td>5-3: MD1ACP
2-0: MD0ACP</td><td>Master Domain Access Control Policy
This field define the access restrictions for respective Master Domain 0 and 1 corresponding to this FRAD region. Write permissions are decided based on secure and privilege attributes of current transaction. Read access is not restricted. This field can not be programmed if MDAD registers are not valid. If EAL is being written in same cycle (10 or 11), or if FRAD_WORD3 [LOCK] bits are written as 11. If LOCK bits are written as 10 then this field can be programmed only if Master ID matches the ID check of respective MDAD queue. If MDAD descriptor is not valid then respective MDxACP also becomes read-only.
Table 767. Field value mapping
<table>
<tr><th>Policy</th><th>Secure privilege</th><th>Secure user</th><th>Non secure privilege</th><th>Non secure user</th></tr>
<tr><td>111</td><td>Read/Write</td><td>Read</td><td>Read/Write</td><td>Read</td></tr>
<tr><td>110</td><td>Read/Write</td><td>Read/Write</td><td>Read/Write</td><td>Read/Write</td></tr>
<tr><td>101</td><td>Read/Write</td><td>Read</td><td>Read</td><td>Read</td></tr>
<tr><td>100</td><td>Read/Write</td><td>Read/Write</td><td>Read</td><td>Read</td></tr>
<tr><td>0xx</td><td>Read</td><td>Read</td><td>Read</td><td>Read</td></tr>
</table>
NOTE
The software must program the value 101 for MDACP fields for FRAD0 as this FRAD has access to all the atomic instructions. For details, see Atomic commands considerations.</td></tr>
</table>

79.13.2.46 Flash Region Lock Control (FRAD0_WORD3 - FRAD7_WORD3)

Offset

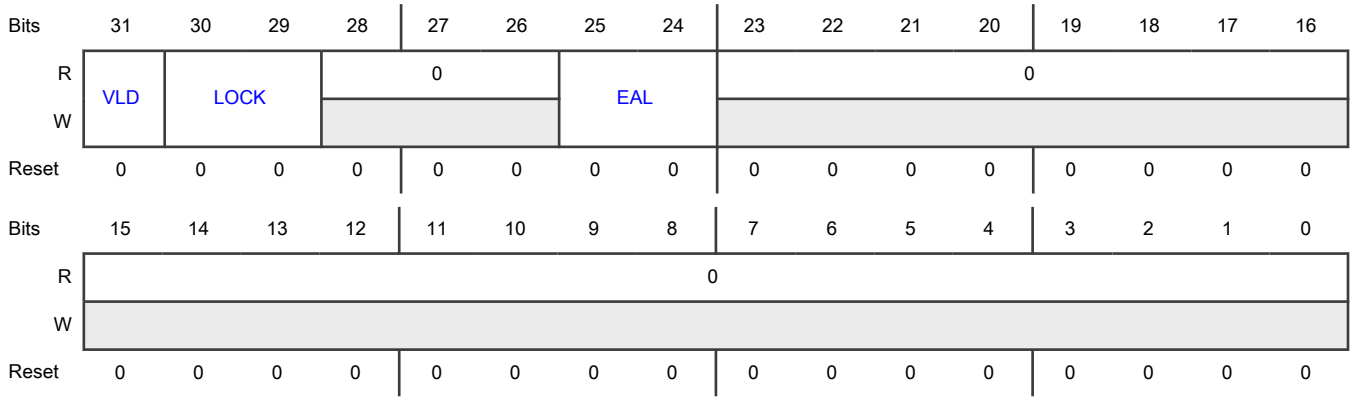
For n = 0 to 7:

Register	Offset
FRADn_WORD3	80Ch + (n × 20h)

Function

This register is access controlled and can only be programmed by privilege masters.

Diagram



Fields

Field	Function
31 VLD	Valid This field indicates whether the FRAD Descriptor for a specific flash region is valid. This field can not be written in same cycle if EAL field is being written as 10 or 11. **NOTE** You must enable this field after programming FRAD0_WORD0, FRAD0_WORD1, FRAD0_WORD2 and FRAD0_WORD3 registers. 0b - FRAD-Assignment is invalid 1b - FRAD-Assignment is valid
30-29 LOCK	Descriptor Lock This field enables masking of accidental write on FRAD registers. Lock is enabled/disabled by Secure/ Privileged master. Lock functionality does not affects the EAL bits of FRAD. They can still be written even if FRAD descriptors are locked. 00b-01b - Lock disabled. Descriptor registers can be written by any master 10b - Lock enabled. Descriptors are read-only. MDnACP fields can be programmed only by the master with ID matching the MID check of their respective Target group MDAD. If the Target group MDAD is not valid then MDnACP fields also become read-only. 11b - Lock enabled. Descriptor registers are read-only.
28-26 —	Reserved
25-24 EAL	Exclusive Access Lock This field provides exclusive write lock over a FRAD region based on MD*n*ACP. 00b - No lock. Write permissions available for all masters based on their MDxACP evaluation. 01b - NA

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10b - Lock enabled. Write permissions revoked for all domains. It can only be changed to 11b and can't directly be changed to 00b. 11b - Lock enabled. Exclusive write permission for master with master ID given in FRADn_WORD3[EALO] fields based on its MDxACP evaluation. Write disabled for other masters. Exclusive lock can be taken by a master only if it its ID matches the master ID check of any of the MDAD target queue, which is valid. MDxACP field corresponding to that target queue is not programmed as 0b. when a master writes 11b on this field it will be written only if above condition is met and then the master ID will be stored in FRADn_WORD3[EALO]. No transfer error will be generated in case the master does not satisfies the check.
23-0 —	Reserved

79.13.2.47 Flash Region Compare Address Status (FRAD0_WORD4 - FRAD7_WORD4)

Offset

For n = 0 to 7:

Register	Offset
FRADn_WORD4	810h + (n × 20h)

Function

This register is access controlled and can only be programmed by privilege masters.

NOTE

Any write access to these registers will result in bus transfer error

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CMP_ADDR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CMP_ADDR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 CMP_ADDR	Capture Address This field indicates the 32-bit start address of the last transaction which lies inside the address range of this FRAD descriptor.

79.13.2.48 Flash Region Compare Status Data (FRAD0_WORD5 - FRAD7_WORD5)

Offset

For n = 0 to 7:

Register	Offset
FRADn_WORD5	814h + (n × 20h)

Function

This register is access controlled and can only be programmed by privilege masters.

NOTE

Any write access to these registers will result in bus transfer error

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	CMPVALID	CMP_ERR	0												
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								CMP_PA	CMP_SA	CMP_MDID					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30 CMPVALID	Comparison Valid This field indicates the validity of flash region specific comparison check.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Access result/status not available 1b - Access result/status is available
29 CMP_ERR	Comparison Error This field indicates the error status (based on secure/privilege attributes checked with MDxACP permissions or exclusive access lock as programmed in FRAD) for flash region specific comparison check for last transaction. When this field is set, it can be cleared by writing '1' to W1C field of ERRSTAT register. It will update new value only after the error field has been cleared. 0b - No error 1b - Access error
28-8 —	Reserved
7 CMP_PA	Capture Privilege Attribute Capture Privilege attribute of last transaction which passed the address check for this FRAD. 0b - Non-privilege transaction 1b - Privilege transaction
6 CMP_SA	Capture Secure Attribute Capture Secure attribute of last transaction which passed the address check for this FRAD. 0b - Non-secure transaction 1b - Secure transaction
5-0 CMP_MDID	Capture MDID Value Capture master ID (MDID) value of the last transaction which passed the address check for this FRAD.

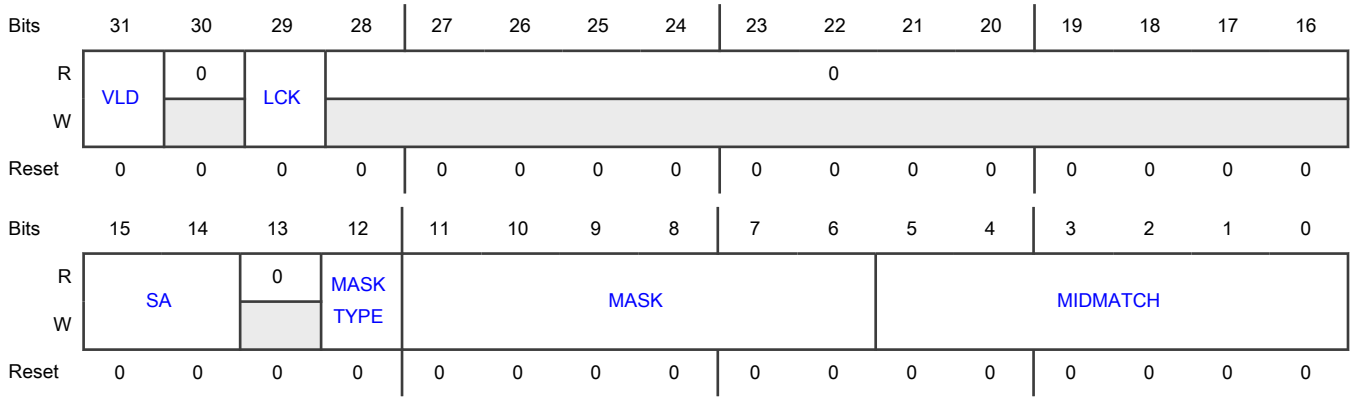
79.13.2.49 Target Group n Master Domain Access Descriptor (TG0MDAD - TG1MDAD)

Offset

Register	Offset
TG0MDAD	900h
TG1MDAD	910h

Function
This register is access controlled and can only be programmed by privilege masters.

Diagram



Fields

Field	Function
31 VLD	Valid Indicates whether MDAD Descriptor for the target group *n* is valid. 0b - MDAD-Assignment is invalid 1b - MDAD-Assignment is valid
30 —	Reserved
29 LCK	Descriptor Lock This field provides a means to make the MDAD descriptor read-only. 0b - Lock disabled. Registers can be written. 1b - Lock enabled. Registers are read-only.
28-16 —	Reserved
15-14 SA	Secure Attribute Defines the secure attribute selection criteria for entry into descriptor queue.. 00b - NA. This option should not be used. Allows the bus attribute for this master to non-secure only 01b - Allow the bus attribute for this master to non-secure only 10b - Allow the bus attribute for this master to secure only 11b - Allow the bus master's attribute: Both secure and non-secure
13 —	Reserved
12	Mask Type

Table continues on the next page...

Table continued from the previous page...

Field	Function
MASKTYPE	0b - ANDed mask 1b - ORed mask
11-6 MASK	Mask Defines the 6-bit mask value for the ID-Match comparison
5-0 MIDMATCH	Master ID Reference Specifies the reference value of the Master-ID (MID) for MID-comparison

79.13.2.50 Target Group n SFAR Address (TG0SFAR - TG1SFAR)

Offset

Register	Offset
TG0SFAR	904h
TG1SFAR	914h

Function
Flash memory start address of the transaction in this target group queue.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SFARADDR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SFARADDR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 SFARADDR	SFAR Address Flash memory start address of the transaction which qualified into this target group queue.

79.13.2.51 Target Group n SFAR Status (TG0SFARS - TG1SFARS)

Offset

Register	Offset
TG0SFARS	908h
TG1SFARS	918h

Function

Target group *n* SFAR status.

NOTE

This register can only be read if the master reading the register passes the security checks set in TGnMDAD[SA] fields or if MDAD descriptor is not valid. Else this register will be read as 0. It will also read as 0 if TGnMDAD[SA] bits in descriptor are set as 00.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	VLD	ERR	CLR	0												
W			W1C													
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0			PA	0	SA	0				TG_MID					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 VLD	Valid Indicates whether the SFAR Address for the target group *n* is valid and queue is busy. This bit will not be set when ERR bit is set. 0b - SFAR-Assignment is invalid 1b - SFAR-Assignment is valid
30 ERR	Error Indicates whether the SFAR address stored by a Master is with required access attributes for this target group descriptor. 0b - SFAR with required attributes 1b - SFAR without required attributes

Table continues on the next page...

Table continued from the previous page...

Field	Function
29 CLR	Clear Clears the SFAR Status register. This register can only be cleared by the master having security attribute defined by TG*n*MDAD[SA]. After clearing if there is any valid transaction present in MDAD queue, it will again get updated in this register else it will be cleared. It will not be cleared if TGxMDAD[SA] bits are set to 00.
28-13 —	Reserved
12 PA	Privileged Attribute Privilege attribute of the master which wrote the SFAR register. 0b - Non-privileged 1b - Privileged
11 —	Reserved
10 SA	Secure Attribute Secure attribute of the master which wrote the SFAR register. 0b - Non-secure 1b - Secure
9-6 —	Reserved
5-0 TG_MID	Transaction Master ID Indicates the Master-ID of a transaction which programmed the SFAR registers with required attributes.

79.13.2.52 Target Group n IPCR Status (TG0IPCRS - TG1IPCRS)

Offset

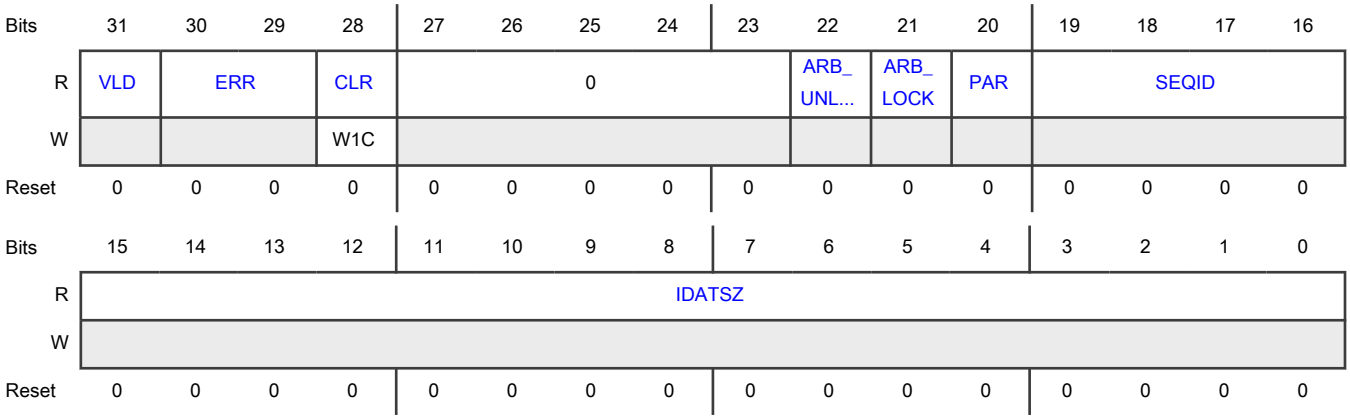
Register	Offset
TG0IPCRS	90Ch
TG1IPCRS	91Ch

Function
Target group *n* IPCR status.

NOTE

This register can only be read if the master reading the register passes the security checks set in TGnMDAD[SA] fields or if MDAD descriptor is not valid. Else this register will be read as 0. It will also read as 0 if TGnMDAD[SA] bits in descriptor are set as 00.

Diagram



Fields

Field	Function
31 VLD	Valid Indicates whether the IPCR-IDATZ, IPCR-SEQID and PAR stored in this target group queue is valid and queue is locked. This bit will not be set in case ERR is set. 0b - IPCR-assignment is invalid 1b - IPCR-assignment is valid and queue is locked.
30-29 ERR	Error Indicates whether the IPCR stored by a Master with required access attributes wrt target group MDAD 00b - IPCR programming with required attributes 01b - IPCR-DATZ programming without required attributes 10b - IPCR-SEQID programming without required attributes 11b - IPCR-DATZ and SEQID both programming without required attributes
28 CLR	Clear Clears the status in IPCR Status register. This register can only be cleared by the master having security attribute defined by TGnMDAD[SA]. After clearing if there is any valid transaction present in MDAD queue it will again get updated in this register else it will be cleared. It will not be cleared if TGxMDAD[SA] bits are set to 00.
27-23 —	Reserved
22	Arbitration Unlock

Table continues on the next page...

Table continued from the previous page...

Field	Function
ARB_UNLOCK	Specifies if the arbitration unlock is requested for TGn. 0b - No effect 1b - Arbitration unlock is requested
21 ARB_LOCK	Arbitration Lock Specifies if the arbitration lock is requested for TGn. 0b - No effect 1b - Arbitration lock is requested
20 PAR	Parallel Mode Enable Value Indicates the parallel mode enable value programmed in the target group queue.
19-16 SEQID	SEQID Value Indicates the SEQID value programmed in target group queue.
15-0 IDATSZ	IDATSZ Value Indicates the IDATSZ value programmed in target group queue.

79.13.2.53 Master Global Configuration (MGC)

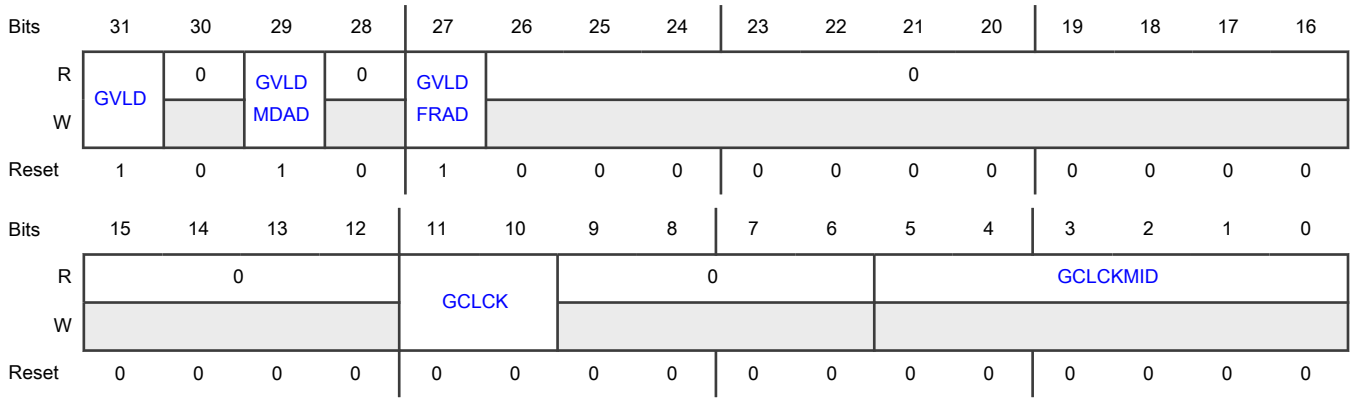
Offset

Register	Offset
MGC	920h

Function

This register is access controlled. It can only be programmed by master with master ID equal to 000011b. For MGC, MRC and MTO registers access checks master ID is being used. For transaction level checks inside MDAD and FRAD domain IDs are being used.

Diagram



Fields

Field	Function
31 GVLD	Global Valid access control 0b - Access controls are disabled. No descriptor comparison check. All transactions are allowed. 1b - Access controls are enabled.
30 —	Reserved
29 GVLDMDAD	Global Valid MDAD 0b - MDADs are disabled 1b - MDADs are enabled
28 —	Reserved
27 GVLDFRAD	Global Valid FRAD 0b - FRADs are disabled 1b - FRADs are enabled
26-12 —	Reserved
11-10 GCLCK	Global Configuration Lock This field provides a mechanism to limit write access to descriptors specific registers (MGC, MRC and MTO registers). 00b - Global Lock disabled. Registers can be written based on individual register access attribute. 01b - NA 10b - Lock enabled. Only the global configuration lock owner can write to the registers. 11b - Lock enabled. All registers are read only until unlocked
9-6 —	Reserved
5-0 GCLCKMID	Global configuration Lock Owner Status This fields specifies the 6-bit Master-ID of Global configuration Lock owner (which set MGC[GCLCK]=10). Input value of Global Master-ID is provided by sideband signals.

79.13.2.54 Master Read Command (MRC)

Offset

Register	Offset
MRC	924h

Function

This register is access controlled. It can only be programmed by master with master ID equal to 000011b.

NOTE

In case of Sequence Auto Join instruction in LUT, SFP will parse only first sequence. Second sequence will not be parsed for checking Read commands.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	VLDCMD03	READ_CMD3						0	VLDCMD02	READ_CMD2					
W																
Reset	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		READ_CMD1						0		READ_CMD0					
W																
Reset	0	0	0	0	1	1	1	0	0	0	0	0	0	1	1	1

Fields

Field	Function
31 —	Reserved
30 VLDCMD03	Valid command 0b - READ_CMD3 value invalid 1b - READ_CMD3 value valid
29-24 READ_CMD3	Read Command 3 Programmes the additional READ Instruction/Command encoding information, if applicable
23 —	Reserved
22 VLDCMD02	Valid command 0b - READ_CMD2 value invalid 1b - READ_CMD2 value valid
21-16 READ_CMD2	Read Command 2 Stores the existing instruction—DATA_LEARN command encoding information. Can be overwritten with additional read command, if required.
15-14 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
13-8 READ_CMD1	Read Command 1 Stores the existing instruction—READ_DDR command encoding information.
7-6 —	Reserved
5-0 READ_CMD0	Read Command 0 Stores the existing instruction—READ command encoding information.

79.13.2.55 Master Timeout (MTO)

Offset

Register	Offset
MTO	928h

Function
This register is access controlled. It can only be programmed by master with master ID equal to 000011b.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	WRITE_TO															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	WRITE_TO															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-0 WRITE_TO	Write Timeout Maximum timeout value to abort the ongoing write or read command. The timeout counter starts after the access from any target queue has won the arbitration and QSPI is IDLE (FSM_STAT state field is not 00). SFP queue is cleared once this timeout value has been reached and interrupt is generated if enabled.

79.13.2.56 FlashSeq Request (FLSEQREQ)

Offset

Register	Offset
FLSEQREQ	92Ch

Function
This register is access controlled and can only be programmed by privilege masters.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	VLD	0	CLR	0	TIMEOUT	0			0	CMD	0		SEQID			
W			W1C													
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	FRAD			0	ARB_LOCK	PA	SA	0	REQ_TG	REQ_MID					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 VLD	Valid Indicates whether the FlashSeq request status is valid 0b - Status is invalid 1b - Status is valid
30 —	Reserved
29 CLR	Clear Clears the status
28 —	Reserved
27 TIMEOUT	Timeout Error Status Timeout error status of last executed FlashSeq request.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Instruction completed without timeout error 1b - Instruction aborted after timeout error
26-25 —	Reserved
24-23 —	Reserved
22 CMD	Instruction Type Instruction type of last executed FlashSeq request. 0b - Read Instruction Sequence 1b - Non-Read Instruction Sequence
21-20 —	Reserved
19-16 SEQID	Sequence ID Sequence ID of last executed FlashSeq request.
15 —	Reserved
14-12 FRAD	Flash Region Descriptor Number Flash Region specific number (FRAD) of last executed FlashSeq request. This will remain 0 in case of read transfers.
11 —	Reserved
10 ARB_LOCK	Arbitration Lock Arbitration lock status for last completed request. 0b - Arbitration was not locked 1b - Arbitration was locked
9 PA	Privilege Attribute Privilege attribute of last executed FlashSeq request. 0b - Non-privilege Transaction 1b - Privilege Transaction
8	Secure Attribute

Table continues on the next page...

Table continued from the previous page...

Field	Function
SA	Secure attribute of last executed FlashSeq request. 0b - Non-secure Transaction 1b - Secure Transaction
7 —	Reserved
6 REQ_TG	FlashSeq Request Target Group Target group queue from which last executed transaction passed. 0b - TG0 1b - TG1
5-0 REQ_MID	FlashSeq Request Master ID Indicates the master-ID of last executed FlashSeq request.

79.13.2.57 FSM Status (FSMSTAT)

Offset

Register	Offset
FSMSTAT	930h

Function
This register is access controlled and can only be programmed by privilege masters.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	VLD	0													ARB_LOCK	CMD
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		MID						0						STATE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 VLD	Valid Indicates whether the FSM Status is valid. 0b - Status is invalid. No IPS transfer is queued. 1b - Status is valid. IPS transfer is queued or execution on QuadSPI.
30-18 —	Reserved
17 ARB_LOCK	Arbitration Lock Arbitration lock status for present request. 0b - Arbitration not locked 1b - Arbitration locked
16 CMD	Command Instruction type of currently initiated Flash transaction on QuadSPI. 0b - Read instruction sequence 1b - Non-read instruction sequence
15-14 —	Reserved
13-8 MID	Master ID Indicates the Master-ID of the currently initiated FlashSeq transaction on QuadSPI.
7-2 —	Reserved
1-0 STATE	FSM State Status 00b - Transaction is Queued, but QuadSPI is busy with AHB transfer, any previous DMA transaction is ongoing, any residue data left in RDBFL or if any interrupt is pending.. 01b - TBDR lock is open. IPS master can write in TBDR. 10b - Write transfer is triggered. SEQID is written to QuadSPI. 11b - Read transfer is triggered. SEQID is written to QuadSPI.

79.13.2.58 IPS Error (IPSERROR)

Offset

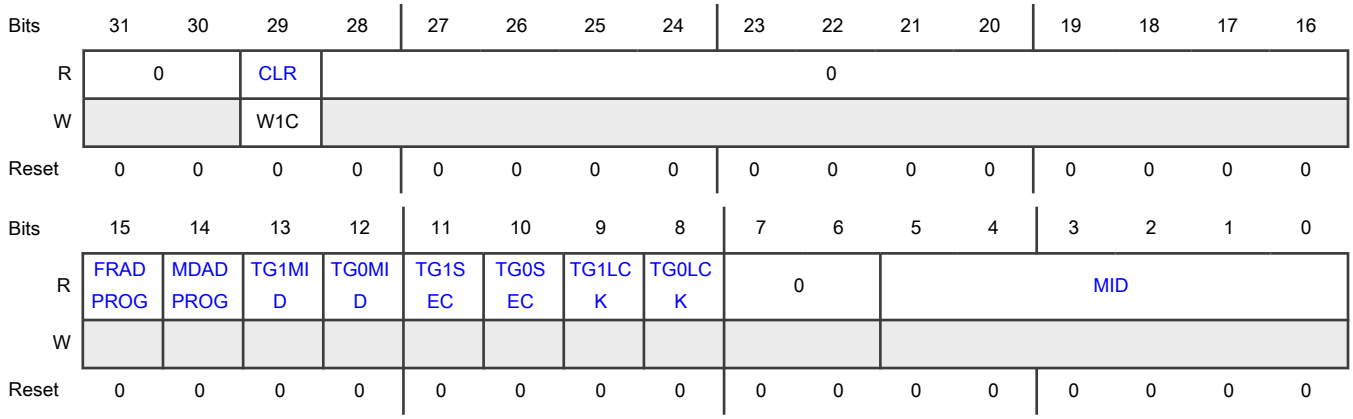
Register	Offset
IPSERROR	934h

Function

This register contains the error status of IPS write on SFAR or IPCR registers if they are not qualified for any of the target group queues.

This register is access controlled and can only be programmed by privilege masters.

Diagram



Fields

Field	Function
31-30 —	Reserved
29 CLR	Clear Clear the status of IPS Error register.
28-16 —	Reserved
15 FRADPROG	FRAD Descriptor Program Status 0b - Some or all of the FRAD descriptors are programmed 1b - None of the FRAD descriptors are programmed
14 MDADPROG	TG/MDAD Descriptor Program Status 0b - One or both of target group descriptors programmed 1b - None of the target group descriptors are programmed and valid
13-12 TGnMID	TGn Master-ID Status 0b - TGn master-ID check passed 1b - TGn master-ID check failed
11-10 TGnSEC	TGn Security Status 0b - Security attribute check passed for TGn

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Security attribute check failed for TGn
9-8 TGnLCK	TGn Lock 0b - TGn queue SEQID is not written yet. 1b - TGn queue SEQID is written and queue is locked
7-6 —	Reserved
5-0 MID	IPS DID Master ID IPS Master ID for the transaction which generated this IPS transfer error.

79.13.2.59 Error Status (ERRSTAT)

Offset

Register	Offset
ERRSTAT	938h

Function
This register is access controlled and can only be programmed by privilege masters.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	TO_ERR	TG1IPCR	TG0IPCR	TG1SFAR	TG0SFAR	IPS_ERR	FRAD7ACC	FRAD6ACC	FRAD5ACC	FRAD4ACC	FRAD3ACC	FRAD2ACC	FRAD1ACC	FRAD0ACC	FRADMTCH
W								W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-15	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
14 TO_ERR	Timeout Error This field is set if any flash transaction generated on QuadSPI results in timeout error and is aborted by SFP module. This field can be cleared by writing 1 to FLSEQREQ[CLR]. 0b - No timeout Error generated 1b - Timeout error is generated
13-12 TGnIPCR	TGn IPCR Error This field is set if any IPCR write generates error while being written to Target group queue. This field can be cleared by writing 1 to TG*n*IPCRS[CLR]. 0b - No Error generated 1b - Error is generated
11-10 TGnSFAR	TGn SFAR Error This field is set if any SFAR write generates error while being written to Target group queue. This field can be cleared by writing 1 to TG*n*SFARS[CLR]. 0b - No Error generated 1b - Error is generated
9 IPS_ERR	IPS Error Some common error occurred and IPS bus transfer error is also generated. The details of the IPS error can be found in IPSERROR register. This field can be cleared by writing 1 to IPSERROR[CLR]. 0b - No Error generated 1b - Error is generated
8-1 FRADnACC	FRADn Access Error This bit is set when the transaction address lies within the address range of this FRAD but it does not qualify the access permission checks for this FRAD or this FRAD was under exclusive lock (FRAD*n*_WORD3[EAL] = 10 or 11) by another master. It may also be set if the transaction address qualifies for multiple FRAD regions. 0b - No valid Error transaction 1b - Transaction is with error and target queue is cleared
0 FRADMTCH	No FRAD Match Error Transaction address does not lie within address range of any FRAD descriptor. 0b - No Error generated

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Transaction does not lie within any FRAD address range and error is generated

79.13.2.60 Interrupt Enable (INT_EN)

Offset

Register	Offset
INT_EN	93Ch

Function

This register is access controlled and can only be programmed by privilege masters.

NOTE

In case Queue Specific Error Interrupt bits (TG0SFAR,TG1SFAR,TG0IPCR,TG1IPCR) are enabled, ensure that the core executing interrupt handler can clear the Queue specific TGxSFARS and TGxIPCRS registers only when core's security attribute matches the queue. In case there is a mismatch in security attribute, you must change the TGMDADx.SA attribute of that queue to match the core SA attribute. After that, core can clear the error and then revert the changes in TGMDADx.SA field.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	TO_ERR	TG1IPCR	TG0IPCR	TG1SFAR	TG0SFAR	IPS_ERR	FRAD7ACC	FRAD6ACC	FRAD5ACC	FRAD4ACC	FRAD3ACC	FRAD2ACC	FRAD1ACC	FRAD0ACC	FRADMTCH
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-15 —	Reserved
14 TO_ERR	Timeout Error Interrupt Enable 0b - Interrupt disabled 1b - Interrupt enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
13-12 TGnIPCR	TGn IPCR Error Interrupt Enable 0b - Interrupt disabled 1b - Interrupt enabled
11-10 TGnSFAR	TGn SFAR Error Interrupt Enable 0b - Interrupt disabled 1b - Interrupt enabled
9 IPS_ERR	IPS Error Interrupt Enable 0b - Interrupt disabled 1b - Interrupt enabled
8-1 FRADnACC	FRADn Access Error Interrupt Enable 0b - Interrupt disabled 1b - Interrupt enabled
0 FRADMTCH	No FRAD Match Error Interrupt Enable 0b - Interrupt disabled 1b - Interrupt enabled

79.13.3 Serial flash memory address assignment

The serial flash memory address assignment can be modified by writing into Serial Flash Memory A1 Top Address Register (SFA1AD) and Serial Flash Memory A2 Top Address Register (SFA2AD) for device A

Table 768. Serial flash memory address assignment

Parameter	Function	Access mode
QuadSPI_AMBA_BASE ((31:10) - 22 bits)	QuadSPI AHB base address First address of the serial flash memory device as presented to the QuadSPI controller. This might be the base of the serial flash memory in the system address map or it may be a remapping, for instance to 0h, performed by the system. (See the system address map file attached to this document)	
QuadSPI_ARDB_BASE	First address of the QuadSPI Rx buffer on system memory map	

Table continues on the next page...

Table 768. Serial flash memory address assignment (continued)

Parameter	Function	Access mode
TOP_ADDR_MEMA1(TPADA1)	Top address for the external flash memory A1 (the first of the two independent flash memories sharing the IOFA)	Any access to the address space between TOP_ADDR_MEMA1 and QuadSPI_AMBA_BASE is routed to serial flash memory A1.
TOP_ADDR_MEMA2(TPADA2)	Top address for the external flash memory A2 (the second of the two independent flash memories sharing the IOFA)	Any access to the address space between TOP_ADDR_MEMA2 and TOP_ADDR_MEMA1 is routed to serial flash memory A2.

79.14 Flash memory mapped AMBA bus

QuadSPI_AMBA_BASE defines the address to be used as the start address of the serial flash memory device, as defined by the system memory map. Note that this may be a remapping of the physical address of the serial flash memory in the system. See the system address map for details.

Table 769. QuadSPI AMBA bus memory map

Address	Register name
QuadSPI_AMBA_BASE to (TOP_ADDR_MEMA2 - 1h)	• See Memory-mapped serial flash memory data—individual flash memory mode on flash memory A. • For information about byte ordering, see Table 732 and Table 733.
QuadSPI_ARDB_BASE to... (32 * 4 Byte) QuadSPI_ARDB_BASE + 1FFh	• See AHB RX Data Buffer Register (ARDB0 - ARDB31). • For information about the byte ordering, see Table 732.

NOTE

Any read access to non-implemented addresses provides undefined results.

In individual flash memory modes, the 3/4 address bytes (as programmed in the instruction/operand in the sequence) available for the flash memory address are determined by SFADR[23:0] or SFADR[31:0] as provided in the table shown above.

79.14.1 AHB bus access read considerations

Note that all logic in the QuadSPI module implementing the AHB bus access is designed to read the content of an external serial flash memory device. Therefore, the following restrictions apply to the QuadSPI module with respect to accesses to the AHB bus:

- AHB bus read access types—NONSEQ and BUSY—are fully supported.
- AHB read access type—SEQ—is treated in the same way as NONSEQ. See Flash memory mapped AMBA bus for details.
- Early burst termination is not supported for AHB transactions.
- An AHB error response is provided on occurrence of:
  - Parity error
  - ECC error from flash memory
  - Data learning failure

  Also, interrupts (if enabled) are provided along with the AHB error response.

- An AHB error response occurs when FR[AITEF] bit is set
- An AHB bus stall along with error response occurs when FR[AAEF] bit is set

79.14.2 Memory-mapped serial flash memory data—individual flash memory mode on flash memory A

Starting with address QuadSPI_AMBA_BASE, the content of the first external serial flash memory device is mapped into the address space of the device containing the QuadSPI module. Serial flash memory byte address 0h corresponds to bus address, QuadSPI_AMBA_BASE, in an increasing order. . See the following table for the address mapping. The byte ordering for 32-bit access is provided in Table 732 and for 64-bit read access, the byte ordering is provided in Table 733.

Table 770. Memory-mapped individual flash memory mode—flash memory A address scheme

Memory-mapped address 32-bit access	Memory-mapped address 64-bit access	Serial flash memory byte address	Flash memory device
QuadSPI_AMBA_BASE + 0h	QuadSPI_AMBA_BASE + 0h	0h to 3h	A1
QuadSPI_AMBA_BASE + 4h		4h to 7h	
...	...	...	
TOP_ADDR_MEMA1 - 8h	TOP_ADDR_MEMA1 - 8h	(TOP_ADDR_MEMA1- 8h) to (TOP_ADDR_MEMA1 - 0h4 -0h1)	
TOP_ADDR_MEMA1 - 4h		(TOP_ADDR_MEMA1 - 4h) to (TOP_ADDR_MEMA1 - 0h1)	
TOP_ADDR_MEMA1 + 4h	TOP_ADDR_MEMA1 + 0h	0h to 3h	A2
TOP_ADDR_MEMA1 + 0h4		4h to 7h	
...	...	...	
TOP_ADDR_MEMA2 - 8h	TOP_ADDR_MEMA2 - 8h	(TOP_ADDR_MEMA2 - 8h) to (TOP_ADDR_MEMA2 - 4h - 1h)	
TOP_ADDR_MEMA2 - 4h		(TOP_ADDR_MEMA2 - 4h) to (TOP_ADDR_MEMA2 - 1h)	

The available address range depends on the size of the external serial flash memory device. Any access beyond the size of the external serial flash memory provides undefined results.

For details concerning the read process, see Flash memory read.

79.14.3 ARDB register descriptions

NOTE

See the system memory map in this document for the base address of the QuadSPI AHB RX data buffer.

79.14.3.1 ARDB memory map

QuadSPI_S32K358_ARDB base address: 6800_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h - 7Ch	AHB RX Data Buffer Register (ARDB0 - ARDB31)	32	R	0000_0000h

79.14.3.2 AHB RX Data Buffer Register (ARDB0 - ARDB31)

Offset

For a = 0 to 31:

Register	Offset
ARDBa	0h + (a × 4h)

Function

This register is used to read the buffer content of the RX buffer from successive addresses. ARDB0 corresponds to the RX buffer register entry corresponding to the current value of the read pointer in an increasing order.

The increment of the read pointer depends on the access scheme (DMA or flag-driven). See Flash memory read, RX buffer, data read through register interface, and AHB read for the description of successive accesses to the RX buffer content. See Byte Ordering of Serial Flash Memory Read Data for the byte ordering scheme.

Valid address range accessible in the ARDB*n* range depends on the number of RX buffer entries implemented and on the number of valid buffer entries available in the RX buffer.

- Example 1 - RX buffer filled completely with 32 words: In this case, the address range for valid read access extends from ARDB0 to ARDB31 .
- Example 2 - RX buffer filled with five valid words; RX buffer fill level RBSR[RDBFL] is 5. In this case, an access to ARDB4 provides the last valid entry.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ARXD															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ARXD															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 ARXD	ARDB provided RX buffer data Byte order (endianness) is identical to the RX buffer data registers.

79.15 Glossary

AHB Advanced high-performance bus, a version of AMBA

AMBA Advanced microcontroller bus architecture

APB Advanced peripheral bus

BE Big endian byte ordering

CRS Center aligned read strobe

CS Chip select

FRAD Flash region access descriptor

I/O Input output, I/O lines are also referred to as pads in this chapter

IFM Individual flash memory mode

LE Little endian

MDAD Master domain access descriptors

MGID Master-Group identifier

PCS Peripheral chip select

SCK Serial communications clock

SFM Serial flash memory

Chapter 80
Quad Serial Peripheral Interface (QuadSPI) for S32K388 and S32K389

80.1 Chip-specific QuadSPI information

80.1.1 QuadSPI configuration

Table 771. QuadSPI instances

Instance	S32K388/S32K389	S32K310/S32K311/S32K312
QuadSPI	Yes	No

Table 772. QuadSPI configuration details

Configuration	S32K388/S32K389
QuadSPI Tx FIFO size	256 words
QuadSPI Rx FIFO size	32 words
Look Up Table Size	16 words
Rx Buffer	1024 Bytes

For supported data rates, see device Datasheet.

NOTE

- Boot from QuadSPI is not supported but execution from external memory is supported.
- SoC generate secure access from all CM7 core to QuadSPI.

Table 773. Features supported

Feature	S32K388/S32K389
AHB Write	No
Data learning feature	No
DLL	Yes
OTFAD (On-the-fly-AES-decryption engine)	No
DDR mode	No
HyperRAM	No
HyperFlash	No
External DQS (Data Strobe)	No
Boot from QuadSPI interface	No

The following block diagram depicts the QuadSPI Flash A Interface.

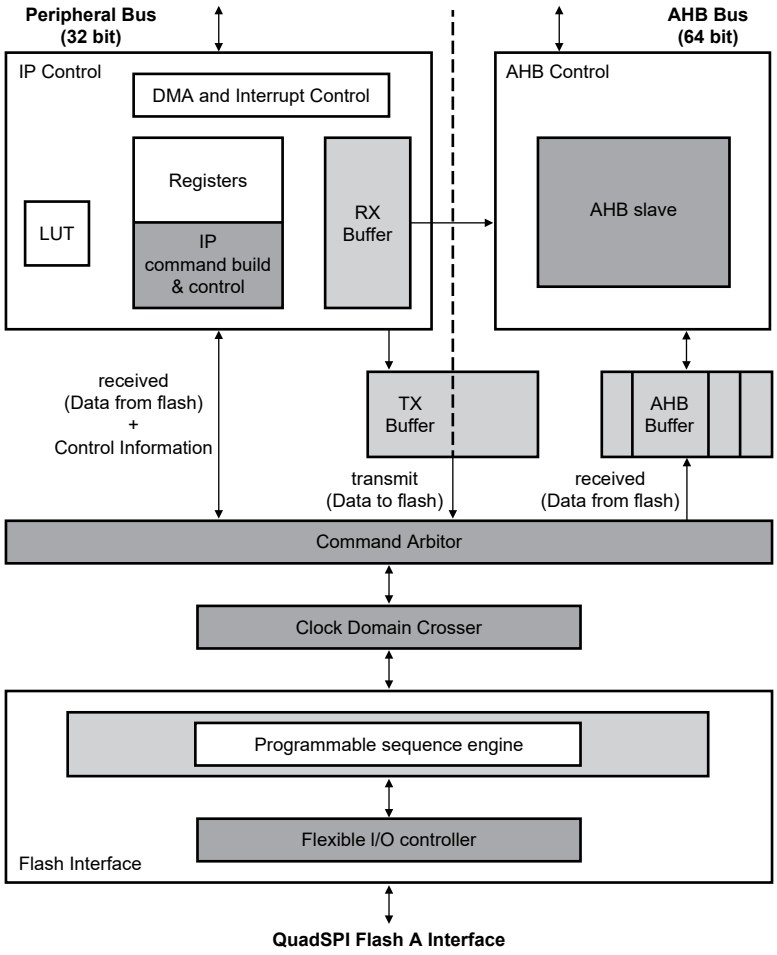


Figure 520. Block diagram

80.1.2 Supported read modes

The table below provides an overview of the QuadSPI read modes.

Table 774. QuadSPI read modes

Read modes		SDR support (QuadSPI_MCR [DDR_EN]=0)	QuadSPI_MCR [DQS_EN]	QuadSPI_MCR [DQS_FA_SEL]	Data learning support	DLL
DQS sampling method	Pad loopback	Yes	1	01	No	Yes

80.1.3 QuadSPI initialization sequence

Following initialization sequence should be followed for proper QuadSPI operation:

- Enable QuadSPI module by MC_ME peripheral clock enable (register PRTNx_COFBy_CLKEN present within MC_ME). Refer MC_ME chapter for peripheral mapping.
- Configure the SIUL2 registers MSCR[OBE] as 1 and MSCR[SSS] as 0 for QuadSPI_SCKFA pin.
- Initialize QuadSPI SCKFA by writing a sequence of 1010 to the SIUL2 register GPDO[PDO_a] for QuadSPI_SCKFA pin.

- Configure the SIUL2 register MSCR[OBE] back as 0 for the pins.
- Initiate a dummy flash read to reset all DQS flops by itself and any crossing from DQS domain to IPS/AHB is taken care by CDC logic.
- Initiate a peripheral software reset to QuadSPI controller by writing to QuadSPI controller's MCR.
- Post this initialization sequence, the QuadSPI will work in intended deterministic manner.

NOTE

QuadSPI initialization is to be done before using QuadSPI after each functional reset.

80.1.4 Pad clock loopback

This chip supports pad clock loopback. The QuadSPI can be configured to use clock loopback to sample input data. SCK is delayed by the SCK pin output delay, plus the SCK pin input delay using pad loopback, and is configured by setting QuadSPI config registers SOCCR[SOCCFG] and MCR[DQS_FA_SEL]. Enabling the loopback version of SCK can improve the setup time of the input data from the Flash.

For details of these register, see QuadSPI register descriptions.

80.1.5 QuadSPI SOC Configuration register SOCCR[SOCCFG] implementation

The QuadSPI SOC Configuration register QuadSPI_SOCCR[SOCCFG] register is used to control dummy loopback pads and obe_pull_timing_relax_b. Below is the description of it's bits:

Table 775. SOCCR[SOCCFG] implementation

Bit	Description
Bit[0]	obe_pull_timing_relax_b : enables the timing relaxation by pulling obe for pad 1 for half cycle, if 0 then enabled else disabled.
Bit[1]	ibe of QSPIA_SCK_DUMMY pad. 0: Disable input receiver 1: Enable input receiver.
Bit[2]	obe of QSPIA_SCK_DUMMY pad. 0: Disable output driver 1: Enable output driver.
Bit[3]	dse of QSPIA_SCK_DUMMY pad. 0: Disable drive strength 1: Enable drive strength.
Bit[4]	pue of QSPIA_SCK_DUMMY pad. 0: Disable internal pullup or pulldown resistor 1: Enable internal pullup or pulldown resistor.
Bit[5]	pus of QSPIA_SCK_DUMMYpad. 0: Enable internal pulldown resistor if pue is set 1: Enable internal pullup resistor if pue is set.
Bit[6]	sre of QSPIA_SCK_DUMMY pad. 0: Disable slew rate control 1: Enable slew rate control.
Bit[31:7]	Reserved

To Enable the Quadspi dummy PAD loopback use following settings

For Flash-A: MCR[DQS_FB_SEL] = 0x2

SOCCR[SOCCFG] = 0x0000000E (ibe=1, obe=1, dse=1 and sre=0)

NOTE

S32K3xx SoCs does not support dual die flashes. Hence, the signal PCSFA2 from QuadSPI is not used.

80.1.6 QuadSPI AHB Buffer write access control

In S32K388/S32K389 the QuadSPI AHB buffer does not support writes. A write to QuadSPI AHB buffer results in error response from QuadSPI.

QuadSPI might behave unexpectedly in case if its AHB buffer is written with QuadSPI MCR[MDIS] =1. In cases wherein QuadSPI is disabled with above MDIS control bit, XRDC must also be appropriately configured to block the accesses to QSPI AHB buffer and indicate an error response.

80.2 Introduction

The QuadSPI module acts as an interface to a single serial flash memory device, with up to four bidirectional data lines.

80.2.1 Features

QuadSPI supports the following features:

- Flexible sequence engine to support various flash memory vendor devices. As there is no specific standard, the module supports various kinds of flash memories from different vendors. See Serial flash memory devices for example sequences.
- Single, dual, and quad modes of operation supported for Quad flash memories
- AHB master to read RX buffer data through AMBA AHB (64-bit width interface) or IPS registers space (32-bit access) and fill TX buffer via IPS registers space (32-bit access)
  - AHB master can be a DMA with a configurable inner loop size
- Multi-master accesses are allowed
  - Flexible and configurable buffer for each master—total available buffer size is 1024 bytes.
- All AHB accesses to flash/RAM memory devices are directly memory mapped to the chip system memory
- Programmable sequence engine to cater to future command/protocol changes and ability to support all existing vendor commands and operations. The software needs to select the corresponding sequence according to the connected flash memory device.
  - Support for all types of addressing

80.2.2 RX buffer push event

To add the valid entries into the RX buffer

By default, each buffer push event adds two entries to the RX buffer because the interface to the serial clock domain is 64 bits in width. Depending on the number of bytes read from the serial flash memory device, it is possible for the very last buffer push event that only one entry is added.

RBSR[RDBFL] is incremented by the number of entries added to the RX buffer.

80.2.3 RX buffer POP event

To remove valid entries from the RX buffer

Each buffer POP event removes (RBCT[WMRK] + 1) valid entries from the buffer. BSR[RDBFL] is decremented by the same number and RBSR[RDCTR] is incremented accordingly.

80.2.4 Block diagram

The following figure shows a block diagram of the QuadSPI module.

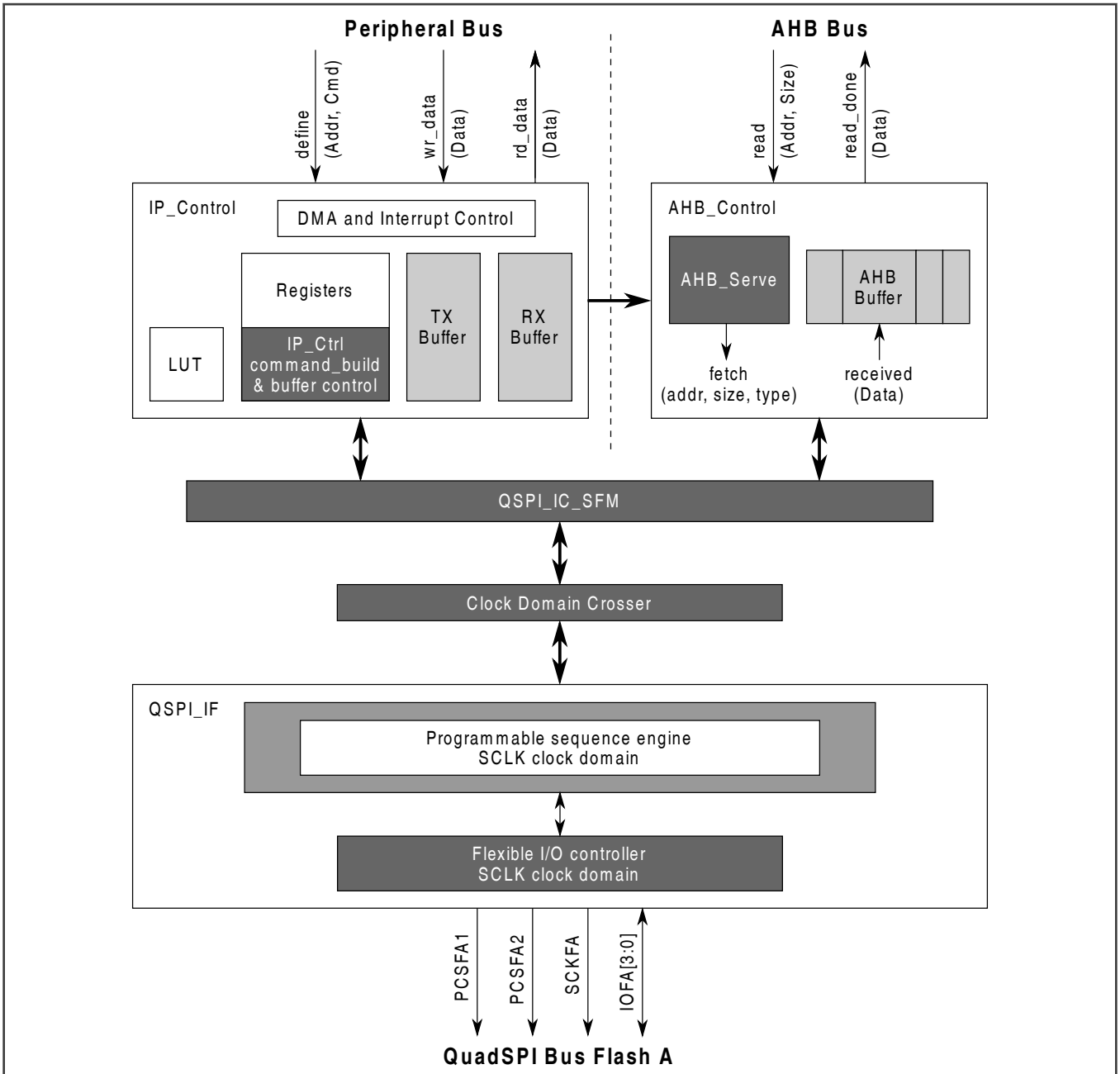


Figure 521. QuadSPI Block Diagram

80.2.5 QuadSPI modes of operation

QuadSPI supports the following modes of operation:

- Normal mode: You can use this mode for write or read accesses to an external serial flash memory device. See Normal mode for details.
  - Serial flash memory write: You can program data into the flash memory through the IP interface only. See Flash memory programming for details.
  - Serial flash memory read: Read the contents of the serial flash memory device. Two separate read channels are available through the RX buffer and AHB buffer. See Flash memory read for details.

- Module Disable mode: You can use this mode for disabling serial flash memory clock and AHB command. The clock to the non-memory mapped logic in QuadSPI can be stopped in the Module Disable mode. The module enters the mode by setting MCR[MDIS].

80.3 External signal description

This section provides the external signal information for the QuadSPI module.

The following table lists the external signals belonging to the module in conjunction with the different modes of operation.

Table 776. Signal properties

Signal name	Function	Direction	Description
PCSFA1	Peripheral Chip Select Flash Memory A1	O	This signal is the chip select for the serial flash memory device A1 that represents the first of the two flash memory devices that share IOFA.
PCSFA2	Peripheral Chip Select Flash Memory A2	O	This signal is the chip select for the serial flash memory device A2 that represents the second of the two flash memory devices that share IOFA.
SCKFA	Serial Clock Flash Memory A	O	This signal is the serial clock output to the serial flash memory device A.
IOFA[3:0]	Serial I/O Flash memory A	I/O	These signals are the data I/O lines to/from the serial flash memory device A. See Driving external signals for details about the signal drive and timing behavior. Note that the signal pins of the serial flash memory device may change their function according to the SFM Command executed, leaving them as control inputs when single and dual instructions are executed. The module supports driving these inputs to dedicated values. In single I/O mode, QuadSPI drives data on IOFA[0] and expects data on IOFA[1].

NOTE

Please refer to chip specific information to check the configuration of QuadSPI block.

80.3.1 Driving external signals

Single/dual/quad instructions

Depending on the serial flash memory device connected to the QuadSPI module, there are instructions using a different number of data lines:

- Single pad: Single line I/O with one data out and one data in line to/from the serial flash memory device
- Dual pad: Dual line I/O with two bidirectional I/O lines, driven alternatively by the serial flash memory device or the QuadSPI module
- Quad pad: Quad line I/O with four bidirectional I/O lines, driven alternatively by the serial flash memory device or the QuadSPImodule

The different phases of the serial flash memory access scheme are shown in the following figure.

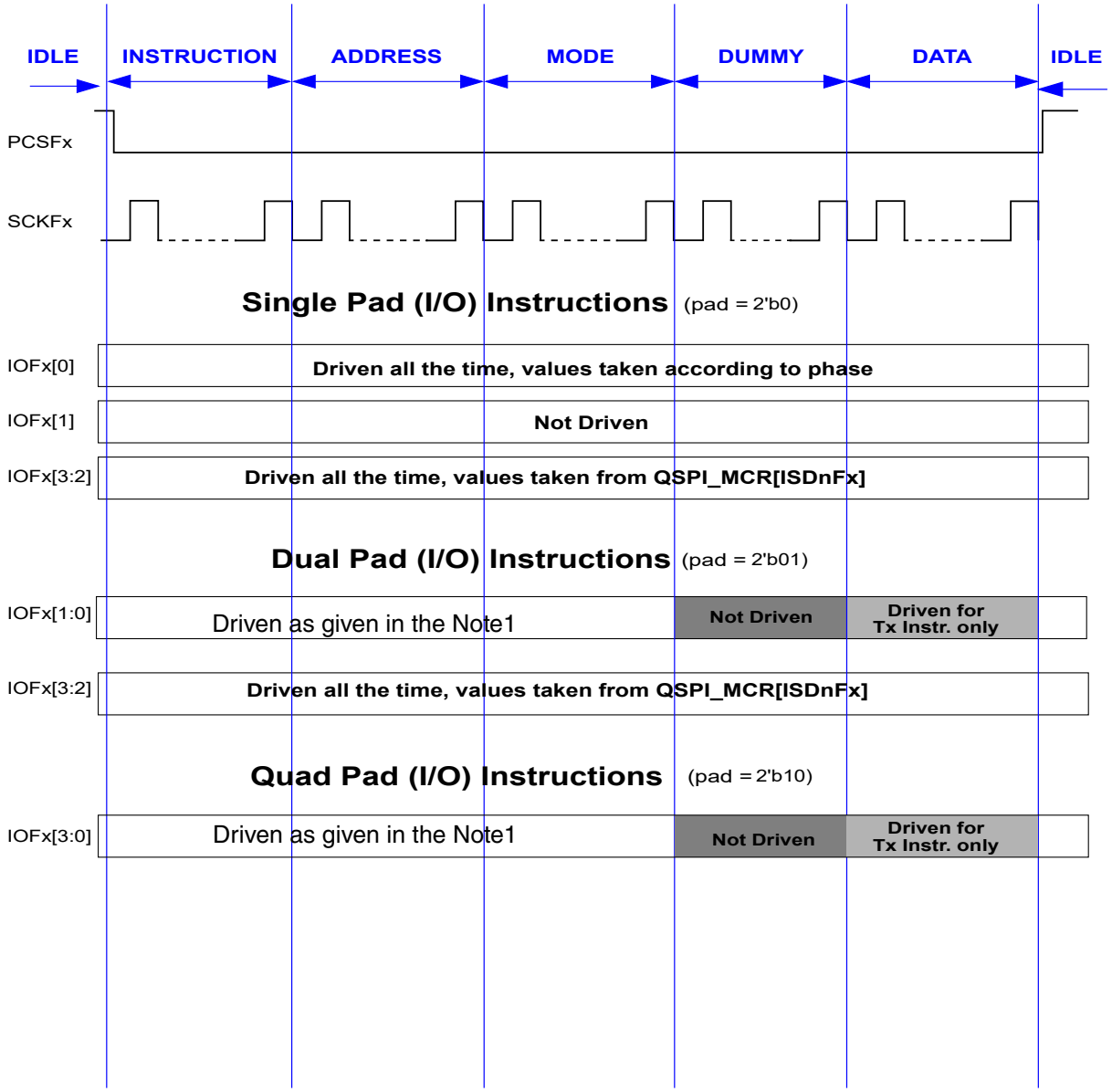


Figure 522. Serial flash memory access scheme

Note1:The IOs are driven from QuadSPI as per the number of pads configured for ongoing phase.

Note:The lines status will change based on command mode in case of instruction, address and mode phases. It can be either 1,2 or 4 lines

Following are the different phases and the I/O driving characteristics of the QuadSPI module:

- Idle: Serial flash memory device not selected – no interaction with the serial flash memory device and the IOFx signals are driven.
- Instruction: Serial flash memory device selected – the instruction is sent to the serial device and all the IOFx signals are driven.
- Address: Serial flash memory address is sent to the device – all the IOFx signals are driven and this phase is not applicable for all SFM commands.
- Mode: Mode bytes are sent to the serial flash memory device – all the IOFx signals are driven and this phase is not applicable for all SFM commands.

- Dummy: Dummy clocks are provided to the serial flash memory device. See Figure 522 for the IOFx signals driven. The actual data lines required for the SFM command executed are not driven for data read commands.

NOTE

— This phase is not applicable for all the SFM commands.

— All read commands in Dual pad, Quad pad modes must use a Dummy phase immediately before the Data phase. The Dummy phase pad configuration in the LUT must use the same number of pads as the subsequent Data phase. Note that this restriction is not applicable to Single-pad mode.

- Data: Serial flash memory data are sent to or received from the serial flash memory device. See the preceding figure for the IOFx signals driven. The actual data lines required for the SFM command executed are not driven for data read commands.

NOTE

This phase is not applicable for all the SFM commands.

The PCSFx and SCKFx signals are driven permanently throughout all the phases. In the individual flash memory mode, this applies to the selected flash memory device.

Access to a single, individual serial flash memory device

See Serial flash memory access schemes for details.

Read access to two serial flash memory devices attached to the QuadSPI module in parallel. See Serial flash memory access schemes for details.

80.4 Functional description

This section provides a functional description of the QuadSPI module.

80.4.1 Serial flash memory access schemes

In the individual flash memory mode, all supported commands are available.

80.4.2 Normal mode

This mode allows communication with an external serial flash memory device. Compared to the standard SPI protocol, this communication method uses up to four bidirectional data lines operating at high-data rates. The communication to the external serial flash memory device consists of an instruction code and optional address, mode, dummy, and data transfers. The flexible programmable core engine described below is immune to a wide variety of command or protocol differences in the serial flash memory devices provided by various flash memory vendors.

80.4.2.1 Programmable sequence engine

The core of the QuadSPI module is a programmable sequence engine that works on "instruction-operand" pairs. The core controller executes each programmed instruction sequentially. The complete list of instructions and the corresponding operands are provided in the following table.

Table 777. Instruction set

Instruction	Instruction encoding	Pins	Operand	Action on serial flash memories
CMD	1d	N={0,1,2}d 0d - One pad 1d - Two pads 2d - Four pads	8-bit command value	Provides the serial flash memory with the SFM command operand (Encoded) on the number of pads specified in STR mode.

Table continues on the next page...

Table 777. Instruction set (continued)

Instruction	Instruction encoding	Pins	Operand	Action on serial flash memories
ADDR	2d		Number of address bits to be sent (for example, 24d => 24 address bits required)	Provide the serial flash memory with address cycles according to the operand on the number of pads specified The actual address to be provided is derived from the incoming address in case of AHB-initiated transactions and the value of SFAR in case of IPS-initiated transactions, if the value of SFACR[CAS] is set to 0. Otherwise, the actual address takes CAS into consideration.
DUMMY	3d		Number of dummy clock cycles (should be <= 64 and > 2 cycles)	Provide the serial flash memory with dummy cycles according the operand The PAD information defines the number of pads in input mode. For example, one pad implies that pad 1 is not driven, rest all are driven. **NOTE** If DLL is enabled and N dummy cycles are needed, you must program two back-to-back DUMMY instructions: DUMMY: N-2 followed by DUMMY:2.
MODE	4d		8-bit mode value	Provide the serial flash memory with 8-bit operand on the number of pads specified
MODE2	5d	N={0,1}d	2-bit mode value	Provide the serial flash memory with 2-bit operand on the number of pads[1] specified
MODE4	6d	N={0,1,2}d	4-bit mode value	Provide the serial flash memory with 4-bit operand on the number of pads[2] specified
READ	7d	N={0,1,2}d 0d - One pad 1d - Two pads 2d - Four pads	Read data size in bytes (for AHB transactions, your application should ensure that data size is a multiple of 8 bytes)	Read data from flash memory on the number of pads specified The data size can be overwritten by writing to the ADATSZ field of the BUFxCR registers for AHB-initiated transactions and to the IDATSZ field of the IP Configuration Register (IPCR) for IPS-initiated transactions.
WRITE	8		Write data size in bytes	Write data on the number of pads specified The data size can be overwritten by writing to the IDATSZ field of IP Configuration Register (IPCR).
JMP_ON_CS[3]	9d	NA	Instruction number	Every time the CS is deasserted, jump to the instruction pointed to by the operand. This instruction allows the programmer

Table continues on the next page...

Table 777. Instruction set (continued)

Instruction	Instruction encoding	Pins	Operand	Action on serial flash memories
				to specify the behavior of the controller when a new read transaction is initiated following a CS deassertion. **NOTE** This instruction is not supported if DLL is enabled.
CMD_DDR	17d	N={0,1,2}d 0d - One pad 1d - Two pads 2d - Four pads	8-bit command value	Provides the serial flash memory with the SFM command operand (Encoded) on the number of pads specified in DTR mode.
CADDR	18d		Number of column address bits to be sent (8d means 8 bits of column address is to be sent)	Provide the serial flash memory with column address cycles according to the operand on the number of pads specified. The actual address to be provided to flash memory depends on the value of SFACR[CAS]. For example, if the value of SFACR[CAS] is 3, then the address to flash memory will be [2:0] of incoming address in case of AHB and the value of SFAR in case of IP. This is appended with 0 if SFACR[CAS] is less than number of pads for a flash memory.
CADDR_DDR	19d		Number of column address bits to be sent(8d means 8 bits of column address is to be sent)	Provide the serial flash memory with column address cycles according to the operand on the number of pads specified at each clock edge of the serial flash memory. The actual address to be provided to flash memory depends on the value of SFACR[CAS]. For example, if CAS is 3, then the address to flash memory will be [2:0] of incoming address in case of AHB and the value of SFAR in case of IP. This is appended with 0 if CAS is less than the number of pads for a flash memory.
STOP[3]	0d	NA	NA	Stop execution; deassert CS

1. For a one-pad instruction, MODE2 takes two serial flash memory clock cycles on the flash memory interface.
2. For a one-pad instruction, MODE4 takes four serial flash memory clock cycles on the flash memory interface. For a four-pad instruction, MODE4 takes one serial flash memory clock cycle on the flash memory interface.
3. Sequence ending with this instruction must have all remaining bits as 0s after it.

The programmable sequence engine allows you to configure the QuadSPI module according to the serial flash memory connected on board. This flexible structure is compatible with new command or protocol changes from different vendors.

80.4.2.2 Flexible read xAHB buffers

To reduce the latency of the reads for AHB masters, the data read from serial flash memory is buffered in flexible AHB buffers. There are four such flexible buffers. The size of each of these buffers is configurable with the minimum size being 0 bytes and maximum size being the size of the complete buffer instantiated (1024 bytes). The size of buffer 0 ranges from 0 to BUF0IND. The size of buffer 1 ranges from BUF0IND to BUF1IND, buffer2 from BUF1IND to BUF2IND and, buffer 3 ranges from BUF2IND to the size of the complete buffer (1024 bytes).

Each flexible AHB buffer is associated with the following:

- An AHB master: Optionally, buffer3 may be configured as an "all master" buffer by writing 1 to BUF3CR[ALLMST]. When buffer3 is configured in such a way, any access from a master not associated with any other buffer is routed to buffer3.

- A datasize field representing the amount of data to be fetched from the flash memory on every missed access.

The master ID of every incoming request is checked and the data is returned or fetched into the corresponding associated buffer. See the chip-specific QuadSPI information for details about master IDs and their corresponding components. Every missed access to the buffer causes the controller to clear the buffer and fetch the BUFxCR[ADATSZ] amount of data from the serial flash memory. As such, you need not configure the buffer size to be greater than ADATSZ because the locations greater than ADATSZ are never used. For any AHB access, the sequence pointed to by BFGENCR[SEQID] is used for the initiated flash memory transaction. The data is returned to the master as soon as the requested amount is read from the serial flash memory. The controller; however, continues to prefetch the rest of the data in anticipation of a next consecutive request. See Figure 523 that shows flexible AHB buffers.

BFGENCR[SEQID] points to an index of the LUT. See LUT for details.

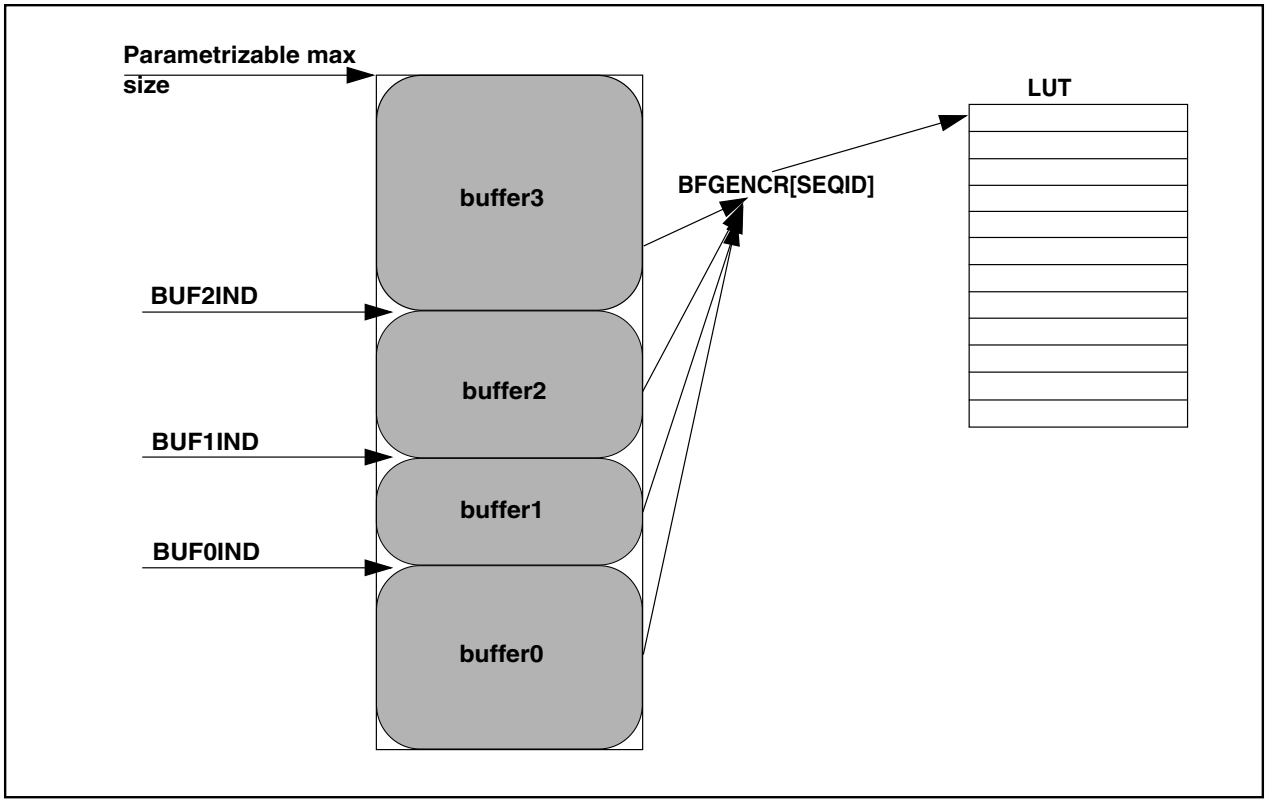


Figure 523. Flexible read AHB buffers

80.4.2.3 Abort mechanism during AHB read

Any ongoing read transaction is aborted if a request from the same master arrives for a location other than the location at which the transaction is going on. The abort can happen at any point of time.

80.4.2.4 HBURST support with AHB read

QuadSPI controller supports HBURST and HSIZE on the AHB read interface. HBURST indicates if the transfer forms part of a burst. Four, eight, and sixteen beat bursts are supported and the bursts might either be incrementing or wrapping. HSIZE indicates the size of the transfer, and supports 8-, 16-, 32-, and 64-bit data sizes. In case of WRAP accesses, QuadSPI generates aligned accesses to serial flash memory if there is no buffer hit for any incoming, non-sequential AHB read access. In case there is a buffer hit, the incoming address in the haddr line is latched as it is. If the total burst size is more than the data prefetch size, an error response is generated and the value of FR[AIBSEF] is configured as 1. The data prefetch size can be defined by BUFxCR[ADATSZ] or data size mentioned in the sequence pointed to by the SEQID field when ADATSZ is programmed as 0. In

case of wrap burst, data prefetch size must be greater than or equal to the wrap burst size + 32 bytes. A few examples are shown in the figure below:

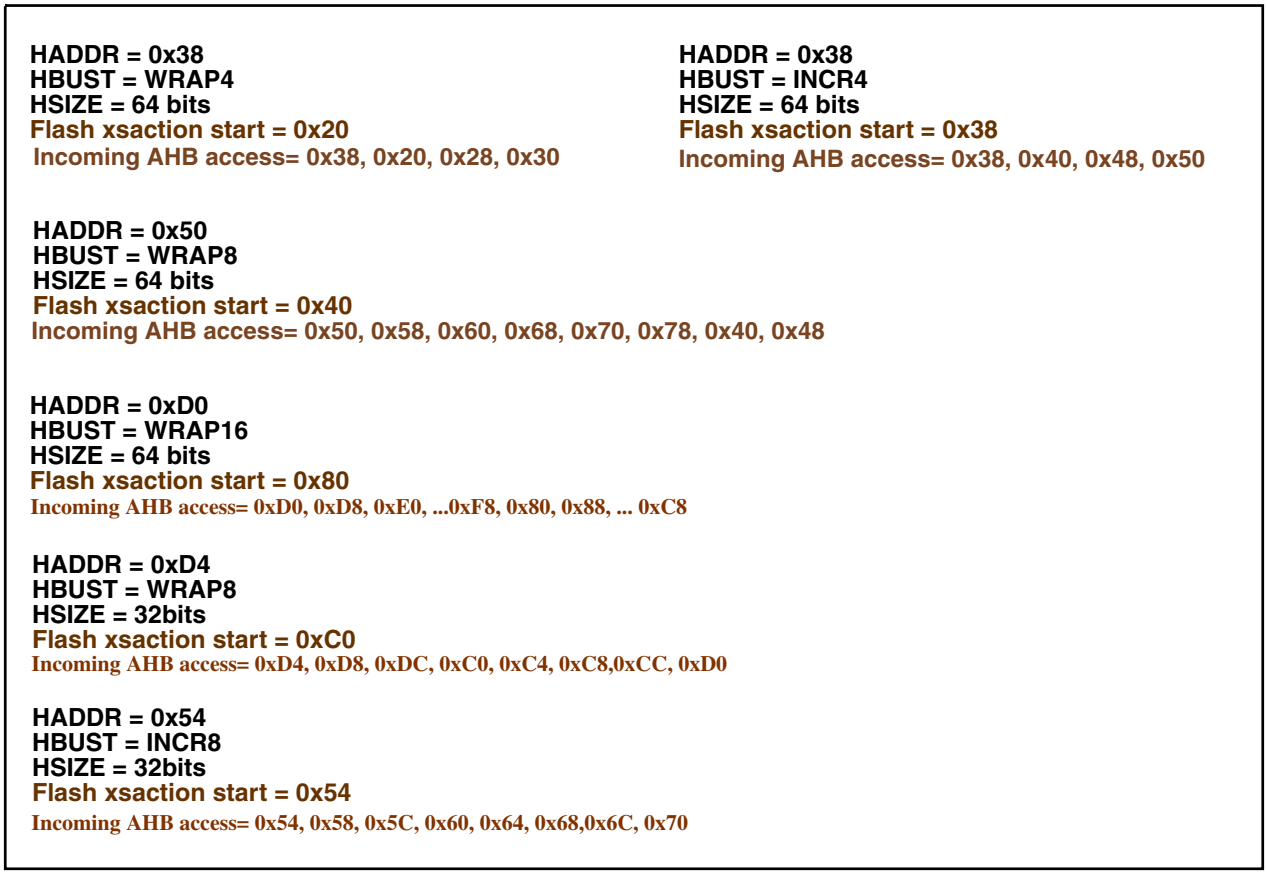


Figure 524. QuadSPI HBURST support

NOTE

The software must take care that the prefetch size should never be set less than the minimum data needed by any external interface to start processing.

NOTE

Whenever a core accesses QuadSPI memory with cache enabled, the prefetch size must be configured as equal or more than the cache line size; otherwise, FR[AIBSEF] error appears.

80.4.2.5 LUT

The LUT consists of a number of pre-programmed sequences. Each sequence is basically a sequence of instruction-operand pairs, which when executed sequentially, generate a valid serial flash memory transaction. Each sequence can have a maximum of 10 instruction-operand pairs. The LUT can hold a maximum of 16 sequences. The figure below shows the basic structure of the sequence in the LUT.

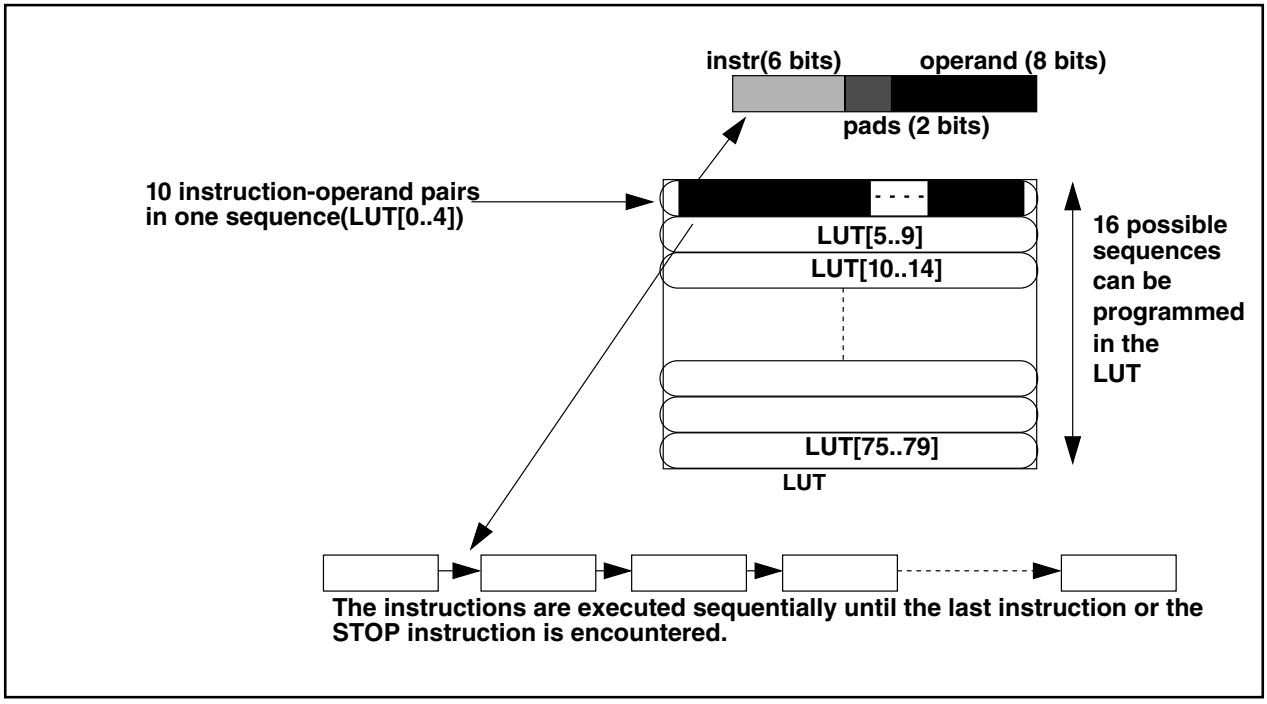


Figure 525. LUT and sequence structure

At reset, the index 0 of the LUT[0..4] is programmed with a basic read sequence as described in Read sequence. After reset, the complete LUT may be reprogrammed according to the chip connected on board. To protect its contents, during a code runover, the LUT might be locked, after which a write to the LUT will not be successful until it has been unlocked again. The key for locking or unlocking the LUT is 5AF05AF0h, and the associated processes are as follows:

Locking the LUT

1. Write the key 5AF05AF0h into the LUT Key Register (LUTKEY).
2. Write 0b01 to the LUT Lock Configuration Register (LCKCR). Note that this IPS transaction should immediately follow the above IPS transaction (no other IPS transaction can be issued). A successful write to this register locks the LUT.

Unlocking the LUT

1. Write the key 5AF05AF0h into the LUT Key Register (LUTKEY).
2. Write 0b10 to the LUT Lock Configuration Register (LCKCR). Note that this IPS transaction should immediately follow the above IPS transaction (no other IPS transaction can be issued in between). A successful write to this register unlocks the LUT.

The lock status of the LUT can be read from the LCKCR[UNLOCK] and LCKCR[LOCK] fields.

Some example sequences are defined in Example sequences. After reset the instruction sequence 0 is populated with the default read sequence shown in the table below.

Table 778. Read sequence

Instruction	Pad	Operand	Comment
CMD	0h	3h	Read data byte command on one pad
ADDR	0h	18h	24 address bits to be sent on one pad

Table continues on the next page...

Table 778. Read sequence (continued)

Instruction	Pad	Operand	Comment
READ	0h	8h	Read 64 bits
JMP_ON_CS	0h	0h	Jump to instruction 0 (CMD)

NOTE

If DLL is disabled then JMP_ON_CS or STOP instruction can be used else only STOP instruction can be used.

80.4.2.6 Issuing SFM commands

Each access to the external device follows this sequence:

1. You must pre-populate the LUT with the serial flash memory command sequences that are required for the flash memory device being used.
2. The module executes the instructions in this sequence one by one. The transaction starts and the module configures the value of SR[BUSY].
3. Communication with the external serial flash memory device starts and the transaction executes.
4. After the transaction is complete (all transmit and receive operations with the external serial flash memory device are complete), the module resets SR[BUSY]. In case of an IP command, FR[TFF] is asserted.

For details, see Flash memory programming and Flash memory read.

You can trigger the processing of SFM commands in the QuadSPI module in one of the following ways:

Using IP commands

For IP commands, the required components need to be written into the following registers and in this sequence:

1. Write the serial flash memory address to be used as provided in the Serial Flash Address Register (SFAR). For IP commands not related to specific addresses, the base address of the related flash memory needs to be programmed. For example, for an instruction which does not require an address (that is, write enable instruction), the SFAR should be programmed with the base address of the memory the command is to be sent to.
2. Write the sequence ID and data size details in the IP Configuration Register (IPCR).
3. Note that writing a value to IPCR[SEQID] must be the last step of the sequence. It is possible to combine all the fields of the IPCR into one single write. See IP Configuration Register (IPCR) for details.

Using AHB commands

Any AHB memory-mapped access is routed to one of the buffers depending on the master ID of the request. If the access is a "miss," a new serial flash memory transaction is initiated. The transaction is based on the sequence pointed to by BFGENCR[SEQID] as described in Flexible read xAHB buffers.

An AHB access is considered memory mapped when the access is to the memory-mapped serial flash memories, as described in Memory Mapped Serial Flash Data - Individual Flash mode on Flash memory A.

80.4.2.7 Flash memory programming

In all NOR Flash devices memory sector to be written needs to be erased first. The programming sequence is then initiated in the following way:

1. Program the FRAD and MDAD registers.
2. Program the address related to the command in SFAR (with access attributes programmed in MDAD). If required, write the desired value to SFACR[CAS], otherwise write 0 to it. Also, write 1 to SFACR[WA] if the serial flash memory is a word addressable flash memory, or write 0 in case serial flash memory is byte addressable

3. Program the IPCR register (with access attributes programmed in MDAD). IPCR[SEQID] should point to an index of the LUT that has the flash memory program sequence pre-programmed. Write an appropriate value to IPCR[IDATSZ] to denote the size of the write in bytes.
4. Check FSM Status (FSMSTAT) register to know status of current transaction. Valid[31] bit should be '1' and domain ID [11:8] should match the ID with which SFAR and IPCR were written. Program the TX watermark in TBCT[WMRK] field and wait for STAT[1:0] bits to turn '01'.
5. Check that the TX buffer is empty. If you need to discard the data present in the TX buffer (SR[TXEDA]) field is set, then the TX buffer must be cleared by writing 1 to MCR[CLR_TXF].
6. Provide data for the program command into the circular buffer through the TBDR. Once the TX buffer is written till watermark level and SR[TXWA] flag is de-asserted this IP command will be triggered. FSMSTAT[STAT] will get set to 10.
7. Repeat step 3, depending on the amount of data required, until all of the required data is written to the TBDR. SR[TXFULL] can be used to check if the buffer is ready to receive more data. At any time, TBSR[TRCTR] can be read to check how many words have been written into the TX buffer.
8. Once the transaction completes FSMSTAT[VLD] bit will set to 0, SR[BUSY] is reset and FR[TFF] is asserted. Please refer to section Secure flash programming for more details.

After writing to IPCR[SEQID] , the module starts executing the programmed sequence when QuadSPI SFM is IDLE. The software ensures that the correct sequence is programmed into the LUT in accordance with the flash memory connected to the module. The data is fetched from the TX buffer. It consists of 256 entries of 32-bit sizes and is organized as a circular FIFO, the read pointer for which is incremented after each fetch. When all the data is transmitted, the QuadSPI module returns from the busy state to the idle state. However, this is not true for the external device because the internal programming is still ongoing. You may monitor the relevant status information available from the serial flash memory device and ensure that the programming is done appropriately.

80.4.2.8 Flash memory read

Host access to the data stored in the external serial flash memory device is performed in two steps. First, the data must be read into the internal buffers and in the second step, these internal buffers can be read by the host.

Reading serial flash memory data into the QuadSPImodule internal buffers

A read access to the external serial flash memory device can be triggered in two different ways:

- IP command read: For reading flash memory data into the RX buffer, you must provide the correct sequence ID in IPCR[SEQID]. The sequence ID points to a sequence in the LUT. The software needs to ensure that a correct read sequence is programmed in the LUT in accordance with the serial flash memory device connected on board. You must program the SFAR , SFACR[CAS], and IPCRs. All available read commands supported by the external serial flash memory are possible.

  Optionally, it is possible to clear the RX buffer pointer prior to triggering the IP command by writing a 1 to MCR[CLR_RXF]. This will invalidate the data currently present in the RX buffer and any new read data will overwrite the old one.

  Using these inputs, the complete transaction is built when IPCR[SEQID] is written to and if MDAD checks are passing (based on the access attributes while writing SFAR and IPCR). The transaction related to the read access starts when QuadSPI is IDLE and FSMSTAT[VLD] bit is 1. The data is fetched for the domain ID that is shown in FSMSTAT[11:8] bits when FSMSTAT[STAT] is set to 11 and the requested number of bytes is fetched from the external serial flash memory device into the RX buffer. As the read access is triggered by an IP command, the value of both SR[IP_ACC] and SR[BUSY] is set to 1.. A count of the number of entries currently in the Rx buffer can be obtained from RBSR[RDBFL].

  Communication with the external serial flash memory stops if the specified number of bytes are read (on successful completion of the transaction).

- AHB command read: For reading flash memory data into the AHB buffer, you must:
  - Set up a read access by a master to the address range in the system memory map, which the external serial flash memory devices are mapped to.
  - Write a desired value to SFACR[CAS], if required, or write 0 to the field.
  - Program the buffer registers corresponding to the AHB master initiating the request.

— Provide the correct sequence ID in the BFGENCR. The software ensures that a correct read sequence is programmed in the LUT in accordance with the serial flash memory device connected on board. Flash memory device selection and access mode are determined by the address accessed in the AHB address space associated with the QuadSPI module (see Memory-mapped serial flash memory data—individual flash memory mode on flash memory A

On each AHB read access to the memory mapped area, the valid data in the AHB buffer is checked against the address requested in the actual read. When the AHB read request cannot be served from the content of the AHB buffer, the buffer is flushed and the controller executes the sequence pointed to by the sequence ID. The requested number of buffer entries defined in BUFxCR[ADATSZ] is then fetched from the external serial flash memory device into the internal AHB buffer. As the read access is triggered through the AHB bus, the value of SR[AHB_ACC] is set, driving SR[BUSY] in turn, until the transaction is complete. Communication with the external serial flash memory stops when the specified number of entries is filled.

Data transfer from the QuadSPI module internal buffers

The data read out from the external serial flash memory device by the QuadSPI module is stored in the internal buffers. The means of accessing the data from the buffer differs depending on which buffer the data is loaded to. See Block diagram for details on the two available buffers, the RX buffer and the AHB buffer, in this module. The buffer appears transparent to you and is non-memory mapped. See the "Flexible AHB Buffer" section for details.

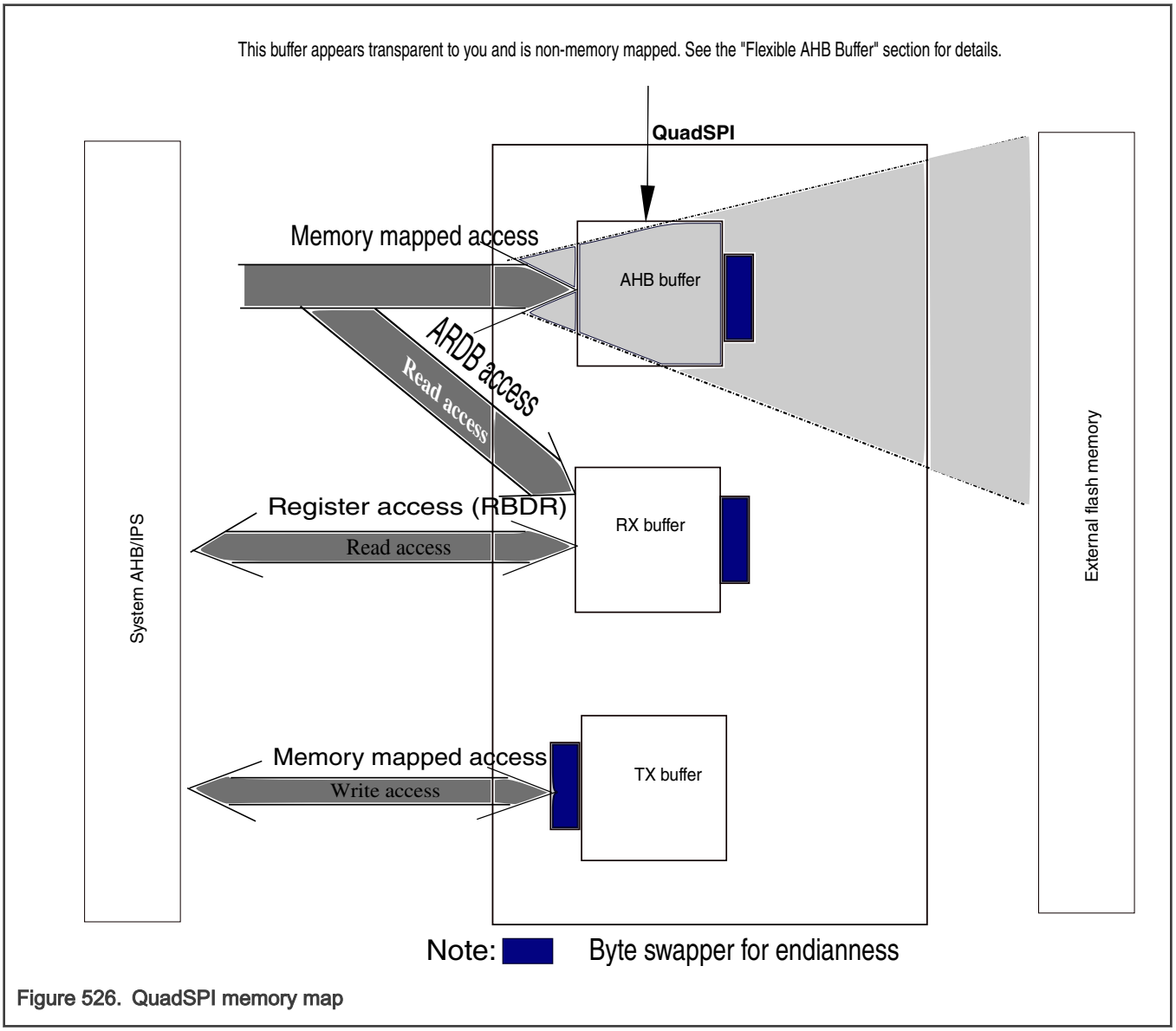


Figure 526. QuadSPI memory map

The RX buffer is implemented as FIFO of depth 32 entries of 4 bytes. Its content is accessible in two different address areas, both referring to identical data and the same physical memory:

- In the IPS address space in the area associated with RX Buffer Data Register (RBDR0 - RBDR31).
- In the AHB address space in the area associated with AHB RX Data Buffer Register (ARDB0 - ARDB31).Two successive entries are accessed with one single 64-bit AHB read operation.

The RX buffer operation can be summarized as follows:

- RBCT[WMRK] determines at which fill level SR[RXWE] is asserted and how many entries are removed from the RX buffer on each buffer POP operation.
- SR[RXWE] indicates that the configured number of data entries is available in the RX buffer and RBSR[RDBFL] indicates how many valid entries are available in total.
- The first entry (RBDR0 or ARDB0) always corresponds to the first valid entry in the RX buffer.

For details, see RX Buffer Data Register (RBDR0 - RBDR31) and AHB RX Data Buffer Register (ARDB0 - ARDB31).

- Flag-based data read of the RX buffer is performed by polling SR[RXWE]. When it is asserted, the valid entries can be read either via the IPS address space (RBDRn) or the AHB address space (ARDBn). A buffer POP operation must be triggered by the application by writing a 1 to FR[RBDF]. This automatically updates the FIFO to point to the next entry as defined by RBCT[WMRK]. For example, if WMRK is set to 3, then the buffer discards 16 bytes of data.
- DMA-controlled data read of the RX buffer is performed by using the DMA module. The application must ensure that the DMA controller of the related chip is programmed appropriately, as described in DMA usage.
- DMA-controlled read out is triggered fully automatically by the assertion of SR[RXWE]. The related buffer POP operation is also handled completely inside the QuadSPI module. As in the case explained here, accessing the RX buffer content either on RBDRn or ARDBn related addresses is equivalent.
- AHB buffer data read via memory-mapped access: This kind of access is performed by reading one of the addresses assigned to the external serial flash memory device(s) within the range specified in Table 807. If this is not the case, a memory-mapped AHB command read is triggered as described above. If the requested data is already available in the AHB buffer, it is provided directly to the host.

When an AHB access is made to the flash memory mapped address, the data is fetched and returned to the AHB interface. The AHB interface is stalled until the data is fetched. As soon as the data from the requested address is read by the QuadSPI module, the AHB read access is served. Therefore, it is possible to run sequential reads from the AHB buffer at arbitrary speed without the need to monitor any information about the availability of the data. Nevertheless, this access scheme stalls the AHB bus for the time required to read the data from the serial flash memory device. If you know that the access is sequential, a better way is to have a prefetch enabled by programming the value of BUFxCR[ADATSZ] so that the data is fetched into the buffer before the next sequential AHB access.

As long as the host restricts its accesses to the data present in the buffer and to the data currently fetched from the serial flash memory, it is possible to run the host read from the AHB buffer simultaneously with the serial flash memory read into the AHB buffer.

80.4.2.9 Byte ordering of serial flash memory read data

The basic scheme is that the first byte read out of the serial flash memory device, which is addressed by SFAR[SFADR], corresponds to RBDR0[31:24] for IP command read. Similarly, to send a single byte it should be positioned in TBDR[0:7]. In contrast to that for AHB command read, the bytes are always positioned according to the byte ordering of the AHB bus.

- Byte ordering in individual flash memory mode

  The following table provides the byte ordering scheme of how the byte oriented data space of the serial flash memory device is mapped into one single 32-bit entry of the RX buffer or the AHB buffer. The table is valid within the following context:

  — Flash memory A in individual flash memory mode

  — All AHB data read commands with 32-bit access size

Table 779. Byte ordering in individual flash memory mode

Serial flash memory byte numbering	3	2	1	0
Buffer entry bit position [31:0] (32-bit data width)	[31:24]	[23:16]	[15:8]	[7:0]

NOTE

For IP commands, the read size can be specified as number of bytes. If this number is not a multiple of four, then the last buffer entry is not completely filled with the missing higher numbered bytes at undefined values.

For AHB commands and reads, starting from an address not aligned to 32-bit boundaries, the requested bytes are given at the appropriate positions according to the AMBA AHB specification.

- Buffer entry ordering for 64-bit read access

  For read access via the AHB interface, a 64-bit access is possible. Each 64-bit access reads two 32-bit entries, simultaneously. The ordering of these 32-bit entries within the 64-bit word is provided in the following table.

Table 780. 64-bit read access buffer entry ordering

AHB read data bit position [63:0]	[63:32]	[31:0]
Buffer entry #	Odd (1, 3, 5, ...)	Even (0, 2, 4, ...)

80.4.2.10 Normal mode interrupt and DMA requests

The QuadSPI module has different flags that can only generate interrupt requests and one flag that can generate an interrupt as well as DMA requests. The following table lists the eight conditions. Note that the flags mentioned in the table are associated with the Flag Register (FR).

Table 781. Interrupt and DMA request conditions

Condition	Flag (FR)	DMA
TX buffer fill	TBFF	-
TX buffer underrun	TBUF	-
Illegal instruction error	ILLINE	-
RX buffer drain	RBDF	X
RX buffer overflow	RBOF	-
AHB buffer overflow	ABOF	-
AHB sequence error	ABSEF	-
AHB illegal transaction error	AITEF	-
AHB illegal burst size error	AIBSEF	-
IP command trigger could not be executed error	IPIEF	-
IP command related transaction finished	TFF	-

Each condition has a corresponding field in Flag Register (FR) and a request enable field in DMA Request Select and Enable Register (RSER). FR[RBDF] has separate enable fields for generating IRQ and DMA requests. Note that not all the fields have an individual IRQ line. See the chip's Interrupt Vector table for details.

- Transmit buffer fill interrupt request

  This indicates that the TX buffer can accept new data. The buffer is asserted if FR[TBFF] is asserted and if the value of the corresponding enable field, RSER[TBFIE], is 1. See TX buffer Operation for details on the assertion of FR[TBFF].

  Apart from IRQ, it is possible to handle the TX buffer fill by using the DMA. If the value of RSER[TBFDE] is 1, a DMA request is triggered when the number of available space in the TX buffer is more than the TBCT[WMRK] valid entries and value of SR[TXWA] is set. The application must configure the environment appropriately (for example, the DMA controller) for the DMA transfer.

  In SFP Enabled configurations, program RSER[TBFDE] bit to 1 only when arbitration is won and TBDR access is unlocked after checking STATE field of FSMSTAT register. After TBDR programming is completed by DMA, reset this bit otherwise, QuadSPI SFM state will remain busy and no further IPS accesses can be served by that Target queue.

- Receive buffer drain interrupt or DMA request

  This is derived from FR[RBDF], indicating that the RX buffer of the QuadSPI module has data available from the serial flash memory device to be read by the host. It remains set as long as RBSR[RXWE] is configured. Also, RSER[RBDIE] enables the related IRQ.

  Apart from the IRQ, it is possible to handle the RX buffer drain by using the DMA. If the value of RSER[RBDDE] is 1, a DMA request is triggered when the RX buffer contains more than RBCT[WMRK] valid entries. The application must set the environment appropriately (for example, the DMA controller) for the DMA transfers.

- Buffer overflow/underrun interrupt request

  This is a combination of the following fields (all located in the Flag Register (FR) with the related enable bits in the DMA Request Select and Enable Register (RSER)):

  — TBUF - TX buffer underrun, enabled by TBUIE

  — RBOF - RX buffer overflow, enabled by RBOIE

  — ABOF - AHB buffer overflow, enabled by ABOIE

  — The transmit buffer underrun indicates that an underrun condition in the TX buffer has occurred. It is generated when a write instruction is triggered whilst the TX buffer is empty and the value of RSER[TBUIE] is 1.

  — The receive buffer overflow indicates that an overflow condition in the RX buffer has occurred. It is generated when the RX buffer is full, an additional read transfer attempts to write into the RX buffer, and the value of RSER[RBOIE] is 1.

  — The AHB buffer overflow indicates that an overflow condition in the AHB buffer has occurred. It is generated when the AHB buffer is full, an additional read transfer attempts to write into the AHB buffer and the value of RSER[ABOIE] is 1.

  — The data from the transfers that generated the individual overflow conditions is ignored.

- Serial flash memory command error interrupt request
- Transaction finished interrupt request

  The IP command transaction finished IRQ indicates the completion of the current IP command. It is triggered by FR[TFF] and is masked by RSER[TFIE].

80.4.2.11 TX buffer operation

The TX buffer provides the data used for page programming. For proper operation, it is required to provide at least one entry in the TX buffer prior to starting the execution of the page programming command. The application must ensure that the required number of data bytes is written into the TX buffer fast enough as long as the command is executed without a TX buffer overflow or underrun.

The QuadSPI module sets the FR[TBFF] field as long as the TX buffer is not full and can accept more data. At the end of write through TX buffer, you must clear FR[TBFF] to avoid unnecessary last TX buffer fill interrupt. However, there would always be a pending request asserted from QuadSPI controller at the end of any DMA transfer. When external DMA finishes transfer iteration, this request from QuadSPI is kept asserted for the next iteration loop.

NOTE

Even if the generation of DMA requests for filling the TX buffer is disabled by using RSER[TBFDE], the TX buffer still accepts a DMA transfer because of the last asserted pending request.

Disabling of DMA transfer should be controlled by an external DMA master.

When the QuadSPI module tries to pull data out of an empty TX buffer, FR[TBUB] signals the TX buffer underrun. The TX buffer underrun flag is also asserted when the TX buffer contains less than 32-bits of data and the QuadSPI module tries to pull out data from it. The current IP command leading to the underrun condition is continued until the specified number of bytes is sent to the serial flash memory device. Also, the data that is transferred is in the Fs[15] format. This means, after the underrun flag is set under this condition, it returns Fs until the required number of bytes are not sent. This has been done to ensure that the software does not erase the whole sector after underrun and just reprogramming from failure point serves the purpose. When this sequence command is complete, FR[TBFF] is asserted, indicating that the TX buffer is ready to be written again.

The TX buffer overflow is not signaled explicitly, but TBSR[TRBFL] can monitor the TX buffer fill level.

For more information, see TX Buffer Status Register (TBSR) and Flag Register (FR).

80.4.2.12 Address scheme

Earlier, serial flash memories supported only a 24-bit address space, restricting the maximum memory size of the serial flash memory to 16 MB. The new memory specification supports two types of 32-bit addressing mode in addition to the legacy 24-bit address mode. It also supports segregation of address programmed into Row address and Column address of the flash memory, as per the requirement.

Extended address mode

In this mode, the legacy 24-bit commands are converted to accept 32-bit address commands. The flash memory needs to be configured for the 32-bit address mode. Also, while programming the LUT sequence in QuadSPI for 32-bit mode, the ADDR commands should be programmed with 32d as the operand value with SFACR[CAS] programmed to 0. If a flash memory needs some bits of the address as its column address, then it must be considered that a total of 12 bits are required by the flash memory; however, the number of bits should not exceed 32 because the maximum address supported by QuadSPI is 32 bits.. Each of the memory vendors have a different way of enabling this mode (see the memory specification from memory vendors). For example, the command B7h sent to the Macronix flash memory enables it for the 32-bit address mode.

Extended address register

In this mode, the upper 8 bits of the 32-bit register are provided by the Extended address register in the memory, which provides a specific register that is updated according to the address to be accessed. This effectively converts the legacy 24-bit address command into 32-bit address commands. The memories greater in size than 16 MB consist of banks of 16 MB each. The 8 bits occupied in the extended address register effectively enable a bank. For example, in Spansion memory, when the extended address register is updated with a value of 1h, with the help of the 17h command, it opens Bank1 of the memory. The consequent 24-bit address commands lead to Bank1. The extended address register needs to be updated with the respective value for access to other banks. This effectively converts the legacy 24-bit address command into 32-bit address commands.

Separation of address into rows and columns

This mode has been introduced for flash memories that need addresses segregated into rows and columns. The value in SFACR[CAS] defines the width of the column address required by a flash memory. The actual address to be provided is derived from the incoming address in case of AHB-initiated transactions and the value of SFAR in case of IPS-initiated transactions, if the value of SFACR[CAS] is 0. Otherwise, the actual address takes CAS into consideration. If the value of SFACR[CAS] is 3, then bits 26-3 of the programmed address are sent to the flash memory as its page address. This is in case the flash memory is operating in a 24-bit mode and bits 2-0 are sent as its column address. If a flash memory requirement for column address is less than the number of pads in which address has to be sent, then QuadSPI appends the remaining bits by 0. You must program the operand value in CADDR and CADDR_DDR command accordingly. It must be ensured that the total number of address bits requested by flash memory as its page and column address must not be more than 32 bits.

Word addressable mode for flash memory

[15] Fs represents a value of FFFF_FFFF.

This mode has been introduced for flash memories, which have a word-addressable memory. This means, each address of the flash memory contains one word (two bytes) of data. The value of SFACR [WA] is configured to 1 to enter this mode. QuadSPI internally divides the incoming address in the AHB bus or the address in the SFAR to map it to a valid flash memory location. For example, if the incoming address is 2004h, the controller re-maps this address to access the flash memory location 1002h. If not in this mode, the incoming address 2004h is mapped to flash memory location 2004h.

80.4.3 Module Disable mode

Module Disable mode is a block-specific mode that the QuadSPI can enter to disable serial flash memory clock and AHB command. This mode can be entered by:

- The host software: by writing a '1' to MCR[MDIS]

Below are the condition that must be fulfilled to enter the Module Disable mode:

- SR[BUSY] = 0
- SR[AHBTRN] = 0
- RBSR[RDBFL] = 0
- SR[RXDMA] = 0
- SR[TXDMA] = 0
- None of the flags in FR are enabled as interrupts is set

The conditions mentioned above ensures the following:

- There is no SFM command currently being executed.
- All the data read into the RX buffer from the serial flash memory have been fetched by the application.
- There is no current AHB access.
- There is no active DMA request.
- There is no enabled interrupt that is pending.

Certain read or write operations have a different effect when the QuadSPI is in the Module Disable mode. In the Module Disable mode, not all of the status and flag bits of the QuadSPI module are updated, and writing to them has no effect. Interrupt and DMA request signals cannot be cleared while in the Module Disable mode.

NOTE

It is illegal to issue a new SFM command starting two clock cycles prior to raising the request of entering the Module Disable mode until the QuadSPI stays in this mode.

80.4.4 Leaving Module Disable mode

In the Module Disable mode, the serial flash memory clock and AHB command to the QuadSPI module are switched off.

After the QuadSPI has left this mode and has returned to Normal mode, the execution of the first SFM command is deferred until the clock to drive that part of the module related to the serial flash memory device is available. Depending upon the point in time when the first SFM command is triggered, the actual execution of the command starts with a delay, respective with the re-enabling of the flash memory clock signal.

80.5 Initialization/application information

This section provides the initialization and application information of the QuadSPI module.

80.5.1 Power up and reset

The serial flash memory devices connected to the QuadSPI module might require special voltage characteristics of their inputs during power up or reset. The application must ensure this.

CAUTION

Erase or program commands should be completed before issuing a reset or power cycle to avoid corrupted flash pages. The application shall ensure there is a backup of critical data stored at a different location to enable recovery from corrupted flash pages.

Example: Flash reset sequence

Use the following sequence to reset flash A:

1. Make sure that the flash supports a reset for the condition CS#=high and IOF[3]=low.
2. Set MCR[SWRSTSD] and MCR[SWRSTHD] fields and then clear them.
3. Set MCR[MDIS] field.
4. Reset MCR[ISD3FA] field for flash A.
5. Clear MCR[MDIS] field.
6. Set MCR[MDIS] field.
7. Set MCR[ISD3FA] field for flash A.
8. Clear MCR[MDIS] field.

80.5.2 Available status/flag information

This section provides an overview of the different flags and statuses available, and their interdependencies for different use cases. The SR and FR are the related registers.

80.5.2.1 IP commands

See IP Configuration Register (IPCR) for additional details not explicitly covered in this paragraph.

- IP commands—normal operation

  Writing to IPCR[SEQID] triggers the execution of a new IP command. Given that this is a legal command, SR[IPACC] and SR[BUSY] are asserted simultaneously, immediately after the execution starts.

  After the instruction on the serial flash memory device is complete, these field deassert and FR[TFF] is configured.

- IP commands—error situations

  See Overview_of_Error_Flags for details.

80.5.2.2 AHB commands

See the "Reading serial flash memory data into the QuadSPI module internal buffers" topic in theFlash Memory Read section for details.

- AHB commands—normal operation

  Memory-mapped read access to a serial flash memory address not contained in the AHB buffer triggers the execution of an AHB command. Given that this is a legal command, SR[AHB_ACC] and SR[BUSY] are asserted simultaneously, immediately after the execution starts. After the instruction on the serial flash memory device is complete, these fields are deasserted.

- IP commands—error situations

  See Overview of FR error flags for details.

80.5.2.3 SFM commands

An SFM command consists of an instruction code and all other parameters (for example, size or mode bytes) needed for that specific instruction code. Triggering a command either initiates a transaction on the external serial flash memory or results in an error. See Table 782 for details on errors.

80.5.2.4 Overview of error flags

The following table provides an overview of the different error flags in the FR and additional error-related details.

Table 782. Overview of FR error flags

Error category	Error flag in FR	Command execution on serial flash memory device TFF behavior (in case of IP commands only)	Description
AHB error flag	ABOF	Flash memory transaction continues until it finishes	Set when the module tries to push data into the AHB buffer that exceeds the size of the AHB buffer. Occurs only because of the wrong programming of BUFxCR[ADATSZ].
AHB error flag	AIBSEF	Flash memory transaction is aborted	Total burst size of the AHB transaction is greater than prefetch data size.
AHB error flag	AITEF	Flash memory transaction is aborted	No response is generated from QuadSPI to AHB bus in case of illegal transaction. Also, the watchdog timer expires.
Miscellaneous error flag	ILLINE	Flash memory transaction aborted	Illegal instruction error flag - set when an illegal instruction is encountered by the controller in any of the sequences.
Command arbitration error	IPIEF	TFF not asserted in conjunction with that command	IP command error - caused when IP access is currently in progress (IP_ACC is set) and during: • Write attempt to RBCT register
Buffer-related error	RBOF	TFF is asserted on completion	• RX buffer overrun
Buffer-related error	TBUF	TFF is asserted on completion	• TX buffer underrun

Note that only the buffer-related errors are associated with a transaction on the external serial flash memory. All the other errors do not trigger an actual transaction.

80.5.2.5 IP bus and AHB access command collisions

There are following flags related to this topic: FR[IPIEF]. See the "Reading serial flash memory data into the QuadSPI module internal buffers" topic of the Flash Memory read section for a description of the flags.

80.5.3 Flash memory device selection

Regardless of the SFM command (IP or AHB), the access mode is selected by specifying the 32-bit address value for the following SFM command.

For IP commands, the access mode is selected with the address programmed into the SFAR register. See Serial Flash Address Register (SFAR) for details.

For AHB commands, the access mode is determined by the memory-mapped address. See AMBA Bus Register Memory Map for details.

80.5.4 DMA usage

For a complete description of the DMA module, see the related DMA Controller chapter. This section only provides QuadSPI-specific DMA usage details.

80.5.4.1 DMA usage in normal mode

80.5.4.1.1 Bandwidth considerations

Careful consideration of the throughput rate of the entire chain (serial flash memory -> AHB bus / IP bus -> DMA controller) involved in the read/write data process is essential for a proper operation. Such analysis must take into account not only the data rate provided by the serial flash memory but also the data rate of the AHB bus and the performance of the DMA controller in reading/writing data from/to the RX/TX buffer.

Two figures must match for a proper operation, which means that the data rate provided by the serial flash memory device must not exceed the average RX buffer readout data rate. Otherwise, the longer this state persists, it results into an RX buffer overflow. Similarly, the data consumed by the serial flash memory device must not exceed the average TX buffer fill rate. If this persists, it leads to an underrun.

AHB bus side (data read)

The total number of bus cycles for each DMA minor loop completion is added from the following components:

The following table provides certain examples for typical use cases:

Case 1: DMA needs to read 4 bytes from SRAM and provide to QuadSPI. It costs total four bus clock cycles. Then, DMA handshake adds additional six bus clock cycles, resulting in a total of [6 + 4 * (32/4) = 38] bus clock cycles.

Table 783. Access duration examples for bus clock side

TBCT[WMRK]	Number of bytes per DMA loop	Number of bus clock cycles for DMA minor loop	Time duration of DMA minor loop for 60 MHz bus clock frequency
3	16	6+(16/4)*4 = 22	~366ns
7	32	6+(32/4)*4 = 38	~633ns
11	48	6+(48/4)*4 = 54	~900ns
15	64	6+(64/4)*4 = 70	~1166ns

Case 2: DMA needs to read 32 bytes from SRAM and provide to QuadSPI. DMA handshake takes an additional six bus cycles, with 32 bytes DMA read from SRAM costs (8 + 3) core clock cycles. DMA writes 32 bytes to QuadSPI, takes 2 * (32/4) = 16 bus cycles with one additional CPU access to QuadSPI, costing two bus clock cycles. This results in a total 6 + (8+3)/2 + 2 * (32/4) +2 = 30 bus clock cycles.

Table 784. Access duration examples for bus clock side

TBCT[WMRK]	Number of bytes per DMA loop >	Number of bus clock cycles for DMA minor loop	Time duration of DMA minor loop for 80 MHz bus clock frequency
3	16	6 + (4+3) /2 + (16/4)*2 + 2 = 20	~333ns
7	32	6 + (8+3) /2 + (32/4)*2 + 2 = 30	~500ns
11	48	6 + (4+3)/2*3 + (48/4)*2 + 2 = 44	~733ns
15	64	6 + (8+3) + (64/4)*2 + 2 = 51	~810ns

NOTE

This table figure represents an ideal scenario; actual performance depends on how the chip integrates DMA and QuadSPI modules.

Serial flash memory device side (data read)

The number of serial flash memory cycles can be determined in the following way:

- Number of serial flash memory clock cycles is required to read 4 bytes, corresponding to one RX buffer entry (setup of command and address not considered): , eight cycles for quad mode (SDR) instructions in individual flash memory mode, and so on.
- Overhead because of clock domain crossing: one cycle

The following table lists the number of clock cycles required to read the data from the serial flash memory corresponding to the different settings of RBCT[WMRK]:

Table 785. Access duration examples for serial flash memory side

RBCT[WMRK] setting	Num bytes per DMA loop [1]	Num SCKFx for 80 MHz SCKFx		Time duration of flash memory data readout for 80 MHz SCKFx (~12.5ns period)	
		IFM [2] quad	IFM quad DDR	IFM quad	IFM quad DDR
0	4	8	4	~100ns	~50ns
1	8	16	8	~200ns	~100ns
3	16	32	16	~400ns	~200ns
7	32	64	32	~800ns	~400ns
11	48	96	48	~1200ns	~600ns

1. DMA loop refers to one minor loop completion that is equivalent to one major loop iteration.
2. Individual flash memory mode

NOTE

The table figure represents an ideal scenario; actual performance depends on how the chip integrates with DMA and QuadSPI modules.

A complementary example is when the watermark is set to be too high. In such a case, the time taken by the DMA to read out the RX buffer entries should be lesser than the time taken by the controller to push in the remaining entries in the buffer.

IPS bus side (data write)

The total number of bus cycles for each DMA minor loop completion are added from the following components:

- Overhead for each minor loop, given by DMA controller: assume 10 cycles
- Overhead because to clock domain crossing: assume two cycles
- Number of bus clock cycles required for 16 bytes (128-bit write size): assume four cycles (read/write sequence of DMA controller)

Note that the size of the minor loop is determined by the size of TBCT[WMRK]; therefore, the overhead specified above distributes among (TBCT[WMRK]+1) write accesses of 32-bit each.

The following table provides some examples for typical use cases:

Table 786. Access duration examples for bus clock side

TBCT[WMRK]	Number of bytes per DMA loop [1]	Number of bus clock cycles for DMA minor loop	Time duration of DMA minor loop for 80 MHz bus clock frequency
3	16	12+4 = 16	~200ns
7	32	12+8 = 20	~250ns

1. DMA loop refers to one minor loop completion that is equivalent to one.

NOTE

The table figure represents an ideal scenario; actual performance depends on how the chip integrates with DMA and QuadSPI modules.

Serial flash memory device side (data write)

The number of serial flash memory cycles can be determined in the following way:

- Number of serial flash memory clock cycles required to write 16 bytes, corresponding to four TX buffer entry (setup of command and address not considered): 32 cycles for quad SDR writes in individual flash memory mode.
- Overhead due to clock domain crossing: one cycle

The following table lists the number of clock cycles required to read the data from the serial flash memory corresponding to the different settings of TBCT[WMRK]:

Table 787. Access duration examples for serial flash memory side

TBCT[WMRK] setting	Num bytes per DMA loop [1]	Num SCKFx		Time duration for consuming data at flash memory interface 100 MHz SCKFx (10 ns period)[2]		Time for FIFO to get empty[3]	
		IFM [4] quad	[5] single IO SDR mode	IFM quad	PFM single IO SDR	IFM quad	PFM single IO SDR
3	16	32	64	320ns	640ns	2240ns	4480ns
7	32	64	128	640ns	1280ns	1920ns	3840ns

1. DMA loop refers to one minor loop completion that is equivalent to one major loop iteration.
2. Not all flash memory devices support writes at 100 MHz. See the flash memory data sheet for the actual page program frequency supported.
3. The assumption for these timings is that the TX Fifo is full when the transaction is initiated
4. Individual flash memory mode
5. Parallel flash memory mode

NOTE

The tables mentioned above are only examples which must be correlated with the DMA in the system.

Considering the examples provided in the two tables above for TX FIFO, it is evident that depending on the relationship between the bus clock and serial flash memory clock frequencies, there are settings possible where the serial flash memory consumes data faster than the IPS bus can write data in TX buffer. In these cases, a TX buffer underrun situation occurs. To avoid TX buffer underrun, the data transaction size should be large enough.

80.5.5 Flash memory devices address mapping

QuadSPI is configured in Single mode for the supported flash memory port A

The sizes of the flash memory devices are mapped with the system memory space based on the configurations of the following registers:

- SFA1AD
- SFA2AD

The total memory region for the flash memory devices is mapped between QuadSPI_AMBA_BASE and TOP_ADDR_MEMA2 such that the corresponding CS is asserted based on SFA1AD and SFA2AD register configurations.

80.5.5.1 Single mode

For single-die flash memories, you must write the same value to SFA2AD register that you write to the SFA1AD register.

Following is a programming example for single mode single-die flash memory:

- QuadSPI_AMBA_BASE - 1000_0000h
- SFA1AD[TPADA1] - 2000_0000h
- SFA2AD[TPADA2] - 2000_0000h

The following figure illustrates the memory mapping for single mode QuadSPI configuration.

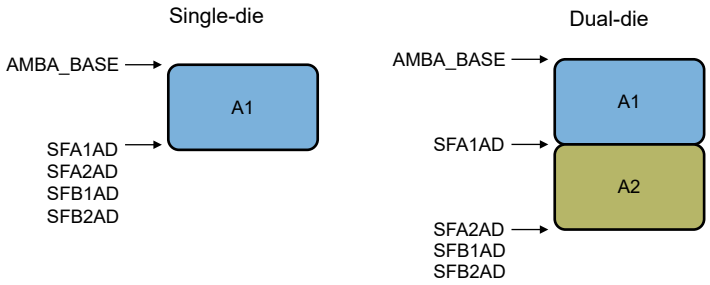


Figure 527. Memory map for Single mode

80.6 Byte ordering – endianness

The following topics show the byte ordering in 64-bit LE configuration for AHB buffer and 32-bit LE for TX/RX buffer.

80.6.1 Programming flash memory data

CPU writes instructions to the TBDR register, such as:

- Write TBDR: 4_03_02_01h
- Write TBDR: 8_07_06_05h

The following table shows the content against each TX buffer entry.

Table 788. Example of QuadSPI TX buffer

TX buffer entry	Content
0	4_03_02_01h
1	8_07_06_05h

Programming the TX buffer into the external serial flash memory device results in the following byte order to be sent to the serial flash memory:

- 01...02...03...04...05...06...07...08

80.6.2 Reading flash memory data into the RX buffer

Reading the content from the same address provides the following sequence of bytes, identical to the write case:

- 01...02...03...04...05...06...07...08

The following table shows the content against each TX buffer entry.

Table 789. Resulting RX buffer content

RX buffer entry	Content
0	4_03_02_01h
1	8_07_06_05h

80.6.2.1 Readout of the RX buffer through RBDRn

The RX buffer content appears at CPU read access through the peripheral bus interface in the following order:

- Read RBDR0: 4_03_02_01h
- Read RBDR1: 8_07_06_05h

80.6.2.2 Readout of the RX buffer through ARDBn

The RX buffer content appears at read access on the AMBA AHB interface at the QuadSPI module boundary:

- 32-bit access: Read ARDB0: 4_03_02_01h
- 32-bit access: Read ARDB1: 8_07_06_05h
- 64-bit access: Read ARDB0: 8_07_06_05_04_03_02_01h

80.6.3 Reading flash memory data into the AHB buffer

Reading the content from the same address as it was written to provides the following sequence of bytes, identical to the write case:

- 01...02...03...04...05...06...07...08

The following table shows the content against each TX buffer entry.

Table 790. Resulting AHB buffer content

AHB buffer entry	Content
0	8_07_06_05_04_03_02_01h

80.6.3.1 Readout of the AHB buffer through memory-mapped read

The AHB buffer content appears at read access on the AMBA AHB interface at the QuadSPI module boundary:

- 32-bit read access: 4_03_02_01h
- 32-bit read access: 8_07_06_05h
- 64-bit read access: 8_07_06_05_04_03_02_01h

80.7 Driving flash memory control signals in single and dual modes

In single and dual modes, the serial flash memory devices that can connect to the QuadSPI module expect additional control signals on the inputs, which are connected to IOFA[3], IOFA[2] in the quad mode. For easy interfacing, the outputs IOFA[3:2] for flash memory A are driven to the logic state given by the configuration fields MCR[ISD3FA], MCR[ISD2FA].

These outputs are driven all the time to the logic level programmed in the MCR except the time when quad commands of the serial flash memory are executed. See the specifications of the related serial flash memory device for details about the inactive level.

80.8 Serial flash memory devices

Several different vendors make flash memory devices with a QuadSPI interface. At present, there is no set standard for the QuadSPI instruction set. The most common commands currently have the same instruction code for all vendors; however, some commands are unique to specific vendors. Some of the example sequences are provided in the following sections.

80.8.1 Example sequences

This section provides the example sequences of the QuadSPI module.

Table 791. Exit 4 x I/O read enhance performance mode (XIP) (Macronix) and read status

Instruction	Pad	Operand	Description
CMD	0h	EBh	4xIO read command
ADDR	2h	18h	24-bit address to be sent on four pads
MODE	2h	0h	2 mode cycles (exit XIP)
DUMMY	2h	4h	4 dummy cycles
READ	2h	8h	Read 64 bits
CMD	0h	5h	Read Status register
READ	0h	1h	Status register data
STOP	0h	0h	Stop, instruction over

80.8.1.1 Fast read sequence (Macronix/Numonyx/Spansion/Winbond)

The following table shows the fast read sequence for Macronix/Numonyx/Spansion/Winbond flash memories.

Table 792. Fast read sequence

Instruction	Pad	Operand	Description
CMD	0h	Bh	Fast read command = 0Bh
ADDR	0h	18h	24 address bits to be sent on one pad
DUMMY	0h	8h	Eight dummy cycles
READ	0h	4h	Read 32 bits on one pad
JMP_ON_CS	0h	0h	Jump to instruction 0 (CMD)

NOTE

If DLL is disabled then JMP_ON_CS or STOP instruction can be used else only STOP instruction can be used.

80.8.1.2 Fast read quad output (Winbond)

The following table shows the fast read quad output sequence for Winbond memories

Table 793. Fast read quad output sequence

Instruction	Pad	Operand	Description
CMD	0h	6Bh	Fast read quad output command = 6Bh
ADDR	0h	18h	24 address bits to be sent on one pad
DUMMY	2h	8h	Eight dummy cycles
READ	2h	4h	Read 32 bits on four pads
JMP_ON_CS	0h	0h	Jump to instruction 0 (CMD)

NOTE

If DLL is disabled then JMP_ON_CS or STOP instruction can be used else only STOP instruction can be used.

80.8.1.3 4 x I/O read enhance performance mode (XIP) (Macronix)

The following table shows the 4 x I/O read enhance performance mode for Macronix flash memories. The enhanced performance mode is also known as XIP mode.

Table 794. Fast read quad output sequence

Instruction	Pad	Operand	Description
CMD	0h	EBh	4xI/O read command = EBh
ADDR	2h	18h	24 address bits to be sent on four pads
MODE	2h	A5h	Two mode cycles
DUMMY	2h	4h	Four dummy cycles
READ	2h	4h	Read 32 bits on four pads
JMP_ON_CS	0h	1h	Jump to instruction 1 (ADDR)

When in XIP mode, the software must ensure that all the flash memories connected to the controller are in this mode. As a part of initializing the controller, all the flash memories might be enabled with XIP by carrying out dummy reads.

80.8.1.4 Dual command page program (Numonyx)

The following table shows the dual command page program sequence for Numonyx flash memories.

Table 795. Dual command page program sequence

Instruction	Pad	Operand	Description
CMD	1h	2h	Dual command page program = 02h on 2 pads
ADDR	1h	18h	24 address bits to be sent on two pads
WRITE	1h	20h	Write 32 bytes on two pads

Table continues on the next page...

Table 795. Dual command page program sequence (continued)

Instruction	Pad	Operand	Description
STOP	0h	0h	Stop, instruction over

80.8.1.5 Sector erase (Macronix/Numonyx/Spansion)

The following table shows the Sector erase sequence for the Macronix/Numonyx/Spansion flash memories.

Table 796. Sector erase sequence

Instruction	Pad	Operand	Description
CMD	0h	20h	Sector erase command = 20h
ADDR	0h	18h	24 address bits to be sent on one pad
STOP	0h	0h	Stop, instruction over

80.8.1.6 Read status register (Macronix/Numonyx/Spansion/Winbond)

The following table shows the read status register sequence for Macronix/Numonyx/Spansion/Winbond flash memories.

Table 797. Read status register sequence

Instruction	Pad	Operand	Description
CMD	0h	0h5	Read status register command = 05h
READ	0h	0h1	Read status register data
STOP	0h	0h	Stop, instruction over

80.9 Sampling of serial flash memory input data

80.9.1 Basic description

QuadSPI is used to read data from the serial flash memory device. Depending on the actual implementation, there is a delay between the internal clocking in the QuadSPI module and the external serial flash memory device. See the following figure for an overview of this scheme.

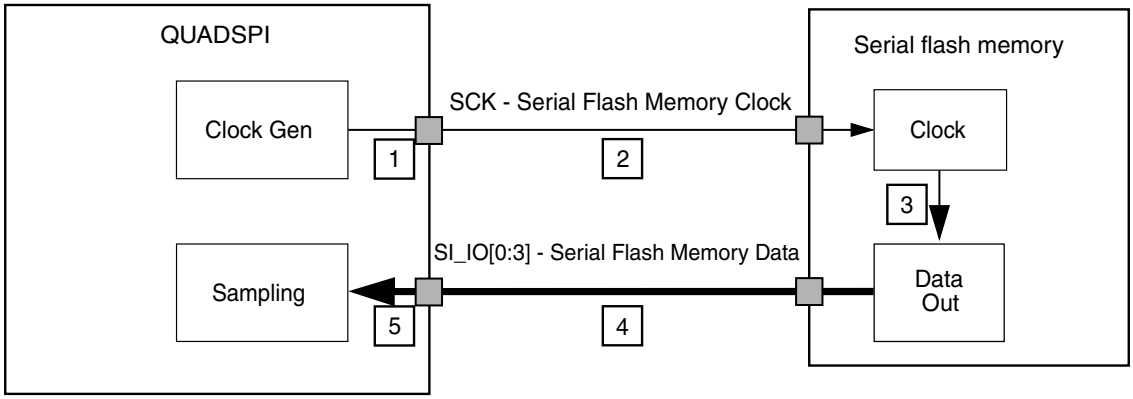


Figure 528. Serial flash memory sampling clock overview

The rising edge of the internal reference clock is taken as timing reference for the data output of the serial flash memory. After a time of t_{total_delay} the data arrives at the internal sampling stage of the QuadSPI module. Considering the figure provided here, the following parts of the delay chain contribute to t_{total_delay}:

- Output delay of the serial flash memory clock output of the device containing the QuadSPI module
- Wire delay of application/PCB from the device containing the QuadSPI module to the external serial flash memory device
- Clock to data out delay of the external serial flash memory device, including input and output delays
- Wire delay of application/PCB from the external serial flash memory device to the device containing the QuadSPI module
- Device delay corresponding to the input data

NOTE

The t_{total_delay} is specific to the characteristics of the actual implementation.

80.9.2 DQS sampling method

80.9.2.1 Basic description

In the DQS mode, the data strobe signal (DQS/RWDS) is used to sample the read data. Here, both DQS and the data sent by the flash memory move in the same direction; therefore, it is relatively easier to achieve at higher frequencies.

When using DQS for SDR reads, QuadSPI internally samples the incoming data on the rising edge of the strobe signal.

The next figure shows the sampling read data in the SDR mode using the DQS.

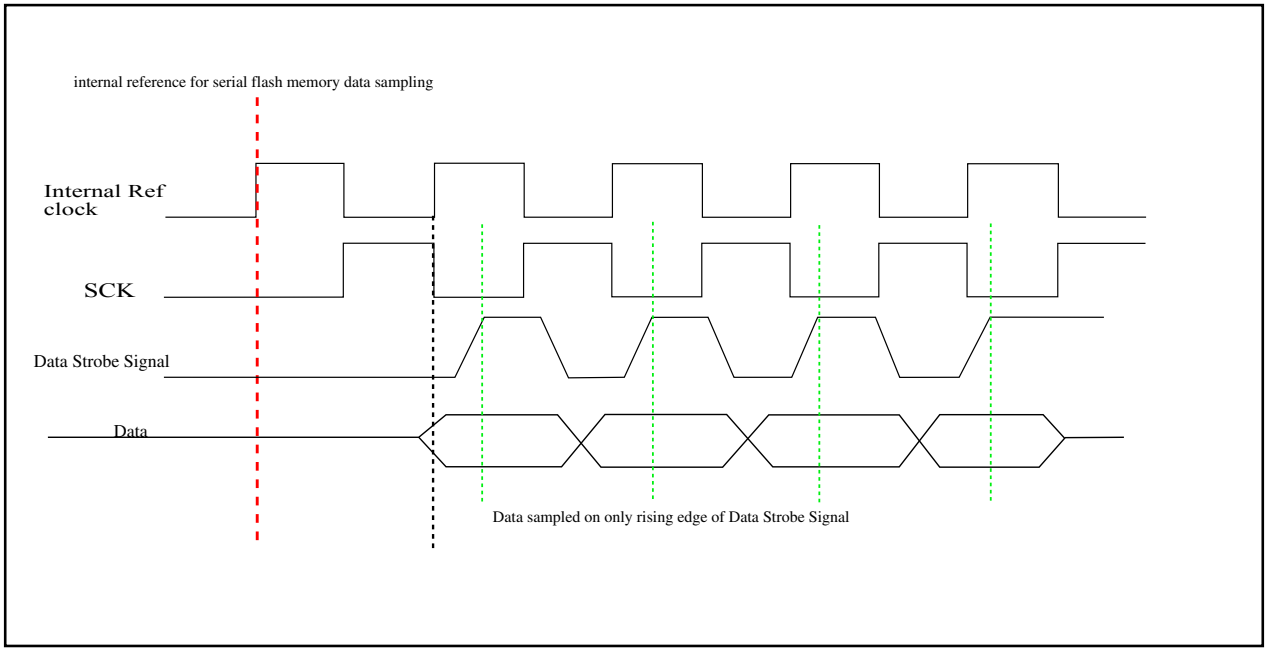


Figure 529. Data strobe functionality in SDR mode for read operation.

NOTE

Consider "Data Strobe Signal" as "Data Strobe Signal driven by memory" and "Data" as "Data from memory".

NOTE

Refer to the Datasheet for specific timing waveforms of QuadSPI

NOTE

For Specific details - Refer to the Data Sheet specification of QuadSPI module.

80.9.2.2 Dummy Pad loopback

The internal clock is loop-backed from the dummy internal pad to compensate data pad delays. This can be enabled by configuring the value of MCR[DQS_FA_SEL] as "01" for flash memory A. This mode can be used with the following configuration:

- High/low frequency delay chain manual programming in bypass mode using DLLCRA[SLV_DLY_COARSE] and DLLCR[FREQEN].
- DLL-assisted sampling only using DLLCRA[DLLEN]

NOTE

Refer to chip-specific QuadSPI section or datasheet for 'Fixed sampling tap specific settings/details' without DLL (in bypass mode).

- DLL-assisted auto data learning using DLLCRA[DLLEN] and along with MCR[DLPEN]

NOTE

Refer to Auto-DataLearning (4x Sampling method) section with DLL for further details

80.10 DLL and delay chain usage

The DLL is a general-purpose, dynamically adaptive clock delay module. It provides the ability to select a quantized delay (in fractions of the clock period) regardless of on-chip variations such as process, voltage, and temperature (PVT). The DLL is suitable for applications where accurate delay adjustment is required, such as in case of DDR interfaces.

DLL working concept

The DLL control loop consists of a counter, reference delay line, and phase detector which operate on the ref_clock reference clock input. The reference clock (clk_ref) is fed into the reference delay line. After reset, a single delay tap is selected. A phase detector is used to detect the condition where a half shift has occurred. In addition to this, signals are generated to either increment or decrement the counter, which controls the delay line (if the half-phase detect condition is not met). Any changes in the delay of the individual elements of the delay chain (because of PVT) automatically cause the phase detector logic to determine if a change in the counter value is required. After the half-phase shift is detected, an internal lock signal is generated, and after a lock event occurs, the slave delay chain is ready to be updated based on a triggering event.

Slave delay chain programming sequence—

Following is the programming sequence for DLL bypass mode.

1. Program DLLCRA[SLV_EN]=1, DLLCRA[SLV_DLL_BYPASS]=1, and DLLCRA[SLAVE_AUTO_UPDT]=0.
2. Program the following fields to provide the desired DQS delay for sampling: DLLCRA[SLV_FINE_OFFSET], DLLCRA[SLV_DLY_COARSE], and DLLCR[FREQEN]. See the chip-specific QuadSPI information for the supported programming settings.
3. Program DLLCRA[SLV_UPD]=1 to load these values in the slave delay chain.
4. Check the slave delay chain update status by polling DLLSR[SLVA_LOCK]=1 and clear DLLCRA[SLV_UPD] after confirming the update state.

Following is the programming sequence for DLL auto update mode.

1. Program DLLCRA[SLV_EN]=1, DLLCRA[SLV_DLL_BYPASS]=0, and DLLCRA[SLAVE_AUTO_UPDT]=1.
2. Program the DLL configuration by using DLLCRA[DLL_REFCNTR] and DLLCRA[DLLRES]. See the chip-specific QuadSPI information for the supported DLL configuration settings.
3. Program the slave settings to delay DQS by using these fields: DLLCRA[SLV_FINE_OFFSET], DLLCRA[SLV_DLY_OFFSET], and DLLCR[FREQEN]. See the chip-specific QuadSPI information for the supported settings.
4. If offset delay needs to be updated on the slave chain, program DLLCRA[SLV_UPD]=1.
5. Enable DLL by programming DLLCRA[DLLEN]=1 and reset DLLCRA[SLV_UPD]=0. Slave delay chain is updated automatically and can be checked by polling DLLSR[SLVA_LOCK]==1

80.11 Data input hold requirement of flash memory

In the DDR mode, the data is valid only for half clock cycle. It is difficult to meet the data input hold time requirement for flash memory. QuadSPI internally delays the data sent to flash memory so that it becomes easy to meet the hold requirement. The FLSHCR[TDH] is used for this purpose. If FLSHCR[TDH] is configured to 0, then the data sent to flash memory is aligned with the internal reference clock of QuadSPI. If it is 1, then the data is aligned to 2x internal reference half clock.

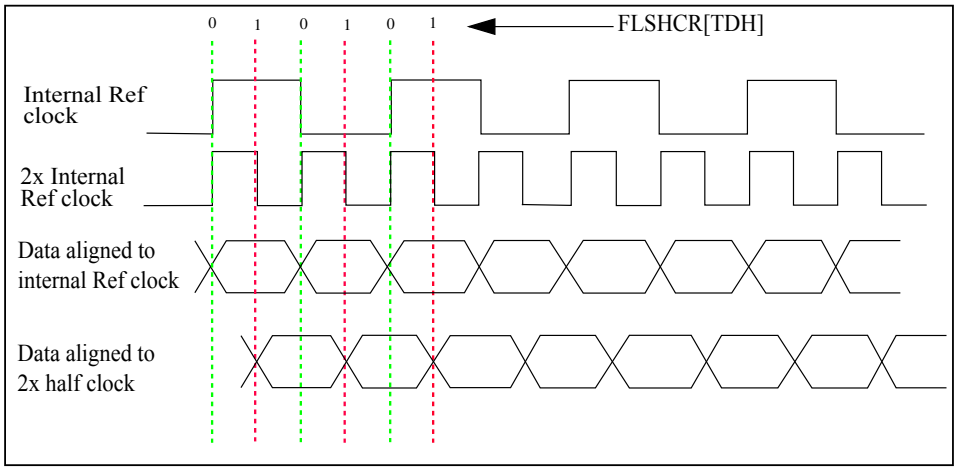


Figure 530. Data hold

80.12 Secure flash protection

This secure flash module enforces access control policies based on the MGID, privilege and secure attributes of the IPS transactions. All accesses throughout the flash device need to be monitored to determine the validity of all accesses. If transaction from a given master has appropriate access rights, it is forwarded to flash, else the access is denied, and an error is generated.

There are two level of checks present in this module, based on MDAD and FRAD descriptors programmed. First level checks the input transactions based on secure attribute and MGID associated with that transaction which are checked according to MDAD. Second level check selects the appropriate FRAD based on address range of transaction. And each FRAD allows the transaction to pass through only when secure, privilege and other MGID related attributes are matching. Else the transaction is not forwarded and IPS bus error/interrupt is generated. Flash regions can be programmed to cover the entire address space to provide a default set of access permissions. For flash read transactions only first level of check is enabled.

Software can bypass both or anyone of these checks by writing to MGC[GVLDFRAD], MGC[GVLDMDAD] and MGC[GVLD] fields.

The IPS masters should write on QuadSPI register (SFAR, IPCR and TBDR) addresses for generating transfers to flash. SFP block monitors these IPS transfers and checks if the write is being done on SFAR or IPCR registers and forwards them for MDAD and FRAD checks. If the transaction passes both checks then it is passed through to QuadSPI register else bus error or interrupt is generated. FRAD check is only done for flash write transactions not for read. Actual write on QuadSPI SFAR and IPCR register is done by SFP module only after the transaction passes MDAD and FRAD checks. There is only a single IPS slave interface which can be used for QuadSPI IPS register access and SFP IPS register access.

An IPS master transaction is secure if IPS_NONSECURE_ACCESS attribute is set to 0 for that transaction. An IPS master transaction is privileged if IPS_SUPERVISOR_ACCESS attribute is set to 1 for that transaction.

QuadSPI SFP related register sets are present here.

80.12.1 MDAD

These descriptors form the first level of checks on the input IPS transaction. There are two target group queues (TG0 and TG1) present in SFP block and each of these queues can be programmed with different set of MGAD descriptors checks in TG*n*MDAD registers. These registers are under access control and can only be programmed by privilege masters. These descriptors (one or both) should be programmed before start of any flash transaction else any IPS write on SFAR, IPCR or TBDR registers will generate an IPS bus error. After programming the target group descriptor the TG*n*MDAD[VLD] should be set to 1 for that register.

There are two target group queues (TG0 and TG1) so two transactions by different master ID can be stored inside SFP block while one of them is being processed by QuadSPI.

80.12.1.1 MDAD checks

- Secure check – This is decided by the TG*n*MDAD[SA] field of the MDAD descriptor. If the fields are set to 01 then queue is reserved for only non-secure transactions. If set to 10 then queue is reserved for only secure transactions. If the bits are set to 11 then this check is bypassed.
- MasterID check – This check compares themasterID of the input transaction with the MID match value set inTG*n*TG*n*MDAD[MIDMATCH]fields and allows only matching transactions inside the queue. There are 6 mask fields (TG*n*TG*n*MDAD[MASK]) inside the descriptor. If the mask type (TG*n*TG*n*MDAD[MASKTYPE]) is set to 0 then the input transactions master ID is ANDed with the mask bits before comparing with the MID match value. If the mask type is set to 1 then the input master ID is ORed with the masks bits and then compared with the MID match bits. Multiple master IDs can be allowed inside the queue using this mask.

When an IPS write is done to SFAR or IPCR registers then SFP block checks the attributes with MDAD descriptors of both the queues. If the attributes match with descriptor of any queue, then the value is written inside that queue and master ID is assigned to that queue. The queue is considered locked once the SEQID bits of IPCR register are written. So software should ensure that IPCR SEQID bits are written only after SFAR and other IPCR bits like DATSZ are written in that queue. The queue will be unlocked when flash transfer is completed for that transaction.

Once a master ID is assigned to a queue then transactions with only that master ID are allowed inside the queue. Any write of SFAR/IPCR is mapped to this queue even if other queue is free. So same master ID cannot have transaction on both queues at a time. The data in the queue is cleared once the flash transaction is finally completed by the QuadSPI.

The data path for input IPS writes to SFAR or IPCR registers are shown in the following figure.

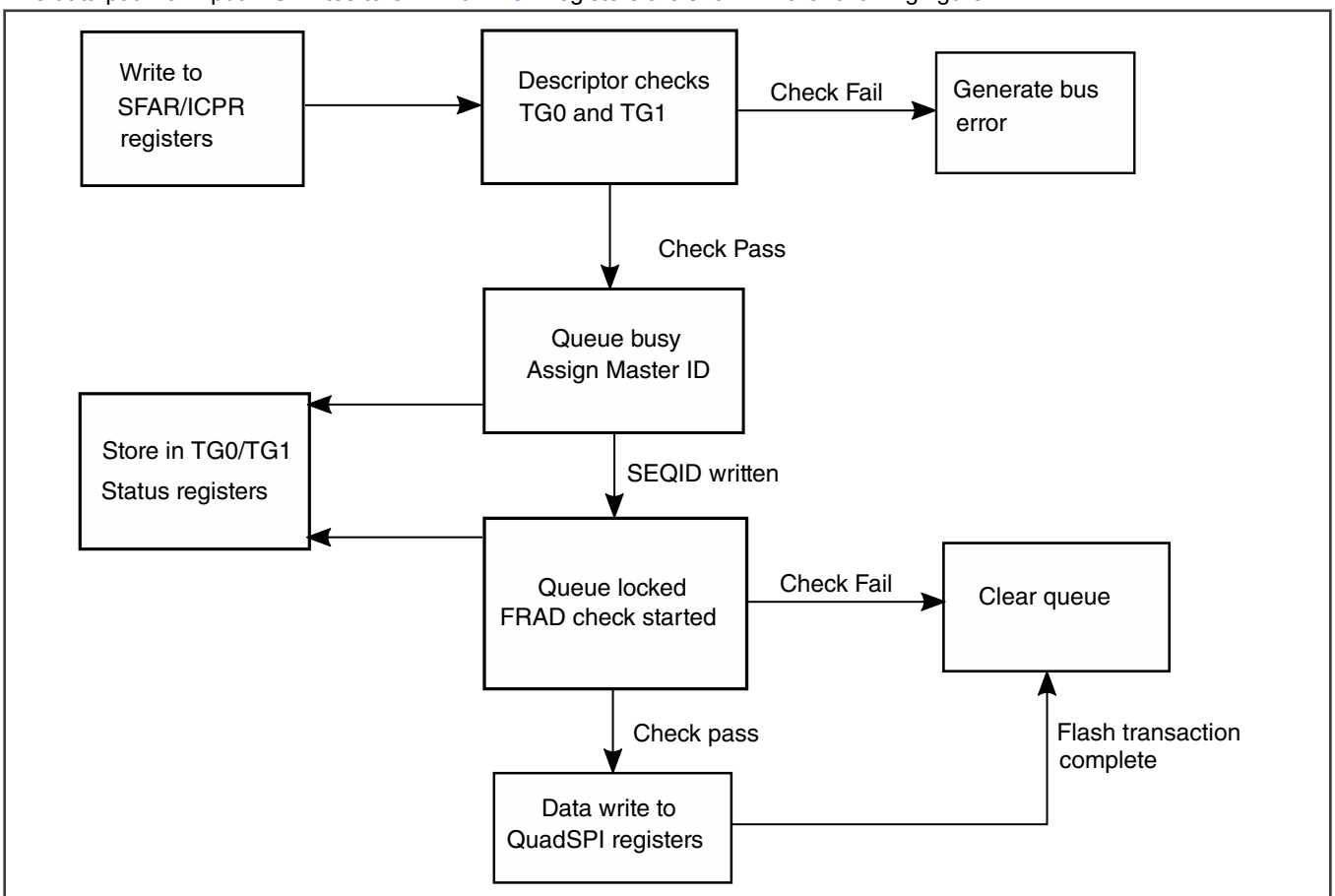


Figure 531. Data path for input IPS write to QuadSPI SFAR or IPCR registers

If the SFAR write passes first level of MDAD checks then the value is written in SFP register TG*n*SFAR and valid bit is set in TG*n*SFARS register and queue is considered as busy. Similarly when ICPR write passes the checks, its value is written into TG*n*IPRCS register and TG*n*IPRCS[VLD] field is set to 1 and queue is considered as locked. It is mandatory that IPCR is

written only after SFAR has been written for a queue and queue is busy. If IPCR is written before SFAR then it will result in IPS transfer error. Master should ensure that it completes the programming sequence for flash access request (write SFAR and IPCR with SEQID) so that target queue is not left after writing SFAR or IPCR without the SEQID. Both queues TG0 and TG1 are processed in round robin arbitration. The queue receiving the data first is processed first and next queue is processed after the first transfer completes.

NOTE

TBDR register write is not allowed until the SFAR and IPCR entries of that master has passed MDAD and FRAD checks and the transaction is queued for writing to QuadSPI. If TBDR registers are written before arbitration win, transfer error will be generated. IPS master can read FSMSTAT register to check the status of their transaction. TBDR access is allowed when FSMSTAT[VLD] field is set and for the master whose master ID matches with fields [11:8] of this register, and field [1:0] are set to 01 or 10. IPCR and SFAR registers are forwarded to QuadSPI only after the TX buffer is filled till watermark level in case of write transactions. IPCR should be written only after SFAR has been written successfully to a queue.

80.12.1.2 Error conditions

The error conditions for first level of MDAD checks and the response is given in table below. These error conditions can occur when SFAR, IPCR or TBDR registers are being written.

NOTE

Queue is considered busy if it has been written successfully once with any of the values SFAR, IDATSZ or PAR. Queue is considered locked when SEQID has been written successfully by passing MDAD checks.

Table 798. MDAD check error conditions

Error condition	Response	Common/Queue error
None of the queue descriptors are programmed	IPS transfer error is generated	Common error
Both queues are locked	IPS transfer error is generated	Common error
Both queues are busy and another master writes whose master ID is not matching with either queue.	IPS transfer error is generated.	Common error
Queue busy and same master writes SFAR or IPCR registers with wrong security attribute	IPS transfer error is generated and queue is cleared (SFAR and IPCR needs to be written again for transfer to start).	Queue error
Queue is locked and same master writes	IPS transfer error is generated	Common error
If master ID of transaction doesn't qualify MID check of any of the queues	IPS transfer error is generated	Common error
If both the queues are empty and security attribute of transaction doesn't match with any of the queues	IPS transfer error is generated	Common error
TBDR register is written before SFAR and IPCR SEQID qualifies MDAD checks	IPS transfer error is generated	-
If none of FRAD descriptors are programmed	IPS transfer error is generated	Common error

Table continues on the next page...

Table 798. MDAD check error conditions (continued)

Error condition	Response	Common/Queue error
IPCR is written before SFAR	IPS transfer error is generated	Common Error - If the transaction doesn't matches the security attribute check or MID check of any MDAD. No common or queue error - If it matches security and MID attribute of any queue.

When a common error is generated its details are latched in IPSERROR register and corresponding error field is set to 1. This register will tell if the transaction failed because of queue locked, MID mismatch or security failure along with the master ID of that transaction. This register will keep the last error transaction latched unless cleared by writing 1 to CLR bit of this register. Then from next cycle onwards this register can latch new errors. Common error will also be generated if none of the FRAD descriptors are programmed. In case a queue is busy and same master writes IPCR or SFAR with wrong security attributes the queue is cleared, the Target Group n SFAR Status (TG0SFARS - TG1SFARS) valid bit goes low. In this case the master should write the SFAR an IPCR again for proper transaction to start.

In case of queue error the details are latched in Target Group n IPCR Status (TG0IPCRS - TG1IPCRS) or Target Group n SFAR Status (TG0SFARS - TG1SFARS) registers. These registers will also keep the error status latched until cleared by writing 1 to CLR bit of respective registers.

80.12.2 FRAD

These descriptors form the second level check for IPS flash write and flash erase transactions. IPS read transaction are not processed by this level of check and are passed through after MDAD checks. There are 8 FRADs which can be programmed by privilege masters. So the access checks can be programmed for up to 8 unique address ranges. Each of the address range is equipped with different access attribute checks and based on this write access can be controlled for the flash region. These descriptors must be programmed before starting transactions on flash.

After programming the FRAD*n*_WORD0, FRAD*n*_WORD1, FRAD*n*_WORD2 and FRAD*n*_WORD3 registers, FRAD*n*_WORD3[VLD] field should be written 1. SFP block doesn't consider the FRAD descriptors which are not valid. If none of the FRAD are programmed, then error interrupt will be generated if enabled, and transaction is not forwarded to QuadSPI.

There is a descriptor lock feature to prevent spurious changes to the FRAD. FRAD*n*_WORD3[LOCK] can be programmed to make all FRAD registers as read only or to allow write access to all or some masters with specific master ID.

80.12.2.1 FRAD checks

- Address range checks – Start and end address for FRAD region are programmed in FRAD*n*_WORD0 and FRAD*n*_WORD1 registers. Start address of IPS flash transaction written in SFAR register and data size programmed in IPCR register, is used to check if the transaction lies within any of the FRAD address ranges. This address range match will be used to select the corresponding FRAD from which other access checks will be done. The start and end address are programmed around 64 KB boundary so 16 MSB are only used for range check..

  The data size should not be programmed as 0 in IPCR register for normal data transfers to flash and actual size should be programmed. In the cases where only flash instruction has to be send and no data has to be written in TBDR register, this DATSZ can be programmed as 0. In case the DATSZ is programmed as 0 in IPCR, SFP module blocks the write access to TBDR register and if any write is done then that generates IPS transfer error.

- Exclusive access lock – Exclusive master access lock can be enabled over any flash address defined under a FRAD. The master ID of the master which enables/owns the exclusive access is captured in FRAD*n*_WORD2[EALO] fields. Once enabled, the exclusive lock can be released only by the same master which enabled the lock by writing on FRAD*n*_WORD3[EAL] fields. The exclusive write access permissions of a flash region are decided as given in table below:

Table 799. Exclusive access lock

FRAD*n*_WORD3[EAL]	Write permissions
00	No lock. The write permissions are decided as set in FRAD*n*_WORD2[MDnACP] fields for respective master domain.
10	Write permissions are revoked for all masters. Any write transaction coming to this flash address region will not be forwarded through and will result in an error.
11	Write permissions are revoked for all masters except the master ID matching the ID specified by FRAD*n*_WORD2[EALO] fields.

- Flash region privilege checks – If the FRAD*n*_WORD3[EAL] fields are set to 00 then flash write access permissions for any IPS master is decided based on the settings done in FRAD*n*_WORD2[MDnACP] fields as given in table below. MD0ACP permission checks are done for transaction coming through target group 0 queue (TG0) and MD1ACP permission checks are for transactions coming through target group 1 queue (TG1).

Table 800. Flash region privilege checks

MDxACP value	Privilege access	Secure access	Write access
111	No	No	Not allowed
	No	Yes	Not allowed
	Yes	No	Allowed
	Yes	Yes	Allowed
110	No	No	Allowed
	No	Yes	Allowed
	Yes	No	Allowed
	Yes	Yes	Allowed
101	No	No	Not allowed
	No	Yes	Not allowed
	Yes	No	Not allowed
	Yes	Yes	Allowed
100	No	No	Not allowed
	No	Yes	Allowed
	Yes	No	Not allowed
	Yes	Yes	Allowed
0xx	-	-	Not allowed

If any transaction fails FRAD check then error interrupt is generated if enabled.

FRAD status registers keep the details of last transaction which lie within the address range of that respective FRAD and will be updated only when a new transaction comes which falls within the same address range.

Flash Region Compare Address Status (FRAD0_WORD4 - FRAD7_WORD4) register stores the flash start address that was compared and Flash Region Compare Status Data (FRAD0_WORD5 - FRAD7_WORD5) contains the master ID, privilege and secure attributes of the transaction. The transaction details are stored in FRAD register for which the address compare check passed. These status registers are not updated in case of read transfers or when FRAD check is disabled. For example, if the transaction address lies in the range set in FRAD2_WORD0 and FRAD2_WORD1 then the transaction details will be stored in

FRAD2_WORD4 and FRAD2_WORD5 registers and valid field [30] will be set. If transaction fails, the security or privilege attribute check for the corresponding FRAD then error field [29] is set in FRADx_WORD5 status register. This bit and other transaction details will remain latched unless cleared by writing '1' to W1C bit of corresponding FRAD in ERRSTAT register.

If transaction address doesn't match any of the FRAD address range or if matches with multiple FRAD address ranges, then also error interrupt is generated. But for no FRAD match, the details will not be stored in any of the FRAD status registers.

FRAD checks are done only for flash write transactions, flash read transactions are passed through to QuadSPI if MDAD checks are passed. At least one FRAD region should be made valid (if the FRAD check is enabled from MGC register) for flash read instructions to go through even when FRAD check is not done for read instructions. SFP module checks if a transaction is read or write from the SEQID written in IPCR register and LUT table programmed inside QuadSPI. Master Read Command (MRC) register shows the read command code which SFP block identifies. There are two command codes which are predefined READ_DDR and READ commands. Software can program two other command code as per its convenience by writing into fields [21:16] or [29:24] and then writing respective valid bit as 1.

80.12.2.2 Error conditions

Any transaction passing the MDAD checks will be checked for FRAD checks and can lead to following error scenarios.

Table 801. Error conditions

Error condition	Response
None of the FRAD are programmed	IPS transfer error will be generated and status will be latched in IPS_ERROR register
Transaction address doesn't fall within any of the FRAD address range	Error interrupt will be generated if enabled
Transaction match with multiple FRAD address range	Error interrupt will be generated if enabled. Error bit will be set, and status will be stored in FRADx_WORD5 status registers for whom the address check passed
Transaction passes FRAD address check but fails privilege or security checks	Error interrupt will be generated if enabled. Error bit will be set, and status will be stored in FRADx_WORD5 status registers for whom the address check passed
Transaction passes FRAD address check but range was locked by EAL bits	Error interrupt will be generated if enabled. Error bit will be set, and status will be stored in FRADx_WORD5 status registers for whom the address check passed
Read transaction	No FRAD check will be done and no error will be generated for FRAD failure. But if FRAD check is enabled in MGC then at least one FRAD should be valid else it will give error during SFAR/IPCR write

80.12.2.3 Atomic commands considerations

For security reasons, the atomic commands (where flash's internal configuration region is accessed) like flash erase etc. should always be programmed in last 6 SEQID locations of LUT. FRAD0 has access to all the LUT locations including these last 6 SEQID. Remaining FRAD regions will not allow write access if the IPCR SEQID is set for any of the last 6 locations of LUT. It will result in FRAD access error and it will set in FRAD status registers. Also the software must set the FRAD0 MDACP to 101 so that write access is allowed only for secure privilege masters. This is done to ensure that no other master can use the atomic commands like flash erase to change the contents of flash. So when any master wants to use these atomic command programmed in last 6 locations of LUT it will have to write FRAD0 start address in SFAR and keep IPCR DATSZ such that it falls under FRAD0 address region and MDACP programmed in FRAD0 are checked.

For flash configuration in case of dual-die the software needs to change the address range of FRAD0 accordingly so that address goes to second die. Similar consideration to be taken in case of dual flash where Port A and B are being used.

80.12.3 TBDR register write lock

SFP module keeps the TBDR write locked for IPS transactions by default. When a IPS master writes SFAR and IPCR transactions and this transaction passes MDAD and FRAD checks then the accesses are unlocked for that master ID. Lock and unlock procedure is shown with an example in figure below:

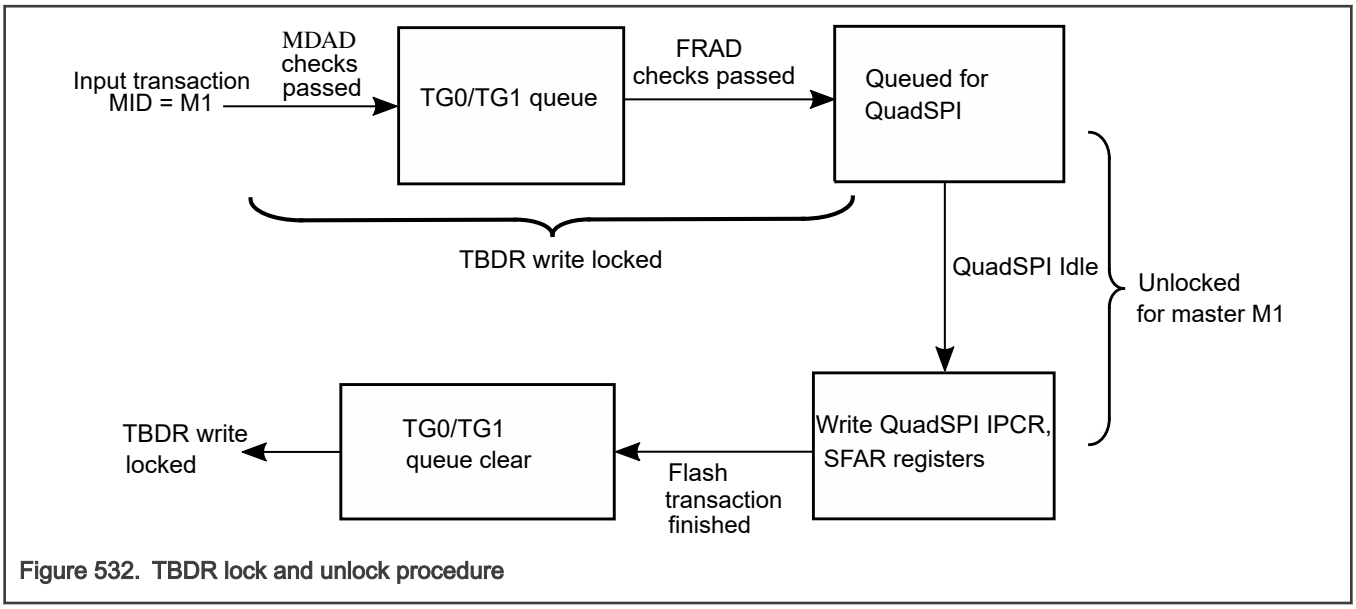


Figure 532. TBDR lock and unlock procedure

If any master writes on TBDR register when it is not unlocked for that master ID it will result in IPS transfer error. IPS master should ensure that it writes the TBDR register once QuadSPI registers have been written by the SFP module. This can be checked by reading FSM Status (FSMSTAT). Bits [11:8] of this register tells which master ID is currently active, and if the value of bits [1:0] is 01 or 10 and bit 31 is set as 1, then the TBDR lock is open for write transactions. It is advisable that for write transactions where IDATSZ is non zero, master clears the TBDR buffer from any residue data and program the TBCT[WMRK] watermark once its request is granted in arbitration (FSMSTAT bit [1:0] are 01). In case of transfers having IDATSZ programmed as 0 the TBDR register remains locked and data can't be written to this register. In case DMA is being used to fill the TBDR buffer, program RSER[TBFDE] field to 1 only when write access is granted and TBDR access is unlocked after checking FSMSTAT[STATE] field. Once DMA completes the TBDR write reset, this field is set back to 0. After this, the SFP will write SFAR and IPCR to QuadSPI triggering the flash transfer. It should be made sure that the DMA should have same master ID as the master which won the arbitration.

80.12.4 Transaction status registers

SFP module has two registers (FSM Status (FSMSTAT) and FlashSeq Request (FLSEQREQ)) which shows the current and last transaction status. These registers are updated only after MDAD and FRAD check are passed.

- FSM Status (FSMSTAT) register shows the current transaction which is queued for QuadSPI when valid field [31] is set. Field [16] of this register if set to 1 then it is a non-read instruction sequence and it is set to 0 then it means that this is a read sequence. Fields [11:8] show the master ID from which this transaction is generated. Fields [1:0] show the current state of active transaction when valid field [31] is set.

  The figure below shows the IPS transaction flow inside SFP block along with various values of bits [1:0]:

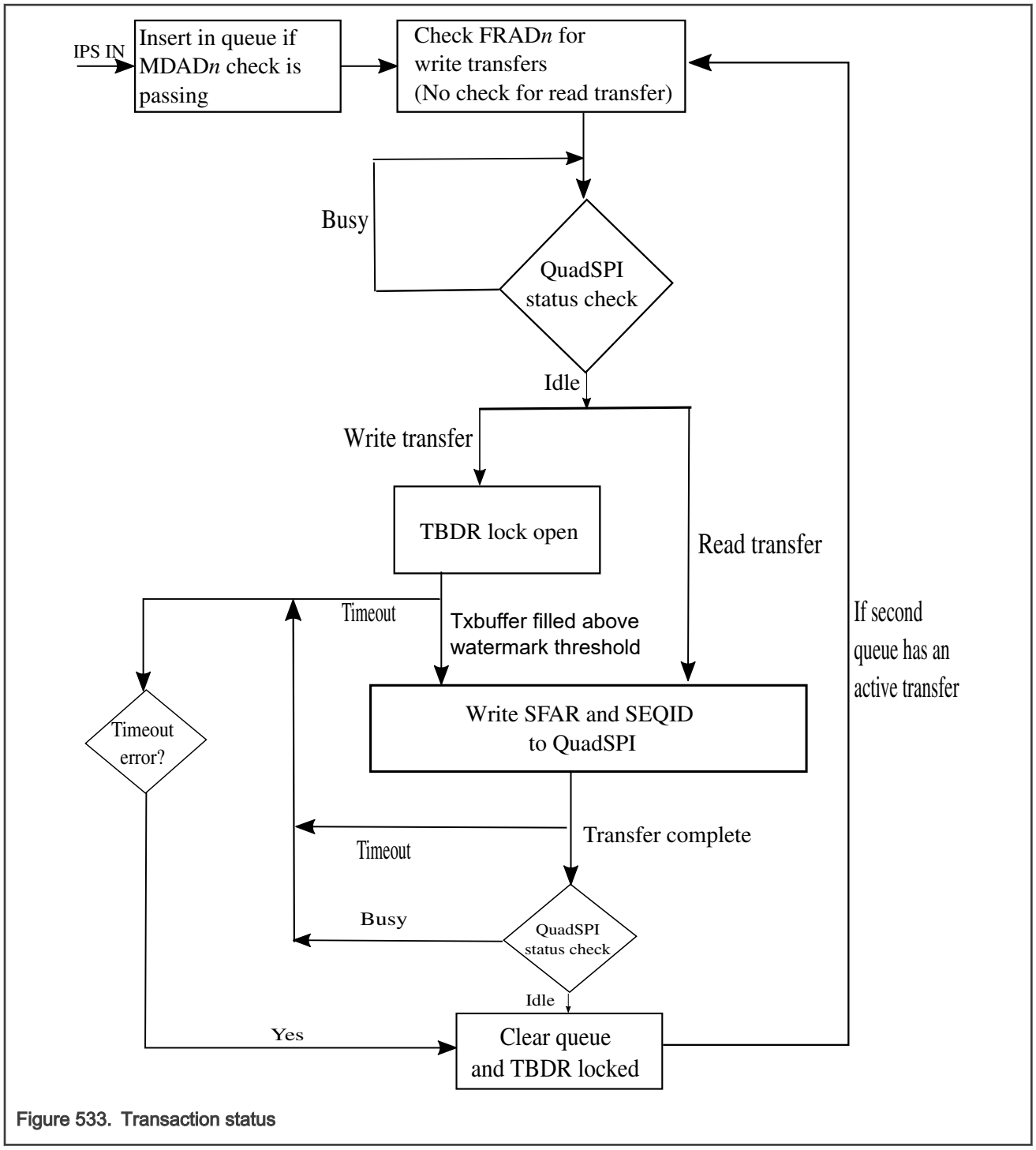


Figure 533. Transaction status

NOTE

If the DATSZ programmed in TG0/TG1 IPCR register is more than the TX buffer watermark threshold value (256-Watermark programmed in TBCT) then it waits for threshold to cross before writing the SFAR and IPCR registers to QuadSPI. Else SFP block waits for one entry to be written to TBDR register and after that it writes the SFAR and IPCR to QuadSPI thus triggering the transfer. In case of read transfers, SFAR and IPCR are written as soon as QSPI is IDLE.

NOTE

QuadSPI is considered IDLE if transaction is finished at flash, No DMA and interrupt pending and RX data is completely read. In case RX data is pending it can be cleared by writing to MCR[CLR_RXF]. Refer to section Timeout error for more details.

FSM state values in FSM Status (FSMSTAT) for different states are in the following table:

Table 802. FSMSTAT register field details

FSMSTAT register fields	
VLD	0 - No IPS transfer is queued for launch to QuadSPI 1 - IPS transfer is queued or running in QuadSPI
STAT	00 - QuadSPI is busy with AHB transfer, any DMA transfer is pending, RDBFL has residue data or any interrupt is pending. 01 - TBDR lock is open. QuadSPI considers IPS transfer. AHB transfer should not start. 10 - TX buffer filled above threshold. Write transfer is triggered. SEQID is written. 11 - Read transfer is triggered. SEQID is written.

For flash write transfers which passes MDAD and FRAD checks and QuadSPI is not busy with any previous transfer, TBDR lock is opened and bits [1:0] shows value of 01. Once the IPS master fills the TX buffer till the level where free entries in TX buffer are less than the watermark threshold set in TX Buffer Control Register (TBCT) and SR[TXWA] flag is de-asserted, SFP block writes the SFAR and IPCR entries to QuadSPI registers and transaction is triggered. To avoid underrun, program the watermark accordingly for size of the data to be transferred. This programming can be done when TBDR access is allowed after arbitration is granted. TBCT Watermark can be programmed as 256-(IDATSZ/4-1) in most of the scenarios. If the DATSZ entry written in IPCR is less such that TX buffer can never be filled till watermark threshold, in such cases SFP block does not wait for the TBCT[WMRK] to trigger and writes the SFAR and ICPR registers as soon as one data entry is written to TBDR register. When the SFAR and IPCR register are written, bits [1:0] are set to 10. In case of flash memory read transfers SFAR and IPCR entries are written to QuadSPI as soon as QuadSPI is IDLE and bits [1:0] are set to 11. QuadSPI is considered IDLE only if no AHB/IPS transfer is ongoing, any DMA transfer is not pending, RDBFL doesn't have residue data and any interrupt is not pending (FR[TFF], FR[TBFF], FR[TBUF], FR[RBDF], FR[RBOF], FR[ABOF], FR[AIBSEF], FR[AITEF], , , FR[IPIEF], , FR[ILLINE], ,). Valid bit [31] goes to 0 once the IPS transfer is completed. IPS master should clear the TX buffer before writing to TBDR register to ensure that no data from previous transaction is left in TX buffer.

If the IPS transaction has passed the MDAD and FRAD checks but QuadSPI is busy with some AHB transfer, then the bits [1:0] are set to 00. Other than ongoing AHB transfer QuadSPI can be busy because of other reasons also like if any TXDMA or RXDMA pending, if read buffer has data pending or if any QuadSPI interrupt is enabled and pending for servicing. So IPS master should ensure that no interrupt is pending or previous read transfer is completed. If the QuadSPI remains busy for a longer time it may result in timeout error.

Software must take care that no AHB transaction is launched when any IPS transaction is ongoing, TBDR lock is open or IPS transaction is getting executed on QuadSPI. It can be checked by reading this FSM status register. if the valid bit is 1 and bits [1:0] are set to 01, 10 or 11 then no AHB transfer should be generated towards QuadSPI.

For any IPS initiated RW transaction which needs two separate transactions at flash interface, like Write-Enable transaction followed by Write-data transaction (Data-Learn flash transaction followed by Read-data transaction) at flash interface must be joined together with JMP_2_SEQ instruction.

- FlashSeq Request (FLSEQREQ) register holds the details of transaction which was last send to QuadSPI. After a flash transaction is completed this register is updated and its valid field [31] is set to 1. This register holds the command type (read or write), transaction master ID (fields [3:0]), target group queue (TG0 or TG1) from which this transaction passed in field [4], privilege and secure attributes for the transaction, SEQID (fields [19:16]) and number of flash descriptor FRAD within whose address range the transaction lies (fields [14:12]).

This register also contains a timeout error field [27] which is set when a timeout error occurs, and transaction is aborted. When the error occurs this register keeps the status latched unless it is cleared by writing 1 to clear field [29].

80.12.5 Timeout error

MTO register can be programmed with a timeout value above which the transaction should be aborted. When the transaction passes MDAD and FRAD level checks and QuadSPI is idle (no AHB transaction is ongoing, no DMA transfer is pending, RDBFL doesn't have any residue data from last transfer and (FR[TFF], FR[TBFF], FR[TBUF], FR[RBDF], FR[RBOF], FR[ABOF], FR[AIBSEF], FR[AITEF], , , FR[IPIEF], , FR[ILLINE], ,) interrupt is not pending) this counter is started. If the transaction is not completed before this timeout counter reaches the value programmed in MTO register the transaction is considered aborted by SFP and an error interrupt is generated if enabled. The error bit is also set in FLSEQREQ register along with the transaction details. After the timeout error, next transaction is written to QuadSPI register which was queued in another target group queue once SR[BUSY] flag goes low.

80.12.6 Arbitration lock

The arbitration lock feature is used to lock the arbitration for a particular queue. This lock can be requested by a master by writing '1' into IPCR[ARB_LOCK] field. Once the transaction passes both the checks (MDAD & FRAD) and wins the arbitration, this lock will be granted. After this, the transaction from the other queue will not be granted unless the arbitration is unlocked by previous queue. The master which locked the arbitration can unlock it by writing '1' to IPCR[ARB_UNLOCK] field. Once the transaction with unlock bit 1 passes the MDAD and FRAD checks, it will unlock the arbitration. After this transaction is finished, the other queue will be granted the arbitration which was waiting.

This arbitration lock feature can be used by software while doing any transaction to flash which may result in a waiting period. For example, in case of NOR flash, after a write is issued to flash there is a waiting period (TPP) for which the flash is inaccessible. In this case, the software should ensure that while writing this SEQID to IPCR register it should lock the arbitration so that transaction coming to other queue is not passed to flash controller which may result in failure. Once the software ensures that waiting period at flash is over, it can read the flash configuration/status registers to confirm. And after getting confirmation, it can send another read command to same configuration register with IPCR[ARB_UNLOCK] field set to '1'. This will ensure that the transaction waiting in another queue are granted. If the flash being used does not have a waiting period attached with transactions, then this ARB_LOCK field can be always kept to 0.

In case software uses this LOCK functionality RWW support will not be available. Because in this case, all the transaction from other queue which was blocked from arbitration will be blocked irrespective of read or write access. So, RWW support (read can go through even when flash is in waiting state) will not be available. The master which has arbitration lock granted can still send read accesses to flash during the waiting period in case the flash supports it.

80.12.7 Soft reset consideration with SFP

If there is any pending transaction in MDAD queue, do not assert QuadSPI reset bits unless you observe unexpected behavior like no response from the module. In that case, assert QSPI reset twice.

In case the soft reset is being used for QSPI, software should take some considerations so that correct functionality is maintained. if the soft reset is asserted when a transaction is being executed by SFP (FSM_STATE [1:0] bits are 01, 10 or 11) then this transaction will be aborted. Different conditions that can occur in case of soft reset are listed in table below:

Table 803. Soft reset conditions

Condition at time of soft reset	FSM state register at time of soft reset	Result	Timeout scenario
Some AHB transaction was ongoing and SFP/IPS is in waiting state.	FSM_STATE[1:0] = 00	After the soft reset if de-asserted the SFP will schedule its transaction over QSPI. FSM_STATE[1:0] will change to other values than 00. Software can fill TBDR in case it is a write transfer or start reading from RBDR.	In this case software doesn’t writes to TBDR or read from RBDR it will result in timeout error.

Table continues on the next page...

Table 803. Soft reset conditions (continued)

Condition at time of soft reset	FSM state register at time of soft reset	Result	Timeout scenario
QSPI was IDLE and SFP transaction has been arbitrated. TBDR is being filled and SFAR, IPCR are not currently written to QSPI.	FSM_STATE[1:0] = 01	Currently TBDR was being filled after soft reset all the entries in TBDR are flushed. So it is expected from software that it will start filling TBDR from start after the reset has de-asserted. SFP will write SFAR and IPCR to QSPI once its normal conditions are met (when number of available spaces in the TX buffer is greater than or equal to the value provided by TBCT[WMRK] or single TBDR write in case watermark check is disabled)	Timeout will occur if TBDR is not filled again after the soft reset is lifted.
SFP has written just SFAR to QSPI IPCR is not written yet. (For a read transfer or write transfer with data size 0)	FSM_STATE[1:0] = 01	After the soft reset is lifted the SFP will write ICPR to QSPI module and FSM_STATE will change to 11 (read) or 10(write). Transaction will finish normally.	Timeout conditions will be same as that for normal transfer.
SFP has written just SFAR to QSPI IPCR is not written yet. (Write data size is non 0)	FSM_STATE[1:0] = 01	After soft reset is lifted the SFP will write IPCR to QSPI module once software fill TBDR (from start) and FSM_STATE will change to 10.	Timeout conditions will be same as that for normal transfer.
SFP has written SFAR and IPCR to QSPI.	FSM_STATE[1:0] = 10 or 11	After the soft reset is de-asserted the current transaction will be deemed aborted. FSM_STATE valid will go low as transfer has finished.	No timeout.

So to conclude in case the soft reset is asserted when FSM_STATE[31] bit is 1 and state bits are 10 or 11 current transaction will be aborted & the next transaction will proceed as normal. In case the state bits are 00 or 11 then it is advisable that software doesn't writes into TBDR after the reset is de-asserted and waits for timeout error to be generated to know when current ongoing SFP transaction is aborted. After clearing the timeout error a new transaction can be launched to SFP or if already a new transaction is present in another queue it will be arbitrated.

80.12.8 Error Interrupt Enable

INT_EN register allow software to disable or enable the interrupt signal generation in case of any of the listed error conditions. This interrupt is generated on 'ipi_int_ored' interface signal of QuadSPI.

- Time out error (INT_EN[TO_ERR])
- IPCR register write error (INT_EN[TG*n*IPCR])
- SFAR register write error (INT_EN[TG*n*SFAR])
- IPS common error (INT_EN[IPS_ERR])
- FRAD access error (INT_EN[FRAD*n*ACC])
- No FRAD address range match error (INT_EN[FRADMTCH])

If the respective field is set to 1 for any of the error conditions, then the interrupt signal is set to 1 when that error occurs and it remains set until the error is cleared by writing 1 to error clear bit as listed in register descriptions.

80.13 Memory map and register definition

This section provides the memory map and register definitions for the QuadSPI module.

80.13.1 Register write access

Following are the write access restriction terms that apply to all the registers:

- Register write access restriction

  For each register field, the write access conditions are specified in the detailed register description.

  The following table provides a description of the write access conditions. If, for a specific register bit or field, none of the given write access conditions is fulfilled, any write attempt to this register bit or field is ignored without any notification. The values of the bits or fields are not changed.

  The condition term [A or B] indicates that the register or field can be written to if at least one of the conditions is fulfilled.

Table 804. Register write access restrictions

Condition	Description
Anytime	No write access restriction
Disabled mode	Write access only if MCR[MDIS] = 1
Normal mode	Write access only if the module is in the normal mode

- Register write access requirements

  You can access all registers using 8-bit, 16-bit, and 32-bit wide operations. For some of the registers, at least a 16-bit or 32-bit wide write access is required to ensure correct operation. This write access requirement is stated in the detailed register description for each affected register.

80.13.2 QUADSPI register descriptions

This section provides the memory map and register definitions for the QuadSPI module.

Access to the following addresses does not result in a transfer error:

- 50h
- 64h
- 120h
- 138h
- 168h
- 188h
- 18Ch

80.13.2.1 QuadSPI memory map

QuadSPI base address: 404C_C000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Module Configuration Register (MCR)	32	RW	000F_404Ch
8h	IP Configuration Register (IPCR)	32	RW	0000_0000h
Ch	Flash Memory Configuration Register (FLSHCR)	32	RW	0000_0303h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
10h	Buffer 0 Configuration Register (BUF0CR)	32	RW	0000_0003h
14h	Buffer 1 Configuration Register (BUF1CR)	32	RW	0000_0002h
18h	Buffer 2 Configuration Register (BUF2CR)	32	RW	0000_0001h
1Ch	Buffer 3 Configuration Register (BUF3CR)	32	RW	8000_0000h
20h	Buffer Generic Configuration Register (BFGENCR)	32	RW	0000_0000h
24h	SOC Configuration Register (SOCCR)	32	RW	0000_0000h
30h	Buffer 0 Top Index Register (BUF0IND)	32	RW	0000_0000h
34h	Buffer 1 Top Index Register (BUF1IND)	32	RW	0000_0000h
38h	Buffer 2 Top Index Register (BUF2IND)	32	RW	0000_0000h
60h	DLL Flash Memory A Configuration Register (DLLCRA)	32	RW	0120_0000h
100h	Serial Flash Memory Address Register (SFAR)	32	RW	0000_0000h
104h	Serial Flash Memory Address Configuration Register (SFACR)	32	RW	0000_0800h
108h	Sampling Register (SMPR)	32	RW	FF00_0000h
10Ch	RX Buffer Status Register (RBSR)	32	R	0000_0000h
110h	RX Buffer Control Register (RBCT)	32	RW	0000_0000h
12Ch	DLL Status Register (DLLSR)	32	R	8000_8000h
134h	Data Learning Status Flash Memory A Register (DLSR_FA)	32	R	0000_0000h
150h	TX Buffer Status Register (TBSR)	32	R	0000_0000h
154h	TX Buffer Data Register (TBDR)	32	RW	0000_0000h
158h	TX Buffer Control Register (TBCT)	32	RW	0000_0000h
15Ch	Status Register (SR)	32	R	0200_3800h
160h	Flag Register (FR)	32	RW	0800_0000h
164h	Interrupt and DMA Request Select and Enable Register (RSER)	32	RW	0000_0000h
16Ch	Sequence Pointer Clear Register (SPTRCLR)	32	RW	0100_0000h
180h	Serial Flash Memory A1 Top Address Register (SFA1AD)	32	RW	7000_0000h
184h	Serial Flash Memory A2 Top Address Register (SFA2AD)	32	RW	7000_0000h
200h - 27Ch	RX Buffer Data Register (RBDR0 - RBDR31)	32	R	0000_0000h
300h	LUT Key Register (LUTKEY)	32	RW	5AF0_5AF0h
304h	LUT Lock Configuration Register (LCKCR)	32	RW	0000_0002h
310h	LUT Register (LUT0)	32	RW	0818_0403h
314h	LUT Register (LUT1)	32	RW	2400_1C08h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
318h - 44Ch	LUT Register (LUT2 - LUT79)	32	RW	0000_0000h
800h	Flash Region Start Address (FRAD0_WORD0)	32	RW	0000_0000h
804h	Flash Region End Address (FRAD0_WORD1)	32	RW	0000_FFFFh
808h	Flash Region Privileges (FRAD0_WORD2)	32	RW	0000_0000h
80Ch	Flash Region Lock Control (FRAD0_WORD3)	32	RW	0000_0000h
810h	Flash Region Compare Address Status (FRAD0_WORD4)	32	R	0000_0000h
814h	Flash Region Compare Status Data (FRAD0_WORD5)	32	R	0000_0000h
820h	Flash Region Start Address (FRAD1_WORD0)	32	RW	0000_0000h
824h	Flash Region End Address (FRAD1_WORD1)	32	RW	0000_FFFFh
828h	Flash Region Privileges (FRAD1_WORD2)	32	RW	0000_0000h
82Ch	Flash Region Lock Control (FRAD1_WORD3)	32	RW	0000_0000h
830h	Flash Region Compare Address Status (FRAD1_WORD4)	32	R	0000_0000h
834h	Flash Region Compare Status Data (FRAD1_WORD5)	32	R	0000_0000h
840h	Flash Region Start Address (FRAD2_WORD0)	32	RW	0000_0000h
844h	Flash Region End Address (FRAD2_WORD1)	32	RW	0000_FFFFh
848h	Flash Region Privileges (FRAD2_WORD2)	32	RW	0000_0000h
84Ch	Flash Region Lock Control (FRAD2_WORD3)	32	RW	0000_0000h
850h	Flash Region Compare Address Status (FRAD2_WORD4)	32	R	0000_0000h
854h	Flash Region Compare Status Data (FRAD2_WORD5)	32	R	0000_0000h
860h	Flash Region Start Address (FRAD3_WORD0)	32	RW	0000_0000h
864h	Flash Region End Address (FRAD3_WORD1)	32	RW	0000_FFFFh
868h	Flash Region Privileges (FRAD3_WORD2)	32	RW	0000_0000h
86Ch	Flash Region Lock Control (FRAD3_WORD3)	32	RW	0000_0000h
870h	Flash Region Compare Address Status (FRAD3_WORD4)	32	R	0000_0000h
874h	Flash Region Compare Status Data (FRAD3_WORD5)	32	R	0000_0000h
880h	Flash Region Start Address (FRAD4_WORD0)	32	RW	0000_0000h
884h	Flash Region End Address (FRAD4_WORD1)	32	RW	0000_FFFFh
888h	Flash Region Privileges (FRAD4_WORD2)	32	RW	0000_0000h
88Ch	Flash Region Lock Control (FRAD4_WORD3)	32	RW	0000_0000h
890h	Flash Region Compare Address Status (FRAD4_WORD4)	32	R	0000_0000h
894h	Flash Region Compare Status Data (FRAD4_WORD5)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
8A0h	Flash Region Start Address (FRAD5_WORD0)	32	RW	0000_0000h
8A4h	Flash Region End Address (FRAD5_WORD1)	32	RW	0000_FFFFh
8A8h	Flash Region Privileges (FRAD5_WORD2)	32	RW	0000_0000h
8ACh	Flash Region Lock Control (FRAD5_WORD3)	32	RW	0000_0000h
8B0h	Flash Region Compare Address Status (FRAD5_WORD4)	32	R	0000_0000h
8B4h	Flash Region Compare Status Data (FRAD5_WORD5)	32	R	0000_0000h
8C0h	Flash Region Start Address (FRAD6_WORD0)	32	RW	0000_0000h
8C4h	Flash Region End Address (FRAD6_WORD1)	32	RW	0000_FFFFh
8C8h	Flash Region Privileges (FRAD6_WORD2)	32	RW	0000_0000h
8CCh	Flash Region Lock Control (FRAD6_WORD3)	32	RW	0000_0000h
8D0h	Flash Region Compare Address Status (FRAD6_WORD4)	32	R	0000_0000h
8D4h	Flash Region Compare Status Data (FRAD6_WORD5)	32	R	0000_0000h
8E0h	Flash Region Start Address (FRAD7_WORD0)	32	RW	0000_0000h
8E4h	Flash Region End Address (FRAD7_WORD1)	32	RW	0000_FFFFh
8E8h	Flash Region Privileges (FRAD7_WORD2)	32	RW	0000_0000h
8ECh	Flash Region Lock Control (FRAD7_WORD3)	32	RW	0000_0000h
8F0h	Flash Region Compare Address Status (FRAD7_WORD4)	32	R	0000_0000h
8F4h	Flash Region Compare Status Data (FRAD7_WORD5)	32	R	0000_0000h
900h	Target Group n Master Domain Access Descriptor (TG0MDAD)	32	RW	0000_0000h
904h	Target Group n SFAR Address (TG0SFAR)	32	R	0000_0000h
908h	Target Group n SFAR Status (TG0SFARS)	32	RW	0000_0000h
90Ch	Target Group n IPCR Status (TG0IPCRS)	32	RW	0000_0000h
910h	Target Group n Master Domain Access Descriptor (TG1MDAD)	32	RW	0000_0000h
914h	Target Group n SFAR Address (TG1SFAR)	32	R	0000_0000h
918h	Target Group n SFAR Status (TG1SFARS)	32	RW	0000_0000h
91Ch	Target Group n IPCR Status (TG1IPCRS)	32	RW	0000_0000h
920h	Master Global Configuration (MGC)	32	RW	A800_0000h
924h	Master Read Command (MRC)	32	RW	0050_0E07h
928h	Master Timeout (MTO)	32	RW	FFFF_FFFFh
92Ch	FlashSeq Request (FLSEQREQ)	32	RW	0000_0000h
930h	FSM Status (FSMSTAT)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
934h	IPS Error (IPSERROR)	32	RW	0000_0000h
938h	Error Status (ERRSTAT)	32	RW	0000_0000h
93Ch	Interrupt Enable (INT_EN)	32	RW	0000_0000h

80.13.2.2 Module Configuration Register (MCR)

Offset

Register	Offset
MCR	0h

Function

This register holds configuration data associated with the QuadSPI operation.

NOTE

When out of reset, after first initial MCR programming and exiting module disable mode (that is, when the value of MCR[MDIS] is 0), you must write 1 to MCR[SWRSTSD] and release it after six (three system and three flash memory) clock cycles.

Special write-access is permitted in different modes:

- DQS_FA_SEL: Disabled mode
- ISD3FA, ISD2FA: Disabled mode
- All other fields: Anytime

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved	Reserved	Reserved		Reserved	Reserved	DQS_FA_SEL		Reserved	Reserved	0		Reserved	Reserved	ISD3FA	ISD2FA
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	MDIS	Reserved	Reserved	0	0	0	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved		SWRSTHD	SWRSTSD
W					CLR_TXF	CLR_RXF										
Reset	0	1	0	0	0	0	0	0	0	1	0	0	1	1	0	0

Fields

Field	Function
31 —	Reserved
30 —	Reserved
29-28 —	Reserved
27 —	Reserved
26 —	Reserved
25-24 DQS_FA_SEL	DQS clock for sampling read data at flash memory A Selects DQS clock for sampling read data at flash memory A QuadSPI port 00b - Reserved 01b - Pad loopback 10b - Reserved 11b - Reserved
23 —	Reserved
22 —	Reserved
21-20 —	Reserved
19 —	Reserved
18 —	Reserved
17 ISD3FA	Idle signal drive IOFA[3] flash memory A Determines the logic level that the IOFA[3] output of the QuadSPI module is driven to in the inactive state. See Driving flash memory control signals in single and dual modes for details. 0b - IOFA[3] is driven to logic L

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - IOFA[3] is driven to logic H
16 ISD2FA	Idle signal drive IOFA[2] flash memory A Determines the logic level that the IOFA[2] output of the QuadSPI module is driven to in the inactive state. See Driving flash memory control signals in single and dual modes for details. 0b - IOFA[2] is driven to logic L. 1b - IOFA[2] is driven to logic H.
15 —	Reserved
14 MDIS	Module disable Allows the clock to the non-memory mapped logic in the QuadSPI to be stopped. 0b - Enable QuadSPI clocks 1b - Allow external logic to disable QuadSPI clocks
13 —	Reserved
12 —	Reserved
11 CLR_TXF	Clear TX FIFO/buffer This is a self-clearing field that invalidates the TX buffer content. **NOTE** Software must wait for at least five system cycles and three flash cycles after writing '1' to this field. 0b - No action 1b - Read and write pointers of the TX buffer are reset to 0 and TBSR[TRCTR] is reset to 0.
10 CLR_RXF	Clear RX FIFO This is a self-clearing field that invalidates the RX buffer content. 0b - No action 1b - Read and write pointers of the RX buffer are reset to 0 and RBSR[RDBFL] is reset to 0.
9 —	Reserved
8 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
7 —	Reserved
6 —	Reserved
5 —	Reserved
4 —	Reserved
3-2 —	Reserved
1 SWRSTHD	Software reset for AHB domain 0b - De-assert Software reset 1b - AHB domain flops are reset. This field does not reset configuration registers. It is advisable to reset both the serial flash memory domain and AHB domain at the same time. Resetting only one domain might lead to side effects. **NOTE** The software resets need the clock to be running to propagate to the design. The value of MCR[MDIS] should be 0 when the software reset bits are asserted. Also, before they can be deasserted again (by setting MCR[SWRSTHD] to 0), it is recommended to set the value of MCR[MDIS] to 1. After the software resets have been deasserted, the normal operation can be started by setting MCR[MDIS] to 0. **NOTE** Software must wait for at least three system cycles and three flash cycles after changing the value of this field.
0 SWRSTSD	Software reset for serial flash memory domain 0b - De-assert Software reset 1b - Serial flash memory domain flops are reset. This field does not reset configuration registers. It is advisable to reset both the serial flash memory domain and AHB domain at the same time. Resetting only one domain might lead to side effects.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE The software resets need the clock to be running to propagate to the design. The value of MCR[MDIS] should therefore be 0 when the software reset bits are asserted. Also, before they can be deasserted again (by specifying 0 as the value for MCR[SWRSTSD]), it is recommended to specify 1 as the value for MCR[MDIS]. After the software resets are deasserted, the normal operation can be started by specifying 0 as the value for MCR[MDIS]. **NOTE** Software must wait for at least three system cycles and three flash cycles after changing the value of this field.

80.13.2.3 IP Configuration Register (IPCR)

Offset

Register	Offset
IPCR	8h

Function

This register provides all the configuration required for an IP-initiated command, which can be triggered by writing in the SEQID field of this register. If the SEQID field is written successfully, a new command to the external serial flash memory is initiated per the sequence pointed to by this field. See Normal mode for details on command triggering and command execution.

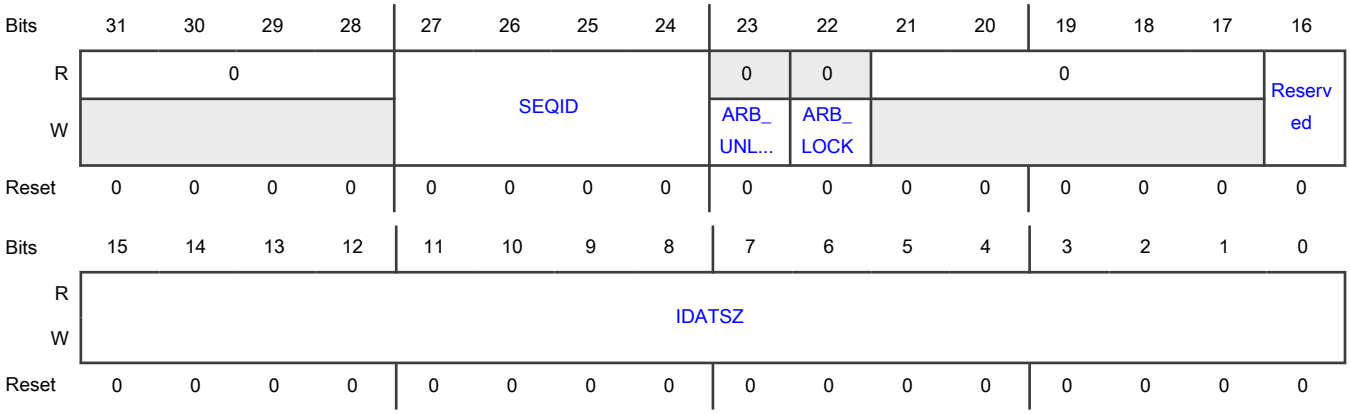
Special write-access is permitted if:

- SR[IP_ACC]=0

NOTE
Only one of the fields should be written '1' at a time out of ARB_LOCK and ARB_UNLOCK. If both the fields are written '1' at same time then it will invert the lock status. If the arbitration was already locked, it will unlock it and vice-versa.

NOTE
If MDAD and FRAD checks are enabled in MGC register but none of the MDAD and FRAD descriptors are valid, then any write on this register will generate a bus transfer error

Diagram



Fields

Field	Function
31-28 —	Reserved
27-24 SEQID	Points to a sequence in the LUT This field contains the sequence index of the LUT. See LUT for details. Each sequence index can accommodate up to 10 instructions (2 instructions per register). In case of read command, a write to this field triggers a transaction on the serial flash memory interface if QSPI SFM is IDLE. In case of write command, a write to this fields unlocks the TBDR access if QSPI SFM is not busy. Refer to section Secure flash protection for more details
23 ARB_UNLOCK	Arbitration Unlock Writing 1 to this bit unlocks the arbitration. Writing 0 will have no effect. The arbitration unlock will be done only if the transaction passes both MDAD and FRAD checks. To lock the arbitration write 1 to ARB_LOCK bit. This bit is always read as 0.
22 ARB_LOCK	Arbitration Lock Writing 1 to this bit locks the arbitration for that specific target queue. Writing 0 will have no effect. The arbitration lock will be granted only if the transaction passes both MDAD and FRAD checks. To unlock the arbitration write 1 to ARB_UNLOCK bit. This bit is always read as 0.
21-17 —	Reserved
16 —	Reserved
15-0 IDATSZ	IP data transfer size This field defines the data transfer size, in bytes, of the IP command.

80.13.2.4 Flash Memory Configuration Register (FLSHCR)

Offset

Register	Offset
FLSHCR	Ch

Function

This register contains the timings that are specific to the flash memory device. The QuadSPI controller must meet these timings for the device to function correctly.

Special write-access is permitted if:

- SR[AHB_ACC] = 0
- SR[IP_ACC] = 0

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0														TDH	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				TCSH				0				TCSS			
W																
Reset	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1	1

Fields

Field	Function
31-18 —	Reserved
17-16 TDH	Serial flash memory data in hold time This field helps in meeting the data in hold time requirement of a flash memory. It is valid only in the DDR mode. **NOTE** This field should be set to 0x00 in the SDR mode (MCR[DDR_EN]=0). See Data input hold requirement of flash memory for details. The valid TDH programming options are shown below and the other combinations are invalid. 00b - Data aligned with the posedge of internal reference clock of QuadSPI 01b - Data aligned with 2x serial flash memory half clock

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-12 —	Reserved
11-8 TCSH	Serial flash memory CS hold time This hold time is in terms of serial flash memory clock cycles, and it must be greater than or equal to two flash memory clock cycles . Refer the chip datasheet for the exact value.
7-4 —	Reserved
3-0 TCSS	Serial flash memory CS setup time This setup time is in terms of serial flash memory clock cycles, and it must be greater than or equal to two flash memory clock cycles. Refer the chip Datasheet for the exact value.

80.13.2.5 Buffer 0 Configuration Register (BUF0CR)

Offset

Register	Offset
BUF0CR	10h

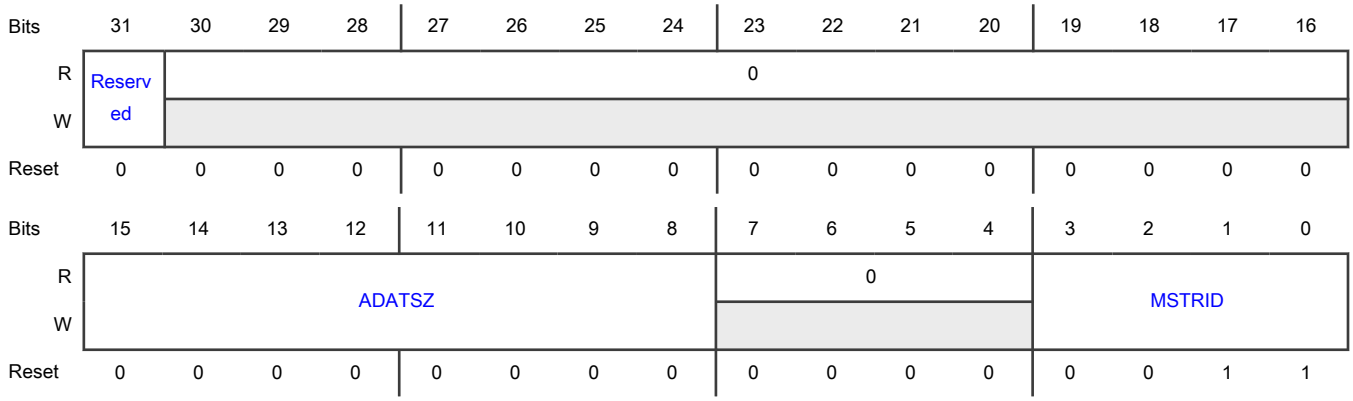
Function

This register provides the configuration for any read access routed to buffer0, which happens when the master ID of the incoming AHB request matches BUF0CR[MSTRID]. Any buffer "miss" leads to a serial flash memory transaction being triggered per the sequence pointed to by BFGENCR[SEQID].

Special write-access is permitted if:

- SR[AHB_ACC] = 0

Diagram



Fields

Field	Function
31 —	Reserved
30-16 —	Reserved
15-8 ADATSZ	AHB data transfer size Defines the read data transfer size in 8 bytes of an AHB triggered read access to serial flash memory. For example, a value of 0x2 sets transfer size to 16 bytes. When ADATSZ = 0, the data size mentioned in the sequence pointed to by the SEQID field overrides this value. The software should ensure that this transfer size is not greater than the size of the buffer.
7-4 —	Reserved
3-0 MSTRID	Master ID ID of the AHB master associated with BUFFER 0 Any AHB read access with this master ID is routed to this buffer. You must ensure that the master IDs associated with all buffers are different. **NOTE** See the chip-specific QuadSPI information for details about master IDs and their corresponding components.

80.13.2.6 Buffer 1 Configuration Register (BUF1CR)

Offset

Register	Offset
BUF1CR	14h

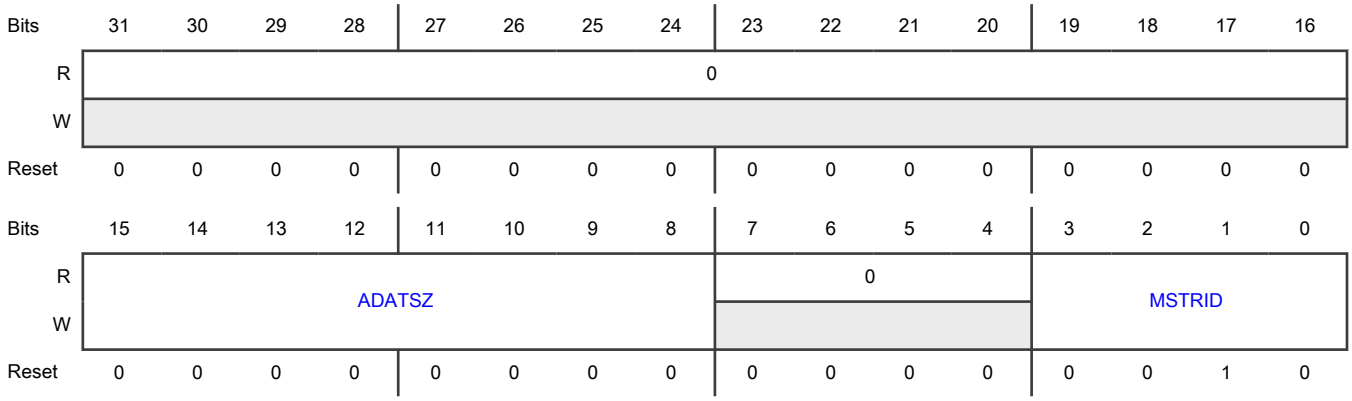
Function

This register provides the configuration for any access routed to buffer 1, which happens when the master ID of the incoming AHB request matches the MSTRID field of this register. Any buffer "miss" leads to the buffer being flushed and a serial flash memory transaction being triggered per the sequence pointed to by BFGENCR[SEQID].

Special write-access is permitted if:

- SR[AHB_ACC] = 0

Diagram



Fields

Field	Function
31-16 —	Reserved
15-8 ADATSZ	AHB data transfer size This field defines the read data transfer size in 8 bytes of an AHB triggered read access to serial flash memory. For example, a value of 0x2 sets the transfer size to 16 bytes. When ADATSZ = 0, the data size mentioned in the sequence pointed to by the SEQID field overrides this value. Software should ensure that this transfer size is not greater than the size of this buffer.
7-4 —	Reserved
3-0 MSTRID	Master ID ID of the AHB master associated with BUFFER 1 Any AHB read access with this master ID is routed to this buffer. You must ensure that the master IDs associated with all buffers are different. **NOTE** See the chip-specific QuadSPI information for details about master IDs and their corresponding components.

80.13.2.7 Buffer 2 Configuration Register (BUF2CR)

Offset

Register	Offset
BUF2CR	18h

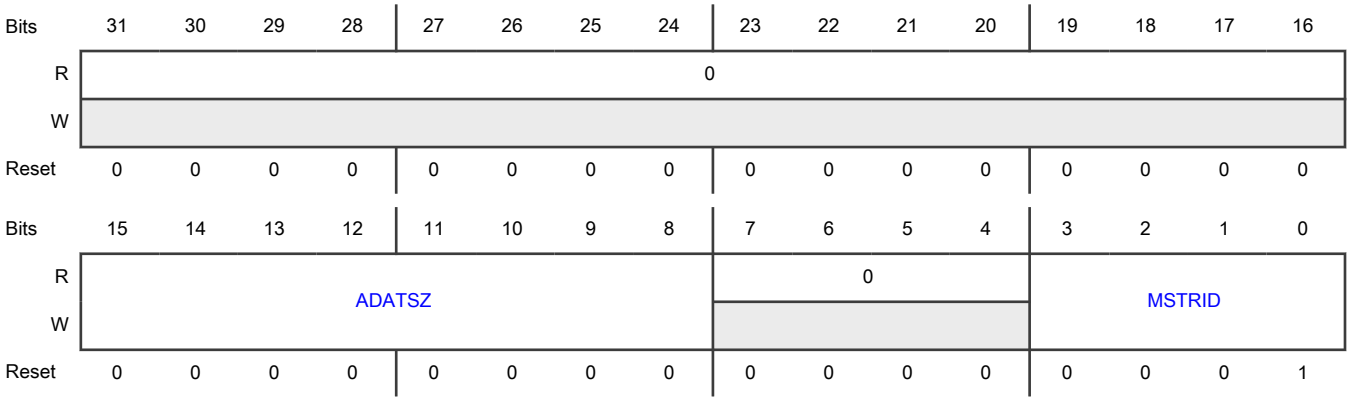
Function

This register provides the configuration for any access routed to buffer 2, which happens when the master ID of the incoming AHB request matches the MSTRID field of this register. Any buffer "miss" leads to the buffer being flushed and a serial flash memory transaction being triggered per the sequence pointed to by BFGENCR[SEQID].

Special write-access is permitted if:

- SR[AHB_ACC] = 0

Diagram



Fields

Field	Function
31-16 —	Reserved
15-8 ADATSZ	AHB data transfer size This field defines the read data transfer size in 8 bytes of an AHB triggered read access to the serial flash memory. For example, a value of 0x2 sets transfer size to 16 bytes. When ADATSZ = 0, the data size mentioned in the sequence pointed to by the SEQID field overrides this value. The software should ensure that this transfer size is not greater than the size of this buffer.
7-4 —	Reserved
3-0 MSTRID	Master ID The ID of the AHB master associated with BUFFER2. Any AHB read access with this master ID is routed to this buffer. It must be ensured that the master IDs associated with all buffers are different. **NOTE** See the chip-specific QuadSPI information for details about master IDs and their corresponding components.

80.13.2.8 Buffer 3 Configuration Register (BUF3CR)

Offset

Register	Offset
BUF3CR	1Ch

Function

This register provides the configuration for any access to buffer 3.

An access is routed to buffer 3 when the master ID of the incoming AHB request matches the MSTRID field of BUF3CR. Any buffer "miss" leads to the buffer being flushed and a serial flash memory transaction being triggered per the sequence pointed to by BFGENCR[SEQID].

In case the value of the ALLMST field is not 1, any such transaction (where master ID does not match any of the MSTRID fields) is returned with an ERROR response.

Special write-access is permitted if:

- SR[AHB_ACC] = 0

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ALLMST	0													Reserved	
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ADATSZ								0				MSTRID			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 ALLMST	All master enable When set, buffer3 acts as an all-master buffer. Any AHB access with a master ID not matching with the master ID of buffer0, buffer1, or buffer2 is routed to buffer3. When set, the MSTRID field of this register is ignored.
30-18 —	Reserved
17-16 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-8 ADATSZ	AHB data transfer size Defines the read data transfer size in 8 bytes of an AHB triggered read access to serial flash memory. When ADATSZ = 0, the data size mentioned in the sequence pointed to by the SEQID field overrides this value.
7-4 —	Reserved
3-0 MSTRID	Master ID ID of the AHB master associated with BUFFER 3. Any AHB read access with this master ID is routed to this buffer. You must ensure that the master IDs associated with all buffers are different. **NOTE** See the chip-specific QuadSPI information for details about master IDs and their corresponding components.

80.13.2.9 Buffer Generic Configuration Register (BFGENCR)

Offset

Register	Offset
BFGENCR	20h

Function

This register provides generic configuration to any of the buffer accesses. Any buffer "miss" leads to the buffer being flushed and a serial flash memory transaction being triggered per the sequence pointed to by the SEQID field.

Special write-access is permitted if:

- SR[AHB_ACC] = 0

This register is access controlled and can only be programmed by privilege masters. Write access is not permitted if MGC[11]=1b.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				0										0	Reserved
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SEQID				0											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved
27-18 —	Reserved
17 —	Reserved
16 —	Reserved
15-12 SEQID	Points to a sequence in the LUT. This field contains the sequence index of the LUT... See LUT. **NOTE** If the sequence pointer differs in the new and the previous sequences, you should reset it. See sequence pointer clear register for more information.
11-0 —	Reserved

80.13.2.10 SOC Configuration Register (SOCCR)

Offset

Register	Offset
SOCCR	24h

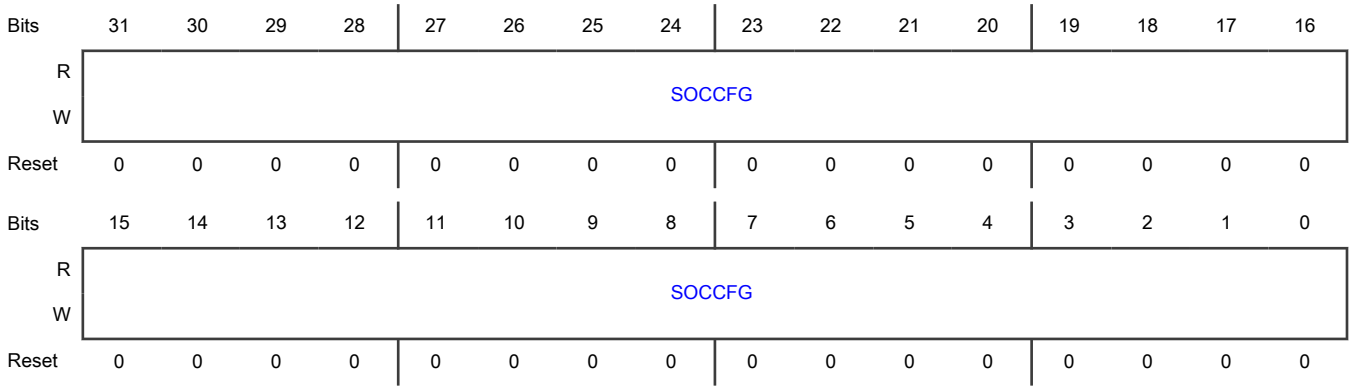
Function

This register is programmed at the chip level for QuadSPI configuration. For details, see chip-specific QuadSPI information.

Special write-access is permitted if:

- SR[AHB_ACC] = 0

Diagram



Fields

Field	Function
31-0 SOCCFG	SOC configuration This field configuration is specific to chip. For details, see chip-specific QuadSPI information.

80.13.2.11 Buffer 0 Top Index Register (BUF0IND)

Offset

Register	Offset
BUF0IND	30h

Function

This register specifies the top index for buffer 0, which defines its size. Note that the three LSBs of this register are set to 0. This ensures that the buffer is 64-bit aligned because each buffer entry is 64-bits long.

The register value should be set to the desired number of bytes. For example, setting BUF0IND[31:3] to 0 gives 0 bytes, setting the value to 1 gives 8 bytes, and so on.

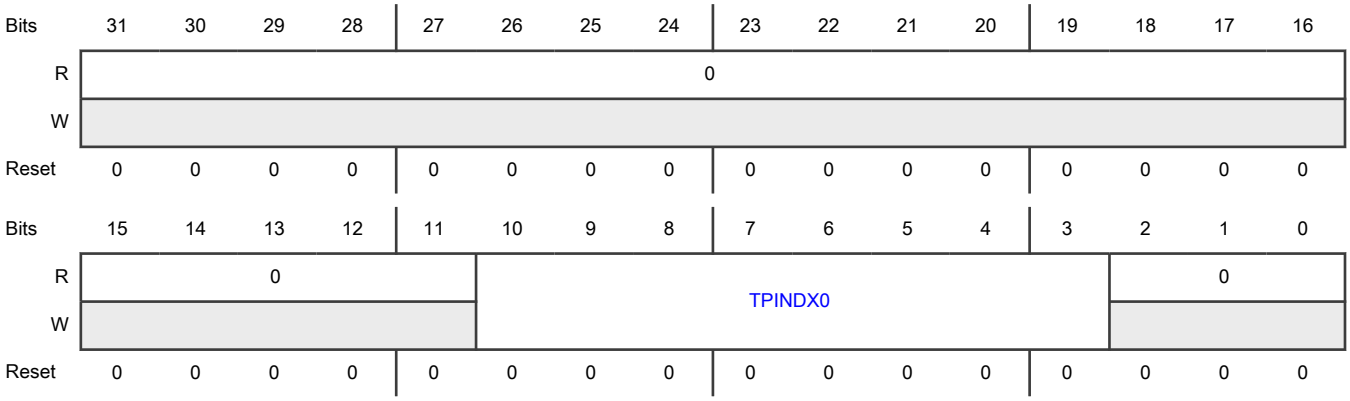
The size of buffer 0 is the difference between BUF0IND and 0.

The software must ensure that the value of TPINDX0 is not greater than the size of buffer 0.

Special write-access is permitted if:

- SR[AHB_ACC] = 0

Diagram



Fields

Field	Function
31-11 —	Reserved
10-3 TPINDX0	Top index of buffer 0
2-0 —	Reserved

80.13.2.12 Buffer 1 Top Index Register (BUF1IND)

Offset

Register	Offset
BUF1IND	34h

Function

This register specifies the top index of buffer 1, which defines its size. Note that the three LSBs of this register are set to 0. This ensures that the buffer is 64-bit aligned because each buffer entry is 64-bits long.

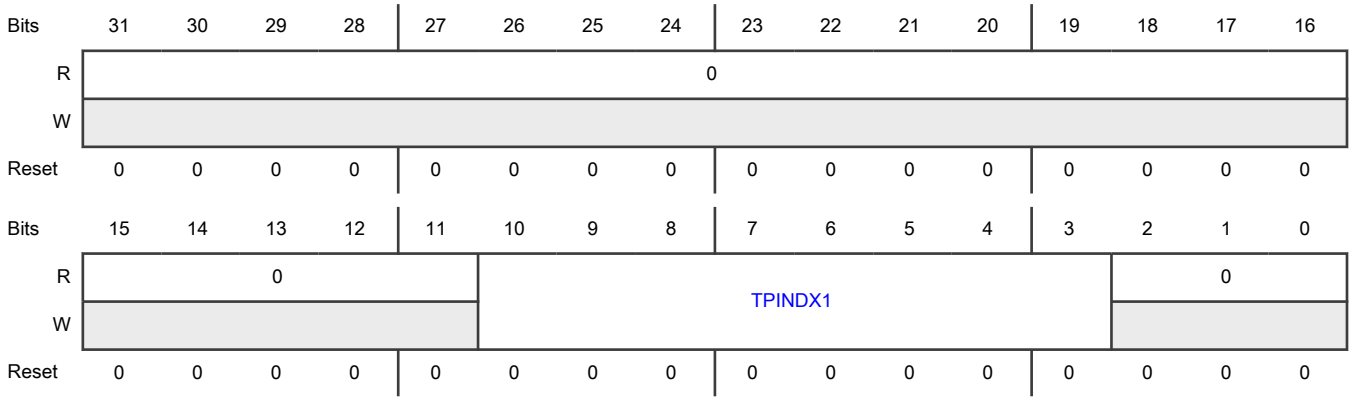
The size of buffer 1 is the difference between BUF1IND and BUF0IND. The register value should be entered in bytes. For example, if BUF0IND = 0x100, then setting BUF1IND = 0x130 sets the size of buffer 1 to 0x30 bytes.

The software must ensure that the value of TPINDX1 is not greater than the size of buffer 1.

Special write-access is permitted if:

- SR[AHB_ACC] = 0

Diagram



Fields

Field	Function
31-11 —	Reserved
10-3 TPINDX1	Top index of buffer 1
2-0 —	Reserved

80.13.2.13 Buffer 2 Top Index Register (BUF2IND)

Offset

Register	Offset
BUF2IND	38h

Function

This register specifies the top index of buffer 2, which defines its size. Note that the three LSBs of this register are set to 0. This ensures that the buffer is 64-bit aligned because each buffer entry is 64-bits long.

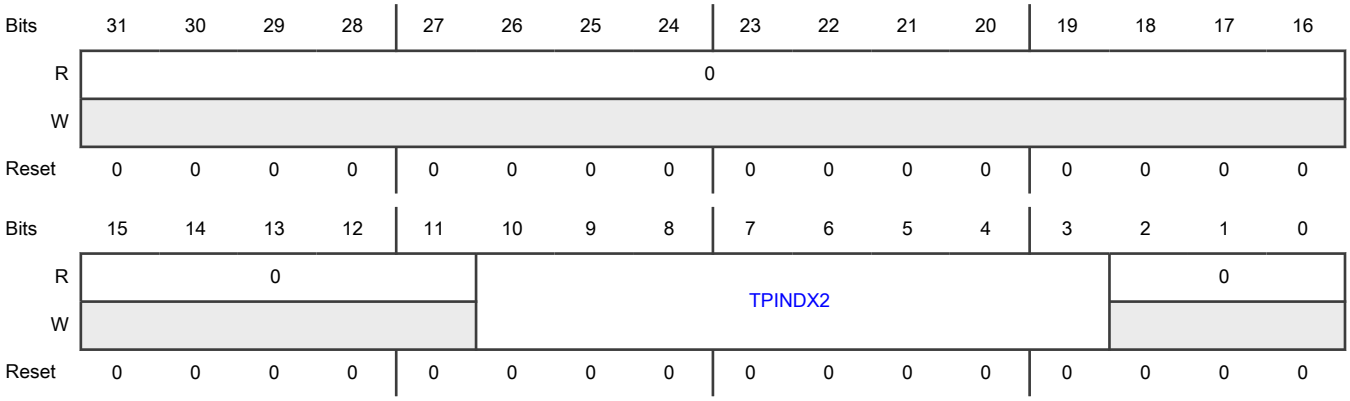
The size of buffer 2 is the difference between BUF2IND and BUF1IND. The register value should be entered in bytes. For example, if BUF1IND = 0x130 then setting BUF2IND = 0x180 sets the size of buffer 2 to 0x50 bytes.

The software must ensure that the value of TPINDX2 is not greater than the size of buffer 2.

Special write-access is permitted if:

- SR[AHB_ACC] = 0

Diagram



Fields

Field	Function
31-11 —	Reserved
10-3 TPINDX2	Top index of buffer 2
2-0 —	Reserved

80.13.2.14 DLL Flash Memory A Configuration Register (DLLCRA)

Offset

Register	Offset
DLLCRA	60h

Function

This register configures DLL and slave delay chain for flash memory A.

The value of the DLLEN field must be 1 after all reference (FREQEN, DLL_REFCNTR, DLLRES, SLAVE_AUTO_UPDT) delay chain configurations are programmed.

See DLL and delay chain usage for the programming sequence.

NOTE

See the chip data sheet for information on programming register fields.

NOTE

Please see the chip specific section of QuadSPI DLLCRA register for delay elements data

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DLLEN	FREQEN	0		DLL_REFCNTR				DLLRES				SLV_FINE_OFFSET			
W																
Reset	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	SLV_DLY_OFFSET			SLV_DLY_COARSE				Reserved			DLL_CDL8	SLAVE_A...	SLV_EN	SLV_DLL...	SLV_UPD
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 DLLEN	DLL enable 0b - DLL reference logic remains in reset and should be 0 for at least three flash memory clock cycles for reset. 1b - Enables DLL logic. Set it to 1 after all the configuration for DLLCR reference settings is complete.
30 FREQEN	Frequency enable 0b - Selects delay chain for low frequency of operation 1b - Selects delay chain for high frequency of operation
29-28 —	Reserved
27-24 DLL_REFCNTR	DLL reference counter Select the "n+1" interval of DLL phase detection and reference delay updating interval (minimum recommended value = 1).
23-20 DLLRES	DLL resolution Minimum resolution for DLL phase detector to remain locked/unlocked based on flash memory clock jitter. The minimum value is 2, and should be programmed to a more suitable value, such as 6.
19-16 SLV_FINE_OFFSET	Fine offset delay elements in incoming DQS This field sets the number of fine offset delay elements up to 16 in incoming DQS, and the default must be 1 element.
15 —	Reserved
14-12	T/16 offset delay elements in incoming DQS This field sets the number of T/16 offset delay elements in incoming DQS; default is 0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
SLV_DLY_OFFSET	
11-8 SLV_DLY_COARSE	Delay elements in each delay tap This field sets the number of delay elements in each delay tap. The field is used to overwrite DLL-generated delay values and works when the value of SLV_DLL_BYPASS is 1. Note : Please refer to the QuadSPI datasheet for more details.
7-5 —	Reserved
4 DLL_CDL8	DLL CDL8 Enable 0b - DLL is implemented to support within 2x variation 1b - DLL is implemented to support within 3x variation (BCS -> WCS)
3 SLAVE_AUTO_UPDT	Slave chain update This field automatically updates the slave chain as soon as DLL is locked. 0b - Auto-update feature is disabled. 1b - Auto-update feature is enabled.
2 SLV_EN	Slave enable 0b - DLL slave logic remains in reset, and its value should be 0 for at least three flash memory clock cycles for reset. 1b - Enables DQS slave delay chain, and should be 1 before any slave configuration settings take place.
1 SLV_DLL_BYPASS	Slave DLL bypass This field enables selection of the number of delays in each slave delay tap. 0b - Disables manual selection of coarse delays in the slave delay chain. 1b - Enables selection of number of delays in each slave delay tap, based on DLLCRA[SLV_DLY_COARSE].
0 SLV_UPD	Slave update You must program this field only after slave delay chain configuration takes place. 0b - Disables any further update on DQS slave delay chain. 1b - Updates the DQS slave delay chain with either ref-delay or bypass slave delay value, and should be set in the absence of the DQS clock.

80.13.2.15 Serial Flash Memory Address Register (SFAR)

Offset

Register	Offset
SFAR	100h

Function

The module automatically translates this address on the memory map to the address on the flash memory. When operating in a 24-bit mode, only bits 23-0 are sent to the flash memory. In the 32-bit mode, bits 27-0 are used with bits 31-28 driven to 0 when the value of SFACR[CAS] is 0. For example, if the value of SFACR[CAS] is 3, then bits 26-3 are sent to the flash memory as its page address in case flash memory is operating in a 24-bit mode. The total number of address bits requested by the flash memory, as its page and column address, must not be more than 32 bits. See Table 806 for the mapping between the access mode and the SFAR content and Normal mode for details on command triggering and command execution. The software must ensure that the serial flash memory address provided in the SFAR register lies in the valid flash memory address range, as defined in Table 806.

Special write-access is permitted if:

- SR[IP_ACC] = 0

NOTE

If MDAD and FRAD checks are enabled in MGC register but none of the MDAD and FRAD descriptors are valid, then any write on this register will generate a bus transfer error

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SFADR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SFADR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 SFADR	Serial flash memory address

80.13.2.16 Serial Flash Memory Address Configuration Register (SFACR)

Offset

Register	Offset
SFACR	104h

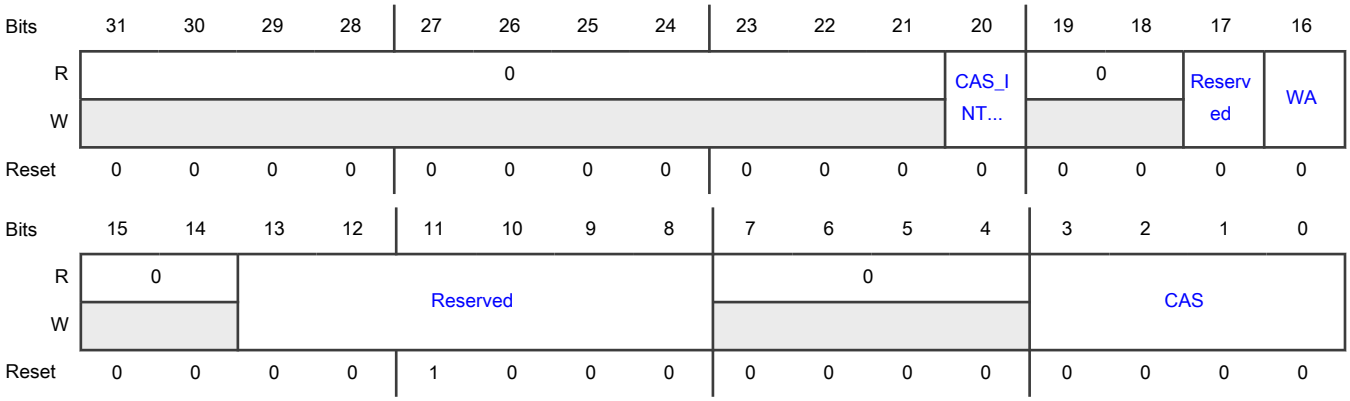
Function

This register contains the address requirements that are specific to serial flash memory. These requirements must be configured according to the connected flash memory, for the controller to function properly. The module automatically translates the address of SFAR on the memory map or the incoming address on the AHB bus to the column address on the flash memory. For example, if a flash memory needs 3 bits as its column address, then only the lower three bits of the SFAR/AHB address are sent to the flash memory as its column address. The software should ensure that the serial flash memory address provided in SFAR or the incoming AHB address lies in the valid flash memory address range.

Special write-access is permitted if:

- SR[IP_ACC] = 0
- SR[AHB_ACC] = 0

Diagram



Fields

Field	Function
31-21 —	Reserved
20 CAS_INTRLVD	CAS Interleaving 0b - CAS interleaving is disabled 1b - CAS interleaving is enabled
19-18 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
17 —	Reserved
16 WA	Word addressable Defines whether the serial flash memory is a byte addressable flash memory or a word addressable flash memory. According to the configuration of this field, the address is remapped to the flash memory interface. See Address scheme for details. 0b - Byte addressable serial flash memory mode 1b - Word (2-byte) addressable serial flash memory mode
15-14 —	Reserved
13-8 —	Reserved
7-4 —	Reserved
3-0 CAS	Column address space Defines the width of the column address. If the column address is, for example, [2:0] of SFAR/AHB address, then CAS must be 3. If there is no column address separation in any serial flash memory, the value of this field must be specified as 0.

80.13.2.17 Sampling Register (SMPR)

Offset

Register	Offset
SMPR	108h

Function

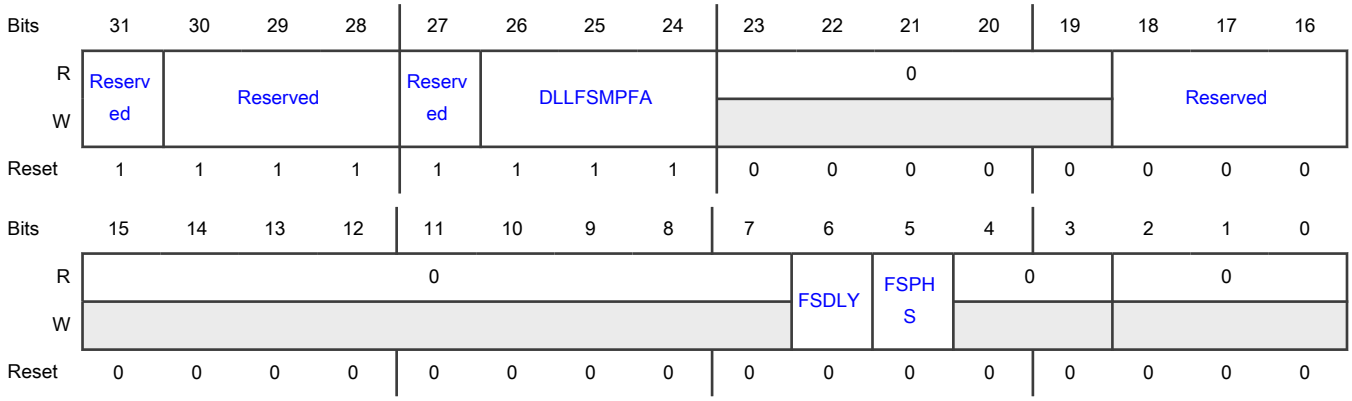
This register allows configuration of how the incoming data from the external serial flash memory devices is sampled in the QuadSPI module.

NOTE

See the chip data sheet for programming the register fields.

Special write-access is permitted in the disabled mode.

Diagram



Fields

Field	Function
31 —	Reserved
30-28 —	Reserved
27 —	Reserved
26-24 DLLFSMPFA	Selects the nth tap provided by slave delay chain for flash memory A The value of *n* can vary from 0 to 7, with each tap delay based on the DLLCRA register.
23-19 —	Reserved
18-16 —	Reserved
15-7 —	Reserved
6 FSDLY	Full-speed delay selection for internal/pad loop back DQS sampling This field selects the delay in accordance with the reference edge for the valid sample point. 0b - Same DQS 1b - Half-cycle early DQS
5 FSPHS	Full-speed phase selection for SDR instructions This field selects the edge of the sampling clock valid for full-speed commands. 0b - Select sampling at non-inverted clock

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Select sampling at inverted clock
4-3 —	Reserved
2-0 —	Reserved

80.13.2.18 RX Buffer Status Register (RBSR)

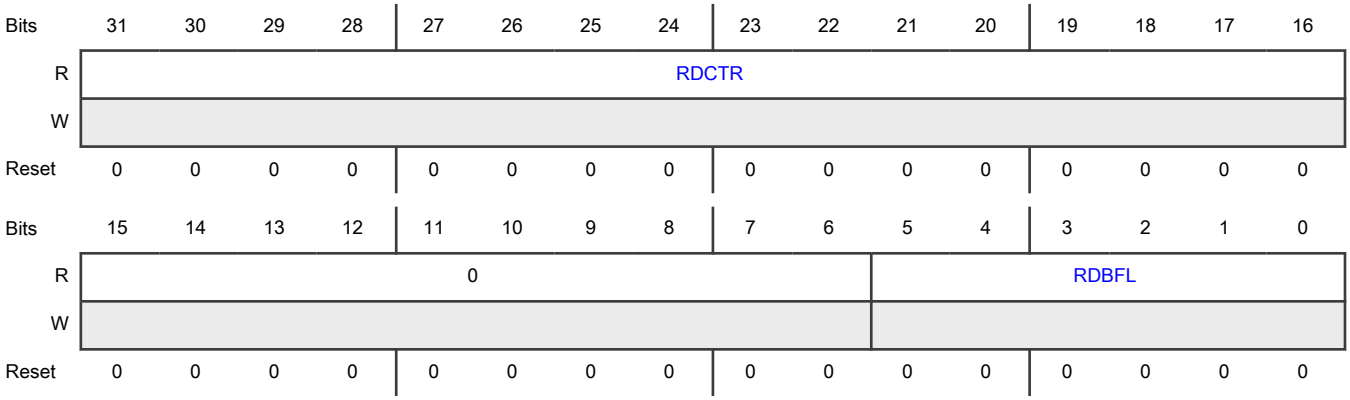
Offset

Register	Offset
RBSR	10Ch

Function

This register contains information related to the receive data buffer.

Diagram



Fields

Field	Function
31-16 RDCTR	Read counter Indicates the number of 4-byte entries removed from the RX buffer. For example, a value of 0x2 indicates that 8 bytes have been removed. It is incremented by the number (RBCT[WMRK] + 1) on RX buffer POP event. The RX buffer can be popped using DMA or FR[RBDF]. The RSER[RBDDE] defines which pop should be pursued. For details, see AHB RX Data Buffer Register (ARDB0 - ARDB31) and Data Transfer from the QuadSPI Module Internal Buffers.

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-6 —	Reserved
5-0 RDBFL	RX buffer fill level Indicates the number of 4-byte entries available in the RX buffer. For example, a value of 0x2 indicates 8 bytes are available.

80.13.2.19 RX Buffer Control Register (RBCT)

Offset

Register	Offset
RBCT	110h

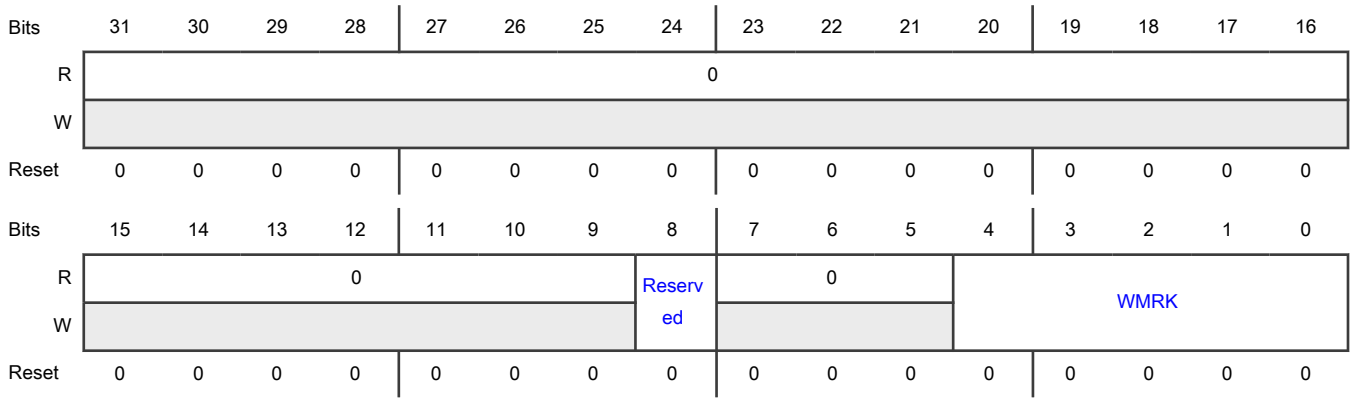
Function

This register contains control data related to the receive data buffer.

Special write-access is permitted if:

- SR[IP_ACC] = 0

Diagram



Fields

Field	Function
31-9 —	Reserved
8	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
7-5 —	Reserved
4-0 WMRK	RX buffer watermark This field determines when the readout action of the RX buffer is triggered. When the number of valid entries in the RX buffer is equal to or greater than the number provided by (WMRK+1), the SR[RXWE] flag is asserted. The value should be entered as the number of 4-byte entries minus 1. For example, a value of 0x0 sets the watermark to 4 bytes, 1 to 8bytes, 2 to 12 bytes, and so on. For details, see DMA usage. **NOTE** This field should never be programmed above 31 because there are only 32 memory mapped RBDR registers. If watermark is programmed above 31, data above 32 words will be lost.

80.13.2.20 DLL Status Register (DLLSR)

Offset

Register	Offset
DLLSR	12Ch

Function

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved				0				Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DLLA_LO...	SLVA_LO...	DLLA_RA...	DLLA_FI...	0				DLLA_SLV_FINE_VAL				DLLA_SLV_COARSE_VAL			
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved
27-24 —	Reserved
23-16 —	Reserved
15 DLLA_LOCK	DLL A lock status
14 SLVA_LOCK	Slave high lock status High-frequency slave delay chain locked. The field is configured when the slave decoder is updated with DLLCRA[SLV_UPD] and is reset after you configure SLV_UPD to 0.
13 DLLA_RANGE_ERR	DLL master delay chain The value 1 indicates that DLL master delay chain is working out of delay range because of incorrect DLL configuration.
12 DLLA_FINE_UNDERFLOW	Fine delay chain underflow The value 1 indicates that fine delay chain underflow has occurred.
11-8 —	Reserved
7-4 DLLA_SLV_FINE_VAL	Fine delay cells in slave delay chain This value indicates the total number of fine delay cells (1 delay unit) selected in the slave delay chain.
3-0 DLLA_SLV_COARSE_VAL	Coarse delay cells in slave delay chain This value indicates the total number of coarse delay cells (16 delay units) selected in the slave delay chain.

80.13.2.21 Data Learning Status Flash Memory A Register (DLSR_FA)

Offset

Register	Offset
DLSR_FA	134h

Function

This register shows sampling point selected by data learning algorithm when the value of DLSR_FA[DLPFFA] is 0. Otherwise, it shows the pattern matching outline.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DLPFFA	0														
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	POS_EDGE								NEG_EDGE							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 DLPFFA	Data learning pattern fail This field asserts when data learning fails at flash memory A.
30-16 —	Reserved
15-8 POS_EDGE	DLP positive edge match signature for flash memory A
7-0 NEG_EDGE	DLP negative edge match signature for flash memory A

80.13.2.22 TX Buffer Status Register (TBSR)

Offset

Register	Offset
TBSR	150h

Function

This register contains information related to the transmit data buffer.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TRCTR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								TRBFL							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 TRCTR	Transmit counter This field indicates how many entries of 4 bytes have been written into the TX buffer by host accesses. It is reset to 0 when a 1 is written to MCR[CLR_TXF]. It is incremented on each write access to the TBDR register when another word has been pushed onto the TX buffer. When it is not cleared, the TRCTR field wraps around to 0. See TX Buffer Data Register (TBDR) for details.
15-9 —	Reserved
8-0 TRBFL	TX buffer fill level This field contains the number of entries of 4 bytes each available in the TX buffer for the QuadSPI module to transmit to the serial flash memory device. The value of this field can reach maximum up to the total TX buffer size.

80.13.2.23 TX Buffer Data Register (TBDR)

Offset

Register	Offset
TBDR	154h

Function

This register provides access to the circular TX buffer of depth 256, so the total size is 256 * 4 bytes. This buffer provides the data written into it as write data for the page programming commands to the serial flash memory device. See Table 779 for the byte ordering scheme. A write transaction on the flash memory with data size of less than 32 bits leads to the removal of one data entry from the TX buffer. The valid bits are used and the rest of the bits are discarded.

Special write-access is permitted if:

- SR[TXFULL] = 0

NOTE

This register can only be written when write access is granted by SFP block else bus transfer error is generated on write. Please refer to section TBDR register write lock

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	TXDATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	TXDATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TXDATA	TX data On write access, the data is written to the next available entry of the TX buffer and TBSR[TRBFL] is updated accordingly. On a read access, the last data written to the register is returned.

80.13.2.24 TX Buffer Control Register (TBCT)

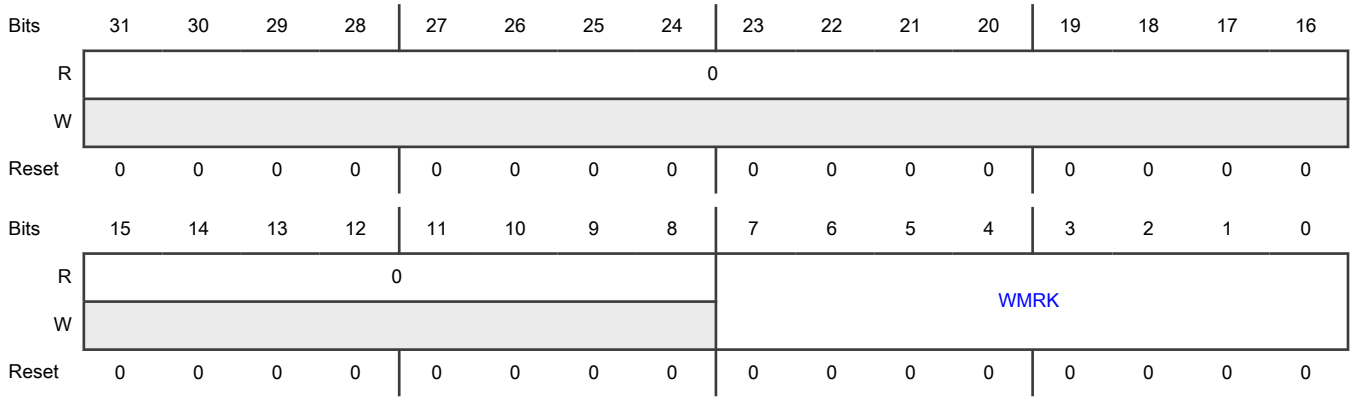
Offset

Register	Offset
TBCT	158h

Function

This register contains control information for transmit data buffer.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 WMRK	Watermark for TX buffer Determines the watermark for the TX buffer When the number of available space in the TX buffer is greater than or equal to the number provided by WMRK (number of 4-byte entries), SR[TXWA] is asserted. For example, a value of 0x1 sets the watermark to 4 bytes, 0x2 sets it to 8 bytes, 0x3 sets it to 12 bytes, and so on. For details, see DMA usage. WMRK = 0 is invalid. **NOTE** For IPS write with SFP enabled, this field must be programmed as per formula to prevent TX underflow: If data size in words >1, watermark=(TX FIFO size in words - data size in words) +1, If data size in words =1, watermark=(TX FIFO size in words - data size in words)

80.13.2.25 Status Register (SR)

Offset

Register	Offset
SR	15Ch

Function

This register provides all the available status information about SFM command execution and arbitration, the RX buffer, TX buffer, and the AHB buffer.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0			0	TXFULL	TXDMA	TXWA	TXNE	RXDMA	0			RXFULL	0		RXWE
W																
Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	AHB3FUL	AHB2FUL	AHB1FUL	AHB0FUL	AHB3NE	AHB2NE	AHB1NE	AHB0NE	AHBTRN	Reserved	0	Reserved	AHB_ACC	IP_ACC	BUSY
W																
Reset	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	Reserved
28 —	Reserved
27 TXFULL	TX buffer full Asserted when the FIFO level reaches TX buffer size
26 TXDMA	TX DMA Asserted when the TXFIFO fill via DMA is active and DMA is requested or running
25 TXWA	TX buffer watermark available Asserted when the number of available spaces in the TX buffer is greater than or equal to the value provided by TBCT[WMRK] Example: When TBCT[WMRK]=1, SR[TXWA] is de-asserted when TX FIFO has 256 entries.
24 TXNE	TX buffer not empty Asserted when TX buffer contains data
23 RXDMA	RX buffer DMA Asserted when RX buffer read out via DMA is active; that is, when DMA is requested or running
22-20 —	Reserved
19 RXFULL	RX buffer full Asserted when the RX buffer is full; that is, when RBSR[RDBFL] is equal to 32

Table continues on the next page...

Table continued from the previous page...

Field	Function
18-17 —	Reserved
16 RXWE	RX buffer watermark exceeded Asserted when the number of valid entries in the RX buffer exceeds the number provided in RBCT[WMRK]
15 —	Reserved
14 AHB3FUL	AHB 3 buffer full Asserted when AHB 3 buffer is full
13 AHB2FUL	AHB 2 buffer full Asserted when AHB 2 buffer is full
12 AHB1FUL	AHB 1 buffer full Asserted when the AHB 1 buffer is full
11 AHB0FUL	AHB 0 buffer full Asserted when the AHB 0 buffer is full
10 AHB3NE	AHB 3 buffer not empty Asserted when the AHB 3 buffer contains data
9 AHB2NE	AHB 2 buffer not empty Asserted when the AHB 2 buffer contains data
8 AHB1NE	AHB 1 buffer not empty Asserted when the AHB 1 buffer contains data
7 AHB0NE	AHB 0 buffer not empty Asserted when the AHB 0 buffer contains data
6 AHBTRN	AHB access transaction pending Asserted when there is a pending request on the AHB interface. See Flash memory mapped AMBA bus.
5 —	Reserved
4 —	Reserved
3	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
2 AHB_ACC	AHB read access Asserted when the currently executed transaction is initiated by the AHB bus
1 IP_ACC	IP access Asserted when transaction currently executed is initiated by the IP bus
0 BUSY	Module busy Asserted when module is currently busy handling a transaction to an external flash memory device

80.13.2.26 Flag Register (FR)

Offset

Register	Offset
FR	160h

Function

This register provides all available flags about SFM command execution and arbitration, which may serve as the source for the generation of interrupt service requests. Note that the error flags in this register do not relate directly to the execution of the transaction in the serial flash memory device itself but only to the behavior and conditions visible in the QuadSPI module.

Special write-access is permitted in the enabled mode.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved	0	Reserved	DLLABRT	TBFF	TBUF	0	DLLUNLCK	ILLINE	0		0	0		RBOF	RBDF
W				W1C	W1C	W1C		W1C	W1C						W1C	W1C
Reset	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	AITEF	AIBSEF	ABOF	Reserved	CRCAEF	0	0	0	IPIEF	0	0	0			TFF
W		W1C	W1C	W1C	W1C	W1C				W1C						W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30 —	Reserved
29 —	Reserved
28 DLLABRT	DLL abort 1b - This field is set whenever DLL is unlocked while reading data from the flash memory.
27 TBFF	TX buffer fill flag Before writing to the TX buffer, this field should be cleared. Then, it should be read back. If it is set, the TX buffer can include more data. If the field remains cleared, the TX buffer can be considered as full. See TX buffer operation for details.
26 TBUF	TX buffer underrun flag This field is set if the module tries to pull data when the TX buffer is empty.. The IP command leading to the TX buffer underrun is continued (data sent to the serial flash memory device is undefined). Here, a valid underrun means that it should have occurred during the transaction so that few bytes (that is, less than 4 bytes) are left in FIFO and the remaining are filled with "FFFFh". This field does not set if transfer is less than 128 bits. The application must clear the TX buffer in response to this event by writing a 1 to MCR[CLR_TXF]. The application must clear the TX buffer in response to this event by writing a 1 to MCR[CLR_TXF].
25 —	Reserved
24 DLLUNLCK	DLL unlock 1b - This field is set whenever DLL unlock event occurs, irrespective of flash memory access.
23 ILLINE	Illegal instruction error flag This field is set when an illegal instruction is encountered by the controller in any of the sequences. As soon as the field is set, you must assert MCR[SWRSTSD] and MCR[SWRSTHD]. That is, reset the flash memory and AHB domain after reconfiguring the correct sequence instruction. See Table 777 for a list of legal instructions.
22-21 —	Reserved
20 —	Reserved
19-18	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
17 RBOF	RX buffer overflow flag This field is set when no more data can be pushed into the RX buffer from the serial flash memory device. The IP command leading to this condition is continued until the number of bytes in IPCR[IDATSZ] are read from the serial flash memory device. The content of the RX buffer remains unchanged.
16 RBDF	RX buffer drain flag This field is set if SR[RXWE] is asserted. Writing 1 to this field triggers one of the following actions: • If the RX buffer has up to RBCT[WMRK] valid entries, then the flag is cleared. • If the RX buffer has more than RBCT[WMRK] valid entries and the RSER[RBDDE] field is not set (flag driven mode), an RX buffer POP event is triggered. The flag remains set if the RX buffer contains more than RBCT[WMRK] valid entries after the RX buffer POP event is complete. The flag is cleared if the RX buffer contains less than or equal to RBCT[WMRK] valid entries after the RX buffer POP event is complete. See the "Receive Buffer Drain Interrupt or DMA Request" section in Normal mode interrupt and DMA requests for details.
15 —	Reserved
14 AITEF	AHB illegal transaction error flag This is set whenever there is no response generated from QuadSPI to AHB bus in case of an illegal transaction and the watchdog timer expires. The timer value is considered as a parameter.
13 AIBSEF	AHB illegal burst size error flag This is set whenever the total burst size (size x beat) of an AHB transaction is greater than the prefetch data size, which is defined by BUFxCR[ADATSZ] or data size mentioned in the sequence pointed to by the SEQID field in case ADATSZ = 0. See HBURST support with AHB read details on HBURST feature.
12 ABOF	AHB buffer overflow flag This is set when the size of the AHB access exceeds the size of the AHB buffer. This condition can occur only if BUFxCR[ADATSZ] is programmed incorrectly. The AHB command leading to this condition is continued until the number of entries according to BUFxCR[ADATSZ] have been read from the serial flash memory device. The content of the AHB buffer is not changed.
11 —	Reserved
10	Sets when there is CRC or ECC error for flash memory A

Table continues on the next page...

Table continued from the previous page...

Field	Function
CRCAEF	0b - CRCEF interrupt is not generated. 1b - CRCEF interrupt is generated.
9 —	Reserved
8 —	Reserved
7 —	Reserved
6 IPIEF	IP command trigger could not be executed error flag This is set when the SR[IP_ACC] and SR[AWRACC] fields are set (that is, an IP triggered command is currently executing) and any of the following conditions occurs: • Write access to the RBCT
5 —	Reserved
4 —	Reserved
3-1 —	Reserved
0 TFF	IP command transaction finished flag This field is set after the QuadSPI module completes a running IP command. If an error occurs, and the related error flags are valid in the same clock cycle, the TFF flag is asserted.

80.13.2.27 Interrupt and DMA Request Select and Enable Register (RSER)

Offset

Register	Offset
RSER	164h

Function

This register provides enables and selectors for the interrupts in the QuadSPI module.

NOTE

Each field of the FR enabled as source for an interrupt prevents the QuadSPI module from entering the Stop mode or Module Disable mode when this flag is set.

Special write-access is permitted in the "Anytime" mode.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved	Reserved		0	TBFIE	TBUIE	TBFDE	DLLULIE	ILLINIE	Reserved	RBDDE	Reserved	0		RBOIE	RBDIE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	AITIE	AIBSIE	ABOIE	Reserved	CRCAIE	Reserved	Reserved	Reserved	IPIEIE	0	Reserved	0			TFIE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30-29 —	Reserved
28 —	Reserved
27 TBFIE	TX buffer fill interrupt enable flag This field indicates the TX buffer fill interrupt enable flag. This interrupt should not be enabled when using SFP functionality because it causes QSPI BUSY SFM to go high and keeps SFP to generate any transaction over to QSPI. 0b - No TBFF interrupt is generated. 1b - TBFF interrupt is generated.
26 TBUIE	TX buffer underrun interrupt enable flag This field indicates the TX buffer underrun interrupt enable flag. 0b - No TBUF interrupt is generated 1b - TBUF interrupt is generated
25 TBFDE	TX buffer fill DMA enable Enables generation of DMA requests for TX buffer fill. When the value of this field is 1, DMA requests are generated as long as number of available spaces in the TX buffer is greater than or equal to the value provided by TBCT[WMRK].

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No DMA request is generated 1b - DMA request is generated
24 DLLULIE	DLL unlock interrupt enable 1b - Write 1 to this to enable generation of interrupt on DLL unlock event.
23 ILLINIE	Illegal instruction error interrupt enable Triggered by the ILLINE flag in FR 0b - No ILLINE interrupt is generated. 1b - ILLINE interrupt is generated.
22 —	Reserved
21 RBDDE	RX buffer drain DMA enable This field enables generation of DMA requests for RX buffer drain. When the value of this field is 1, the DMA requests are generated as long as SR[RXWE] is set. 0b - No DMA request is generated. 1b - DMA request is generated.
20 —	Reserved
19-18 —	Reserved
17 RBOIE	RX buffer overflow interrupt enable This field indicates the RX buffer overflow interrupt enable flag. 0b - No RBOF interrupt is generated. 1b - RBOF interrupt is generated.
16 RBDIE	RX buffer drain interrupt enable This field enables generation of IRQ requests for RX buffer drain. When the value of this field is 1, the interrupt is asserted as long as SR[RBDF] is set. 0b - No RBDF interrupt is generated. 1b - RBDF Interrupt is generated.
15 —	Reserved
14	AHB illegal transaction interrupt enable flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
AITIE	This field indicates the AHB illegal transaction interrupt enable flag. 0b - No AITEF interrupt is generated. 1b - AITEF interrupt is generated.
13 AIBSIE	AHB illegal burst size interrupt enable flag This field indicates the AHB illegal burst size interrupt enable flag. 0b - No AIBSEF interrupt is generated. 1b - AIBSEF interrupt is generated.
12 ABOIE	AHB buffer overflow interrupt enable flag This field indicates the AHB buffer overflow interrupt enable flag. 0b - No ABOF interrupt is generated. 1b - ABOF interrupt is generated.
11 —	Reserved
10 CRCAIE	CRC and ECC interrupt enable for flash memory A 0b - CRCAEF interrupt is not generated. 1b - CRCAEF interrupt is generated.
9 —	Reserved
8 —	Reserved
7 —	Reserved
6 IPIEIE	IP command trigger during IP access error interrupt enable flag This field indicates IP command trigger during IP access error interrupt enable flag. 0b - No IPIEF interrupt is generated 1b - IPIEF interrupt is generated
5 —	Reserved
4 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
3-1 —	Reserved
0 TFIE	Transaction finished interrupt enable flag This field indicates the transaction finished interrupt enable flag. 0b - No TFF interrupt is generated. 1b - TFF interrupt is generated.

80.13.2.28 Sequence Pointer Clear Register (SPTRCLR)

Offset

Register	Offset
SPTRCLR	16Ch

Function

This register provides fields to reset the IP and buffer sequence pointers. The sequence pointer contains the index of instructions within the LUT entry that is to be executed next. For example, if the LUT entry ends on a JMP_ON_CS value of 2, the index is stored as 2.

The software should reset the sequence pointers defined by JMP_ON_CS operand whenever the sequence ID is required to be changed by updating the SEQID field in the IPCR or BFGENCR.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0						Reserved		0				Reserved	Reserved	Reserved	
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0							0	0							0
W								IPPTRC								BFPTRC
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-26	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
25-24 —	Reserved
23-20 —	Reserved
19 —	Reserved
18 —	Reserved
17-16 —	Reserved
15-9 —	Reserved
8 IPPTRC	IP pointer clear This is a self-clearing field. 1b - Clears the sequence pointer for IP accesses as defined in IPCR.
7-1 —	Reserved
0 BFPTRC	Buffer pointer clear This is a self-clearing field. 1b - Clears the sequence pointer for AHB read accesses as defined in BFGENCR.

80.13.2.29 Serial Flash Memory A1 Top Address Register (SFA1AD)

Offset

Register	Offset
SFA1AD	180h

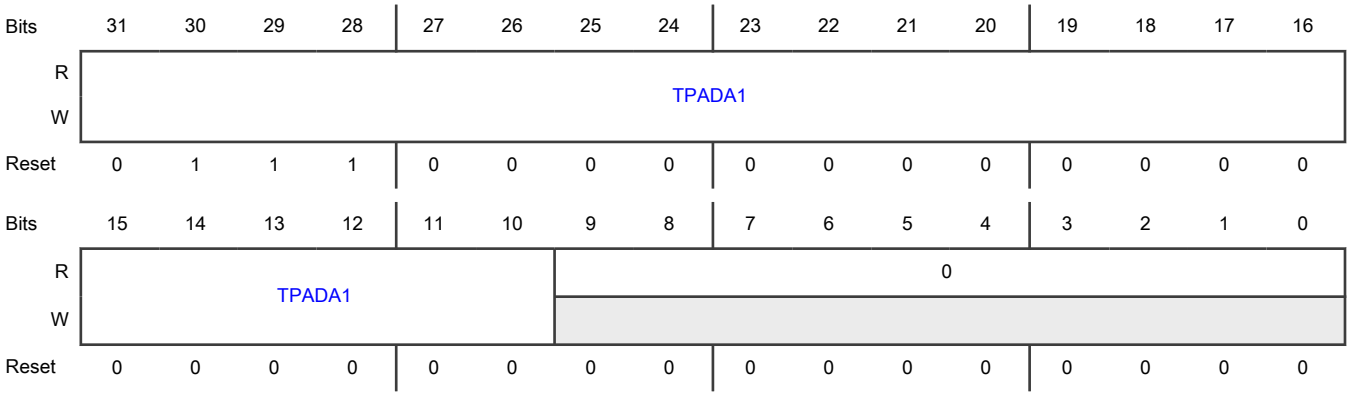
Function

This register provides the address mapping for serial flash memory A1. The difference between SFA1AD[TPADA1] and AMBA_BASE defines the size of the memory map for serial flash memory A1.

Special write-access is permitted if:

- SR[IP_ACC] = 0
- SR[AHB_ACC] = 0

Diagram



Fields

Field	Function
31-10 TPADA1	Top address for serial flash memory A1 In effect, TPADxx is the first location of the next memory.
9-0 —	Reserved

80.13.2.30 Serial Flash Memory A2 Top Address Register (SFA2AD)

Offset

Register	Offset
SFA2AD	184h

Function

This register provides the address mapping for serial flash memory A2. The difference between SFA2AD[TPADA2] and SFA1AD[TPADA1] defines the size of the memory map for serial flash memory A2.

Special write-access is permitted if:

- SR[IP_ACC] = 0
- SR[AHB_ACC] = 0

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TPADA2															
W																
Reset	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TPADA2						0									
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-10 TPADA2	Top address for serial flash memory A2 In effect, TPxxAD is the first location of the next memory.
9-0 —	Reserved

80.13.2.31 RX Buffer Data Register (RBDR0 - RBDR31)

Offset

For a = 0 to 31:

Register	Offset
RBDRa	200h + (a × 4h)

Function

These registers provide access to individual entries in the RX buffer. See Table 779 for the byte ordering scheme.

RBDR0 corresponds to the actual position of the read pointer within the RX buffer. The number of valid entries available depends on the number of RX buffer entries implemented and on the number of valid buffer entries available in the RX buffer.

Example 1 - RX buffer filled completely with 32 words: In this case, the address range for valid read access extends from RBDR0 to RBDR31 RBDR63.

Example 2 - RX buffer filled with five valid words: RX buffer fill level of RBSR[RDBFL] is 5. In this case, access to RBDR4 provides the last valid entry.

Any access beyond the range of valid RX buffer entries provides undefined results.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RXDATA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXDATA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 RXDATA	RX data This field contains the data associated with the related RX buffer entry. For data format and byte ordering, see Byte ordering of serial flash memory read data.

80.13.2.32 LUT Key Register (LUTKEY)

Offset

Register	Offset
LUTKEY	300h

Function

This register contains the key to lock and unlock the LUT. See LUT for details.

Special write-access is permitted in the "Anytime" mode. Write access is not permitted if MGC[11]=1b.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	KEY															
W																
Reset	0	1	0	1	1	0	1	0	1	1	1	1	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	KEY															
W																
Reset	0	1	0	1	1	0	1	0	1	1	1	1	0	0	0	0

Fields

Field	Function
31-0 KEY	Key to lock or unlock the LUT The key is 0x5AF05AF0 and the read value is always 0x5AF05AF0. Write access is not permitted if MGC[11]=1b.

80.13.2.33 LUT Lock Configuration Register (LCKCR)

Offset

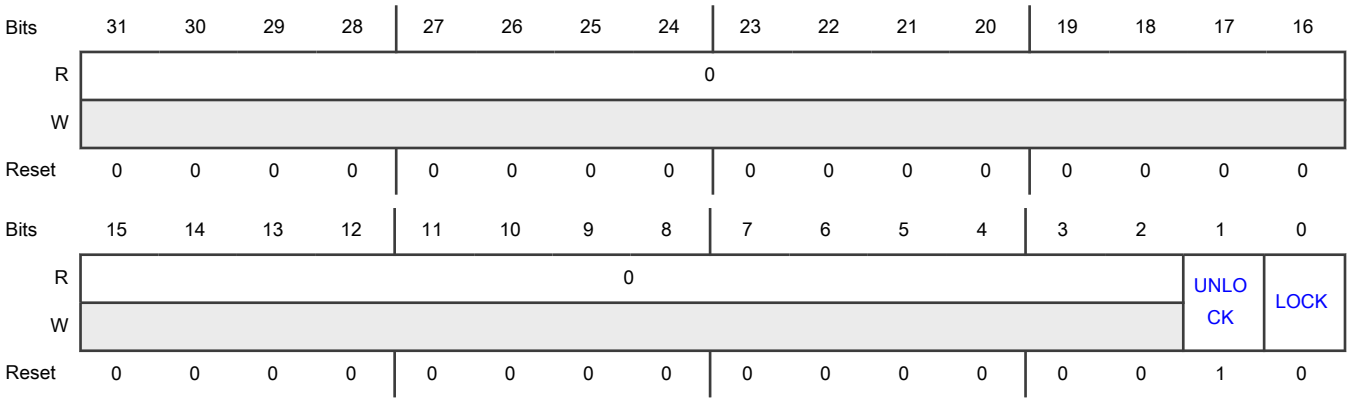
Register	Offset
LCKCR	304h

Function

This register is used along with the LUTKEY register to lock or unlock the LUT. This register should be written immediately after the LUTKEY register for the lock or unlock operation to be successful. See LUT for details. Setting both the LOCK and UNLOCK bits as "00" or "11" is not allowed.

Special write access is permitted after writing the LUT key register. Write access is not permitted if MGC[11]=1b.

Diagram



Fields

Field	Function
31-2 —	Reserved
1 UNLOCK	Unlock LUT Unlocks the LUT when the following two conditions are met:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• This register is written just after the LUT Key Register (LUTKEY). • The LUT key register was written with the 0x5AF05AF0 key.
0 LOCK	Lock LUT Locks the LUT when the following conditions are met: • This register is written just after the LUT Key Register (LUTKEY). • The LUT key register is written with the 0x5AF05AF0 key.

80.13.2.34 LUT Register (LUT0)

Offset

Register	Offset
LUT0	310h

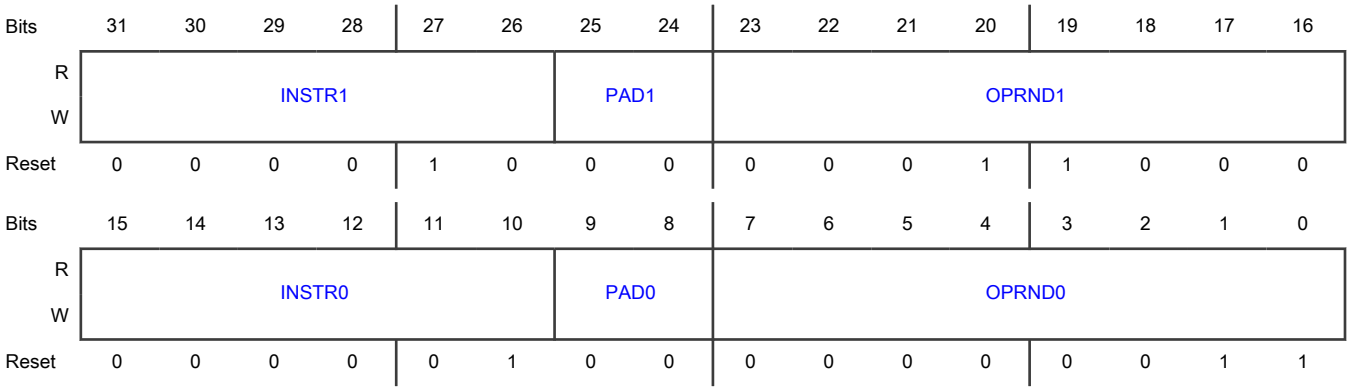
Function

A sequence of instruction-operand pairs may be pre-populated in the LUT according to the device connected on board. Each instruction-operand pair is of 16 bits (2 bytes) each. Every sequence preprogrammed by Program Sequence Engine in the LUT is referred to by its index. The LUT registers act as lookup tables for sequences of instructions. The programmable sequence engine executes the instructions in these sequences to generate a valid serial flash memory transaction. There are a total of 80 LUT registers. These 80 registers are divided into groups of 5 registers that make a valid sequence. Therefore, LUT[0], LUT[5], LUT[10] ..... LUT[75] are the starting registers of a valid sequence. Each of these sets of 5 registers can have a maximum of 10 instructions. Reset value of the register shown below is only applicable to LUT2 to LUT79. A maximum of 16 sequences can be defined at one time. See LUT that describes the LUT registers in detail.

Special write-access is permitted if the LUT is unlocked.

This register is access controlled and can only be programmed by privilege masters. Last six LUT SEQID locations should be used for programming SEQID for atomic commands (where flash's internal config region is accessed) like flash erase etc. where data transfer is not required. These 6 SEQID can be send to QSPI using only FRAD0 and can't be qualified using any other FRAD. For details, see Atomic commands considerations. Write access is not permitted if MGC[11]=1b.

Diagram



Fields

Field	Function
31-26 INSTR1	Instruction 1
25-24 PAD1	Pad information for INSTR1 00b - 1 Pad 01b - 2 Pads 10b - 4 Pads 11b - NA
23-16 OPRND1	Operand for INSTR1
15-10 INSTR0	Instruction 0
9-8 PAD0	Pad information for INSTR0 00b - 1 Pad 01b - 2 Pads 10b - 4 Pads 11b - NA
7-0 OPRND0	Operand for INSTR0

80.13.2.35 LUT Register (LUT1)

Offset

Register	Offset
LUT1	314h

Function

A sequence of instruction-operand pairs may be pre-populated in the LUT according to the device connected on board. Each instruction-operand pair is of 16 bits (2 bytes) each. Every sequence preprogrammed by Program Sequence Engine in the LUT is referred to by its index. The LUT registers act as lookup tables for sequences of instructions. The programmable sequence engine executes the instructions in these sequences to generate a valid serial flash memory transaction. There are a total of 80 LUT registers. These 80 registers are divided into groups of 5 registers that make a valid sequence. Therefore, LUT[0], LUT[5], LUT[10] ..... LUT[75] are the starting registers of a valid sequence. Each of these sets of 5 registers can have a maximum of 10 instructions. Reset value of the register shown below is only applicable to LUT2 to LUT79. A maximum of 16 sequences can be defined at one time. See LUT that describes the LUT registers in detail.

Special write-access is permitted if the LUT is unlocked.

This register is access controlled and can only be programmed by privilege masters. Last six LUT SEQID locations should be used for programming SEQID for atomic commands (where flash's internal config region is accessed) like flash erase etc. where data transfer is not required. These 6 SEQID can be send to QSPI using only FRAD0 and can't be qualified using any other FRAD. For details, see Atomic commands considerations. Write access is not permitted if MGC[11]=1b.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	INSTR1						PAD1		OPRND1							
Reset	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	INSTR0						PAD0		OPRND0							
Reset	0	0	0	1	1	1	0	0	0	0	0	0	1	0	0	0

Fields

Field	Function
31-26 INSTR1	Instruction 1
25-24 PAD1	Pad information for INSTR1 00b - 1 Pad 01b - 2 Pads 10b - 4 Pads 11b - NA
23-16 OPRND1	Operand for INSTR1
15-10 INSTR0	Instruction 0
9-8 PAD0	Pad information for INSTR0 00b - 1 Pad 01b - 2 Pads 10b - 4 Pads 11b - NA
7-0 OPRND0	Operand for INSTR0

80.13.2.36 LUT Register (LUT2 - LUT79)

Offset

For a = 2 to 79:

Register	Offset
LUTa	310h + (a × 4h)

Function

A sequence of instruction-operand pairs may be pre-populated in the LUT according to the device connected on board. Each instruction-operand pair is of 16 bits (2 bytes) each. Every sequence preprogrammed by Program Sequence Engine in the LUT is referred to by its index. The LUT registers act as lookup tables for sequences of instructions. The programmable sequence engine executes the instructions in these sequences to generate a valid serial flash memory transaction. There are a total of 80 LUT registers. These 80 registers are divided into groups of 5 registers that make a valid sequence. Therefore, LUT[0], LUT[5], LUT[10] ..... LUT[75] are the starting registers of a valid sequence. Each of these sets of 5 registers can have a maximum of 10 instructions. Reset value of the register shown below is only applicable to LUT2 to LUT79. A maximum of 16 sequences can be defined at one time. See LUT that describes the LUT registers in detail.

Special write-access is permitted if the LUT is unlocked.

This register is access controlled and can only be programmed by privilege masters. Last six LUT SEQID locations should be used for programming SEQID for atomic commands (where flash's internal config region is accessed) like flash erase etc. where data transfer is not required. These 6 SEQID can be send to QSPI using only FRAD0 and can't be qualified using any other FRAD. For details, see Atomic commands considerations. Write access is not permitted if MGC[11]=1b.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	INSTR1						PAD1		OPRND1							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	INSTR0						PAD0		OPRND0							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-26 INSTR1	Instruction 1
25-24 PAD1	Pad information for INSTR1 00b - 1 Pad

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01b - 2 Pads 10b - 4 Pads 11b - NA
23-16 OPRND1	Operand for INSTR1
15-10 INSTR0	Instruction 0
9-8 PAD0	Pad information for INSTR0 00b - 1 Pad 01b - 2 Pads 10b - 4 Pads 11b - NA
7-0 OPRND0	Operand for INSTR0

80.13.2.37 Flash Region Start Address (FRAD0_WORD0 - FRAD7_WORD0)

Offset

For n = 0 to 7:

Register	Offset
FRADn_WORD0	800h + (n × 20h)

Function

This register is access controlled and can only be programmed by privilege masters.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	STARTADR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 STARTADR	Start Address Specifies the specific flash memory region starting address (around 64 KB boundary)
15-0 —	Reserved

80.13.2.38 Flash Region End Address (FRAD0_WORD1 - FRAD7_WORD1)

Offset

For n = 0 to 7:

Register	Offset
FRADn_WORD1	804h + (n × 20h)

Function

This register is access controlled and can only be programmed by privilege masters.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ENDADR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-16 ENDADR	End Address Specifies the specific flash memory region end address (around 64 KB boundary)
15-0 —	Reserved

80.13.2.39 Flash Region Privileges (FRAD0_WORD2 - FRAD7_WORD2)

Offset

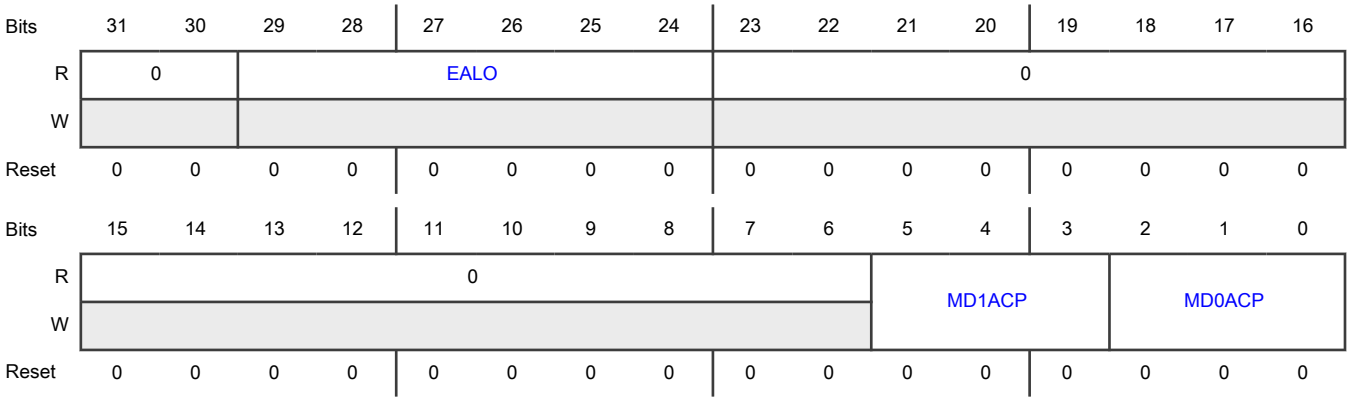
For n = 0 to 7:

Register	Offset
FRADn_WORD2	808h + (n × 20h)

Function

This register is access controlled and can only be programmed by privilege masters.

Diagram



Fields

Field	Function
31-30 —	Reserved
29-24 EALO	Exclusive Access Lock Owner When FRAD*n*_WORD3[EAL] = 11, this field indicates the domain/master ID that owns the exclusive access lock.
23-6	Reserved

Table continues on the next page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>—</td><td></td></tr>
<tr><td>5-3: MD1ACP
2-0: MD0ACP</td><td>Master Domain Access Control Policy
This field define the access restrictions for respective Master Domain 0 and 1 corresponding to this FRAD region. Write permissions are decided based on secure and privilege attributes of current transaction. Read access is not restricted. This field can not be programmed if MDAD registers are not valid. If EAL is being written in same cycle (10 or 11), or if FRAD_WORD3 [LOCK] bits are written as 11. If LOCK bits are written as 10 then this field can be programmed only if Master ID matches the ID check of respective MDAD queue. If MDAD descriptor is not valid then respective MDxACP also becomes read-only.
Table 805. Field value mapping
<table>
<tr><th>Policy</th><th>Secure privilege</th><th>Secure user</th><th>Non secure privilege</th><th>Non secure user</th></tr>
<tr><td>111</td><td>Read/Write</td><td>Read</td><td>Read/Write</td><td>Read</td></tr>
<tr><td>110</td><td>Read/Write</td><td>Read/Write</td><td>Read/Write</td><td>Read/Write</td></tr>
<tr><td>101</td><td>Read/Write</td><td>Read</td><td>Read</td><td>Read</td></tr>
<tr><td>100</td><td>Read/Write</td><td>Read/Write</td><td>Read</td><td>Read</td></tr>
<tr><td>0xx</td><td>Read</td><td>Read</td><td>Read</td><td>Read</td></tr>
</table>
NOTE
The software must program the value 101 for MDACP fields for FRAD0 as this FRAD has access to all the atomic instructions. For details, see Atomic commands considerations.</td></tr>
</table>

80.13.2.40 Flash Region Lock Control (FRAD0_WORD3 - FRAD7_WORD3)

Offset

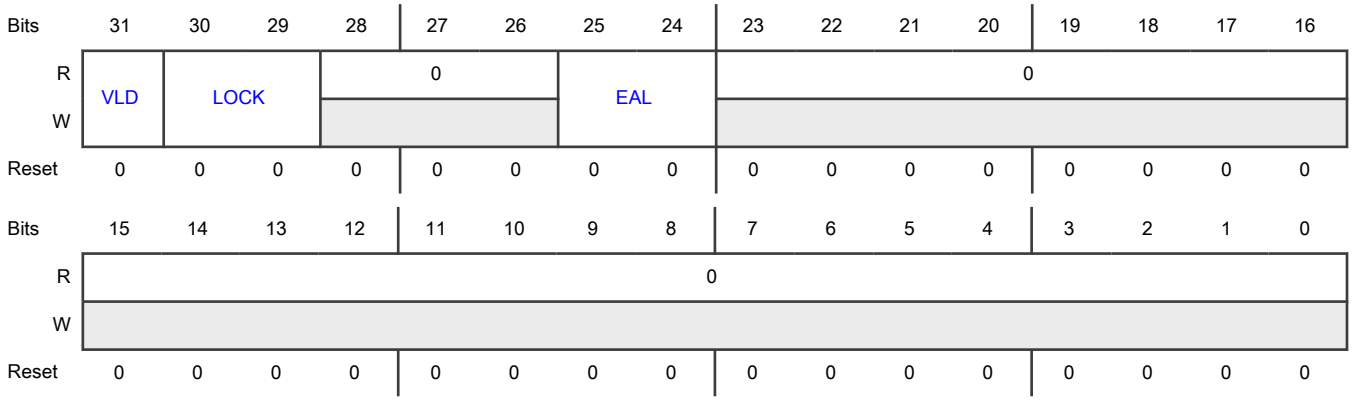
For n = 0 to 7:

Register	Offset
FRADn_WORD3	80Ch + (n × 20h)

Function

This register is access controlled and can only be programmed by privilege masters.

Diagram



Fields

Field	Function
31 VLD	Valid This field indicates whether the FRAD Descriptor for a specific flash region is valid. This field can not be written in same cycle if EAL field is being written as 10 or 11. **NOTE** You must enable this field after programming FRAD0_WORD0, FRAD0_WORD1, FRAD0_WORD2 and FRAD0_WORD3 registers. 0b - FRAD-Assignment is invalid 1b - FRAD-Assignment is valid
30-29 LOCK	Descriptor Lock This field enables masking of accidental write on FRAD registers. Lock is enabled/disabled by Secure/Privileged master. Lock functionality does not affects the EAL bits of FRAD. They can still be written even if FRAD descriptors are locked. **NOTE** These bits are 'sticky' in nature. After each bit is written with 1, writing 0 has no effect. It remains 1 until after a hard reset. 00b-01b - Lock disabled. Descriptor registers can be written by any master 10b - Lock enabled. Descriptors are read-only. MDnACP fields can be programmed only by the master with ID matching the MID check of their respective Target group MDAD. If the Target group MDAD is not valid then MDnACP fields also become read-only. 11b - Lock enabled. Descriptor registers are read-only.
28-26 —	Reserved
25-24 EAL	Exclusive Access Lock This field provides exclusive write lock over a FRAD region based on MD*n*ACP.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00b - No lock. Write permissions available for all masters based on their MDxACP evaluation. 01b - NA 10b - Lock enabled. Write permissions revoked for all domains. It can only be changed to 11b and can't directly be changed to 00b. 11b - Lock enabled. Exclusive write permission for master with master ID given in FRADn_WORD3[EALO] fields based on its MDxACP evaluation. Write disabled for other masters. Exclusive lock can be taken by a master only if it its ID matches the master ID check of any of the MDAD target queue, which is valid. MDxACP field corresponding to that target queue is not programmed as 0b. when a master writes 11b on this field it will be written only if above condition is met and then the master ID will be stored in FRADn_WORD3[EALO]. No transfer error will be generated in case the master does not satisfies the check.
23-0 —	Reserved

80.13.2.41 Flash Region Compare Address Status (FRAD0_WORD4 - FRAD7_WORD4)

Offset

For n = 0 to 7:

Register	Offset
FRADn_WORD4	810h + (n × 20h)

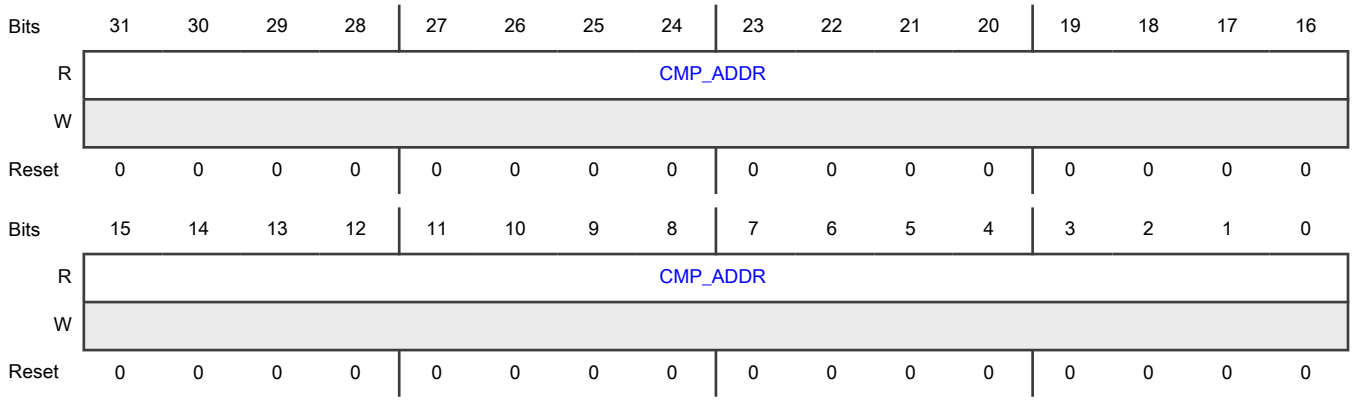
Function

This register is access controlled and can only be programmed by privilege masters.

NOTE

Any write access to these registers will result in bus transfer error

Diagram



Fields

Field	Function
31-0 CMP_ADDR	Capture Address This field indicates the 32-bit start address of the last transaction which lies inside the address range of this FRAD descriptor.

80.13.2.42 Flash Region Compare Status Data (FRAD0_WORD5 - FRAD7_WORD5)

Offset

For n = 0 to 7:

Register	Offset
FRADn_WORD5	814h + (n × 20h)

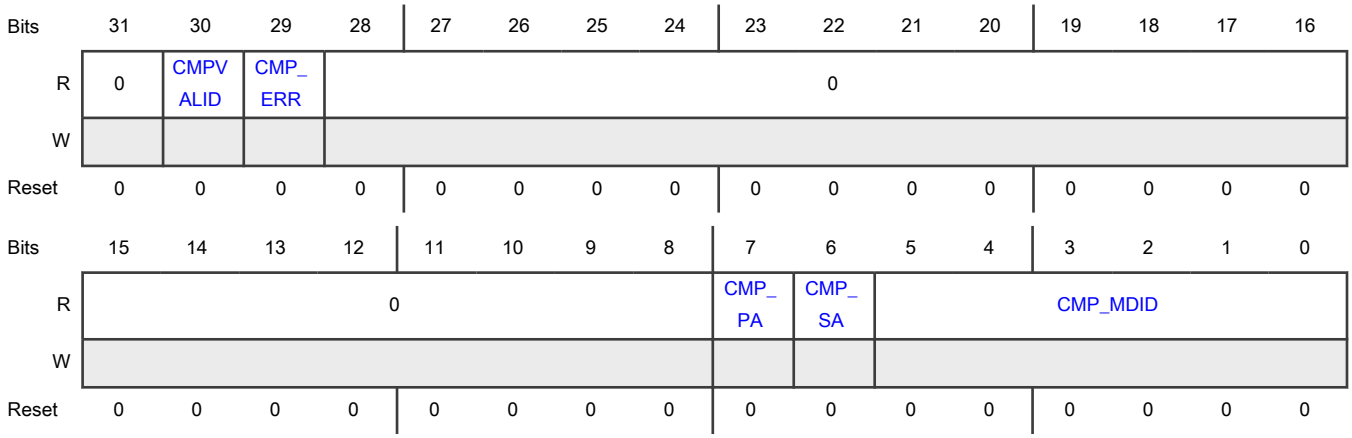
Function

This register is access controlled and can only be programmed by privilege masters.

NOTE

Any write access to these registers will result in bus transfer error

Diagram



Fields

Field	Function
31 —	Reserved
30 CMPVALID	Comparison Valid This field indicates the validity of flash region specific comparison check.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Access result/status not available 1b - Access result/status is available
29 CMP_ERR	Comparison Error This field indicates the error status (based on secure/privilege attributes checked with MDxACP permissions or exclusive access lock as programmed in FRAD) for flash region specific comparison check for last transaction. When this field is set, it can be cleared by writing '1' to W1C field of ERRSTAT register. It will update new value only after the error field has been cleared. 0b - No error 1b - Access error
28-8 —	Reserved
7 CMP_PA	Capture Privilege Attribute Capture Privilege attribute of last transaction which passed the address check for this FRAD. 0b - Non-privilege transaction 1b - Privilege transaction
6 CMP_SA	Capture Secure Attribute Capture Secure attribute of last transaction which passed the address check for this FRAD. 0b - Non-secure transaction 1b - Secure transaction
5-0 CMP_MDID	Capture MDID Value Capture master ID (MDID) value of the last transaction which passed the address check for this FRAD.

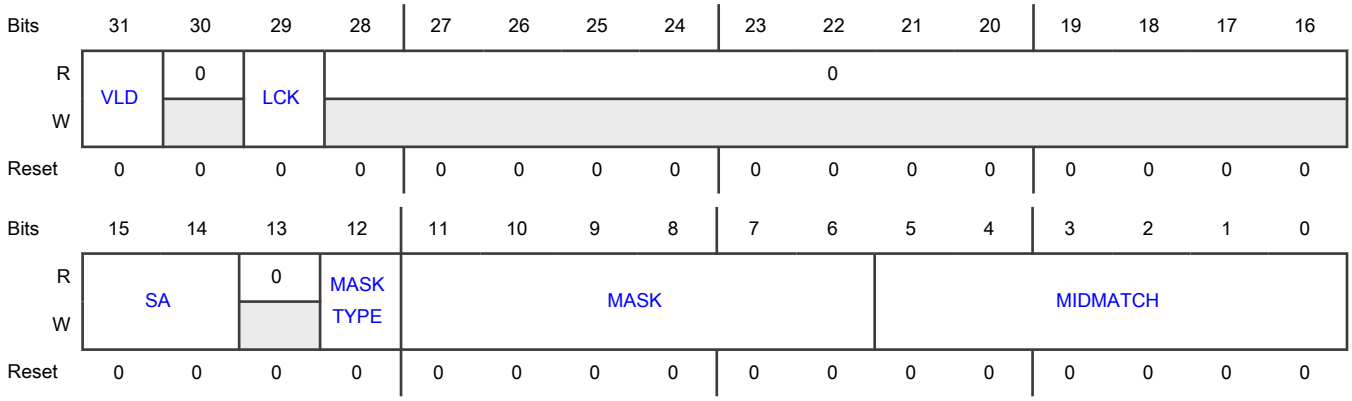
80.13.2.43 Target Group n Master Domain Access Descriptor (TG0MDAD - TG1MDAD)

Offset

Register	Offset
TG0MDAD	900h
TG1MDAD	910h

Function
This register is access controlled and can only be programmed by privilege masters.

Diagram



Fields

Field	Function
31 VLD	Valid Indicates whether MDAD Descriptor for the target group *n* is valid. 0b - MDAD-Assignment is invalid 1b - MDAD-Assignment is valid
30 —	Reserved
29 LCK	Descriptor Lock This field provides a means to make the MDAD descriptor read-only. **NOTE** This bit is 'sticky' in nature. After it is written with 1, writing 0 has no effect. It remains 1 until after a hard reset. 0b - Lock disabled. Registers can be written. 1b - Lock enabled. Registers are read-only.
28-16 —	Reserved
15-14 SA	Secure Attribute Defines the secure attribute selection criteria for entry into descriptor queue.. 00b - NA. This option should not be used. Allows the bus attribute for this master to non-secure only 01b - Allow the bus attribute for this master to non-secure only 10b - Allow the bus attribute for this master to secure only 11b - Allow the bus master's attribute: Both secure and non-secure

Table continues on the next page...

Table continued from the previous page...

Field	Function
13 —	Reserved
12 MASKTYPE	Mask Type 0b - ANDed mask 1b - ORed mask
11-6 MASK	Mask Defines the 6-bit mask value for the ID-Match comparison
5-0 MIDMATCH	Master ID Reference Specifies the reference value of the Master-ID (MID) for MID-comparison

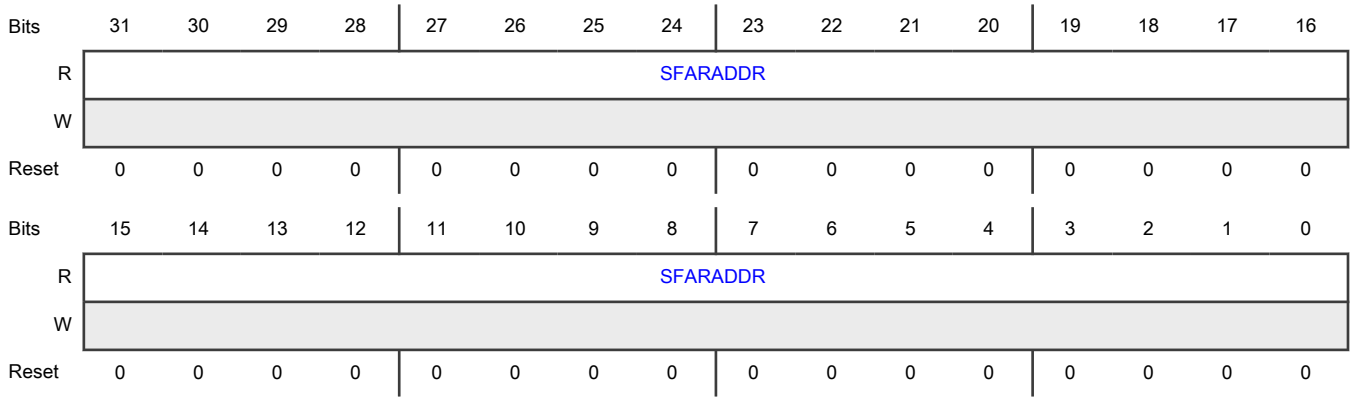
80.13.2.44 Target Group n SFAR Address (TG0SFAR - TG1SFAR)

Offset

Register	Offset
TG0SFAR	904h
TG1SFAR	914h

Function
Flash memory start address of the transaction in this target group queue.

Diagram



Fields

Field	Function
31-0 SFARADDR	SFAR Address Flash memory start address of the transaction which qualified into this target group queue.

80.13.2.45 Target Group n SFAR Status (TG0SFARS - TG1SFARS)

Offset

Register	Offset
TG0SFARS	908h
TG1SFARS	918h

Function
Target group *n* SFAR status.

NOTE

This register can only be read if the master reading the register passes the security checks set in TGnMDAD[SA] fields or if MDAD descriptor is not valid. Else this register will be read as 0. It will also read as 0 if TGnMDAD[SA] bits in descriptor are set as 00.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	VLD	ERR	CLR	0												
W			W1C													
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0			PA	0	SA	0				TG_MID					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 VLD	Valid Indicates whether the SFAR Address for the target group *n* is valid and queue is busy. This bit will not be set when ERR bit is set. 0b - SFAR-Assignment is invalid

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - SFAR-Assignment is valid
30 ERR	Error Indicates whether the SFAR address stored by a Master is with required access attributes for this target group descriptor. 0b - SFAR with required attributes 1b - SFAR without required attributes
29 CLR	Clear Clears the SFAR Status register. This register can only be cleared by the master having security attribute defined by TG*n*MDAD[SA]. After clearing if there is any valid transaction present in MDAD queue, it will again get updated in this register else it will be cleared. It will not be cleared if TGxMDAD[SA] bits are set to 00.
28-13 —	Reserved
12 PA	Privileged Attribute Privilege attribute of the master which wrote the SFAR register. 0b - Non-privileged 1b - Privileged
11 —	Reserved
10 SA	Secure Attribute Secure attribute of the master which wrote the SFAR register. 0b - Non-secure 1b - Secure
9-6 —	Reserved
5-0 TG_MID	Transaction Master ID Indicates the Master-ID of a transaction which programmed the SFAR registers with required attributes.

80.13.2.46 Target Group n IPCR Status (TG0IPCRS - TG1IPCRS)

Offset

Register	Offset
TG0IPCRS	90Ch
TG1IPCRS	91Ch

Function

Target group *n* IPCR status.

NOTE

This register can only be read if the master reading the register passes the security checks set in TGnMDAD[SA] fields or if MDAD descriptor is not valid. Else this register will be read as 0. It will also read as 0 if TGnMDAD[SA] bits in descriptor are set as 00.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	VLD	ERR		CLR	0					ARB_ UNL...	ARB_ LOCK	PAR	SEQID			
W				W1C												
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	IDATSZ															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 VLD	Valid Indicates whether the IPCR-IDATZ, IPCR-SEQID and PAR stored in this target group queue is valid and queue is locked. This bit will not be set in case ERR is set. 0b - IPCR-assignment is invalid 1b - IPCR-assignment is valid and queue is locked.
30-29 ERR	Error Indicates whether the IPCR stored by a Master with required access attributes wrt target group MDAD 00b - IPCR programming with required attributes 01b - IPCR-DATZ programming without required attributes

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10b - IPCR-SEQID programming without required attributes 11b - IPCR-DATZ and SEQID both programming without required attributes
28 CLR	Clear Clears the status in IPCR Status register. This register can only be cleared by the master having security attribute defined by TGnMDAD[SA]. After clearing if there is any valid transaction present in MDAD queue it will again get updated in this register else it will be cleared. It will not be cleared if TGxMDAD[SA] bits are set to 00.
27-23 —	Reserved
22 ARB_UNLOCK	Arbitration Unlock Specifies if the arbitration unlock is requested for TGn. 0b - No effect 1b - Arbitration unlock is requested
21 ARB_LOCK	Arbitration Lock Specifies if the arbitration lock is requested for TGn. 0b - No effect 1b - Arbitration lock is requested
20 PAR	Parallel Mode Enable Value Indicates the parallel mode enable value programmed in the target group queue.
19-16 SEQID	SEQID Value Indicates the SEQID value programmed in target group queue.
15-0 IDATSZ	IDATSZ Value Indicates the IDATSZ value programmed in target group queue.

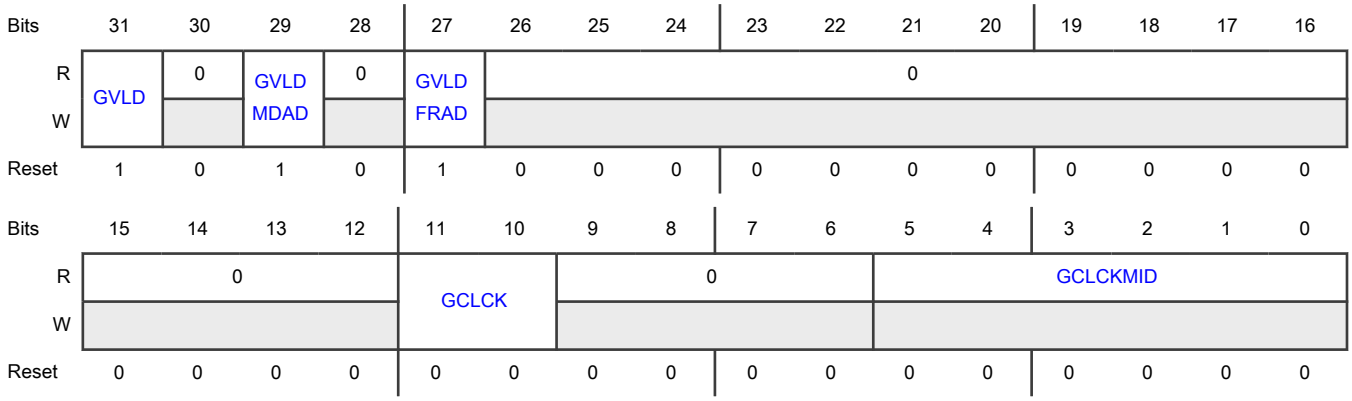
80.13.2.47 Master Global Configuration (MGC)

Offset

Register	Offset
MGC	920h

Function
This register is access controlled. It can only be programmed by master with master ID equal to 000011b. For MGC, MRC and MTO registers access checks master ID is being used. For transaction level checks inside MDAD and FRAD domain IDs are being used.

Diagram



Fields

Field	Function
31 GVLD	Global Valid access control 0b - Access controls are disabled. No descriptor comparison check. All transactions are allowed. 1b - Access controls are enabled.
30 —	Reserved
29 GVLDMDAD	Global Valid MDAD 0b - MDADs are disabled 1b - MDADs are enabled
28 —	Reserved
27 GVLDFRAD	Global Valid FRAD 0b - FRADs are disabled 1b - FRADs are enabled
26-12 —	Reserved
11-10 GCLCK	Global Configuration Lock This field provides a mechanism to limit write access to descriptors specific registers (MGC, MRC and MTO registers). If lock is enabled (MGC[11]=1b), this field also restricts write access to BFGENCR and LUT registers. 00b - Global Lock disabled. Registers can be written based on individual register access attribute. 01b - NA 10b - Lock enabled. Only the global configuration lock owner can write to the registers.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	11b - Lock enabled. All registers are read only until unlocked
9-6 —	Reserved
5-0 GCLCKMID	Global configuration Lock Owner Status This fields specifies the 6-bit Master-ID of Global configuration Lock owner (which set MGC[GCLCK]=10). Input value of Global Master-ID is provided by sideband signals.

80.13.2.48 Master Read Command (MRC)

Offset

Register	Offset
MRC	924h

Function

This register is access controlled. It can only be programmed by master with master ID equal to 000011b.

NOTE

In case of Sequence Auto Join instruction in LUT, SFP will parse only first sequence. Second sequence will not be parsed for checking Read commands.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	VLDC MD03	READ_CMD3						0	VLDC MD02	READ_CMD2					
W																
Reset	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		READ_CMD1						0		READ_CMD0					
W																
Reset	0	0	0	0	1	1	1	0	0	0	0	0	0	1	1	1

Fields

Field	Function
31 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
30 VLDCMD03	Valid command 0b - READ_CMD3 value invalid 1b - READ_CMD3 value valid
29-24 READ_CMD3	Read Command 3 Programmes the additional READ Instruction/Command encoding information, if applicable
23 —	Reserved
22 VLDCMD02	Valid command 0b - READ_CMD2 value invalid 1b - READ_CMD2 value valid
21-16 READ_CMD2	Read Command 2 Stores the existing instruction—DATA_LEARN command encoding information. Can be overwritten with additional read command, if required.
15-14 —	Reserved
13-8 READ_CMD1	Read Command 1 Stores the existing instruction—READ_DDR command encoding information.
7-6 —	Reserved
5-0 READ_CMD0	Read Command 0 Stores the existing instruction—READ command encoding information.

80.13.2.49 Master Timeout (MTO)

Offset

Register	Offset
MTO	928h

Function
This register is access controlled. It can only be programmed by master with master ID equal to 000011b.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	WRITE_TO															
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	WRITE_TO															
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-0 WRITE_TO	Write Timeout Maximum timeout value to abort the ongoing write or read command. The timeout counter starts after the access from any target queue has won the arbitration and QSPI is IDLE (FSM_STAT state field is not 00). SFP queue is cleared once this timeout value has been reached and interrupt is generated if enabled.

80.13.2.50 FlashSeq Request (FLSEQREQ)

Offset

Register	Offset
FLSEQREQ	92Ch

Function

This register is access controlled and can only be programmed by privilege masters.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	VLD	0	CLR	0	TIMEOUT	0			0		CMD	0	SEQID			
W			W1C													
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	FRAD			0	ARB_LOCK	PA	SA	0	REQ_TG	REQ_MID					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 VLD	Valid Indicates whether the FlashSeq request status is valid 0b - Status is invalid 1b - Status is valid
30 —	Reserved
29 CLR	Clear Clears the status
28 —	Reserved
27 TIMEOUT	Timeout Error Status Timeout error status of last executed FlashSeq request. 0b - Instruction completed without timeout error 1b - Instruction aborted after timeout error
26-25 —	Reserved
24-23 —	Reserved
22 CMD	Instruction Type Instruction type of last executed FlashSeq request. 0b - Read Instruction Sequence 1b - Non-Read Instruction Sequence
21-20 —	Reserved
19-16 SEQID	Sequence ID Sequence ID of last executed FlashSeq request.
15 —	Reserved
14-12 FRAD	Flash Region Descriptor Number Flash Region specific number (FRAD) of last executed FlashSeq request.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This will remain 0 in case of read transfers.
11 —	Reserved
10 ARB_LOCK	Arbitration Lock Arbitration lock status for last completed request. 0b - Arbitration was not locked 1b - Arbitration was locked
9 PA	Privilege Attribute Privilege attribute of last executed FlashSeq request. 0b - Non-privilege Transaction 1b - Privilege Transaction
8 SA	Secure Attribute Secure attribute of last executed FlashSeq request. 0b - Non-secure Transaction 1b - Secure Transaction
7 —	Reserved
6 REQ_TG	FlashSeq Request Target Group Target group queue from which last executed transaction passed. 0b - TG0 1b - TG1
5-0 REQ_MID	FlashSeq Request Master ID Indicates the master-ID of last executed FlashSeq request.

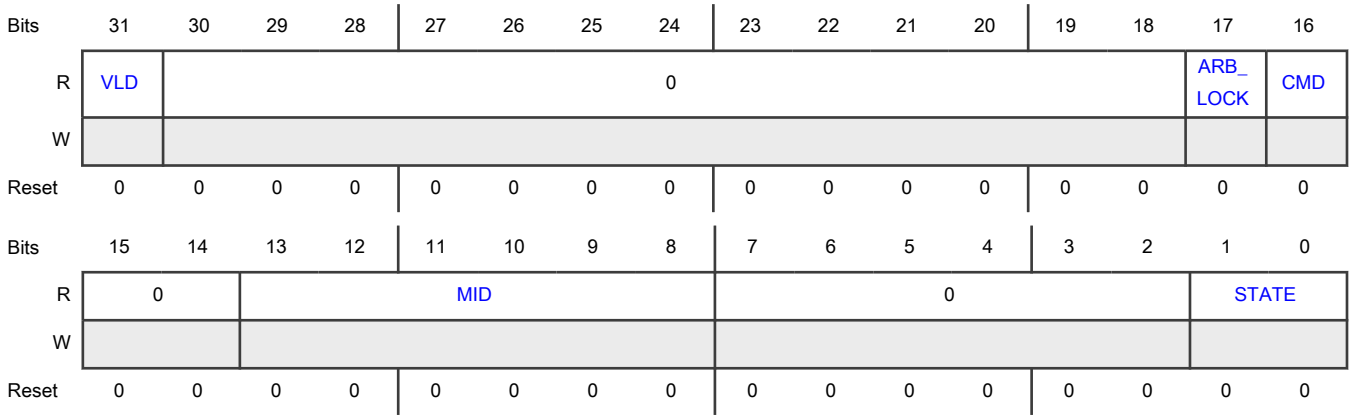
80.13.2.51 FSM Status (FSMSTAT)

Offset

Register	Offset
FSMSTAT	930h

Function
This register is access controlled and can only be programmed by privilege masters.

Diagram



Fields

Field	Function
31 VLD	Valid Indicates whether the FSM Status is valid. 0b - Status is invalid. No IPS transfer is queued. 1b - Status is valid. IPS transfer is queued or execution on QuadSPI.
30-18 —	Reserved
17 ARB_LOCK	Arbitration Lock Arbitration lock status for present request. 0b - Arbitration not locked 1b - Arbitration locked
16 CMD	Command Instruction type of currently initiated Flash transaction on QuadSPI. 0b - Read instruction sequence 1b - Non-read instruction sequence
15-14 —	Reserved
13-8 MID	Master ID Indicates the Master-ID of the currently initiated FlashSeq transaction on QuadSPI.
7-2 —	Reserved
1-0	FSM State Status

Table continues on the next page...

Table continued from the previous page...

Field	Function
STATE	00b - Transaction is Queued, but QuadSPI is busy with AHB transfer, any previous DMA transaction is ongoing, any residue data left in RDBFL or if any interrupt is pending.. 01b - TBDR lock is open. IPS master can write in TBDR. 10b - Write transfer is triggered. SEQID is written to QuadSPI. 11b - Read transfer is triggered. SEQID is written to QuadSPI.

80.13.2.52 IPS Error (IPSERROR)

Offset

Register	Offset
IPSERROR	934h

Function

This register contains the error status of IPS write on SFAR or IPCR registers if they are not qualified for any of the target group queues.

This register is access controlled and can only be programmed by privilege masters.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		CLR		0											
W			W1C													
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FRADPROG	MDADPROG	TG1MID	TG0MID	TG1SEC	TG0SEC	TG1LCK	TG0LCK	0		MID					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29	Clear

Table continues on the next page...

Table continued from the previous page...

Field	Function
CLR	Clear the status of IPS Error register.
28-16 —	Reserved
15 FRADPROG	FRAD Descriptor Program Status 0b - Some or all of the FRAD descriptors are programmed 1b - None of the FRAD descriptors are programmed
14 MDADPROG	TG/MDAD Descriptor Program Status 0b - One or both of target group descriptors programmed 1b - None of the target group descriptors are programmed and valid
13-12 TGnMID	TGn Master-ID Status 0b - TGn master-ID check passed 1b - TGn master-ID check failed
11-10 TGnSEC	TGn Security Status 0b - Security attribute check passed for TGn 1b - Security attribute check failed for TGn
9-8 TGnLCK	TGn Lock 0b - TGn queue SEQID is not written yet. 1b - TGn queue SEQID is written and queue is locked
7-6 —	Reserved
5-0 MID	IPS DID Master ID IPS Master ID for the transaction which generated this IPS transfer error.

80.13.2.53 Error Status (ERRSTAT)

Offset

Register	Offset
ERRSTAT	938h

Function
This register is access controlled and can only be programmed by privilege masters.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	TO_ERR	TG1IPCR	TG0IPCR	TG1SFAR	TG0SFAR	IPS_ERR	FRAD7ACC	FRAD6ACC	FRAD5ACC	FRAD4ACC	FRAD3ACC	FRAD2ACC	FRAD1ACC	FRAD0ACC	FRADMTCH
W								W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-15 —	Reserved
14 TO_ERR	Timeout Error This field is set if any flash transaction generated on QuadSPI results in timeout error and is aborted by SFP module. This field can be cleared by writing 1 to FLSEQREQ[CLR]. 0b - No timeout Error generated 1b - Timeout error is generated
13-12 TGnIPCR	TGn IPCR Error This field is set if any IPCR write generates error while being written to Target group queue. This field can be cleared by writing 1 to TG*n*IPCRS[CLR]. 0b - No Error generated 1b - Error is generated
11-10 TGnSFAR	TGn SFAR Error This field is set if any SFAR write generates error while being written to Target group queue. This field can be cleared by writing 1 to TG*n*SFARS[CLR]. 0b - No Error generated 1b - Error is generated
9 IPS_ERR	IPS Error Some common error occurred and IPS bus transfer error is also generated. The details of the IPS error can be found in IPSERROR register. This field can be cleared by writing 1 to IPSERROR[CLR].

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No Error generated 1b - Error is generated
8-1 FRADnACC	FRADn Access Error This bit is set when the transaction address lies within the address range of this FRAD but it does not qualify the access permission checks for this FRAD or this FRAD was under exclusive lock (FRAD*n*_WORD3[EAL] = 10 or 11) by another master. It may also be set if the transaction address qualifies for multiple FRAD regions. 0b - No valid Error transaction 1b - Transaction is with error and target queue is cleared
0 FRADMTCH	No FRAD Match Error Transaction address does not lie within address range of any FRAD descriptor. 0b - No Error generated 1b - Transaction does not lie within any FRAD address range and error is generated

80.13.2.54 Interrupt Enable (INT_EN)

Offset

Register	Offset
INT_EN	93Ch

Function

This register is access controlled and can only be programmed by privilege masters.

NOTE

In case Queue Specific Error Interrupt bits (TG0SFAR,TG1SFAR,TG0IPCR,TG1IPCR) are enabled, ensure that the core executing interrupt handler can clear the Queue specific TGxSFARS and TGxIPCRS registers only when core's security attribute matches the queue. In case there is a mismatch in security attribute, you must change the TGMDADx.SA attribute of that queue to match the core SA attribute. After that, core can clear the error and then revert the changes in TGMDADx.SA field.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	TO_ERR	TG1IPCR	TG0IPCR	TG1SFAR	TG0SFAR	IPS_ERR	FRAD7ACC	FRAD6ACC	FRAD5ACC	FRAD4ACC	FRAD3ACC	FRAD2ACC	FRAD1ACC	FRAD0ACC	FRADMTCH
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-15 —	Reserved
14 TO_ERR	Timeout Error Interrupt Enable 0b - Interrupt disabled 1b - Interrupt enabled
13-12 TGnIPCR	TGn IPCR Error Interrupt Enable 0b - Interrupt disabled 1b - Interrupt enabled
11-10 TGnSFAR	TGn SFAR Error Interrupt Enable 0b - Interrupt disabled 1b - Interrupt enabled
9 IPS_ERR	IPS Error Interrupt Enable 0b - Interrupt disabled 1b - Interrupt enabled
8-1 FRADnACC	FRADn Access Error Interrupt Enable 0b - Interrupt disabled 1b - Interrupt enabled
0 FRADMTCH	No FRAD Match Error Interrupt Enable 0b - Interrupt disabled 1b - Interrupt enabled

80.13.3 Serial flash memory address assignment

The serial flash memory address assignment can be modified by writing into Serial Flash Memory A1 Top Address Register (SFA1AD) and Serial Flash Memory A2 Top Address Register (SFA2AD) for device A

Table 806. Serial flash memory address assignment

Parameter	Function	Access mode
QuadSPI_AMBA_BASE ((31:10) - 22 bits)	QuadSPI AHB base address First address of the serial flash memory device as presented to the QuadSPI controller. This might be the base of the serial flash memory in the system address map or it may be a remapping, for instance to 0h, performed by the system. (See the system address map file attached to this document)	
QuadSPI_ARDB_BASE	First address of the QuadSPI Rx buffer on system memory map	
TOP_ADDR_MEMA1(TPADA1)	Top address for the external flash memory A1 (the first of the two independent flash memories sharing the IOFA)	Any access to the address space between TOP_ADDR_MEMA1 and QuadSPI_AMBA_BASE is routed to serial flash memory A1.
TOP_ADDR_MEMA2(TPADA2)	Top address for the external flash memory A2 (the second of the two independent flash memories sharing the IOFA)	Any access to the address space between TOP_ADDR_MEMA2 and TOP_ADDR_MEMA1 is routed to serial flash memory A2.

80.14 Flash memory mapped AMBA bus

QuadSPI_AMBA_BASE defines the address to be used as the start address of the serial flash memory device, as defined by the system memory map. Note that this may be a remapping of the physical address of the serial flash memory in the system. See the system address map for details.

Table 807. QuadSPI AMBA bus memory map

Address	Register name
QuadSPI_AMBA_BASE to (TOP_ADDR_MEMA2 - 1h)	• See Memory-mapped serial flash memory data—individual flash memory mode on flash memory A. • For information about byte ordering, see Table 779 and Table 780.
QuadSPI_ARDB_BASE to... (32 * 4 Byte) QuadSPI_ARDB_BASE + 1FFh	• See AHB RX Data Buffer Register (ARDB0 - ARDB31). • For information about the byte ordering, see Table 779.

NOTE

Any read access to non-implemented addresses provides undefined results.

In individual flash memory modes, the 3/4 address bytes (as programmed in the instruction/operand in the sequence) available for the flash memory address are determined by SFADR[23:0] or SFADR[31:0] as provided in the table shown above.

80.14.1 AHB bus access read considerations

Note that all logic in the QuadSPI module implementing the AHB bus access is designed to read the content of an external serial flash memory device. Therefore, the following restrictions apply to the QuadSPI module with respect to accesses to the AHB bus:

- Any AHB command resulting in the assertion of the FR[ABSEF] flag is answered with the ERROR condition according to the AMBA_AHB specification. The resulting AHB command is ignored.
- AHB bus read access types—NONSEQ and BUSY—are fully supported.
- AHB read access type—SEQ—is treated in the same way as NONSEQ. See Flash memory mapped AMBA bus for details.
- Early burst termination is not supported for AHB transactions.
- An AHB error response occurs when FR[AITEF] bit is set
- An AHB bus stall along with error response occurs when FR[AAEF] bit is set

80.14.2 Memory-mapped serial flash memory data—individual flash memory mode on flash memory A

Starting with address QuadSPI_AMBA_BASE, the content of the first external serial flash memory device is mapped into the address space of the device containing the QuadSPI module. Serial flash memory byte address 0h corresponds to bus address, QuadSPI_AMBA_BASE, in an increasing order. . See the following table for the address mapping. The byte ordering for 32-bit access is provided in Table 779 and for 64-bit read access, the byte ordering is provided in Table 780.

Table 808. Memory-mapped individual flash memory mode—flash memory A address scheme

Memory-mapped address 32-bit access	Memory-mapped address 64-bit access	Serial flash memory byte address	Flash memory device
QuadSPI_AMBA_BASE + 0h	QuadSPI_AMBA_BASE + 0h	0h to 3h	A1
QuadSPI_AMBA_BASE + 4h		4h to 7h	
...	...	...	
TOP_ADDR_MEMA1 - 8h	TOP_ADDR_MEMA1 - 8h	(TOP_ADDR_MEMA1- 8h) to (TOP_ADDR_MEMA1 - 0h4 -0h1)	
TOP_ADDR_MEMA1 - 4h		(TOP_ADDR_MEMA1 - 4h) to (TOP_ADDR_MEMA1 - 0h1)	
TOP_ADDR_MEMA1 + 4h	TOP_ADDR_MEMA1 + 0h	0h to 3h	A2
TOP_ADDR_MEMA1 + 0h4		4h to 7h	
...	...	...	
TOP_ADDR_MEMA2 - 8h	TOP_ADDR_MEMA2 - 8h	(TOP_ADDR_MEMA2 - 8h) to (TOP_ADDR_MEMA2 - 4h - 1h)	
TOP_ADDR_MEMA2 - 4h		(TOP_ADDR_MEMA2 - 4h) to (TOP_ADDR_MEMA2 - 1h)	

The available address range depends on the size of the external serial flash memory device. Any access beyond the size of the external serial flash memory provides undefined results.

For details concerning the read process, see Flash memory read.

80.14.3 ARDB register descriptions

NOTE

See the system memory map in this document for the base address of the QuadSPI AHB RX data buffer.

80.14.3.1 ARDB memory map

QuadSPI_ARDB base address: 6800_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h - 7Ch	AHB RX Data Buffer Register (ARDB0 - ARDB31)	32	R	0000_0000h

80.14.3.2 AHB RX Data Buffer Register (ARDB0 - ARDB31)

Offset

For a = 0 to 31:

Register	Offset
ARDBa	0h + (a × 4h)

Function

This register is used to read the buffer content of the RX buffer from successive addresses. ARDB0 corresponds to the RX buffer register entry corresponding to the current value of the read pointer in an increasing order.

The increment of the read pointer depends on the access scheme (DMA or flag-driven). See Flash memory read, RX buffer, data read through register interface, and AHB read for the description of successive accesses to the RX buffer content. See Byte Ordering of Serial Flash Memory Read Data for the byte ordering scheme.

Valid address range accessible in the ARDB*n* range depends on the number of RX buffer entries implemented and on the number of valid buffer entries available in the RX buffer.

- Example 1 - RX buffer filled completely with 32 words: In this case, the address range for valid read access extends from ARDB0 to ARDB31 .
- Example 2 - RX buffer filled with five valid words; RX buffer fill level RBSR[RDBFL] is 5. In this case, an access to ARDB4 provides the last valid entry.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ARXD															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ARXD															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 ARXD	ARDB provided RX buffer data Byte order (endianness) is identical to the RX buffer data registers.

80.15 Glossary

AHB	Advanced high-performance bus, a version of AMBA
AMBA	Advanced microcontroller bus architecture
APB	Advanced peripheral bus
BE	Big endian byte ordering
CRS	Center aligned read strobe
CS	Chip select
FRAD	Flash region access descriptor
I/O	Input output, I/O lines are also referred to as pads in this chapter
IFM	Individual flash memory mode
LE	Little endian
MDAD	Master domain access descriptors
MGID	Master-Group identifier
PCS	Peripheral chip select
SCK	Serial communications clock
SFM	Serial flash memory

Chapter 81
Ultra Secured Digital Host Controller (uSDHC)

81.1 Chip-specific uSDHC information

81.1.1 uSDHC instances and configuration

This chip supports up to one instance of uSDHC module.

Table 809. uSDHC instances

Instance	S32K358/S32K356/S32K348/S32K338/ S32K328	S32K310/S32K311/S32K312/S32K314/ S32K322/S32K324/S32K341/S32K342/S32K344/ S32K388/S32K389
uSDHC	Yes	No

Table 810. Features supported

Features	Notes
MMC	• System specification version 4.2/4.3/4.4/4.41 • Bit Widths x1/x4/x8 • Full Speed Mode upto 25MHz • High Speed SDR upto 50MHz • High Speed DDR mode (50MHz both edges)
SD/ SDIO	• Card Specification version 2.0/3.0 • Bit Widths x1/x4 • Full Speed Mode upto 25MHz • High Speed Mode upto 50MHz
FIFO	128 x 32 external FIFO supported
External DMA	Supported
Write Protect WP system implementation	Supported
Voltage support	3.3 V

NOTE

- uSDHC cannot directly write QSPI if flash page is not 512 bytes.
- Card clock pin should have weak pull down enabled when configured for uSDHC function.

This chip doesn't support the following features:

- MMC HS400 MODE
- Card Detection (CD) System Implementation
- LED control (LCTL) output signal
- IO power Voltage selection Signal (VSELECT)
- Input clock for eMMC HS400 mode (STROBE)

- DLL tuning is not supported in S32K358 since SDHC has 50 MHz frequency. Whereas, ideal tuning frequency requirement is 200 MHz.

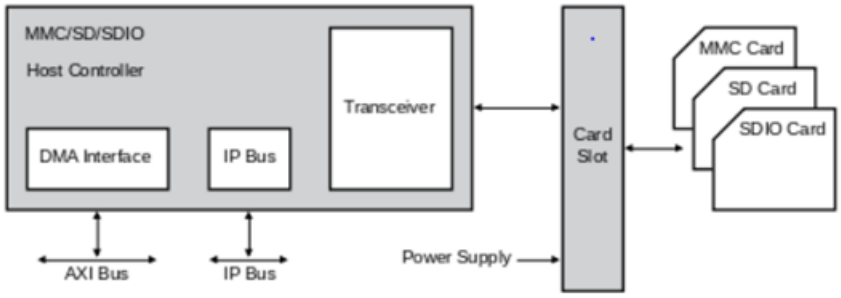


Figure 534. System connection of uSDHC

81.1.2 Sequence for DLL implementation in SDR25 mode

To implement DLL in SDR25 mode the following steps need to be done without using CMD19 and CMD21:

1. Set TUNING_CTRL[STD_TUNING_EN] to 0.
2. Set MIX_CTRL[EXE_TUNE] to 0.
3. Set SYS_CTRL[RSTA] to reset all.
4. Set MIX_CTRL[SMP_CLK_SEL] to 1.
5. Set bit[14:0] of CLK Tuning Control and Status (CLK_TUNE_CTRL_STATUS) with the 32'h400. This needs to be configured as per the number of delay cells.
6. Poll bit[30:16] of CLK Tuning Control and Status (CLK_TUNE_CTRL_STATUS) until the value written in Step 5 above is read.

81.2 Overview

uSDHC provides the interface between the host system and the eMMC, SD card, and SDIO as shown in Figure 536. The module acts as a bridge, passing host bus transactions to the eMMC, SD card, and SDIO by sending commands and performing data accesses to/from the cards. It handles the SD card/SDIO/eMMC protocols at the transmission level.

81.2.1 Block diagram

The following figure illustrates the block diagram for uSDHC. Dual port SRAM is FIFO for write/read. For more information, see Data buffer.

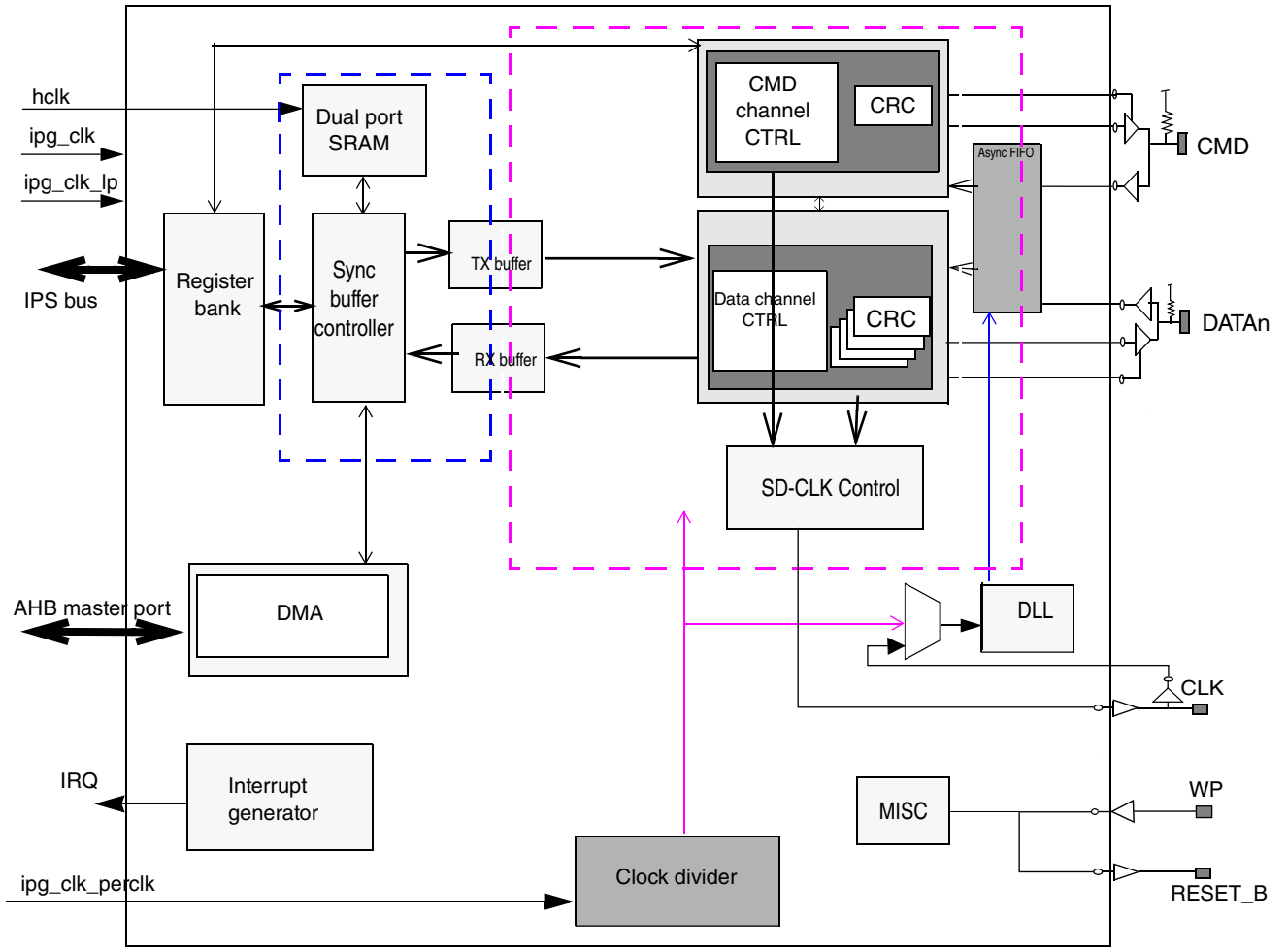


Figure 535. uSDHC block diagram

The figure below shows the System connection of uSDHC:

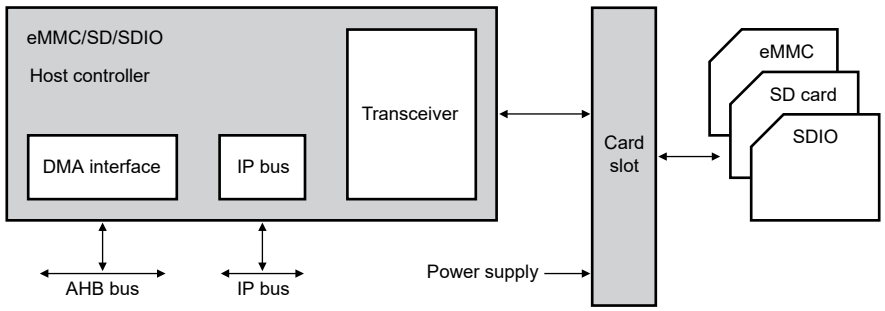


Figure 536. System connection of uSDHC

The following are brief descriptions of the cards supported by uSDHC:

- The Embedded MultiMediaCard (eMMC) is a universal low-cost data storage and communication media designed to cover a wide array of applications including mobile video and gaming. Previous eMMC were based on a 7-pin serial bus with a single data pin, while the new high speed eMMC communication is based on an advanced 11-pin serial bus designed to operate in the low-voltage range.

- The Secure Digital Card (SD) is an evolution of the old eMMC technology. It is specifically designed to meet the security, capacity, performance, and environmental requirements inherent in newly emerging audio and video consumer electronic devices. The physical form factor, pin assignment, and data transfer protocol are backward compatible with the old eMMC (with some additions).
- Under the SD protocol, system connection can be categorized into memory card, I/O card, and combo card, which have both memory and I/O functions. The memory card invokes a copyright protection mechanism that complies with the security of the SDMI (Secure Digital Music Initiative) standard. The I/O card, which is also known as SDIO, provides high-speed data I/O with low-power consumption for mobile electronic devices.

81.2.2 Features

The list given below shows the features of the uSDHC module:

- Conforms to the SD Host Controller Standard Specification version 2.0
- Compatible with the eMMC System Specification version 4.2/4.3/4.4/4.41
- Compatible with the SD Memory Card Specification version 3.0 and supports the Extended Capacity SD Memory Card
- Compatible with the SDIO Specification version 2.0
- Designed to work with SD Memory, miniSD Memory, SDIO, miniSDIO, SD Combo, eMMC, MMC plus, and MMC RS cards
- For card bus clock frequency: See the product Data Sheet for the clock frequency of each supported mode.
- Supports 1-bit/4-bit SD and SDIO modes, and 1-bit/4-bit/8-bit eMMC modes
  - Up to 200 Mbps of data transfer for SDIO using four parallel data lines in the Single Data Rate (SDR) mode
  - Up to 400 Mbps of data transfer for SDIO using four parallel data lines in the Dual Data Rate (DDR) mode
  - Up to 200 Mbps of data transfer for SDXC cards using four parallel data lines in the Single Data Rate (SDR) mode
  - Up to 400 Mbps of data transfer for SDXC card using four parallel data lines in the Dual Data Rate (DDR) mode
  - Up to 416 Mbps of data transfer for eMMC using eight parallel data lines in the Single Data Rate (SDR) mode
  - Up to 832 Mbps of data transfer for eMMC using eight parallel data lines in the Dual Data Rate (DDR) mode
- Supports single block/multi-block read and write
- Supports block sizes of 1 ~ 4096 bytes
- Supports the write protection switch for write operations
- Supports both synchronous and asynchronous abort
- Supports pause during the data transfer at block gap
- Supports SDIO Read Wait and Suspend Resume operations
- Supports Auto CMD12 for multi-block transfer
- Host can initiate non-data transfer command while data transfer is in progress
- Allows cards to interrupt the host in 1-bit and 4-bit SDIO modes; also supports interrupt period
- Embodies a fully configurable 128x32-bit FIFO for read/write data
- Supports internal and external DMA capabilities
- Supports voltage selection by configuring vendor-specific register bit
- Supports Advanced DMA to perform linked memory access

NOTE

This block can support all the above-listed speed mode and maximum data throughput. However, these may be specific to the device. See the corresponding chip-specific information or the device data sheet for accurate details.

81.3 Functional description

The following sections provide a brief functional description of the major system blocks, including the data buffer, DMA AHB interface, register bank as well as IP Bus interface, dual-port memory wrapper, data/command controller, clock and reset manager, and clock generator.

81.3.1 Modes and operations

81.3.1.1 Data transfer modes

The uSDHC module can select the following modes for data transfer:

- SD 1-bit
- SD 4-bit
- eMMC 1-bit
- eMMC 4-bit
- eMMC 8-bit
- Identification mode (up to 400 kHz)
- eMMC full-speed mode (up to 26 MHz)
- eMMC high-speed mode (up to 52 MHz)
- eMMC DDR mode (52 MHz both edges)
- SD card or SDIO full-speed mode (up to 25 MHz)
- SD card or SDIO high-speed mode (up to 50 MHz)

NOTE

This block can support all the above listed speed mode and maximum clock frequency. However, these may be specific to the device. See the corresponding chip-specific information or the device data sheet for accurate details.

81.3.2 Data buffer

The uSDHC module uses one configurable data buffer to transfer data between the system bus (IP bus or advanced high-performance bus (AHB) bus) and the SD card in an optimized manner, maximizing throughput between the two clock domains (IP peripheral clock and the master clock). The buffer is used as a temporary storage for transferring data between the host system and the card. The watermark levels for read and write are both configurable and can range between 1 to 128 words. The burst lengths for read and write are also configurable and can range between 1 to 31 words. The next figure provides the uSDHC buffer scheme as buffer control and buffer RAM wrapper.

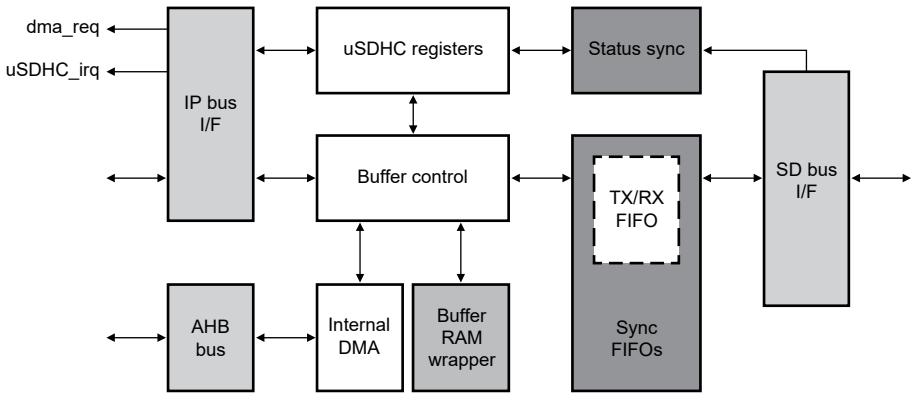


Figure 537. uSDHC buffer scheme

Here are 3 transfer modes to access the data buffer:

- CPU polling mode:
  - For a host-read operation, when the number of words received in the buffer meets or exceeds the RD_WML watermark value, by polling the BRR bit, the host driver can read the Buffer Data Port register to fetch the amount of words set in the RD_WML register from the buffer. The write operation is similar. For more information on the process of writing operation, see Write operation sequence.
- External DMA mode:
  - For a read operation, when there are more words received in the buffer than the amount set in WTMK_LVL[RD_WML], a DMA request is sent out to inform the external DMA to fetch the data. The request will be immediately de-asserted when there is an access on the Data Buffer Access Port (DATA_BUFF_ACC_PORT). If the number of words in the buffer after the current burst meets or exceeds RD_WML value, the DMA request is asserted again. For instance, if there are twice as many words in the buffer as there are in the RD_WML value, there are two successive DMA requests with only one cycle of de-assertion between. The write operation is similar. Note the accesses CPU polling mode and external DMA mode both use the IP bus, and if the external DMA is enabled, in both modes an external DMA request is sent when the buffer is ready.
- Internal DMA mode (includes simple and advanced DMA accesses):
  - The internal DMA access, either by simple or advanced DMA, is over the AHB bus. For internal DMA access mode, the external DMA request will never be sent out.

For a read operation, when there are more words in the buffer than the amount set in WTMK_LVL[RD_WML], the internal DMA starts fetching data over the AHB bus. Except for INCR4 and INCR8, the burst type is always the INCR mode and the burst length depends on the shortest of the following factors:

- Burst length configured in the burst length field of the Watermark Level register
- Watermark level boundary
- Block size boundary
- Data boundary configured in the current descriptor (if the ADMA is active)
- 1 KB address boundary defined in the AHB protocol

The Write operation functions in a similar manner—sequential and contiguous access is necessary to ensure that the pointer address value is correct. Random or skipped access is not possible. The byte order, by reset, is little endian mode. The actual byte order is swapped inside the buffer, according to the endian mode configured by software (see the following figures). For a host write operation, the byte order is swapped after the data is fetched from the buffer and ready to send to the SD Bus. For a host read operation, the byte order is swapped before the data is stored in the buffer.

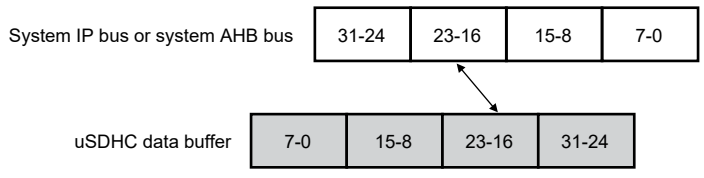


Figure 538. Data swap between system bus and uSDHC data buffer in the byte little endian mode

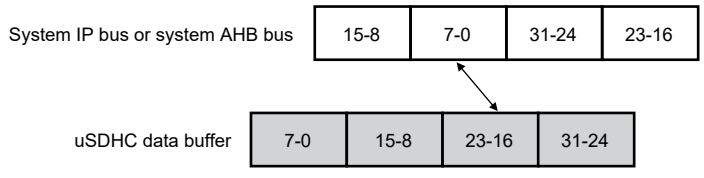


Figure 539. Data swap between system bus and uSDHC data buffer in half word big endian mode

81.3.2.1 Write operation sequence

There are 3 ways to write data into the buffer when the user transfers data to the card:

- External DMA through the uSDHC DMA request signal
- Processor core polling through the BWR bit in the Interrupt Status register (interrupt or polling)
- Internal DMA

When the internal DMA is not used, the DMAEN bit in the Transfer Type register is not set when the command is sent, uSDHC asserts a DMA request when the amount of buffer space exceeds the value set in WTMK_LVL[WR_WML] and is ready for receiving new data. At the same time, uSDHC sets the BWR bit. The buffer write ready interrupt is generated if it is enabled by software.

When internal DMA is used, uSDHC does not inform the system before all the required number of bytes are transferred (if no error is encountered). When an error occurs during the data transfer, uSDHC aborts the data transfer and abandons the current block. The host driver should read the contents of the DMA System Address register to obtain the starting address of the abandoned data block. If the current data transfer is in multi-block mode, uSDHC does not automatically send CMD12, even though the AC12EN bit in the Transfer Type register is set. The host driver sends CMD12 in this scenario and restarts the write operation from that address. It is recommended that a software reset for Data be applied before the transfer is restarted.

The uSDHC module does not start data transmission until the buffer has been filled with the number of words set in the WR_WML register. If the buffer is empty and the host system does not write data in time, uSDHC stops the CLK to avoid the data buffer underrun situation.

81.3.2.2 Read operation sequence

There are 3 ways to read data from the buffer when the data is received from the card:

- External DMA through uSDHC DMA request signal
- Processor core polling through the BRR bit in the Interrupt Status register (interrupt or polling)
- Internal DMA

When internal DMA is not used (DMAEN bit in Transfer Type register is not set when the command is sent), uSDHC asserts a DMA request when the amount of data exceeds the value set in the RD_WML register, which is available and ready for system fetching data. At the same time, uSDHC sets the BRR bit. The buffer read ready interrupt is generated if it is enabled by the software.

When internal DMA is used, uSDHC does not inform the system before all the required number of bytes are transferred (if no error is encountered). When an error occurs during the data transfer, uSDHC aborts the data transfer and abandons the current block. The host driver should read the content of the DMA System Address register to get the starting address of the abandoned data block. If the current data transfer is in multi-block mode, uSDHC does not automatically send CMD12, even though the AC12EN bit in the Transfer Type register is set. The host driver sends CMD12 in this scenario and restarts the read operation from that address. It is recommended that a software reset (in register RSTD) for data be applied before the transfer is restarted.

For any read transfer mode, uSDHC does not start data transmission until the number of words set in the RD_WML register are in buffer. If the buffer is full and the host system does not read data in time, uSDHC stops the CLK to avoid the data buffer overrun situation.

81.3.2.3 Data buffer and block size

In the uSDHC module, the data buffer can hold up to 128 words (32-bit) and the watermark levels for write and read can be configured accordingly. The user needs to know the buffer size for the buffer operation during a data transfer to utilize it in the most optimized way. For both read and write, the watermark level can range between 1 to 128 words. For both read and write, the burst length can range between from 1 to 31 words. The host driver may configure the value according to the system and requirement.

During a multi-block data transfer, the block length can be set to any value between 1 and 4096 bytes, satisfying the requirements of the external card. The only restriction is from the external card, which can be limited in size or support of a partial block access (which is not the integer times of 512 bytes). That means, the largest block size is 512 bytes.

As uSDHC treats each block individually, for block sizes which are not multiples of four (not word-aligned), stuffed bytes are required at the end of each block. For example, if the block size is 7 bytes and there are 12 blocks to write, the software side must write twice for each block. For each block, the ending byte is abandoned by uSDHC because it only sends 7 bytes to the card and picks data from the following system write, resulting in 24 beats of write access in total.

81.3.2.4 Dividing large data transfer

This SDIO command CMD53 definition limits the maximum data size of data transfers according to the following formula:

Max data size = Block size x Block count

The length of a multiple block transfer needs to be in multiples of block size. If the total data length cannot be divided evenly into a multiple of the block size, then there are two options to transfer the data. These two options depend on the function and the card design. Option 1 is for the host driver to split the transaction. The remainder of the block size data is then transferred by using a single block command at the end. Option 2 is to add dummy data in the last block to fill the block size. For option 2, the card must manage the removal of the dummy data. Only for write, uSDHC sends data to the card.

See Figure 540 for an example showing the division of large data transfers, assuming a kind of WLAN SDIO that only supports a block size of up to 64 bytes. Although uSDHC supports a block size of up to 4096 bytes, SDIO can only accept a block size of less than 64 bytes, so the data must be divided (see the example below).

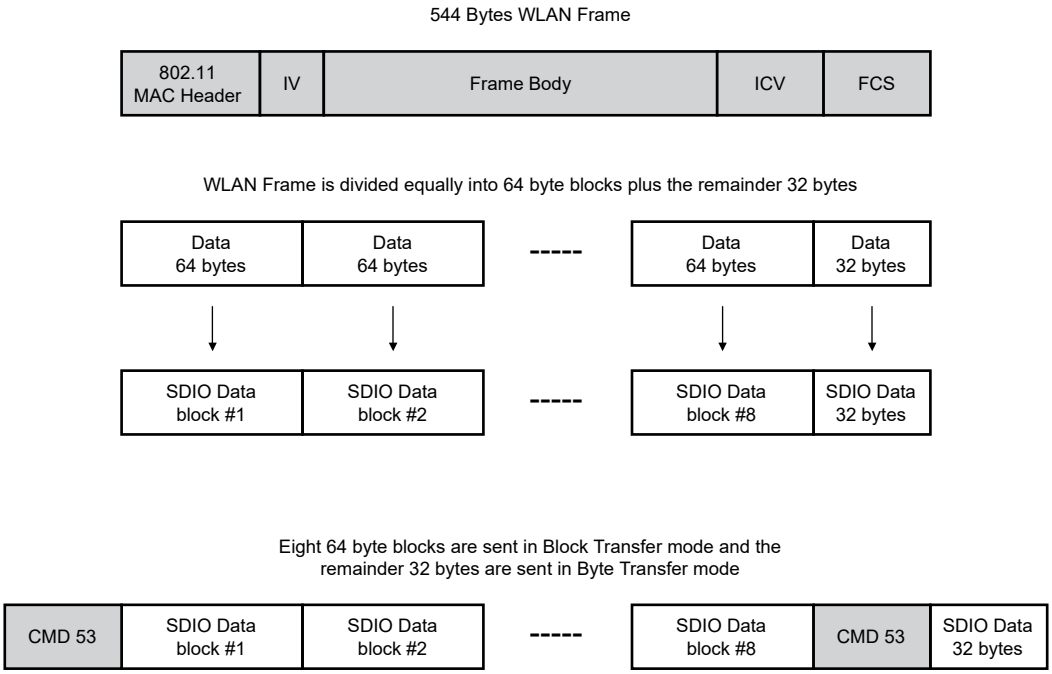


Figure 540. Example for dividing large data transfers

81.3.2.5 External DMA request

When the internal DMA is not in use and external DMA is enabled, the Data Buffer will generate a DMA request to the system. During a write operation, when the number of WR_WML words can be held in the buffer free space, the signal uSDHC_dreq_b is asserted to 0, informing the Host System of a DMA write.

The BWR bit in the Interrupt Status register is also set, as long as the BWRSEN bit in the Interrupt Status Enable register is set. The DMA request is de-asserted after several accesses to the Data Buffer Access Port (DATA_BUFF_ACC_PORT) are made while the buffer's free space can't meet the watermark condition (free space> write watermark level).

On read operation, when the number of RD_WML words are already in the buffer, the signal uSDHC_dreq_b is asserted to 0, informing the Host System for a DMA read. The BRR bit in the Interrupt Status register is also set, as long as the BRRSEN bit in

the Interrupt Status Enable register is set. The DMA request is de-asserted after several accesses to the Data Port register are made while the buffer's data can't meet the watermark condition (the number of data in buffer > read watermark level).

If the DMA burst length can't change during a data transfer for an external DMA transfer, the watermark level (read or write) must be a divisor of the block size. If it is not, transferring the block may cause buffer under-run (read operation) or over-run (write operation). For example, if the block size is 512 bytes, the watermark level of read (or write) must be a power of two between 1 and 128. For processor core polling access there is no such issue, as the last access in the block transfer can be controlled by software. The watermark level can be any value, even larger than the block size (but no greater than 128 words) because the actual number of bytes transferred by the software can be controlled and does not exceed the block size in each transfer.

The uSDHC also supports non-word aligned block size, as long as the card supports that block size. In this case, the watermark level should be set as the number of words. For example, if the block size is 31 bytes, the watermark level can be set to any number of words. For this case, the BLK_ATT[BLKSIZE] will be set as 1fh. For the CPU polling access, the burst length can be 1 to 128 words, without restriction. This is because the software will transfer 8 words, and the uSDHC will also set the BWR or BRR bits when the remaining data does not violate data buffer. See DMA burst length for more details about the dynamic watermark level of the data buffer.

For the above example, even though 8 words are transferred via Data Buffer Access Port (DATA_BUFF_ACC_PORT), the uSDHC will transfer only 31 bytes over the SD Bus, as required by the BLK_ATT[BLKSIZE] fields. In this data transfer, with non-word aligned block size, the endian mode should be set cautiously or invalid data will be transferred to and from the card.

81.3.3 DMA AHB interface

The internal DMA implements a DMA engine and the AHB master. When the internal DMA is enabled, but the BWR and BRR bits are set if the BWRSEN and BRRSEN bits have been set in the Interrupt Status Enable register. See Figure 541 for an illustration of the DMA AHB interface block.

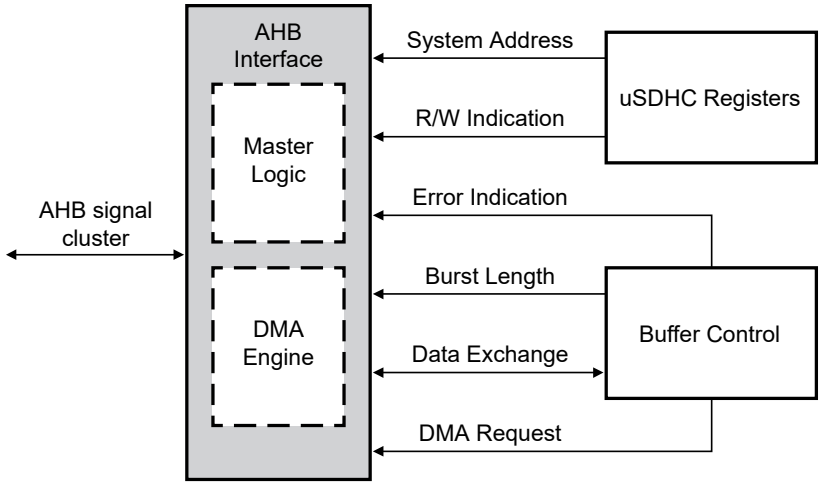


Figure 541. DMA AHB interface block

81.3.3.1 Internal DMA request

If the watermark level requirement is met in data transfer or if the last data of current block is ready in the data buffer, and the Internal DMA is enabled, the data buffer block sends a DMA request to AHB interface. Meanwhile, the external DMA request signal (uSDHC_dreq_b) is disabled.

The delay in response from the internal DMA engine depends on the system AHB bus loading and the priority assigned to uSDHC. The DMA engine does not respond to the request during its burst transfer, but is ready to serve as soon as the burst is over. The data buffer de-asserts the request if the data buffer space (for write) or bytes in data buffer is smaller than the watermark level. Upon access to the buffer by internal DMA, the data buffer updates its internal buffer pointer, and when the watermark level is satisfied or the last data of the current block is ready in the data buffer, another DMA request is sent.

The data transfer is in the block unit, and the subsequent watermark level is always automatically set as the remaining number of words. For instance, for a multi block data read with each block size of 31 bytes, and the burst length set to six words (24 bytes). After the first burst transfer, if there are greater than or equal to two words in the buffer, which might contain some data of the next block), another DMA request is sent. This is because the remaining number of words to transfer for the current block is ceiling((31 - 6 * 4) / 4) = 2. The uSDHC module reads two words out of the buffer, with seven valid bytes and one stuffed byte.

81.3.3.2 DMA burst length

Just like a CPU polling access, the DMA burst length for the internal DMA engine can range between 1 to 16 words. The actual burst length for the DMA depends on the lesser of the configured burst length or the remaining words of the current block. See the example in Internal DMA request. After six words are read, the burst length is two words, then the next burst length is six words. This is because the next block starts, which is 31 bytes, more than six words. The host driver may take this variable burst length into account. It is also acceptable to configure the burst length as the divisor of the block size, so that each time the burst length is the same.

81.3.3.3 AHB master interface

It is possible that the internal AHB DMA engine could fail during the data transfer. Upon detection of an AHB bus error during DMA transfer, the DMA engine stops the transfer and goes to the idle state. At that point, the internal data buffer stops receiving incoming data and sending out data. The DMAE bit in the Interrupt Status register is generated to host CPU to report a bus error condition.

After the DMAE interrupt is received, the software sends a CMD12 to abort the current transfer and read the DS_ADDR bits of the DMA System Address register to get the starting address of the corrupted block. After the DMA error is fixed, the software should apply a data reset and restart the transfer from this address to recover the corrupted block.

81.3.3.4 ADMA engine

In the SD host controller standard, a new DMA transfer algorithm called the Advanced DMA (ADMA) is defined. For simple DMA, after the page boundary is reached, a DMA interrupt is generated and the new system address is programmed by the host driver. The ADMA defines the programmable descriptor table in the system memory. The host driver can calculate the system address at the page boundary and program the descriptor table before executing ADMA. It reduces the frequency of interrupts to the host system. Therefore, higher speed DMA transfers could be realized because the host MCU intervention is not be needed during long DMA-based data transfers.

There are two types of ADMA in host controller: ADMA1 and ADMA2. ADMA1 can support data transfer of 4KB aligned data in system memory, and ADMA2 eliminates the restriction so that the data of any location and any size can be transferred in system memory. Their formats of Descriptor Table are different.

ADMA can recognize all kinds of descriptors defined in SD host controller Standard, and if the "End" flag is detected in the descriptor, ADMA stops after this descriptor is processed.

81.3.3.4.1 ADMA concept and descriptor format

ADMA1 includes the following descriptors:

- Valid/invalid descriptor
- Nop descriptor
- Set data length descriptor
- Set data address descriptor
- Link descriptor
- Interrupt flag and end flag in descriptor

ADMA2 includes the following descriptors:

- Valid/invalid descriptor
- Nop descriptor

- Rsv descriptor (new in ADMA2)
- Set data length and address descriptor
- Link descriptor
- Interrupt flag and end flag in descriptor

ADMA starts read/write operation after it reaches the tran state, using the data length and data address analyzed from most recent descriptor(s).

For ADMA1, the valid data length descriptor is the last set type descriptor before the tran type descriptor. Every tran type descriptor triggers a transfer, and the transfer data length is extracted from the most recent set type descriptor. If there is no set type descriptor after the previous Trans descriptor, the data length is the value for previous transfer, or 0 if no set descriptor is ever met.

For ADMA2, the tran type descriptor contains both data length and transfer data address, so the tran type descriptor itself can start a data transfer.

See Figure 542 for the format of the descriptor table for ADMA1.

Address/ Page Field		Address/ Page Field		Attribute Field					
31	12	11	6	5	4	3	2	1	0
Address or Data Length		000000		Act 2	Act 1	0	Int	End	Valid

Act 2	Act1	Symbol	Comment	31- 28	27- 12
0	0	Nop	No Operation	Don't Care	
0	1	Set	Set Data Length	0000	Data Length
1	0	Tran	Transfer Data	Data Address	
1	1	Link	Link Descriptor	Descriptor Address	

Valid	Valid = 1 indicates this line of descriptor is effective. If Valid = 0 generate ADMA Error Interrupt and stop ADMA.
End	End = 1 indicates current descriptor is the ending one.
Int	Int = 1 generates DMA Interrupt when this descriptor is processed.

Figure 542. Format of the ADMA1 descriptor table

See Figure 543 for the concept and access method of the descriptor table for ADMA1.

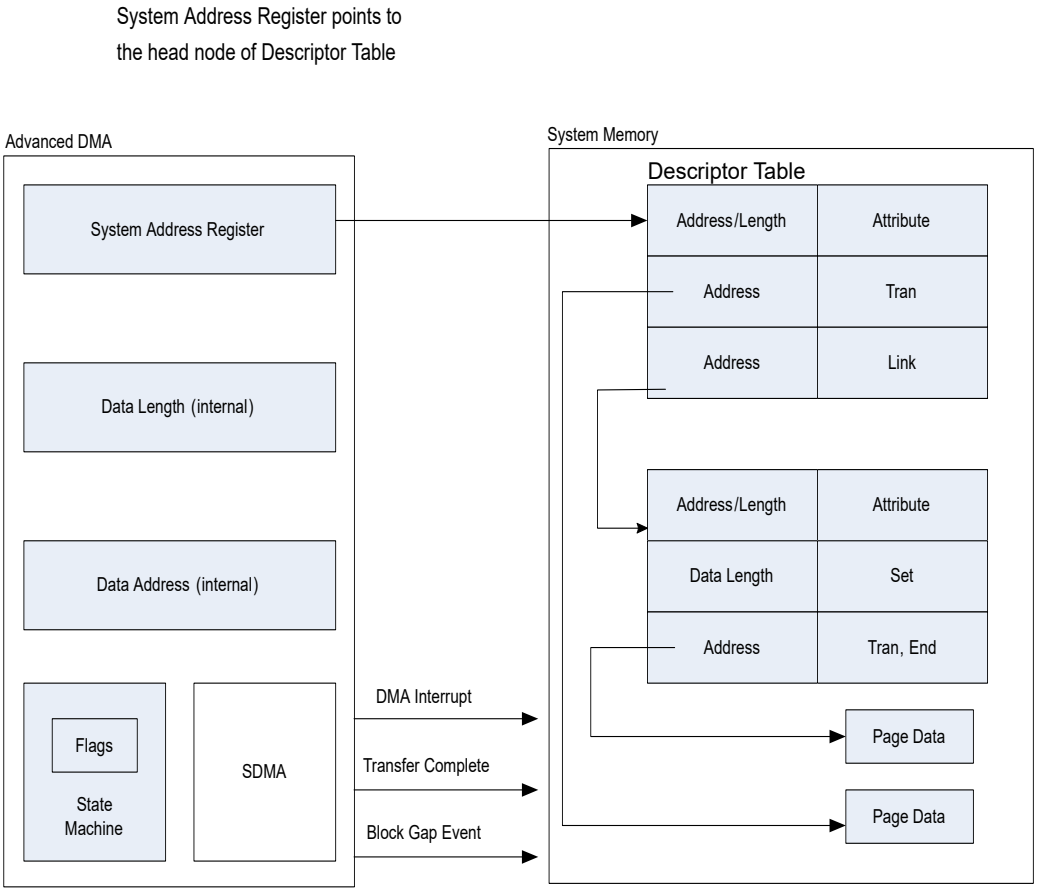


Figure 543. Concept and access method of the ADMA1 descriptor table

The figure below explains the ADMA2 format. ADMA2 deals with the lower 32-bit first, and then the higher 32-bit. If the 'Valid' flag of descriptor is 0, it ignores the higher 32-bit. The Address field should be set to word aligned (lower 2-bit is always set to 0). Data length is in byte unit.

Address Field		Length		Reserved		Attribute Field					
63	32	31	16	15	06	05	04	03	02	01	00
32-bit Address		16-bit length		0000000000		Act 2	Act 1	0	Int	End	Valid

Act 2	Act1	Symbol	Comment	Operation
0	0	Nop	No Operation	Don't Care
0	1	Rsv	Reserved	Same as Nop. Read this line and go to next one
1	0	Tran	Transfer Data	Transfer data with address and length set in this descriptor line
1	1	Link	Link Descriptor	Link to another descriptor

Valid	Valid = 1 indicates this line of descriptor is effective. If Valid = 0 generate ADMA Error Interrupt and stop ADMA.
End	End = 1 indicates current descriptor is the ending one.
Int	Int = 1 generates DMA Interrupt when this descriptor is processed.

Figure 544. Format of the ADMA2 descriptor table

See Figure 545 for the concept and access method of the descriptor table for ADMA2.

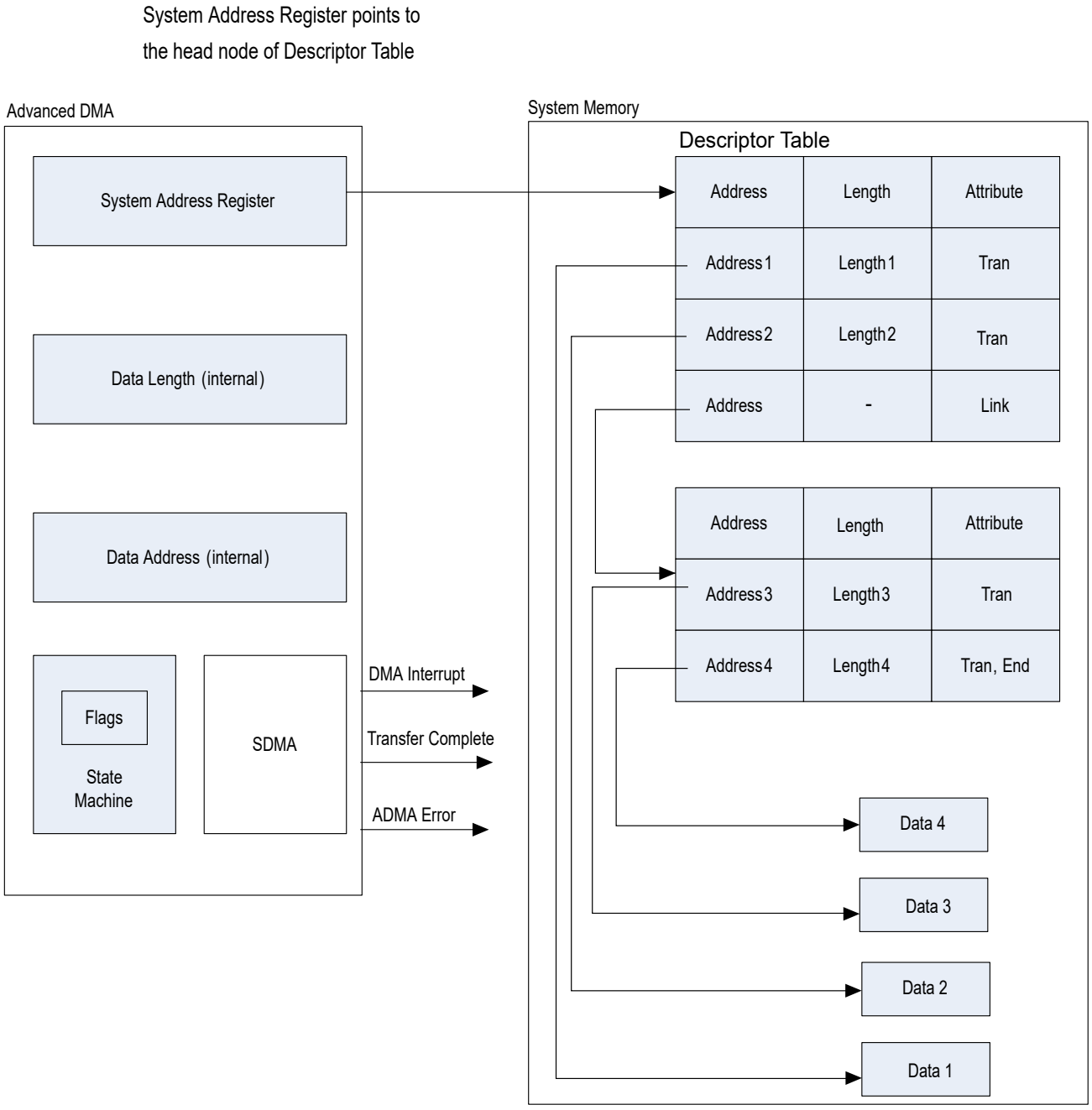


Figure 545. Concept and access method of the ADMA2 descriptor table

81.3.3.4.2 ADMA interrupt

If the interrupt flag descriptor is set, ADMA generates an interrupt according to various types of descriptors.

For ADMA1:

- Set type of descriptor: interrupt is generated when data length is set.
- Tran type descriptor: interrupt is generated when this transfer is complete.
- Link type of descriptor: interrupt is generated when new descriptor address is set.

- Nop type of descriptor: interrupt is generated just after this descriptor is fetched.

For ADMA2:

- Tran type of descriptor: interrupt is generated when this transfer is complete.
- Link type of descriptor: interrupt is generated when new descriptor address is set.
- Nop/Rsv type of descriptor: interrupt is generated just after this descriptor is fetched.

81.3.3.4.3 ADMA error

The ADMA stops whenever an error is encountered. These errors include:

- Fetching descriptor error
- AHB response error
- Data length mismatch error

An ADMA descriptor error is generated when it fails to detect a "valid" flag in the descriptor. If an ADMA descriptor error occurs, the interrupt is not generated even if the "Interrupt" flag of this descriptor is set.

When BLKCNTEN bit is set, data length set in register BLK_ATT must be equal to the whole data length set in descriptor, otherwise data length mismatch error is generated.

When BLKCNTEN bit is not set, then the whole data length set in descriptor is a multiple of block lengths; otherwise, when data set in the descriptor nodes are not performed at block boundaries, then data mismatch errors occur.

81.3.4 Register bank with IP bus interface

Register accesses through the IP bus interface are on the register bank. See Figure 546 for the block diagram.

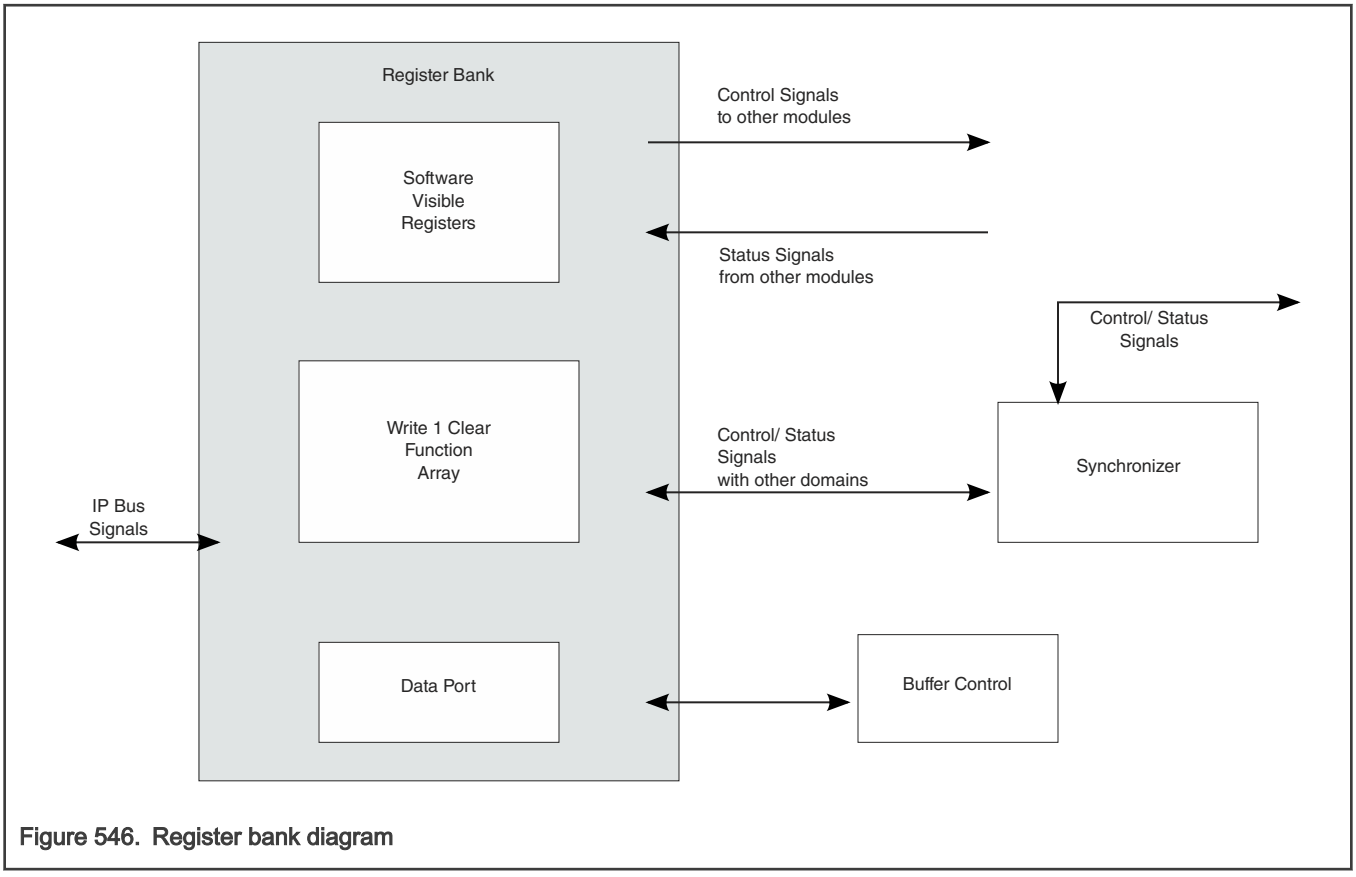


Figure 546. Register bank diagram

Only a 32-bit access is allowed, and no partial read/write is supported; therefore, all accesses are word aligned.

Access to an unimplemented address within the register memory map space does not generate a transfer error.

81.3.4.1 SD protocol unit

The SD protocol unit deals with all SD protocol affairs.

The SD protocol unit performs the following functions:

- Acts as the bridge between the internal buffer and the SD bus
- Sends the command data as well as its argument serially
- Stores the serial response bit stream into corresponding registers
- Detects the bus state on the CMD/DAT lines
- Monitors the interrupt from SDIO
- Asserts the read wait signal
- Gates off the SD clock when buffer is announcing danger status
- Detects the write protect state

The SD protocol unit consists of four sub-modules:

- SD control misc
- Command control
- Data control
- Clock control

81.3.4.2 SD control miscellaneous

In the SD control miscellaneous unit:

- The card detection (including DATA3 for card detection) and card interrupt are implemented.
- The module monitors the signal level on all the eight data lines and the command lines. It directly routes the level values into the register bank.
- The module also detects the Write Protect (WP) line. If WP is active, writes to memory and combo cards are ignored.

81.3.4.3 SD clock control

If the internal data buffer is near full (for read) or near empty (for write), the SD clock must be gated off to avoid buffer over/under-run. This module asserts the gate of the output SD clock to shut the clock off. After the buffer has space (for read) or has data (for write), the clock gate of this module opens, and the SD clock is active again.

81.3.4.4 Command control

The Command Control module deals with the transactions on the CMD line.

See Figure 547 for an illustration of the structure for the Command CRC shift register.

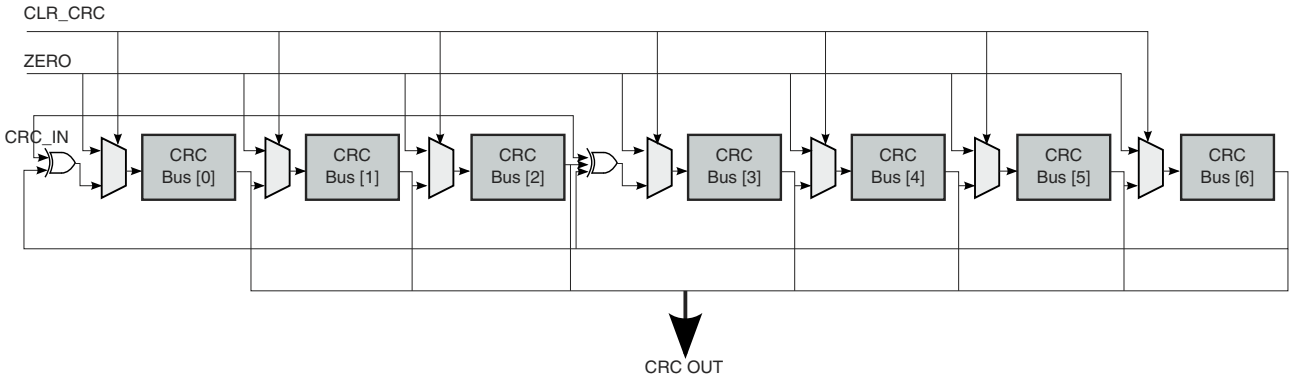


Figure 547. Command CRC shift register

The CRC polynomials for the CMD are as follows:

```
Generator polynomial: G(x) = x^7 + x^3 + 1
M(x) = (first bit) * x^n + (second bit) * x^n-1 +...+ (last bit) * x^0
CRC[6:0] = Remainder [(M(x) * x^7) / G(x)]
```

81.3.4.5 Data control

The data agent deals with the transactions on the eight data lines. Moreover, this module also detects the busy state on the DATA0 line and generates the read wait state by the request from the transceiver.

The CRC polynomials for the data are as follows:

```
Generator polynomial: G(x) = x^16 + x^12 + x^5 +1

M(x) = (first bit) * x^n + (second bit) * x^n-1 +...+ (last bit) * x^0

CRC[15:0] = Remainder [(M(x) * x^16) / G(x)]
```

81.3.5 Card insertion and removal detection

The uSDHC module uses the DATA3 pin to detect card insertion or removal. It is controlled by SoC pad or other logic. When there is no card on the eMMC/SD bus, the DATA3 is pulled to a low voltage level by default.

When any card is inserted to or removed from the socket, uSDHC detects the logic value changes on the DATA3 pin and generates an interrupt.

81.3.6 Power management and wakeup events

When there is no operation between uSDHC and the card through the SD bus, the user can completely disable the peripheral clock and base clock in the chip-level clock control module to save power. When the user needs to use uSDHC to communicate with the card, it can enable the clock and start the operation.

In some circumstances, when the clocks to uSDHC are disabled, for instance, when the system is in low-power mode, there are some events for which the user needs to enable the clock and handle the event. These events are called wakeup interrupts. The uSDHC module can generate these interrupts even when there are no clocks enabled. The three interrupts that can be used as wakeup events are these:

- Card removal interrupt
- Card insertion interrupt

- Interrupt from SDIO (not available in multiple block data transfers)

These three wakeup events (or wakeup interrupts) can also be used to wakeup the system from low-power modes.

NOTE

To make the interrupt a wakeup event, when all the clocks to uSDHC are disabled or when the entire system is in low power mode, the corresponding wakeup enabled bit needs to be set. Refer to Protocol Control (PROT_CTRL) for more information on the uSDHC Protocol Control register.

81.3.6.1 Setting wakeup events

For uSDHC to respond to a wakeup event, the software must set the respective wakeup enable bit before the CPU enters the sleep mode. Before the software disables the host clock, it should ensure that all the following conditions have been met:

- No read or write transfer is active
- Data and command lines are not active
- No interrupts are pending
- Internal data buffer is empty

81.3.7 eMMC fast boot

The Embedded MultiMediaCard (eMMC4.3 or later) specification adds a fast boot feature that requires hardware support. There are two types of fast boot modes in the eMMC4.3 or later specification: boot operation and alternative boot operation. Each type also has with-acknowledge and without-acknowledge modes.

In the boot operation mode, the master (eMultiMediaCard host) can read boot data from the slave (eMMC device) by keeping CMD line low after power-on, or sending CMD0 with argument + 0xFFFFFFFA (optional for slave), before issuing CMD1.

NOTE

For the eMMC4.3 setting, please see the eMMC4.3 specification.

81.3.7.1 Boot operation

If the CMD line is held low for 74 clock cycles and more after power-up before the first command is issued, the slave recognizes that boot mode is being initiated and starts preparing boot data internally.

Within one second after the CMD line goes low, the slave starts to send the first boot data to the master on the DATA line(s). The master must keep the CMD line low to read all of the boot data.

NOTE

For the purposes of this documentation, fast boot is called "normal fast boot mode".

If boot acknowledge is enabled, the slave has to send acknowledge pattern '010' to the master within 50ms after the CMD line goes low. If boot acknowledge is disabled, the slave does not send out acknowledge pattern '010'.

The master can terminate the boot mode with the CMD line high.

The boot operation is terminated when all the contents of the enabled boot data are sent to the master. After the boot operation is executed, the slave gets ready for the CMD1 operation and the master needs to start a normal eMMC initialization sequence by sending CMD1.

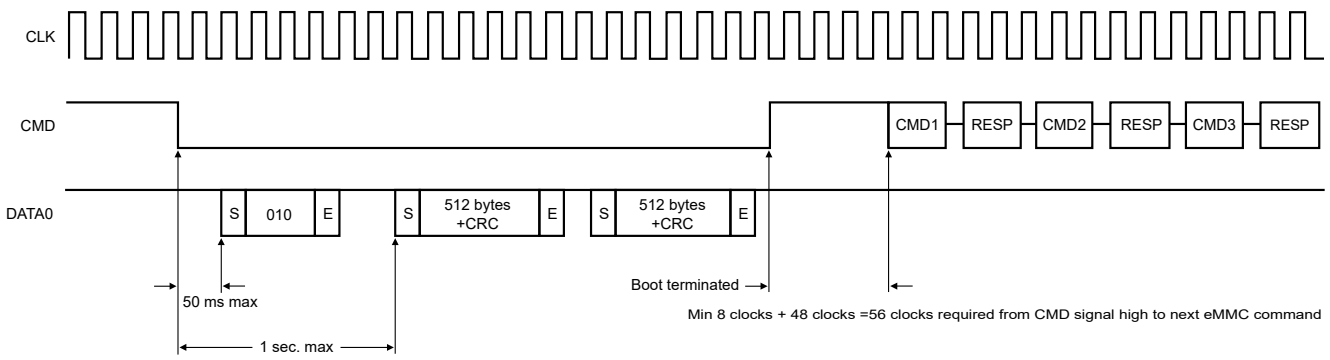


Figure 548. Embedded MultiMediaCard state diagram (normal boot mode)

81.3.7.2 Alternative boot operation

This boot function is optional for the device. If bit 0 in the extended CSD byte[228] is set to '1', the device supports the alternative boot operation.

After power-up, if the host issues CMD0 with the argument of 0xFFFFFFFA after 74 clock cycles, before CMD1 is issued or the CMD line goes low, the slave recognizes that boot mode is being initiated and starts preparing boot data internally.

Within one second after CMD0 with the argument of 0xFFFFFFFA is issued, the slave starts to send the first boot data to the master on the DATA line(s).

If boot acknowledge is enabled, the slave must send the acknowledge pattern '010' to the master within 50ms after the CMD0 with the argument of 0xFFFFFFFA is received. If boot acknowledge is disabled, the slave does not send out acknowledge pattern '010'.

The master can terminate the boot mode by issuing CMD0 (Reset).

Boot operation is terminated when all the contents of the enabled boot data are sent to the master. After the boot operation is executed, the slave gets ready for the CMD1 operation and the master needs to start a normal eMMC initialization sequence by sending CMD1.

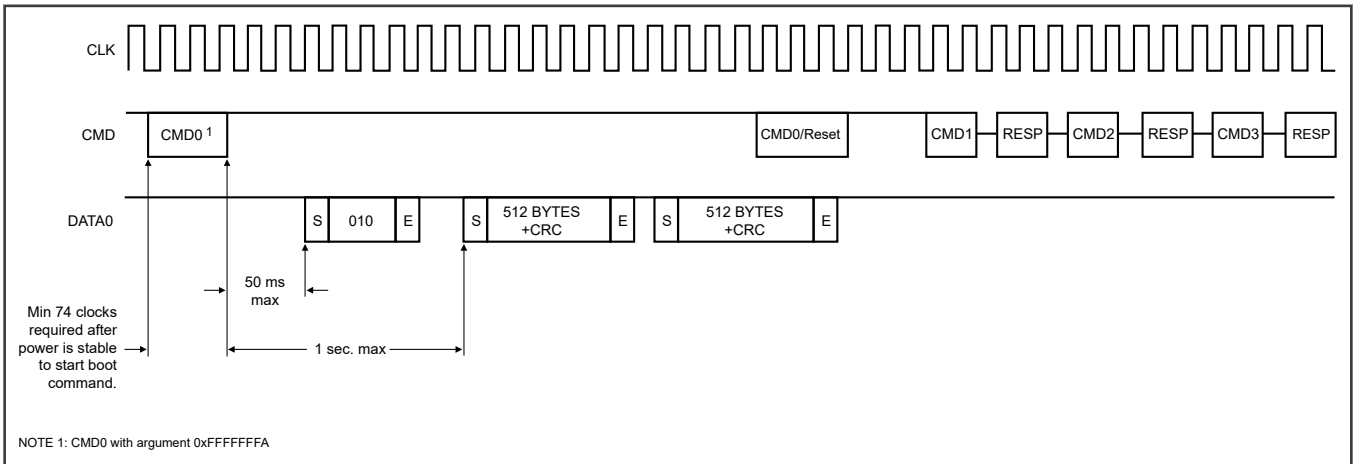


Figure 549. Embedded MultiMediaCard state diagram (alternative boot mode)

81.3.8 Commands for SD card, SDIO, and eMMC

A table containing the list of commands for the eMMC/SD card/SDIO is provided here.

Refer to the corresponding specifications for more details about the command information.

There are four kinds of commands defined to control the SD card, SDIO, and eMMC:

- Broadcast commands (bc), no response
- Broadcast commands with response (bcr), response from all cards simultaneously
- Addressed (point-to-point) commands (ac), no data transfer on the DATA
- Addressed (point-to-point) data transfer commands (adtc)

Response: A response is a token that is sent from the card to the host as an answer to a previously received command. A response is transferred serially on the CMD line.

Table 811. Commands for eMMC/SD card/SDIO

CMD INDEX	Type	Argument	Response type	Abbreviation	Description
CMD0	bc	[31:0] stuff bits	-	GO_IDLE_STATE	Resets all eMMC and SD memory cards to idle state.
CMD1	bcr	[31:0] OCR without busy	R3	SEND_OP_COND	Asks all eMMC and SD Memory cards in idle state to send their operation conditions register contents in the response on the CMD line.
CMD2	bcr	[31:0] stuff bits	R2	ALL_SEND_CID	Asks all cards to send their CID numbers on the CMD line.
CMD3[1]	ac	[31:6] RCA [15:0] stuff bits	R1 R6 (SDIO)	SET/ SEND_RELATIVE_ADDR	Assigns relative address to the card.
CMD4	bc	[31:0] DSR [15:0] stuff bits	-	SET_DSR	Programs the DSR of all cards.
CMD5	bc	[31:0] OCR without busy	R4	IO_SEND_OP_COND	Asks all SDIO in idle state to send them operation conditions register contents in the response on the CMD line.
CMD6[2]	adtc	[31] Mode 0: Check function 1: Switch function [30:8] Reserved for function groups 6 ~ 3 (All 0 or 0xFFFF) [7:4] Function group1 for command system [3:0] Function group2 for access mode	R1	SWITCH_FUNC	Checks switch ability (mode 0) and switch card function (mode 1). Refer to "SD Physical Specification V1.1" for more details.
CMD6[3]	ac	[31:26] Set to 0 [25:24] Access	R1b	SWITCH	Switches the mode of operation of the selected card or modifies the EXT_CSD registers. Refer to "The

Table continues on the next page...

Table 811. Commands for eMMC/SD card/SDIO (continued)

CMD INDEX	Type	Argument	Response type	Abbreviation	Description
		[23:16] Index [15:8] Value [7:3] Set to 0 [2:0] Cmd Set			Embedded MultiMediaCard System Specification Version 4.0 Final draft 2" for more details.
CMD7	ac	[31:6] RCA [15:0] stuff bits	R1b	SELECT/ DESELECT_CARD	Toggles a card between the stand-by and transfer states or between the programming and disconnect states. In both cases, the card is selected by its own relative address and gets deselected by any other address. Address 0 deselects all.
CMD8	adtc	[31:0] stuff bits	R1	SEND_EXT_CSD	The card sends its EXT_CSD register as a block of data, with a block size of 512 bytes.
CMD9	ac	[31:6] RCA [15:0] stuff bits	R2	SEND_CSD	Addressed card sends its card-specific data (CSD) on the CMD line.
CMD10	ac	[31:6] RCA [15:0] stuff bits	R2	SEND_CID	Addressed card sends its card-identification (CID) on the CMD line.
CMD11	adtc	[31:0] data address	R1	READ_DAT_UNTIL_STOP	Reads data stream from the card, starting at the given address, until a STOP_TRANSMISSION follows.
CMD12	ac	[31:0] stuff bits	R1b	STOP_TRANSMISSION	Forces the card to stop transmission.
CMD13	ac	[31:6] RCA [15:0] stuff bits	R1	SEND_STATUS	Addressed card sends its status register.
CMD14	Reserved				
CMD15	ac	[31:6] RCA [15:0] stuff bits	-	GO_INACTIVE_STATE	Sets the card to inactive state in order to protect the card stack against communication breakdowns.
CMD16	ac	[31:0] block length	R1	SET_BLOCKLEN	Sets the block length (in bytes) for all following block commands (read and write). Default block length is specified in the CSD.

Table continues on the next page...

Table 811. Commands for eMMC/SD card/SDIO (continued)

CMD INDEX	Type	Argument	Response type	Abbreviation	Description
CMD17	adtc	[31:0] data address	R1	READ_SINGLE_BLOCK	Reads a block of the size selected by the SET_BLOCKLEN command.
CMD18	adtc	[31:0] data address	R1	READ_MULTIPLE_BLOCK	Continuously transfers data blocks from card to host until interrupted by a stop command.
CMD19	Reserved				
CMD20	adtc	[31:0] data address	R1	WRITE_DAT_UNTIL_STOP	Writes data stream from the host, starting at the given address, until a STOP_TRANSMISION follows.
CMD21	Reserved				
CMD22	Reserved				
CMD23	ac	[31] reliable write request [30:16] set to 0 [15:0] number of blocks	R1	SET_BLOCK_COUNT	Defines the number of blocks (read/write) and the reliable writer parameter (write) for a block read or write command.
CMD24	adtc	[31:0] data address	R1	WRITE_BLOCK	Writes a block of the size selected by the SET_BLOCKLEN command.
CMD25	adtc	[31:0] data address	R1	WRITE_MULTIPLE_BLOCK	Continuously writes blocks of data until a STOP_TRANSMISSION follows.
CMD26	adtc	[31:0] stuff bits	R1	PROGRAM_CID	Programming of the card identification register. This command is issued only once per card. The card contains hardware to prevent this operation after the first programming. Normally this command is reserved for the manufacturer.
CMD27	adtc	[31:0] stuff bits	R1	PROGRAM_CSD	Programming of the programmable bits of the CSD.
CMD28	ac	[31:0] data address	R1b	SET_WRITE_PROT	If the card has write protection features, this command sets the write protection bit of the addressed group. The properties of write protection are coded in the card specific data (WP_GRP_SIZE).

Table continues on the next page...

Table 811. Commands for eMMC/SD card/SDIO (continued)

CMD INDEX	Type	Argument	Response type	Abbreviation	Description
CMD29	ac	[31:0] data address	R1b	CLR_WRITE_PROT	If the card provides write protection features, this command clears the write protection bit of the addressed group.
CMD30	adtc	[31:0] write protect data address	R1	SEND_WRITE_PROT	If the card provides write protection features, this command asks the card to send the status of the write protection bits.
CMD31	Reserved				
CMD32	ac	[31:0] data address	R1	TAG_SECTOR_START	Sets the address of the first sector of the erase group.
CMD33	ac	[31:0] data address	R1	TAG_SECTOR_END	Sets the address of the last sector in a continuous range within the selection of a single sector to be selected for erase.
CMD34	ac	[31:0] data address	R1	UNTAG_SECTOR	Removes one previously selected sector from the erase selection.
CMD35	ac	[31:0] data address	R1	TAG_ERASE_GROUP_START	Sets the address of the first erase group within a range to be selected for erase.
CMD36	ac	[31:0] data address	R1	TAG_ERASE_GROUP_END	Sets the address of the last erase group within a continuous range to be selected for erase.
CMD37	ac	[31:0] data address	R1	UNTAG_ERASE_GROUP	Removes one previously selected erase group from the erase selection.
CMD38	ac	[31:0] stuff bits	R1b	ERASE	Erase all previously selected sectors.
CMD39	ac	[31:0] RCA [15] register write flag [14:8] register address [7:0] register data	R4	FAST_IO	Used to write and read 8-bit (register) data fields. The command addresses a card, and a register, and provides the data for writing if the write flag is set. The R4 response contains data read from the address register. This command accesses application dependent registers which are not defined in the eMMC standard.

Table continues on the next page...

Table 811. Commands for eMMC/SD card/SDIO (continued)

CMD INDEX	Type	Argument	Response type	Abbreviation	Description
CMD40	bcr	[31:0] stuff bits	R5	GO_IRQ_STATE	Sets the system into interrupt mode.
CMD41	Reserved				
CDM42	adtc	[31:0] stuff bits	R1b	LOCK_UNLOCK	Used to set/reset the password or lock/unlock the card. The size of the data block is set by the SET_BLOCK_LEN command.
CMD43~51	Reserved				
CMD52	ac	[31:0] stuff bits	R5	IO_RW_DIRECT	Access a single register within the total 128k of register space in any I/O function.
CMD53	ac	[31:0] stuff bits	R5	IO_RW_EXTENDED	Accesses a multiple I/O register with a single command. Allows the reading or writing of a large number of I/O registers.
CMD54	Reserved				
CMD55	ac	[31:16] RCA [15:0] stuff bits	R1	APP_CMD	Indicates to the card that the next command is an application specific command rather than a standard command.
CMD56	adtc	[31:1] stuff bits [0]: RD/WR	R1b	GEN_CMD	Used either to transfer a data block to the card or to get a data block from the card for general purpose / application specific commands. The size of the data block is set by the SET_BLOCK_LEN command.
CMD57-59	Reserved				
CMD60	adtc	[31] WR [30:24] stuff bits [23:16] address [15:8] stuff bits [7:0] byte count	R1b	RW_MULTIPLE_REGISTER	These registers are used to control the behavior of the device and to retrieve status information regarding the operation of the device. All Status and Control registers are WORD (32-bit) in size and are WORD aligned. CMD60 is used to read and write these registers.
CMD61	adtc	[31] WR [30:16] stuff bits	R1b	RW_MULTIPLE_BLOCK	The host issues a RW_MULTIPLE_BLOCK (CMD61) to begin the data transfer.

Table continues on the next page...

Table 811. Commands for eMMC/SD card/SDIO (continued)

CMD INDEX	Type	Argument	Response type	Abbreviation	Description
		[15:0] data unit count			
CMD62-63	Reserved				
ACMD6[4]	ac	[31:2] stuff bits [1:0] bus width	R1	SET_BUS_WIDTH	Defines the data bus width ('00'=1bit or '10'=4bit bus) to be used for data transfer. The allowed data bus widths are given in SCR register.
ACMD13[4]	adtc	[31:0] stuff bits	R1	SD_STATUS	Send the SD Memory Card status.
ACMD22[4]	adtc	[31:0] stuff bits	R1	SEND_NUM_WR_SECTORS	Send the number of the written sectors (without errors). Responds with 32-bit plus the CRC data block.
ACMD23[4]	ac	[31:23] stuff bits [22:0] Number of blocks	R1	SET_WR_BLK_ERASE_COUNT	Set the number of write blocks to be pre-erased before writing (to be used for fast Multiple Block WR command). "1"=default(one write block).
ACMD41[4]	bcr	[31:0] OCR	R3	SD_APP_OP_COND	Asks the accessed card to send its operating condition register (OCR) contents in the response on the CMD line.
ACMD42[4]	ac	[31:1] stuff bits [0] set_cd	R1	SET_CLR_CARD_DETECT	Connect(1)/Disconnect(0) the 50KOhm pull-up resistor on DATA3 of the card.
ACMD51[4]	adtc	[31:0] stuff bits	R1	SEND_SCR	Reads the SD Configuration Register (SCR).

1. CMD3 differs for eMMC and SD cards. For eMMC, it is referred to as SET_RELATIVE_ADDR, with a response type of R1. For SD cards, it is referred to as SEND_RELATIVE_ADDR, with a response type of R6 (with RCA inside).
2. CMD6 differs completely between high speed eMMC and high-speed SD cards. Command SWITCH_FUNC is for high-speed SD cards.
3. Command SWITCH is for high speed eMMC. The Index field can contain any value from 0-255, but only values 0-191 are valid. If the Index value is in the 192-255 range the card does not perform any modification and the SWITCH_ERROR status bit in the EXT_CSD register is set. The Access Bits are shown in Table 812.
4. ACMDs is preceded with the APP_CMD command. Commands listed are used for SD only, other SD commands not listed are not supported on this module.

The access bits for the EXT_CSD access modes are listed in the following table.

Table 812. EXT_CSD access modes

Bits	Access name	Operation

Table continues on the next page...

Table 812. EXT_CSD access modes (continued)

00	Command set	The command set is changed according to the Cmd Set field of the argument.
01	Set bits	The bits in the pointed byte are set, according to the bits set to 1 in the Value field.
10	Clear bits	The bits in the pointed byte are cleared, according to the bits set to 1 in the Value field.
11	Write byte	The Value field is written into the pointed byte.

81.3.9 SDIO interrupt

Information on interrupts in 1-bit mode, interrupts in 4-bit mode, and card interrupt handling are detailed in the sections below.

81.3.9.1 Interrupts in 1-bit mode

In this case, the DATA1 pin provides the interrupt function. An interrupt is asserted by pulling the DATA1 low from SDIO, until the interrupt service is finished to clear the interrupt.

81.3.9.2 Interrupt in 4-bit mode

As the interrupt and data line 1 share pin 8 in a 4-bit mode, an interrupt is only sent by the card and recognized by the host during a specific time. This is known as the interrupt period. The uSDHC module only provides samples the level on pin 8 during the interrupt period. At all other times, the host treats it as the data signal. The definition of the interrupt period is different for operations with single block and multiple block data transfers.

In the case of normal single data block transmissions, the interrupt period becomes active two clock cycles after the completion of a data packet. This interrupt period lasts until after the card receives the end bit of the next command that has a data block transfer associated with it.

For multiple block data transfers in a 4-bit mode, there is only a limited period of time that the interrupt period can be active because of the limited period of data line availability between the multiple blocks of data. This requires stricter definition of the interrupt period. For this case, the interrupt period is limited to two clock cycles. This begins two clocks after the end bit of the previous data block. During this 2-clock cycle interrupt period, if an interrupt is pending, the DATA1 line holds low for one clock cycle, then pulls high for the next clock cycle . On completion of the interrupt period, the card releases the DATA1 line into the high Z state. The uSDHC module provides sample of the DATA1 during the interrupt period when the IABG bit in the Protocol Control register is set.

Refer to SDIO Specification v1.10 for further information about the SDIO interrupt.

81.3.9.3 Card interrupt handling

When the CINTIEN bit in the Interrupt Signal Enable Register is set to 0, uSDHC clears the interrupt request to the host system. The host driver should clear this bit before servicing the SDIO interrupt, and should set this bit again after all interrupt requests from the card are cleared to prevent inadvertent interrupts.

The SDIO Interrupt Status can be cleared by writing 1 to this bit. But as the interrupt source from SDIO does not clear, this bit is set again. To clear this bit, it is required to reset the interrupt source from the external card followed by writing 1 to this bit. In a 1-bit mode, uSDHC detects the SDIO interrupt with or without the SD clock (to support wakeup). In a 4-bit mode, the interrupt signal is sampled during the interrupt period, so there are some sample delays between the interrupt signal from SDIO and the interrupt to the Host System Interrupt Controller. When the SDIO status is set and the host driver needs to service this interrupt, the SDIO bit in the Interrupt Control Register of SDIO is cleared. This is required to clear the SDIO interrupt status latched in uSDHC and to stop driving the interrupt signal to the System Interrupt Controller. The host driver must issue a CMD52 to clear the card interrupt. After completion of the card interrupt service, the SDIO Interrupt Status Enable bit is set to 1 and uSDHC starts sampling the interrupt signal again.

See Figure 550 for an illustration of the SDIO interrupt scheme and for the sequences of software and hardware events that take place during a card interrupt handling procedure.

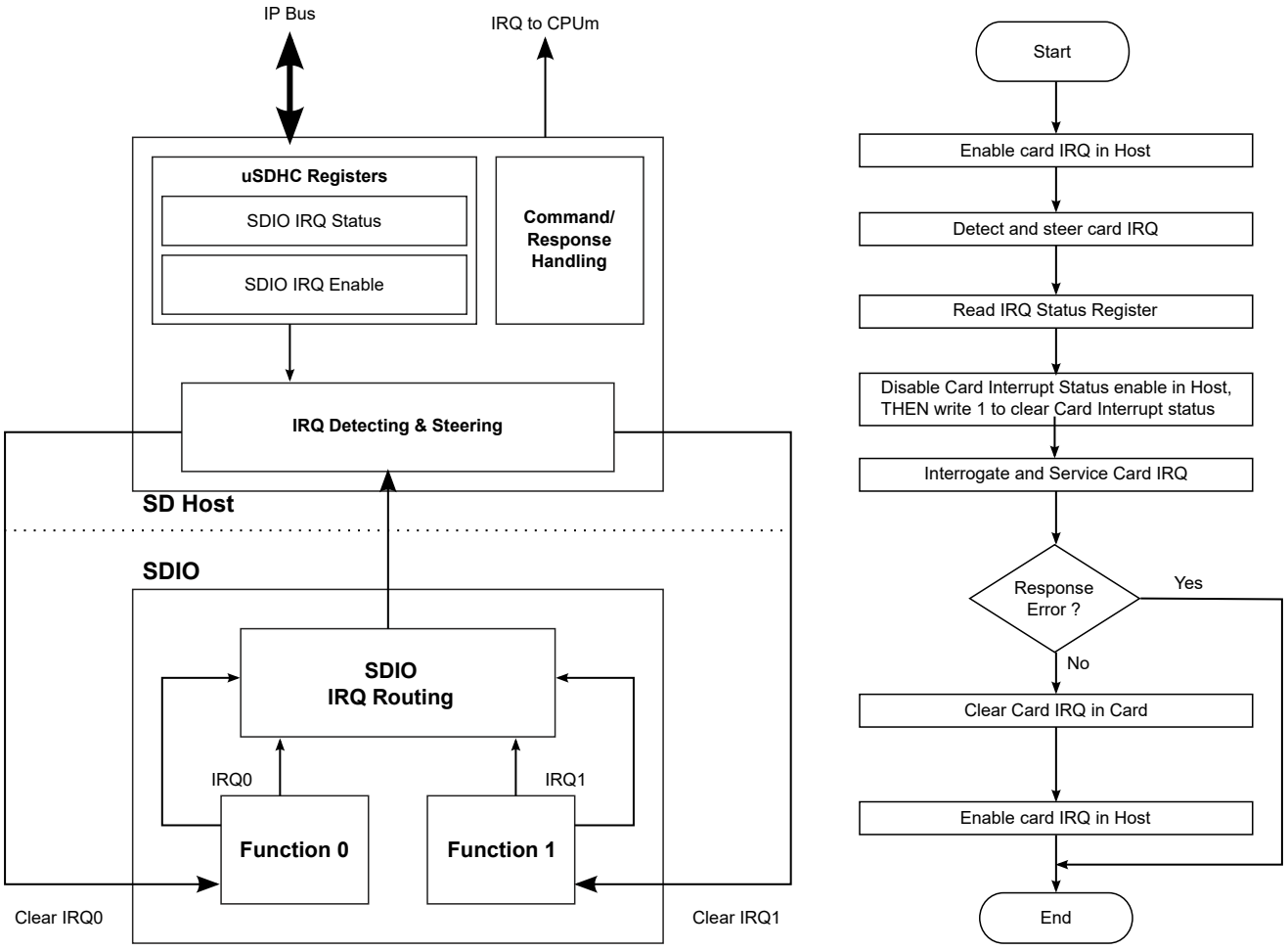


Figure 550. Card interrupt scheme, card interrupt detection, and handling procedure

81.3.10 Software restrictions

81.3.10.1 Initialization active

The driver cannot set INITA bit in System Control register when any of the command line or data lines are active, so the driver must ensure both CDIHB and CIHB bits are cleared.

81.3.10.2 Software polling procedure

For polling read or write, after the software begins a buffer read or write, it must access exactly the number of times as the values set in the Watermark Level Register; moreover, if the block size is not a multiple of the value in the Watermark Level Register (read and write respectively), the software must access exactly the remaining number of words at the end of each block. For example, for a read operation, if the RD_WML is 4, indicating the watermark level is 16 bytes, block size is 40 bytes, and the block number is 2, then the access times for the burst sequence in the whole transfer process must be 4, 4, 2, 4, 4, 2.

81.3.10.3 Suspend operation

To suspend the data transfer, the software must inform uSDHC that the suspend command is successfully accepted. To achieve this, after the Suspend command is accepted by SDIO, software must send another normal command marked as suspend command (CMDTYP bits set as '01') to inform uSDHC that the transfer is suspended.

If software needs to resume the suspended transfer, it should read the value in BLKCNT register to save the remaining number of blocks before sending the normal command marked as suspend, otherwise on sending such a 'suspend' command, uSDHC treats the current transfer is aborted and change the BLKCNT register to its original value, instead of retaining the remaining number of blocks.

81.3.10.4 Data length setting

For either ADMA (ADMA1 or ADMA2) transfer, the data in the data buffer must be word aligned, so the data length set in the descriptor must be a multiple of 4.

ADMA1 is 4KB aligned.

81.3.10.5 (A)DMA address setting

To configure the ADMA1/ADMA2/DMA address register, when TC bit is set, the register always updates itself with the internal address value to support dynamic address synchronization, so the software must ensure that the TC bit is cleared prior to configuring the ADMA1/ADMA2/DMA address register.

81.3.10.6 Data port access

Data port does not support parallel access. For example, during an internal DMA access, it is not allowed to write any data to the data port by CPU; or during a CPU read operation, it is also prohibited to write any data to the data port, by either CPU or internal DMA. Otherwise the data is corrupted inside the uSDHC buffer.

81.3.10.7 Change clock frequency

The uSDHC module does not automatically gate off the card clock when the host driver changes the clock frequency. To prevent possible glitch on the card clock, clear the FRC_SDCLK_ON bit when changing the clock divisor value (SDCLKFS or DVS in System Control Register) or setting the RSTA bit.

Also, before changing the clock divisor value, the host driver should make sure that the SDSTB bit is high.

81.3.10.8 Multi-block read

For pre-defined multi-block read operation, that is, the number of blocks to read has been defined by previous CMD23 for eMMC, or pre-defined number of blocks in CMD53 for SDIO/SDCombo, or whatever multi-block read without abort command at card side, an abort command, either automatic or manual CMD12/CMD52, is still required by uSDHC after the pre-defined number of blocks are done, to drive the internal state machine to idle mode. In this case, the card may not respond to this extra abort command and uSDHC gets response timeout. It is recommended to manually send an abort command with RSPTYP[1:0] both bits cleared.

81.3.11 Clocking

The table found here describes the clock sources for uSDHC. For more information on clocking, see the Clocking chapter.

Table 813. Clocks

Clock name	Description
hclk	Bus clock
ipg_clk	Peripheral clock
ipg_clk_perclk	Base clock
ipg_clk_lp	Low power clock

81.3.11.1 Clock and reset manager

This module controls all four kinds reset signals within uSDHC:

- Hardware reset
- Software reset for all logic
- Software reset for the data logic
- Software reset for the command logic

All these signals are fed into this module and stable signals are generated inside the module to reset all other modules. The module also gates off all the inside signals.

81.3.11.2 Clock generator

The clock generator generates the card CLK by peripheral source clock in two stages. The clock divisor can be configured through register SYS_CTRL[SDCLKFS] for prescaler configuration while the [DVS] for divisor configuration. Details can be found in the register function description. See the following figure for the structure of the divider. The term "Base" represents the frequency of peripheral source clock (see ipg_clk_perclk in Table 813).

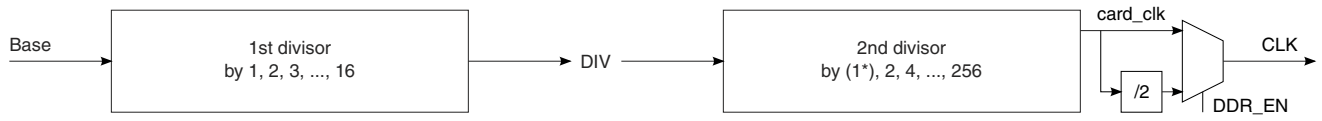


Figure 551. Two stages of the clock divider

The first stage outputs an intermediate clock (DIV) that can be Base, Base/2, Base/3, ..., or Base/16.

The second stage is a prescaler and outputs the actual internal working clock (card_clk). This clock is the driving clock for all the sub modules of the SD protocol unit, and helps in syncing FIFOs (see Figure 537) to synchronize with the data rate from the internal data buffer. The frequency of the clock output from this stage can be DIV, DIV/2, DIV/4,..., or DIV/256. Therefore, the highest frequency of the card_clk is base, and the next highest is Base/2, while the lowest frequency is Base/4096. If the duty cycle of the base clock is 50%, the duty cycle of card_clk is also 50%, even when the compound divisor is an odd value.

NOTE

CLK is different for the SDR and DDR modes.

- In the SDR mode, CLK is equal to the internal working clock (card_clk).

$$f_{card_clk} = \frac{f_{Base}}{SYS_CTRL[DVS] \times SYS_CTRL[SDCLKFS]}$$

$$f_{CLK} = f_{card_clk}$$

Equation 32. Equation for f_{card_clk} in SDR mode

$$f_{card_clk} = \frac{400MHz}{1x(2)} = 200MHz$$

$$f_{CLK} = f_{card_clk} = 200\ MHz$$

- In the DDR mode, CLK is equal to card_clk/2

$$f_{card_clk} = \frac{f_{Base}}{SYS_CTRL[DVS] \times (SYS_CTRL[SDCLKFS]/2)}$$

$$f_{CLK} = \frac{f_{\text{card_clk}}}{2}$$

Equation 33. Equation for $f_{\text{card_clk}}$ in DDR mode

$$f_{\text{card_clk}} = \frac{400MHz}{1x(2/2)} = 400MHz$$

$$f_{CLK} = \frac{f_{\text{card_clk}}}{2} = \frac{400MHz}{2} = 200MHz$$

81.4 External signals

The following table describes the uSDHC external signals:

Table 814. uSDHC external signals

Signal	Description	Direction
CLK	Is an Internally generated clock used to drive the eMMC, SD card, and SDIO.	O
CMD	Is used to send commands and receive responses to and from the card.	I/O
DAT7	DAT7 line in the 8-bit mode — Not used in other modes	I/O
DAT6	DAT6 line in the 8-bit mode — Not used in other modes	I/O
DAT5	DAT5 line in the 8-bit mode — Not used in other modes	I/O
DAT4	DAT4 line in the 8-bit mode — Not used in other modes	I/O
NOTE If uSDHC needs to support a 4-bit data transfer, DAT7~DAT4 can also be optional and tied to high.		
DAT3	DAT3 line in the 4/8-bit mode or configured as card detection pin. The bit may be configured as card detection pin in the 1-bit mode.	I/O
DAT2	DAT2 line or Read Wait in the 4-bit mode Read Wait in 1-bit mode	I/O
DAT1	DAT1 line in the 4/8-bit mode Also, used to detect interrupt in 1/4-bit mode	I/O
DAT0	DAT0 line in all the modes	I/O

Table continues on the next page...

Table 814. uSDHC external signals (continued)

Signal	Description	Direction
	Also, used to detect busy state	
WP	Write protection signals directly routed from the socket. Low value (0) indicates it is not write protected. In the case the pin is not used (for the embedded memory), tie low. Optional for system implementation.	I
RESET_B	Is used to reset the eMMC.	O

81.5 Application information

All communication between the system and cards are controlled by the host. The host sends commands of two types: broadcast and addressed (point-to-point).

Broadcast commands are intended for all cards, such as GO_IDLE_STATE, SEND_OP_COND, and ALL_SEND_CID. In the Broadcast mode, eMMC are in the open-drain mode to avoid bus contention. See Commands for SD card, SDIO, and eMMC for the commands of bc and bcr categories.

After the broadcast command CMD3 is issued, the cards enter the standby mode. Addressed type commands are used from this point. In this mode, for eMMC, the CMD/DATA I/O pads turns to the push-pull mode to have the driving capability for maximum frequency operation. See Commands for SD card, SDIO, and eMMC for the commands of ac and adtc categories.

81.5.1 Command send and response receive basic operation

Assuming that the data type WORD is an unsigned 32-bit integer, the flow indicated below presents a guideline for sending a command to the card(s):

```
send_command(cmd_index, cmd_arg, other requirements)

{

    WORD wCmd; // 32-bit integer to make up the data to write into Transfer Type register, it is
recommended to implement in a bit-field manner

    wCmd = (<cmd_index> & 0x3f) << 24; // set the first 8 bits as '00'+<cmd_index>

    set CMDTYP, DPSEL, CICEN, CCCEN, RSTTYP, DTDSEL according to the command index;

    if (internal DMA is used) wCmd |= 0x1;

    if (multi-block transfer) {

        set MSBSEL bit;

        if (finite block number) {

            set BCEN bit;

            if (auto12 command is to use) set AC12EN bit;

        }

    }
```

```
    write_reg(CMDARG, <cmd_arg>); // configure the command argument

    write_reg(XFERTYP, wCmd); // set Transfer Type register as wCmd value to issue the command

}

wait_for_response(cmd_index)

{

    while (CC bit in IRQ Status register is not set); // wait until Command Complete bit is set

    read IRQ Status register and check if any error bits about Command are set

    if (any error bits are set) report error;

    write 1 to clear CC bit and all Command Error bits;

}
```

For the sake of simplicity, the function wait_for_response is implemented here by means of polling. For an effective and formal way, the response is usually checked after the Command Complete Interrupt is received. When doing this, make sure that the corresponding interrupt status bits are enabled.

For some scenarios, the response time-out is expected. For instance, after all cards respond to CMD3 and move to the Standby state no response to the Host when CMD2 is sent. The host driver deals with "fake" errors like this with caution.

81.5.2 Card identification mode

When a card is inserted to the socket or the card is reset by the host, the host needs to validate the operation voltage range, identify the cards, request the cards to publish the Relative Card Address (RCA) or to set the RCA for eMMC.

81.5.2.1 Card detect

See Figure 552 for a flow diagram showing the detection of SD card and SDIO using uSDHC.

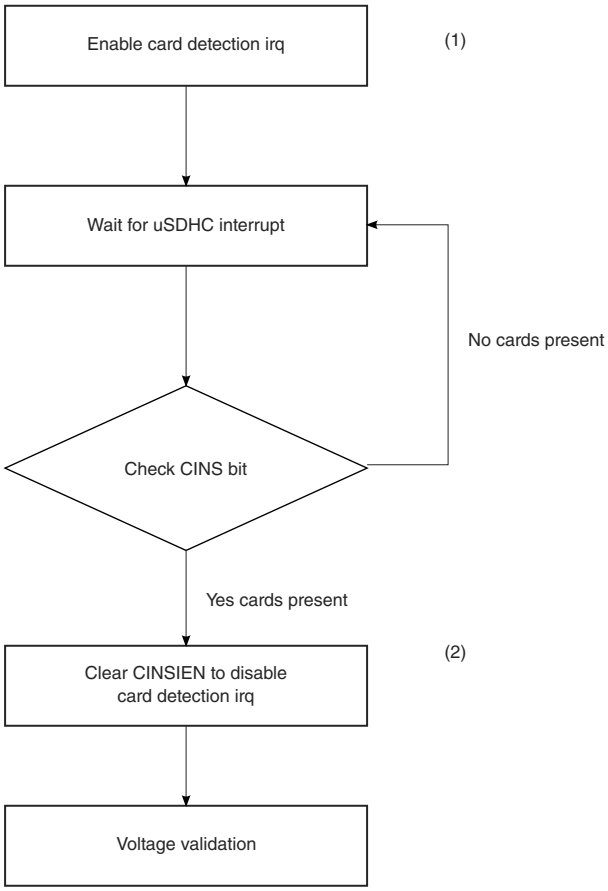


Figure 552. Flow diagram for card detection

Here is the card detect sequence:

- Set the CINSIEN bit to enable card detection interrupt.
- When an interrupt from uSDHC is received, check the CINS bit in the Interrupt Status register to see if it was caused by card insertion.
- Clear the CINSIEN bit to disable the card detection interrupt and ignore all card insertion interrupts afterwards.

81.5.2.2 Reset

The host consists of three types of resets:

- Hardware reset (Card and Host) that is driven by Power On Reset (POR).
- Software reset (Host only) is initiated by the write operation on the RSTD, RSTC, or RSTA bits of the System Control register to reset the data part, command part, or all parts of the host controller, respectively.
- Card reset (Card only): The command, "Go_Idle_State" (CMD0), is the software reset command for all types of eMMC and SD memory cards. This command sets each card into the idle state regardless of the current card state. For an SDIO, CMD52 is used to write an I/O reset in CCCR. The cards are initialized with a default relative card address (RCA=0x0000) and with a default driver stage register setting (lowest speed, highest driving current capability).

After the card is reset, the host needs to validate the voltage range of the card. See the figure below for the software flow to reset both uSDHC and the card.

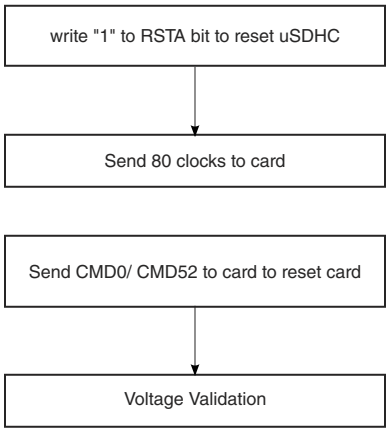


Figure 553. Flow chart for resetting uSDHC and SD card or SDIO

```
software_reset() {
    set_bit(SYSCTRL, RSTA); // software reset the Host set DTOCV and SDCLKFS fields to get the CLK of
    //frequency around 400kHz
    //configure IO pad to set the power voltage of external card to around 3.0V
    //poll bits CIHB and CDIHB
    //bits of PRSSTAT to wait both fields are cleared
    set_bit(SYSCTRL, INTIA); // send 80 clock ticks for card to power up
    send_command(CMD_GO_IDLE_STATE, <other parameter>); // reset the card with CMD0 or
    send_command(CMD_IO_RW_DIRECT, <other parameter>);
}
```

81.5.2.3 Voltage validation

All cards should be able to establish communication with the host using any operation voltage in the maximum allowed voltage range specified in the card specification. However, the supported minimum and maximum values for VDD are defined in the Operation Conditions Register (OCR) and may not cover the whole range. Cards that store the CID and CSD data in the preload memory are only able to communicate this information under data transfer VDD conditions. This means that if the host and card have non-common VDD ranges, the card is neither able to complete the identification cycle nor able to send CSD data.

Therefore, special commands Send_Op_Cont (CMD1 for eMMC), SD_Send_Op_Cont (ACMD41 for SD Memory), and IO_Send_Op_Cont (CMD5 for SD I/O) are used. The voltage validation procedure is designed to provide a mechanism to identify and reject cards that do not match the VDD range(s) desired by the host. This is accomplished when the host sends the desired VDD voltage window as the operand of this command. Cards that cannot perform the data transfer in the specified range must discard themselves from further bus operations and go into the Inactive state. By omitting the voltage range in the command, the host can query each card and determine the common voltage range before sending out-of-range cards into the Inactive state. This query should be used if the host is able to select a common voltage range or if a notification is sent to the system when a non-usable card in the stack is detected.

The following steps show how to perform voltage validation when a card is inserted:

```
voltage_validation(voltage_range_argument) {
    label the card as UNKNOWN;
    send_command(IO_SEND_OP_COND, 0x0, <other parameters are omitted>); // CMD5, check SDIO operation
voltage, command argument is zero
    if (RESP_TIMEOUT != wait_for_response(IO_SEND_OP_COND)) { // SDIO command is accepted
        if (0 < number of IO functions) {
            label the card as SDIO;
            IORDY = 0;
```

```
            while (!(IORDY in IO OCR response)) { // set voltage range for each IO function
                send_command(IO_SEND_OP_COND, <voltage range>, <other parameter>);
                wait_for_response(IO_SEND_OP_COND);
            } // end of while ...
        } // end of if (0 < ...
        if (memory part is present inside SDIO)
            Label the card as SDCombo; //this is an SD-Combo card
    } // end of if (RESP_TIMEOUT ...
    if (the card is labeled as SDIO)
        return; // card type is identified and voltage range is set, so exit the function;
    send_command(APP_CMD, 0x0, <other parameter>); // CMD55, Application specific CMD prefix
    if (no error calling wait_for_response(APP_CMD, <...>) { // CMD55 is accepted
        send_command(SD_APP_OP_COND, <voltage range>, <...>); // ACMD41, to set voltage range for
memory part or SD card
        wait_for_response(SD_APP_OP_COND); // voltage range is set
        if (card type is UNKNOWN)
           label the card as SD;
        return; //
    } // end of if (no error ...
    else if (errors other than time-out occur) { // command/response pair is corrupted
        deal with it by program specific manner;
    } // of else if (response time-out
    else { // CMD55 is refuse, it must be eMMC
        if (card is already labeled as SDCombo) { // change label
            re-label the card as SDIO;
            ignore the error or report it;
            return; // card is identified as SDIO
        } // end of if (card is ...
        send_command(SEND_OP_COND, <voltage range>, <...>);
        if(RESP_TIMEOUT == wait_for_response(SEND_OP_COND)) { // CMD1 is not accepted,
            either label the card as UNKNOWN;
            return;
        } // end of if (RESP_TIMEOUT ...
    } // end of else
}
```

81.5.2.4 Card registry

This section briefly describes the registry flow. For details, please refer to the card specifications. Card registry for the eMMC and SD/SDIO/SD combo cards are different. For the SD card, the identification process starts at a clock rate lower than 400 kHz and the power voltage higher than 2.7 V (as defined by the card spec). Currently, the CMD line output drives are push-pull drivers instead of open-drain. After the bus is activated, the host requests the card to send their valid operation conditions. The response to ACMD41 is the operation condition register of the card. The same command is sent to all the new cards in the system. Incompatible cards are put into the Inactive state. The host then issues the command, All_Send_CID (CMD2), to each card to get its unique card identification (CID) number. Cards that are currently unidentified (in the Ready state), send their CID number as the response. After the CID is sent by the card, the card goes into the Identification state.

The host then issues Send_Relative_Addr (CMD3), requesting the card to publish a new relative card address (RCA) that is shorter than the CID. This RCA is used to address the card for future data transfer operations. After the RCA is received, the card changes its state to the Standby state. At this point, if the host wants the card to have an alternative RCA number, it may ask the card to publish a new number by sending another Send_Relative_Addr command to the card. The last published RCA is the actual RCA of the card.

The host repeats the identification process with CMD2 and CMD3 for each card in the system until the last CMD2 gets no response from any of the cards in the system.

For eMMC operation, the host starts the card identification process in the open-drain mode with the identification clock rate lower than 400 kHz and the power voltage higher than 2.7 V. The open drain driver stages on the CMD line to allow parallel card operation during card identification. After the bus is activated, the host requests the cards to send their valid operation conditions

(CMD1). The response to CMD1 is the "wired AND " operation on the condition restrictions of all cards in the system. Incompatible cards are sent into the Inactive state. The host then issues the broadcast command All_Send_CID (CMD2), asking all cards for their unique card identification (CID) number. All unidentified cards (the cards in the Ready state) simultaneously start sending their CID numbers serially, while bit-wise monitoring their outgoing bit stream. Those cards, whose outgoing CID bits do not match the corresponding bits on the command line in any one of the bit periods, stop sending their CID immediately and must wait for the next identification cycle. As the CID is unique for each card, only one card can be successfully sent its full CID to the host. This card then goes into the Identification state. Thereafter, the host issues Set_Relative_Addr (CMD3) to assign a relative card address (RCA) to the card. After the RCA is received, the state of the card changes to standby, and the card does not react in further identification cycles. Also, its output driver switches from open-drain to push-pull. The host repeats the process, mainly CMD2 and CMD3, until the host receives a time-out condition to recognize the completion of the identification process.

The following steps show how to perform an operation using eMMC:

```
card_registry() {
    do { // decide RCA for each card until response time-out
        if(card is labelled as SDCombo or SDIO) { // for SDIO like device
            send_command(SET_RELATIVE_ADDR, 0x00, <...>); // ask SDIO to publish its RCA
            retrieve RCA from response;
        } // end if (card is labelled as SDCombo ...
        else if (card is labelled as SD) { // for SD card
            send_command(ALL_SEND_CID, <...>);
            if(RESP_TIMEOUT == wait_for_response(ALL_SEND_CID))
                break;
            send_command(SET_RELATIVE_ADDR, <...>);
            retrieve RCA from response;
        } // else if (card is labelled as SD ...
        else if (card is labelled as eMMC) {
            send_command(ALL_SEND_CID, <...>);
            rca = 0x1; //arbitrarily set RCA, 1 here for example
            send_command(SET_RELATIVE_ADDR, 0x1 << 16, <...>); // send RCA at upper 16 bits
        } // end of else if (card is labelled as eMMC...
    } while (response is not time-out);
}
```

81.5.3 Card access

Information about Block Write, Block Read, Suspense Resume, ADMA Usage, Transfer Error, and Card Interrupt are detailed in the sections below.

81.5.3.1 Block write

Information on Normal Write, DDR Write, and Write with Pause are detailed in the sections below.

81.5.3.1.1 Normal Write

During a block write (CMD24 - 27, CMD60, CMD61), one or more blocks of data are transferred from the host to the card with a CRC appended to the end of each block by the host. If the CRC fails, the card indicates that the failure on the DATA line. The transferred data is discarded and not written, and all further transmitted blocks (in multiple block write mode) are ignored.

If the host uses partial blocks whose accumulated length is not block aligned and block misalignment is not allowed (CSD parameter WRITE_BLK_MISALIGN is not set), the card detects the block misalignment error and aborts the programming before the beginning of the first misaligned block. The card sets the ADDRESS_ERROR error bit in the status register, and while ignoring all further data transfer, waits in the Receive-data-State for a stop command. The write operation is also aborted if the host tries to write over a write-protected area.

For eMMC and SD cards, programming of the CID and CSD registers does not require a previous block length setting. The transferred data is also CRC protected. If a part of the CSD or CID register is stored in ROM, then this unchangeable part must match the corresponding part of the receive buffer. If this match fails, then the card reports an error and does not change any register contents.

For all types of cards, some may require long and unpredictable periods of time to write a block of data. After receiving a block of data and completing the CRC check, the card begins writing and holds the DATA line low if its write buffer is full and unable to accept new data from a new WRITE_BLOCK command. The host may poll the status of the card with a SEND_STATUS command (CMD13) or other means for SDIO at any time, and the card responds with its status. The responded status indicates whether the card can accept new data or whether the write process is still in progress. The host may deselect the card by issuing a CMD7 (to select a different card) to place the card into the Standby state and release the DATA line without interrupting the write operation. When re-selecting the card, it reactivates the busy indication by pulling data to low if the programming is still in progress and the write buffer is unavailable.

The software flow to write to a card that incorporates the internal DMA and the write operation is a multi-block write with the Auto CMD12 enabled. For the other two methods (by means of external DMA or CPU polling status with different transfer methods, the internal DMA parts should be removed and the alternative steps should be straightforward.

The software flow to write to a card is described below:

1. Check the card status, wait until the card is ready for data.
2. Set the card block length/size:
   - For eMMC and SD cards, use SET_BLOCKLEN (CMD16).
   - For SDIO cards or the I/O portion of SDCombo cards, use IO_RW_DIRECT (CMD52) to set the I/O Block Size bit field in the CCCR register (for function 0) or FBR register (for functions 1~7).
3. Set the uSDHC block length register to be the same as the block length set for the card in step 2.
4. Set the uSDHC number block register (NOB), where nob is 5, for instance.
5. Disable the buffer write ready interrupt, configure the DMA settings and enable the uSDHC DMA when sending the command with data transfer. The AC12EN bit should also be set.
6. Wait for the Transfer Complete interrupt.
7. Check the status bit to see if a write CRC error occurred, or another error that occurred during the auto12 command sending and response receiving.

81.5.3.1.2 DDR write

uSDHC supports the dual data rate mode.

The software flow to write to a card in the DDR mode is described as below:

1. Check the card status and wait until the card is ready for data.

   NOTE

   For eMMC, block length only can be set to 512byte.

2. Set the uSDHC number block register (NOB), where nob is 5, for instance.
3. Set the eMMC, SD card, or SDIO to high-speed mode and use SWITCH(CMD6).
4. Set the eMMC bus or SD with 4-bit/8-bit DDR mode and use SWITCH(CMD6).
5. Disable the buffer write ready interrupt, configure the DMA settings and enable the uSDHC DMA when sending the command with data transfer. The DDR_EN and AC12EN bits should be set.
6. Wait for the Transfer Complete interrupt.
7. Check the status bit to see if a write CRC error occurred or another error that occurred during the auto12 command sending and response receiving.

81.5.3.1.3 Write with Pause

The write operation can be paused during the transfer. Instead of stopping the CLK at any time to pause all the operations, which is also inaccessible to the host driver, the driver can set the Stop At Block Gap Request (SABGREQ) bit in the Protocol

Control register to pause the transfer between the data blocks. As there is no time-out condition in a write operation during the data blocks, a write to all types of cards can be paused in this way, and if the DATA0 line is not required to de-assert to release the busy state, no suspend command is needed.

Similar to the flow described in Normal Write, the write with pause is shown with the same kind of write operation:

1. Check the card status and wait until the card is ready for data.
2. Set the card block length/size:
   - For eMMC and SD cards, use SET_BLOCKLEN (CMD16).
   - For SDIO or the I/O portion of SDCombo cards, use IO_RW_DIRECT(CMD52) to set the I/O Block Size bit field in the CCCR register (for function 0) or FBR register (for functions 1~7).
3. Set the uSDHC block length register to be the same as the block length set for the card in step 2.
4. Set the uSDHC number block register (NOB), where nob is 5, for instance.
5. Disable the buffer write ready interrupt, configure the DMA settings and enable the uSDHC DMA when sending the command with data transfer. The AC12EN bit should also be set.
6. Set the SABGREQ bit.
7. Wait for the Transfer Complete interrupt.
8. Clear the SABGREQ bit.
9. Check the status bit to see if a write CRC error occurred.
10. Set the CREQ bit to continue the write operation.
11. Wait for the Transfer Complete interrupt.
12. Check the status bit to see if a write CRC error occurred or some another error that occurred during the auto12 command sending and response receiving.

The number of blocks left during the data transfer is accessible by reading the contents of the BLKCNT field in the Block Attribute register. As the data transfer and the setting of the SABGREQ bit are concurrent, and the delay of register read and the register setting, the actual number of blocks left may not be exactly the value read earlier. The host driver reads the value of BLKCNT after the transfer is paused and the Transfer Complete interrupt is received.

It is also possible that the last block has begun when the Stop At Block Gap Request is sent to the buffer. In this case, the next block gap is the end of the transfer. These types of requests are ignored, the driver should treat these as a non-pause transfer, and deal with it as a common write operation.

When the write operation is paused, the data transfer inside the host system is not stopped, and the transfer is active until the data buffer is full. Because of this, it is recommended to avoid using the Suspend command for SDIO. This is because when such a command is sent, uSDHC interprets the system that switches to another function on SDIO and flush the data buffer. uSDHC takes the Resume command as a normal command with data transfer, and it is left for the host driver to set all the relevant registers before the transfer is resumed. If there is only one block to send when the transfer is resumed, the MSBSEL and BCEN bits of the Transfer Type register are set as well as the AC12EN bit. However, the uSDHC module automatically sends a CMD12 to mark the end of the multi-block transfer.

81.5.3.2 Block read

Information about Normal read, DDR read, Read with Pause, and Delay Line (DLL) in Read Path are detailed in the sections below.

81.5.3.2.1 Normal read

For block reads, the basic unit of data transfer is a block whose maximum size is stored in areas defined by the corresponding card specification. A CRC is appended to the end of each block, ensuring data transfer integrity. The CMD17, CMD18, CMD53, CMD60, CMD61, and so on, can initiate a block read. After completing the transfer, the card returns to the Transfer state. For multi blocks read, data blocks are continuously transferred until a stop command is issued.

The software flow to read from a card that incorporates the internal DMA and the read operation is a multi-block read with the Auto CMD12 enabled. For the other two methods (by means of external DMA or CPU polling status with different transfer methods, the internal DMA parts should be removed and the alternative steps should be straightforward.

The software flow to read from a card is described below:

1. Check the card status and wait until card is ready for data.
2. Set the card block length/size:
   - For eMMC and SD cards, use SET_BLOCKLEN (CMD16).
   - For SDIO or the I/O portion of SDCombo cards, use IO_RW_DIRECT(CMD52) to set the I/O Block Size bit field in the CCCR register (for function 0) or FBR register (for functions 1~7).
3. Set the uSDHC block length register to be the same as the block length set for the card in step 2.
4. Set the uSDHC number block register (NOB), where nob is 5, for instance.
5. Disable the buffer read ready interrupt, configure the DMA settings, and enable the uSDHC DMA when sending the command with data transfer. The AC12EN bit should also be set.
6. Wait for the Transfer Complete interrupt.
7. Check the status bit to see if a read CRC error occurred, or another error occurred during the auto12 command sending and response receiving.

81.5.3.2.2 DDR read

The uSDHC module supports dual data rate mode.

The software flow to write to a card in the DDR mode is described below:

1. Check the card status and wait until the card is ready for data.

NOTE

For eMMC, block length can be set to only 512 bytes.

2. Set the uSDHC number block register (NOB) where nob is 5, for instance.
3. Set the eMMC, SD card, or SDIO to high-speed mode and use SWITCH (CMD6).
4. Set the eMMC bus or SD with 4-bit /8-bit DDR mode and use SWITCH(CMD6).
5. Disable the buffer write ready interrupt, configure the DMA settings, and enable the uSDHC DMA when sending the command with data transfer. The DDR_EN and AC12EN bits should be set.
6. Wait for the Transfer Complete interrupt.
7. Check the status bit to see if a write CRC error occurred, or another error that occurred during the auto12 command sending and response receiving.

81.5.3.2.3 Read with Pause

The read operation is not generally able to pause. Only SDIO (and SDCombo card working under I/O mode) supporting the Read Wait feature can pause during the read operation. If SDIO supports Read Wait (SRW bit in CCCR register is 1), the host driver can set the SABGREQ bit in the Protocol Control register to pause the transfer between the data blocks. Before setting the SABGREQ bit, ensure that the RWCTL bit in the Protocol Control register is set, otherwise uSDHC does not assert the Read Wait signal during the block gap and data corruption occurs. It is recommended to set the RWCTL bit after the Read Wait capability of SDIO is recognized.

Similar to the flow described in Normal read, the read with pause is shown with the same kind of read operation:

1. Check the SRW bit in the CCR register on SDIO to confirm that the card supports the Read Wait mode.
2. Set the RWCTL bit.

3. Check the card status and wait until the card is ready for data.
4. Set the card block length/size:
   - For eMMC and SD cards, use SET_BLOCKLEN (CMD16).
   - For SDIO or the I/O portion of SDCombo cards, use IO_RW_DIRECT(CMD52) to set the I/O Block Size bit field in the CCCR register (for function 0) or FBR register (for functions 1~7).
5. Set the uSDHC block length register to be the same as the block length set for the card in step 2.
6. Set the uSDHC number block register (NOB), where nob is 5, for instance.
7. Disable the buffer read ready interrupt, configure the DMA setting, and enable the uSDHC DMA when sending the command with data transfer. The AC12EN bit should also be set.
8. Set the SABGREQ bit.
9. Wait for the Transfer Complete interrupt.
10. Clear the SABGREQ bit.
11. Check the status bit to see if read CRC error occurred.
12. Set the CREQ bit to continue the read operation.
13. Wait for the Transfer Complete interrupt.
14. Check the status bit to see if a read CRC error occurred, or another error occurred during the auto12 command sending and response receiving.

Similar to the Write operation, it is possible to meet the ending block of the transfer when paused. In this case, uSDHC ignores the Stop At Block Gap Request and treats it as a command read operation.

Unlike the write operation, there is no remaining data inside the buffer when the transfer is paused. All data received before the pause is transferred to the host system. No matter if the Suspend command is sent or not, the internal data buffer is not flushed.

If the Suspend command is sent and the transfer is later resumed by means of a Resume command, uSDHC takes the command as a normal one accompanied with data transfer. It is left for the host driver to set all the relevant registers before the transfer is resumed. If there is only one block to send when the transfer is resumed, the MSBSEL and BCEN bits of the Transfer Type register are set, as well as the AC12EN bit. However, the uSDHC automatically sends CMD12 to mark the end of multi-block transfer.

81.5.3.2.4 Delay Line (DLL) in read path

The DLL is newly added to assist in sampling read data. The DLL provides the ability to programmatically select a quantized delay (in fractions of the clock period) regardless of on-chip variations such as process, voltage, and temperature (PVT).

The reasons why DLL is needed for uSDHC are these:

- The path of read data traveling from card to host varies.
- In the eMMC and SD cards DDR mode, the minimum input setup and hold time are both at 2.5 ns.

The data sampling window is so small that the delay of loopback clock needs to be accurate and consistent regardless of PVT. The DLL takes the divided card_clk as the reference clock and loopback clock as the input clock. It then generates a delayed version of the input clock according to the programmed target delay.

The DLL can be disabled or bypassed, and it can also be manually set for a fixed delay in the override mode. The override value set is the number of delay cells. In the override mode, there is no need to set the DLL_enable. Another DLL mode is target value mode. In this mode, the DLL automatically adjusts the number of delay cells according to the target value set by the user and PVT changes. Be aware that the target value is in units of 1/32 of the clock reference period. If the card_clk is 100Mhz, then the reference clock period is 10ns; setting target value of 16 means 5ns =(16/32)*10ns. The software can disable automatic update by the setting dll_gate_update bit.

As you might change the frequency of card_clk from time to time by changing SDCLKFS[7:0]/DVS[3:0], the software must adjust the delay value to ensure it works correctly when the reference clock (card_clk) is changed. If DLL is to be used, make sure

SDCLKFS is used (at least be set to divided by 2) so that the REF_CLK generated are the same frequency as card_clk. There are two DLLs, PARTS DLL and STROBE DLL.

Step 1: Set the DLL_CTRL_RESET and DLL_CTRL_ENABLE fields

Step 2: Configure the SDCLKFS[7:0] and DVS[3:0]

Step 3: Wait until SDSTB is asserted

Step 4: Clear the DLL_CTRL_RESET field

Step 5: Wait until both the DLL_STS_SLV_LOCK and DLL_STS_REF_LOCK are asserted

Step 6: Set the DLL_CTRL_SLV_FORCE_UPD

Step 7: Clear the DLL_CTRL_SLV_FORCE_UPD

NOTE

The software should make sure that the DLL_CTRL_SLV_FORCE_UPD lasts for at least one card_clk. So, the software may need to add some delay between step 6 and step 7.

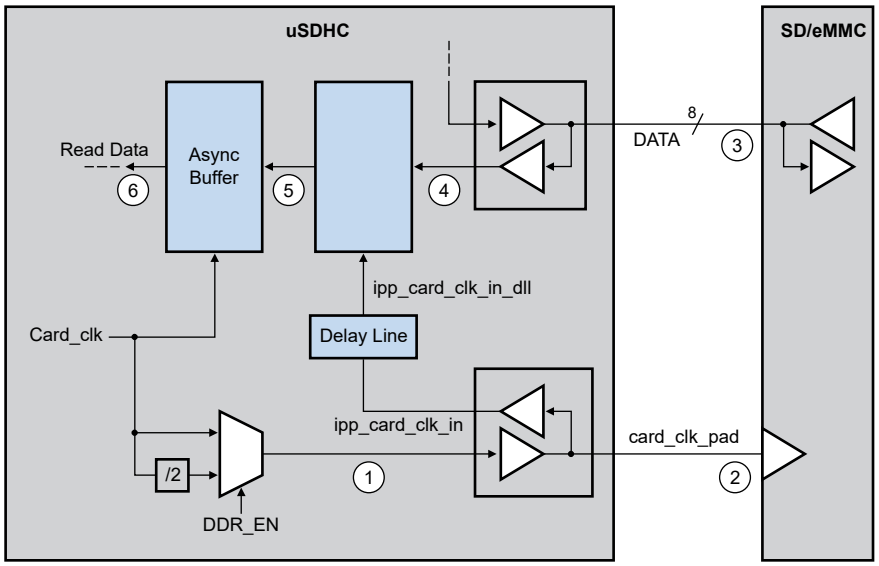


Figure 554. DLL in read path

81.5.3.3 Suspend Resume

The uSDHC module supports the Suspend Resume operations of SDIO, although slightly different than the suggested implementation of Suspend in the SDIO specification.

81.5.3.3.1 Suspend

After setting the SABGREQ bit, the host driver may send a Suspend command to switch to another function of SDIO. The uSDHC module does not monitor the content of the response, so it does not know if the Suspend command succeeded or not. Accordingly, it does not de-assert Read Wait for read pause. To solve this problem, the host driver does not mark the Suspend command as "Suspend", (that is, setting the CMDTYP bits to 01). Instead, the driver sends this command as if it were a normal command, and only when the command succeeds, and the BS bit is set in the response, can the Driver send another command marked as "Suspend" to inform uSDHC that the current transfer is suspended. Here is the sequence for the Suspend operation:

1. Set the SABREQ bit to pause the current data transfer at block gap.
2. After the BGE bit is set, send the Suspend command to suspend the active function. The CMDTYP bit field must be 2'b00.

3. Check the BS bit of the CCCR in the response. If it is 1, repeat this step until the BS bit is cleared or abandon the suspend operation according to the Driver strategy.
4. Send another normal I/O command to the suspended function. The CMDTYP of this command must be 2'b01, so uSDHC can detect this special setting and be informed that the paused operation has successfully suspended. If the paused transfer is a read operation, uSDHC stops driving DATA2 and goes to the idle state.
5. Save the context registers in the system memory for later use, including the DMA System Address Register (for internal DMA operation), and the Block Attribute Register.
6. Begin operation for another function on SDIO.

81.5.3.3.2 Resume

To resume the data transfer, a Resume command is issued:

1. To resume the suspended function, restore the context register with the saved value in step #5 of the Suspend operation.
2. Send the Resume command: In the Transfer Type register, all the bit fields are set to the value as if this were another ordinary data transfer instead of a transfer resume (except the CMDTYP is set to 2'b10).
3. If the Resume command has responded, the data transfer is resumed.

81.5.3.4 ADMA usage

To use the ADMA in a data transfer, the host driver must prepare the correct descriptor chain prior to sending the read/write command. The steps to prepare the correct descriptor chain are these:

1. Create a descriptor to set the data length that the current descriptor group is about to transfer. The data length should be even numbers of the block size.
2. Create another descriptor to transfer the data from the address setting in this descriptor. The data address must be at a page boundary (4KB address aligned).
3. If necessary, create a Link descriptor containing the address of the next descriptor. The descriptor group is created in steps 1 ~ 3.
4. Repeat steps 1 ~ 3 until all descriptors are created.
5. In the last descriptor, set the End flag to 1 and make sure the total length of all descriptors matches the product of the block size and block number configured in the Block Attribute Register.
6. Set the ADMA System Address Register to the address of the first descriptor and set the DMAS field in the Protocol Control Register to 01 to select the ADMA.
7. Issue a write or read command with the DMAEN bit set to 1 in the Transfer Type Register.

Steps 1 ~ 5 are independent of step 6, so step 6 can finish before steps 1 ~ 5. Regarding the descriptor configuration, it is recommended not to use the Link descriptor as it requires extra system memory access.

81.5.3.5 Transfer error

Information about CRC, Internal DMA, Transfer ADMA, and Auto CMD12 errors are detailed in the sections below.

81.5.3.5.1 CRC error

It is possible at the end of a block transfer that a write CRC status error or read CRC error occurs. For this type of error, the latest block received is discarded. This is because the integrity of the data block is not guaranteed. It is recommended to discard the following data blocks and re-transfer the block from the corrupted one. For a multi-block transfer, the host driver issues a CMD12 to abort the current process and start the transfer by a new data command. In this scenario, even when the AC12EN and BCEND bits are set, uSDHC does not automatically send a CMD12 because the last block is not transferred. On the other hand, if it is within the last block that the CRC error occurs, an Auto CMD12 is sent by uSDHC. In this case, the host driver re-sends or re-obtains the last block with a single block transfer.

81.5.3.5.2 Internal DMA error

During the data transfer with internal Simple DMA, if the DMA engine encounters an error on the AHB bus, the DMA operation is aborted, and the DMA error interrupt is sent to the host system. When acknowledged by such an interrupt, the host driver calculates the start address of data block in which the error occurs. The start address can be calculated by either:

- Reading the DMA System Address register: The error occurs during the previous burst. Considering the block size, the previous burst length and the start address of the next burst transfer, it is straight forward to obtain the start address of the corrupted block.
- Reading the BLKCNT field of the Block Attribute register: Considering the number of blocks left, the total number to transfer, the start address of transfer, and the size of each block, the start address of corrupted block can be determined.To use this method, MIX_CTRL[BCEN] bit has to be set to enable Block Attribute register update.

When a DMA error occurs, it is recommended to abort the current transfer by means of a CMD12 (for multi block transfer), apply a reset for data, and restart the transfer from the corrupted block to recover from the error.

81.5.3.5.3 Transfer ADMA error

There are three kinds of possible ADMA errors: The AHB transfer, invalid descriptor, and data-length mismatch. Whenever these errors occur, the DMA transfer stops and the corresponding error status bit is set. For acknowledging the status, the host driver should recover the error as shown below and re-transfer from the place of interruption.

- AHB transfer error: Such errors may occur during data transfer or descriptor fetch. For either scenario, it is recommended to retrieve the transfer context, reset for the data part and re-transfer the block that was corrupted, or to the next block if no block is corrupted.
- Invalid descriptor error: For such errors, it is recommended to retrieve the transfer context, reset for the data part and recreate the descriptor chain from the invalid descriptor and issue a new transfer. As the data to transfer now may be less than the previous setting, the data length configured in the new descriptor chain should match the new value.
- Data-length mismatch error: It is similar to recover from this error. The Host Driver polls relating registers to retrieve the transfer context, apply a reset for the data part, configure a new descriptor chain and make another transfer if there is data left. Like the previous scenario of the invalid descriptor error, the data length must match the new transfer.

81.5.3.5.4 Auto CMD12 error

After the last block of the multi-block transfer is sent or received, and the AC12EN bit is set when the data transfer is initiated by the data command, uSDHC automatically sends a CMD12 to the card to stop the transfer. When errors with this command occur, it is recommended to the host driver to deal with the situations in the following manner:

- Auto CMD12 response time-out: It is not certain whether the command is accepted by the card or not. The host driver clears the auto CMD12 error status bits and re-send CMD12 until it is accepted by the card.
- Auto CMD12 response CRC error: As the card responds to CMD12, it aborts the transfer. The host driver may ignore the error and clear the error status bit.
- Auto CMD12 conflict error or not sent: The command is not sent; therefore, the host driver sends a CMD12 manually.

81.5.3.6 Card interrupt

The external cards can inform the host controller by means of some special signals. For SDIO, it can be the low-level on the DATA1 line during some special period. The uSDHC module only monitors the DATA1 line and supports the SDIO interrupt.

When the SDIO interrupt is captured by uSDHC, and the host system is informed by uSDHC asserting the uSDHC interrupt line, the interrupt service from the host driver is called.

As the interrupt source is controlled by the external card, the interrupt from SDIO must be serviced before the CINT bit is cleared. Refer to Card interrupt handling for the card interrupt handling flow.

81.5.4 Switch function

A switch command is issued by the host driver to enable new features added to the eMMC and SD cards. The eMMC and SD cards can transfer data at bus widths other than 1-bit. Different speed modes are also defined. To enable these features, a switch command is issued by the host driver.

For SDIO, the high-speed mode are enabled by writing the EHS bit in the CCCR register after the SHS bit is confirmed as high. For SD cards, the high-speed mode are queried and enabled by a CMD6 (with the mnemonic symbol as SWITCH_FUNC). For eMMC, the high-speed mode is queried by a CMD8 and enabled by a CMD6 (with the mnemonic symbol as SWITCH).

The SDR4-bit, SDR8-bit, DDR4-bit, and DDR8-bit width of eMMC is also enabled by the SWITCH command, but with a different argument.

These new functions can also be disabled by a software reset. For SDIO, it can be done by setting the RES bit in the CCCR register. For other cards, it can be accomplished by issuing a CMD0. This method of restoring to the normal mode is not recommended because a complete identification process is needed before the card is ready for data transfer.

For the sake of simplicity, the following pseudo-code examples do not show current capability check, which is recommended in the function switch process.

81.5.4.1 Query, enable, and disable SDIO high-speed mode

```
enable_sdio_high_speed_mode(void)
{
send CMD52 to query bit SHS at address 0x13;
if (SHS bit is '0') report SDIO does not support high speed mode and return;
send CMD52 to set bit EHS at address 0x13 and read after write to confirm EHS bit is set;
change clock divisor value or configure the system clock feeding into uSDHC to generate the card_clk
of around 50MHz;
(data transactions like normal peers)
}
disable_sdio_high_speed_mode(void)
{
send CMD52 to clear bit EHS at address 0x13 and read after write to confirm EHS bit is cleared;
change clock divisor value or configure the system clock feeding into uSDHC to generate the card_clk
of the desired value below 25MHz;
(data transactions like normal peers)
}
```

81.5.4.2 Query, enable, and disable SD high-speed mode

```
enable_sd_speed_mode(void)
{
set BLKCNT field to 1 (block), set BLKSIZE field to 64 (bytes);
send CMD6, with argument 0xFFFFFx and read 64 bytes of data accompanying the R1 response;(high speed
mode,x=1;)
wait data transfer done bit is set;
check if the bit x of received 512 bits is set;
if (bit 401 is '0') report the SD card does not support high speed mode and return;



      send CMD6, with argument 0x80FFFFFx and read 64 bytes of data accompanying the R1 response;
(high speed mode,x=1;)
check if the bit field 379~376 is 0xF;
if (the bit field is 0xF) report the function switch failed and return;
change clock divisor value or configure the system clock feeding into uSDHC to generate the card_clk
of around 50MHz for high speed mode
(data transactions like normal peers)
}
```

```
disable_sd_speed_mode(void)
{
set BLKCNT field to 1 (block), set BLKSIZE field to 64 (bytes);
send CMD6, with argument 0x80FFFFF0 and read 64 bytes of data accompanying the R1 response;
check if the bit field 379~376 is 0xF;
if (the bit field is 0xF) report the function switch failed and return;
change clock divisor value or configure the system clock feeding into uSDHC to generate the
card_clk of the desired value below 25MHz;
(data transactions like normal peers)
}
```

81.5.4.3 Query, enable, and disable eMMC high-speed mode

```
enable_mmc_high_speed_mode(void)
{
send CMD9 to get CSD value of eMMC;
check if the value of SPEC_VER field is 4 or above;
if (SPEC_VER value is less than 4) report the eMMC does not support high speed mode and return;
set BLKCNT field to 1 (block), set BLKSIZE field to 512 (bytes);
send CMD8 to get EXT_CSD value of eMMC;
extract the value of CARD_TYPE field to check the 'high speed mode' in this eMMC is 26MHz or
52MHz;
send CMD6 with argument 0x1B90100;
send CMD13 to wait card ready (busy line released);
send CMD8 to get EXT_CSD value of eMMC;
check if HS_TIMING byte (byte number 185) is 1;
if (HS_TIMING is not 1) report eMMC switching to high speed mode failed and return;
change clock divisor value or configure the system clock feeding into uSDHC to generate the
card_clk of around 26MHz or 52MHz according to the CARD_TYPE;
(data transactions like normal peers)
}
disable_mmc_high_speed_mode(void)
{
send CMD6 with argument 0x2B90100;
set BLKCNT field to 1 (block), set BLKSIZE field to 512 (bytes);
send CMD8 to get EXT_CSD value of eMMC;
check if HS_TIMING byte (byte number 185) is 0;
if (HS_TIMING is not 0) report the function switch failed and return;
change clock divisor value or configure the system clock feeding into uSDHC to generate the
card_clk of the desired value below 20MHz;
(data transactions like normal peers)
}
```

81.5.4.4 Set eMMC bus width

```
change_mmc_bus_width(void)
{
send CMD9 to get CSD value of eMMC;
check if the value of SPEC_VER field is 4 or above;
if (SPEC_VER value is less than 4) report the eMMC does not support multiple bit width and return;
send CMD6 with argument 0x3B70x00; (8-bit(dual data rate), x=6; 4-bit(dual data rate), x=5;8-bit,
x=2; 4-bit, x=1; 1-bit, x=0)
send CMD13 to wait card ready (busy line released);
(data transactions like normal peers)
}
```

81.5.5 ADMA operation

Here are the codes for the ADMA1 and ADMA2 operations.

81.5.5.1 ADMA1 operation

```
Set_adma1_descriptor
{
if (to start data transfer) {
// Make sure the address is 4KB align.
Set 'Set' type descriptor;
{
Set Act bits to 01;
Set [31:12] bits data length (byte unit);
}
Set 'Tran' type descriptor;
{
Set Act bits to 10;
Set [31:12] bits address (4KB align);
}
}
else if (to fetch descriptor at non-continuous address) {
Set Act bits to 11;
Set [31:12] bits the next descriptor address (4KB aligned);
}
else { // other types of descriptor
Set Act bits accordingly
}
if (this descriptor is the last one) {
Set End bit to 1;
}
if (to generate interrupt for this descriptor) {
Set Int bit to 1;
}
Set Valid bit to 1;
}
```

81.5.5.2 ADMA2 operation

```
Set_adma2_descriptor
{
if (to start data transfer) {
// Make sure the address is a 32-bit boundary (lower 2-bit are always '00').
Set higher 32-bit of descriptor for this data transfer initial address;
Set [31:16] bits data length (byte unit);
Set Act bits to '10';
}
else if (to fetch descriptor at non-continuous address) {
Set Act bits to '11';
// Make sure the address is 32-bit boundary (lower 2-bit are always set to '00').
Set higher 32-bit of descriptor for the next descriptor address;
}
else { // other types of descriptor
Set Act bits accordingly
}
if (this descriptor is the last one) {
Set 'End' bit '1';
}
if (to generate interrupt for this descriptor) {
```

```
Set 'Int' bit '1';
}
Set the 'Valid' bit to '1';
}
```

81.5.6 Fast boot operation

81.5.6.1 Normal fast boot flow

Here are the steps for normal fast boot flow:

1. Software must configure the SYS_CTRL[INITA] bit to make sure that 74 card clocks are finished.
2. Software must configure the eMMC Boot Register (offset 0xc4) bit 6 to 1 (enable boot), and bit 5 to 0 (normal fast boot), and bit 4 to select the ack mode. If the data is sent through the DMA mode, the software should configure bit 7 to enable the automatic stop at block gap feature, and configure bit 3-bit 0 to select the ack timeout value according to the SD CLK frequency.
3. Software then needs to configure the Block Attributes Register to set the block size and count. If in DDR fast boot mode, the block size only can be configured to 512 bytes.
4. Software must configure the Protocol control register to set Data Transfer Width (DTW). If in the DDR fast boot mode, DTW only can be configured to 4-bit/8-bit dataline mode.
5. Software needs to configure the Command Argument Register to set argument if needed (no need in normal fast boot).
6. Software must configure the Transfer Type Register to start the boot process. In normal boot mode, CMDINX, CMDTYP, RSPTYP, CICEN, CCCEN, AC12EN, BCEN and DMAEN retain the default value, where DPSEL bit is set to 1, DTDSEL is set to 1 and MSBSEL is set to 1.
7. DMAEN should be configured as 0 in the polling mode and if BCEN is configured as 1, it is recommended to configure the number of blocks in the Block Attributes Register to the maximum value. If in DDR fast boot mode, DDR_EN needs to be set to 1.
8. When step 6 is configured, the boot process begins. The software needs to poll the data buffer ready status to read the data from the buffer in time. If a boot timeout happens (ack times out or the first data read times out), an interrupt is triggered, and the software must configure eMMC Boot Register to bit 6 to 0 to disable boot. This makes CMD high, then after at least 56 clocks, it is ready to begin a normal initialization process.
9. If there is no timeout, software needs to determine when the data read is finished and then configure eMMC Boot Register bit 6 to 0 to disable boot. This render CMD line high and command completed asserted. After at least 56 clocks, it is ready to begin the normal initialization process.
10. You must reset the host and begin the normal process.

81.5.6.2 Alternative fast boot flow

Here are the steps for alternative fast boot flow:

1. Software needs to configure SYS_CTRL[INITA] to make sure 74 card clocks are finished.
2. Software needs to configure eMMC Boot Register (offset 0xc4) bit 6 to 1 (enable boot), and bit 5 to 1 (alternative boot), and bit 4 to select the ack mode or not. If data needs to be sent through the DMA mode, then configure bit 7 to enable the automatic stop at block gap feature. Software should also configure bit 3-bit 0 to select the ack timeout value according to the SD clock frequency.
3. Software then needs to configure the Block Attributes Register to set the block size and count. If in the DDR fast boot mode, the block size only can be configured to 512 bytes.
4. Software needs to configure the Protocol control register to set the data transfer width (DTW). If in the DDR fast boot mode, DTW only can be configured to 4-bit/8-bit dataline mode.
5. Software needs to configure Command Argument Register to set argument to 0xFFFFFFFA.

6. Software needs to configure the Transfer Type Register to start the boot process by CMD0 with the 0xFFFFFFFA argument. In alternative boot, CMDINX, CMDTYP, RSPTYP, CICEN, CCCEN, AC12EN, BCEN, and DMAEN retain the default value. DPSEL bit is set to 1, DTDSEL is set to 1, and MSBSEL is set to 1. Note DMAEN should be configured as 0 in the polling mode, and if BCEN is configured as 1 in polling mode, it is recommended to configure the block count in the Block Attributes Register to the maximum value. If in the DDR fast boot mode, DDR_EN needs to be set to 1.
7. When step 6 is configured, the boot process begins. Software needs to poll the data buffer ready status to read the data from the buffer in time. If there is a boot timeout (ack data timeout in 50ms or data timeout in 1s), the host sends out the interrupt and the software needs to send CMD0 with reset and then configure the boot enable bit to 0 to stop this process.
8. If there is no time out, the software needs to decide when to stop the boot process, and send out the CMD0 with reset and then after the command is completed, configure the eMMC Boot Register bit 6 to stop the process. After 8 clocks from the command completion, the slave (card) is ready for the identification step.
9. You must reset the host and begin the normal process.

81.5.6.3 Fast boot application case (in DMA mode)

In the boot application case, because the image destination and the image size are contained in the beginning of the image, it is necessary to switch DMA parameters on the fly during eMMC fast boot.

In fast boot, the host can use Advanced DMA2 (ADMA2) with two destinations.

The detailed flow is described below:

1. The software needs to configure INIT_ACTIVE bit (system control register bit 27) to make sure that 74 card clocks are finished.
2. The software needs to configure the eMMC Boot Register (offset 0xc4) bit 6 to 1 (enable boot); and bit 5 to 0 (normal fast boot) or 1 (alternative boot); and bit 4 to select the ack mode. In DMA mode, configure bit 7 to 1 to enable the automatic stop at block gap feature. Also configure bits[31-16] to set the (BLK_CNT - VALUE1). Here VALUE1 is the value of the block count that needs to transfer the first time, so that the host stops at the block gap when the uSDHC controller gets VAULE1 blocks from the device. Also, configure bits[3-0] to select the ack timeout value according to the SD clock frequency.
3. The software then needs to configure the Block Attributes Register to set block size and count. If in DDR fast boot mode, the block size only can be configured to 512 bytes. In DMA mode, it is recommended to set the block count (BLK_CNT) to the max value (16'hffff).
4. The software needs to configure Protocol Control Register to set DTW (data transfer width). If in DDR fast boot mode, the DTW only can be configured to 4-bit/8-bit dataline mode.
5. Software enable ADMA2 by configuring Protocol Control Register bits [9-8].
6. The software needs to set at least three pairs of ADMA2 descriptor in boot memory (that is, in IRAM, at least six words). The first pair descriptor defines the start address (that is, IRAM) and data length (that is,512byte*VALUE1) of the first part boot code. The software also needs to set the second pair descriptor, the second start address (any value that is writable), and data length is suggested to set 1~2word (record as VALUE2). Note that the second couple desc also transfers useful data even at lease 1 word, because our ADMA2 cannot support 0 data_length data transfer descriptor.
7. The software needs to configure Command Argument Register to set argument to 0xFFFFFFFA in alternative fast boot and do not need to be set in normal fast boot.
8. The software needs to configure Transfer Type Register to start the boot process. CMDINX, CMDTYP, RSPTYP, CICEN, CCCEN, AC12EN, BCEN, and DMAEN retain the default value. DPSEL bit is set to 1, DTDSEL is set to 1, and MSBSEL is set to 1. DMAEN is configured as 1 in the DMA mode. And, if BCEN is configured as 1, then configure blk no in Bock Attributes Register to the max value. And, if in the DDR fast boot mode, DDR_EN needs to be set to 1.
9. When step 8 is configured, boot process begins, the first VALUE1 block number data gets transferred. The software needs to poll the TC bit (bit1 in Interrupt Status Register) to determine first transfer is ended. Also, the software needs to polling the BGE bit (bit2 in Interrupt Status Register) to determine if the first transfer stops at the block gap.

10. When TC and BGE bits are set to 1, the software can analyze the first code of VALUE1 block, initializes the new memory device, if required, and sets the third pair of descriptors to define the start address and length of the remaining part of the boot code (VALUE3, the remain boot code block). Remember to set the last descriptor with END.
11. The software needs to configure the eMMC Boot Register (offset 0xc4) again. Set bit 6 to 1 (enable boot); and bit 5 to 0 (normal fast boot), to 1 (alternative boot); and bit 4 to select the ack mode or not. In the DMA mode, configure bit 7 to 1 for enabling the automatically stop at block gap feature. Also, configure bit31-bit16 to set the (BLK_CNT - (VALUE1+1+VALUE3)), that host stops at block gap when the uSDHC controller gets (VALUE1+1+VALUE3)) blocks from device totally include the blocks received in step 9. And need to configure bit 3-bit0 to select the ack timeout value according to the sd clk frequency. Note that the software does not need to configure the BLK_CNT again, because it is counted down automatically by the uSDHC controller.
12. The software needs to clear the TC and BGE bits and the software needs to clear SABGREQ (bit 16 in the Protocol control register) and set CREQ (bit17 in the Protocol control register) to 1 to resume the data transfer. Host transfers the VALUE2 and VALUE3 data to the destination that is set by descriptor.
13. The software needs to do poll BGE bit to determine if the fast boot is over.

Note:

- When ADMA boot flow starts, for uSDHC, it is like a normal ADMA read operation. So, set ADMA2 descriptor as the normal ADMA2 transfer.
- Need a few words length memory to keep descriptor.
- For the 1~2-word data in second descriptor setting, it is a useful data, so the software needs to deal the data because of the application case.

81.6 uSDHC memory map and register definition

81.6.1 uSDHC register descriptions

This section includes the module memory map and detailed descriptions of all registers.

See the table below for the register memory map of uSDHC. All these registers only support 32-bit accesses.

NOTE

The uSDHC registers are 32-bit wide and only support 32-bit access.

81.6.1.1 uSDHC memory map

usdhc base address: 404E_4000h

Offset	Register	Width (In bits)	Access	Reset value
0h	DMA System Address (DS_ADDR)	32	RW	0000_0000h
4h	Block Attributes (BLK_ATT)	32	RW	0001_0000h
8h	Command Argument (CMD_ARG)	32	RW	0000_0000h
Ch	Command Transfer Type (CMD_XFR_TYP)	32	RW	0000_0000h
10h	Command Response0 (CMD_RSP0)	32	R	0000_0000h
14h	Command Response1 (CMD_RSP1)	32	R	0000_0000h
18h	Command Response2 (CMD_RSP2)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1Ch	Command Response3 (CMD_RSP3)	32	R	0000_0000h
20h	Data Buffer Access Port (DATA_BUFF_ACC_PORT)	32	RW	0000_0000h
24h	Present State (PRES_STATE)	32	R	See section
28h	Protocol Control (PROT_CTRL)	32	RW	0880_0020h
2Ch	System Control (SYS_CTRL)	32	RW	0080_800Fh
30h	Interrupt Status (INT_STATUS)	32	RW	0000_0000h
34h	Interrupt Status Enable (INT_STATUS_EN)	32	RW	0000_0000h
38h	Interrupt Signal Enable (INT_SIGNAL_EN)	32	RW	0000_0000h
3Ch	Auto CMD12 Error Status (AUTOCMD12_ERR_STATUS)	32	R	0000_0000h
40h	Host Controller Capabilities (HOST_CTRL_CAP)	32	R	03F3_B404h
44h	Watermark Level (WTMK_LVL)	32	RW	0810_0810h
48h	Mixer Control (MIX_CTRL)	32	RW	8000_0000h
50h	Force Event (FORCE_EVENT)	32	RW	0000_0000h
54h	ADMA Error Status (ADMA_ERR_STATUS)	32	R	0000_0000h
58h	ADMA System Address (ADMA_SYS_ADDR)	32	RW	0000_0000h
60h	DLL (Delay Line) Control (DLL_CTRL)	32	RW	0000_0000h
64h	DLL Status (DLL_STATUS)	32	R	0000_0200h
C0h	Vendor Specific Register (VEND_SPEC)	32	RW	3000_7809h
C4h	eMMC Boot (MMC_BOOT)	32	RW	0000_0000h
C8h	Vendor Specific 2 Register (VEND_SPEC2)	32	RW	0001_9006h

81.6.1.2 DMA System Address (DS_ADDR)

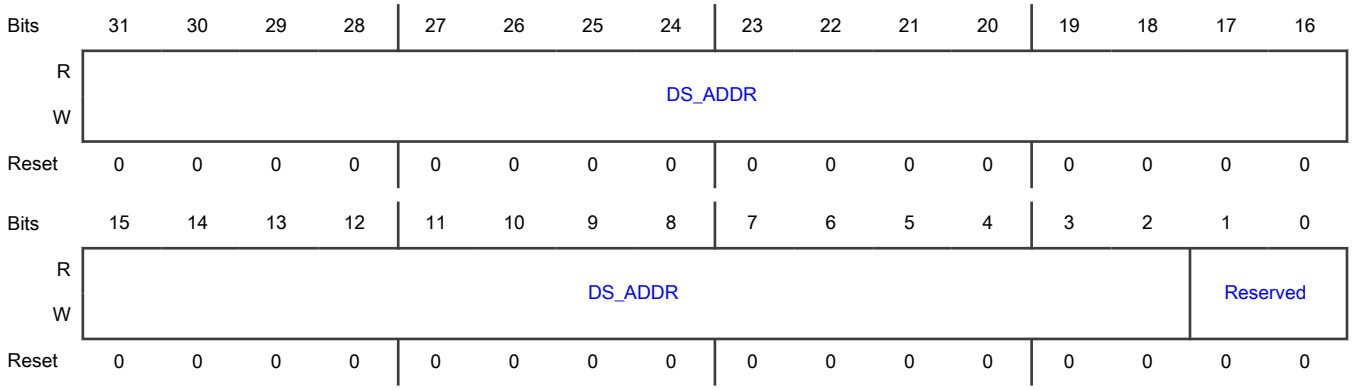
Offset

Register	Offset
DS_ADDR	0h

Function

This register contains the physical system memory address used for DMA transfers.

Diagram



Fields

Field	Function
31-2 DS_ADDR	DMA system address This field contains the memory address for a DMA transfer. Because the address must be word (4 bytes) aligned, the least 2 bits are reserved, always 0. When uSDHC stops a DMA transfer, this field points to the system address of the next contiguous data position. It can be accessed only when no transaction is executing (that is, after a transaction has stopped). Read operation during transfers may return an invalid value. The host driver initializes this field before starting a DMA transaction. After DMA has stopped, the system address of the next contiguous data position can be read from this field. This field is protected during a data transfer. When data lines are active, write to this field is ignored. The host driver waits until the DLA field in the Present State register is cleared, before writing to this field. The uSDHC internal DMA does not support a virtual memory system. It only supports continuous physical memory access. Also, because AHB burst limitations, if the burst must cross the 1 KB boundary, uSDHC automatically changes SEQ burst type to NSEQ. Because this field supports dynamic address reflecting, when TC field is set, it automatically alters the value of internal address counter, so the software cannot change this field when TC field is set. Such restriction is also listed in Software restrictions.
1-0 —	Reserved

81.6.1.3 Block Attributes (BLK_ATT)

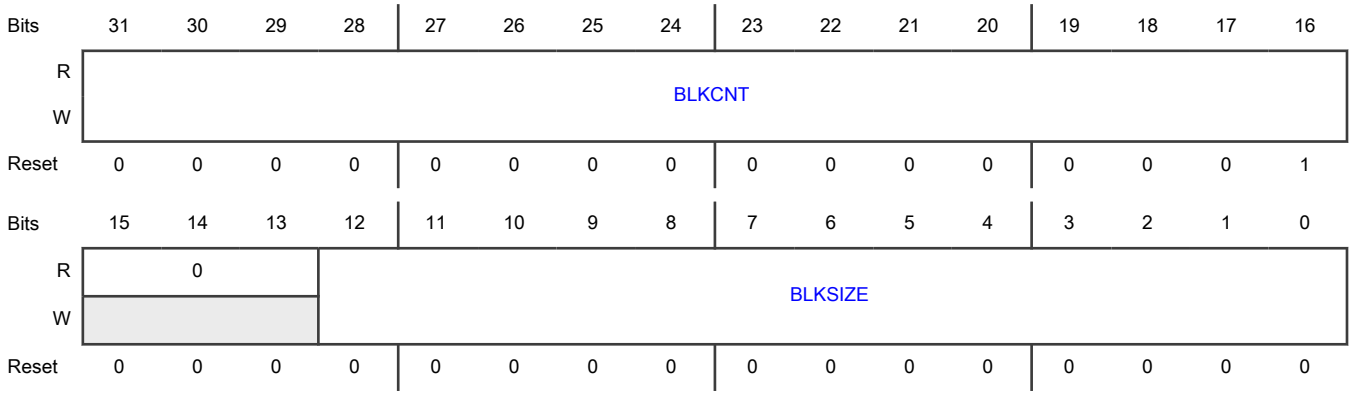
Offset

Register	Offset
BLK_ATT	4h

Function

This register is used to configure the number of data blocks and the number of bytes in each block.

Diagram



Fields

Field	Function
31-16 BLKCNT	Blocks count for current transfer This field is enabled when the Block Count Enable field in the Transfer Mode register is set to 1 and is valid only for multiple block transfers. For single block transfer, this field always reads as 1. The host driver sets this field to a value between 1 and the maximum block count. The uSDHC module decrements the block count after each block transfer and stops when the count reaches zero. Setting the block count to zero results in no data blocks being transferred. This field should be accessed only when no transaction is executing (that is, after transactions are stopped). During data transfer, read operations on this field may return an invalid value and write operations are ignored. When saving transfer content because of a Suspend command, the number of blocks yet to be transferred can be determined by reading this field. The reading of this field should be applied after transfer is paused by stop at block gap operation and before sending the command marked as suspend. This is because when the Suspend command is sent out, uSDHC treats the current transfer as aborted and change the BLKCNT field back to its original value instead of keeping the dynamical indicator of the remaining block count. When restoring transfer content prior to issuing a Resume command, the host driver restores the previously saved block count. **NOTE** Although the BLKCNT field is 0 after reset, the read of reset value is 0x1. This is because when MSBSEL field is indicating a single block transfer, the read value of BLKCNT is always 1. 0000_0000_0000_0000b - Stop count 0000_0000_0000_0001b - 1 block 0000_0000_0000_0010b - 2 blocks 1111_1111_1111_1111b - 65535 blocks
15-13 —	Reserved
12-0	Transfer block size

Table continues on the next page...

Table continued from the previous page...

Field	Function
BLKSIZE	This field specifies the block size for block data transfers. Values ranging from 1 byte up to the maximum buffer size can be set. It can be accessed only when no transaction is executing (that is, after a transaction has stopped). Read operations during transfers may return an invalid value, and write operations are ignored. 0_0000_0000_0000b - No data transfer 0_0000_0000_0001b - 1 byte 0_0000_0000_0010b - 2 bytes 0_0000_0000_0011b - 3 bytes 0_0000_0000_0100b - 4 bytes 0_0001_1111_1111b - 511 bytes 0_0010_0000_0000b - 512 bytes 0_1000_0000_0000b - 2048 bytes 1_0000_0000_0000b - 4096 bytes

81.6.1.4 Command Argument (CMD_ARG)

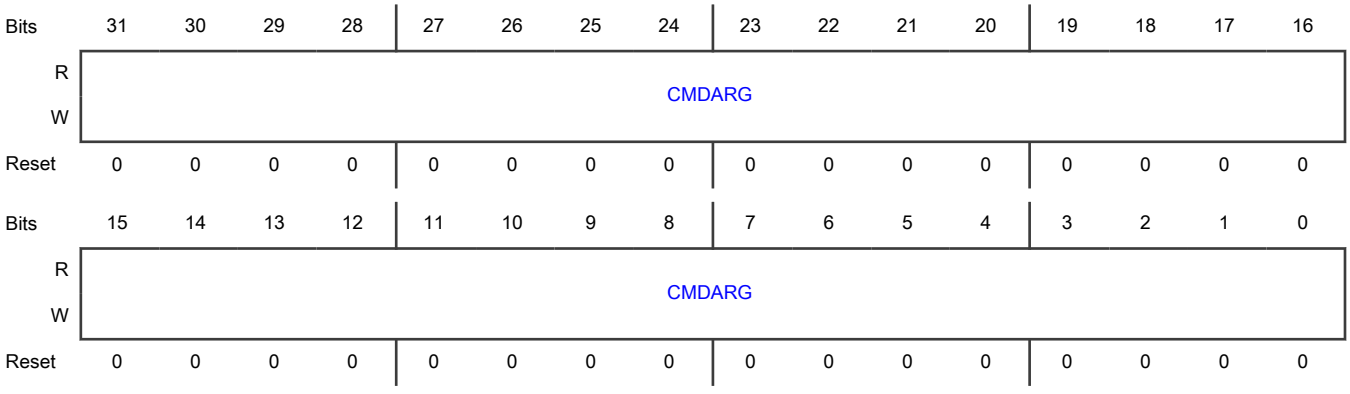
Offset

Register	Offset
CMD_ARG	8h

Function

This register contains the SD/eMMC command argument.

Diagram



Fields

Field	Function
31-0 CMDARG	Command argument The SD/eMMC command argument is specified as bits 39-8 of the command format in the SD or eMMC specification. This field is write protected when the Command Inhibit (CMD) field in the Present State register is set.

81.6.1.5 Command Transfer Type (CMD_XFR_TYP)

Offset

Register	Offset
CMD_XFR_TYP	Ch

Function

This register is used to control the operation of data transfers. The host driver sets this register before issuing a command followed by a data transfer or before issuing a Resume command. To prevent data loss, uSDHC prevents writing to the bits, which are involved in the data transfer of this register, when data transfer is active. These fields are DPSEL, MBSEL, DTDSEL, AC12EN, BCEN, and DMAEN.

The host driver checks the Command Inhibit DAT field (PRES_STATE[CDIHB]) and the Command Inhibit CMD field (PRES_STATE[CIHB]) in the Present State register before writing to this register. When the CDIHB field in the Present State register is set, any attempt to send a command with data by writing to this register is ignored; when the CIHB field is set, any write to this register is ignored.

On sending commands with data transfer involved, it is mandatory that the block size is non-zero. Block count must also be non-zero, or indicated as a single block transfer (bit 5 of this register is '0' when written), or block count is disabled (bit 1 of this register is '0' when written), otherwise uSDHC ignores the sending of this command and do nothing. For write command, with all above restrictions, it is also mandatory that the write protect switch is not active (PRES_STATE[WPSPL] field of Present State register is '1'); otherwise, uSDHC also ignores the command.

If the commands with data transfer do not receive the response in 64 clock cycles, that is, if response time-out happens, uSDHC treats the external device, does not accept the command, and aborts the data transfer. In this scenario, the driver should issue the command again to retry the transfer. It is also possible that for some reason the card responds to the command but uSDHC does not receive the response, and if it is internal DMA (either simple DMA or ADMA) read operation, the external system memory is over-written by the internal DMA with data sent back from the card.

The table below shows the summary of how register settings determine the type of data transfer.

Table 815. Transfer type register setting for various transfer types

Multi/single block select	Block count enable	Block count	Function
0	Do not care	Do not care	Single transfer
1	0	Do not care	Infinite transfer
1	1	Positive number	Multiple transfer
1	1	Zero	No data transfer

The table below shows the relationship between the Command Index Check Enable and the Command CRC Check Enable, regarding the Response Type bits as well as the name of the response type.

Table 816. Relationship between parameters and the name of the response type

Response type	Index check enable	CRC check enable	Name of response type
00	0	0	No response
01	0	1	R2
10	0	0	R3, R4
10	1	1	R1, R5, R6
11	1	1	R1b,R5b

- In the SDIO specification, response type notation for R5b is not defined. R5 includes R5b in the SDIO specification, but R5b is defined in this specification to specify that uSDHC checks the busy status after receiving a response. For example, usually CMD52 is used with R5, but the I/O abort command is used with R5b.
- The CRC fields for R3 and R4 are expected to be all 1 bits. The CRC check is disabled for these response types.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				CMDINX				CMDTYP		DPSEL	CICEN	CCCEN	Reserved	RSPTYP	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								AC23EN	NIBBLE_...	MSBSEL	DTDSEL	DDR_EN	AC12EN	BCEN	DMAEN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29-24 CMDINX	Command index These fields are set to the command number that is specified in bits 45-40 of the command-format in the SD Memory Card Physical Layer Specification and SDIO Specification.
23-22 CMDTYP	Command type There are three types of special commands: Suspend, Resume, and Abort. These fields are set to 00b for all other commands.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• Suspend command: If the Suspend command succeeds, uSDHC assumes that the card bus has been released and that it is possible to issue the next command that uses the DATA line. Because uSDHC does not monitor the content of command response, it does not know if the Suspend command succeeded or not. It is the host driver's responsibility to check the status of the Suspend command and send another command marked as Suspend to inform uSDHC that a Suspend command was successfully issued. See Suspend Resume for more details. After the end bit of command is sent, uSDHC deasserts Read Wait for read transactions and stops checking busy for write transactions. In a 4-bit mode, the interrupt cycle starts. If the Suspend command fails, uSDHC maintains its current state, and the host driver restarts the transfer by setting the Continue Request field in the Protocol Control register. • Resume command: The host driver re-starts the data transfer by restoring the registers saved before sending the Suspend command and then sends the Resume command. The uSDHC module checks for a pending busy state before starting write transfers. • Abort command: If this command is set when executing a read transfer, uSDHC stops reads to the buffer. If this command is set when executing a write transfer, uSDHC stops driving the DATA line. After issuing the Abort command, the host driver should issue a software reset (Abort Transaction). 00b - Normal other commands 01b - Suspend CMD52 for writing bus suspend in CCCR 10b - Resume CMD52 for writing function select in CCCR 11b - Abort CMD12, CMD52 for writing I/O Abort in CCCR
21 DPSEL	Data present select This field is set to 1 to indicate that data is present and is transferred using the DATA line. It is set to 0 for the following: • Commands using only the CMD line (for example, CMD52) • Commands with no data transfer, but using the busy signal on DATA0 line (R1b or R5b (for example, CMD38)) **NOTE** In resume command, this field is set, and other bits in this register is set the same as when the transfer was initially launched. When the write protect switch is on, (that is, the WPSPL field is active as '0'), any command with a write operation ignored. When this field is set, while the DTDSEL field is 0, writes to the register Transfer Type are ignored. 0b - No data present 1b - Data present
20 CICEN	Command index check enable If this field is set to 1, uSDHC checks the Index field in the response to see if it has the same value as the command index. If it is not, it is reported as a Command Index Error. If this field is set to 0, the Index field is not checked. 0b - Disable command index check 1b - Enables command index check

Table continues on the next page...

Table continued from the previous page...

Field	Function
19 CCCEN	Command CRC check enable If this field is set to 1, uSDHC checks the CRC field in the response. If an error is detected, it is reported as a Command CRC Error. If this field is set to 0, the CRC field is not checked. The number of bits checked by the CRC field value changes according to the length of the response. See RSPTYP[1:0] and Command Transfer Type (CMD_XFR_TYP). 0b - Disables command CRC check 1b - Enables command CRC check
18 —	Reserved
17-16 RSPTYP	Response type select 00b - No response 01b - Response length 136 10b - Response length 48 11b - Response length 48, check busy after response
15-8 —	Reserved
7 AC23EN	AC23EN This field is read when VEND_SPEC[CMD_BYTE_EN] is enabled; otherwise, this field is tied to '0'. When this field is set to 1, the host controller issues a CMD23 automatically before issuing a command specified in the Command Register. 0b - Disable 1b - Enable
6 NIBBLE_POS	NIBBLE_POS This field indicates the nibble position in the DDR 4-bit mode. This field is read/write when VEND_SPEC[CMD_BYTE_EN] is enabled; otherwise, this field is read-only. 0- the sequence is 'odd high nibble -> even high nibble -> odd low nibble -> even low nibble'; 1- the sequence is 'odd high nibble -> odd low nibble -> even high nibble -> even low nibble'. 0b - Disable 1b - Enable
5 MSBSEL	MSBSEL This field enables multiple block DATA line data transfers. This field is read/write when VEND_SPEC[CMD_BYTE_EN] is enabled; otherwise, this field is read-only. For any other commands, this field can be set to 0. If this field is 0, it is not necessary to set the Block Count register. See Command Transfer Type (CMD_XFR_TYP). 0b - Disable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Enable
4 DTDSEL	DTDSEL This field defines the direction of DATA line data transfers. This field is read/write when VEND_SPEC[CMD_BYTE_EN] is enabled; otherwise, this field is read-only. The field is set to 1 by the host driver to transfer data from the SD card to uSDHC and is set to 0 for all other commands. 0b - Disable 1b - Enable
3 DDR_EN	DDR_EN Dual data rate mode selection. This field is read/write when VEND_SPEC[CMD_BYTE_EN] is enabled; otherwise, this field is read-only. 0b - Disable 1b - Enable
2 AC12EN	AC12EN Multiple block transfers for memory require a CMD12 to stop the transaction. This field is read/write when VEND_SPEC[CMD_BYTE_EN] is enabled; otherwise, this field is read-only. When this field is set to 1, uSDHC issues a CMD12 automatically when the last block transfer has completed. The host driver is not set this field to issue commands that do not require CMD12 to stop a multiple block data transfer. In particular, secure commands defined in File Security Specification (see reference list) do not require CMD12. In single block transfer, uSDHC ignores this field no matter it is set or not. 0b - Disable 1b - Enable
1 BCEN	BCEN This field is used to enable the Block Count register, which is only relevant for multiple block transfers. This field is read/write when VEND_SPEC[CMD_BYTE_EN] is enabled; otherwise, this field is read-only. When this field is 0, the internal counter for block is disabled, which is useful in executing an infinite transfer. 0b - Disable 1b - Enable
0 DMAEN	DMAEN This field enables DMA functionality. This field is read/write when VEND_SPEC[CMD_BYTE_EN] is enabled; otherwise, this field is read-only. If this field is set to 1, a DMA operation begins when the host driver sets the DPSEL field of this register. Whether the simple DMA or the advanced DMA is active depends on the DMA Select field of the Protocol Control register. 0b - Disable 1b - Enable

81.6.1.6 Command Response0 (CMD_RSP0)

Offset

Register	Offset
CMD_RSP0	10h

Function

This register is used to store part 0 of the response bits from the card.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CMDRSP0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CMDRSP0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 CMDRSP0	Command response 0 See Command Response3 (CMD_RSP3) for the mapping of command responses from the SD bus to this field for each response type.

81.6.1.7 Command Response1 (CMD_RSP1)

Offset

Register	Offset
CMD_RSP1	14h

Function

This register is used to store part 1 of the response bits from the card.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CMDRSP1															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CMDRSP1															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 CMDRSP1	Command response 1 See Command Response3 (CMD_RSP3) for the mapping of command responses from the SD bus to this field for each response type.

81.6.1.8 Command Response2 (CMD_RSP2)

Offset

Register	Offset
CMD_RSP2	18h

Function

This register is used to store part 2 of the response bits from the card.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CMDRSP2															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CMDRSP2															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 CMDRSP2	Command response 2 See Command Response3 (CMD_RSP3) for the mapping of command responses from the SD bus to this field for each response type.

81.6.1.9 Command Response3 (CMD_RSP3)

Offset

Register	Offset
CMD_RSP3	1Ch

Function

This register is used to store part 3 of the response bits from the card.

The table below describes the mapping of command responses from the SD bus to Command Response registers for each response type. In this table, R[ ] refers to a bit range within the response data as transmitted on the SD bus.

Table 817. Response bit definition for each response type

Response type	Meaning of response	Response field	Response register
R1,R1b (normal response)	Card status	R[39:8]	CMDRSP0
R1b (auto CMD12 response)	Card status for auto CMD12	R[39:8]	CMDRSP3
R2 (CID, CSD register)	CID/CSD register [127:8]	R[127:8]	{CMDRSP3[23:0], CMDRSP2, CMDRSP1, CMDRSP0}
R3 (OCR register)	OCR register for memory	R[39:8]	CMDRSP0
R4 (OCR register)	OCR register for I/O etc.	R[39:8]	CMDRSP0
R5, R5b	SDIO response	R[39:8]	CMDRSP0
R6 (publish RCA)	New published RCA[31:16] and card status[15:0]	R[39:9]	CMDRSP0

This table shows that most responses with a length of 48 (R[47:0]) have 32-bits of the response data (R[39:8]) stored in the CMDRSP0 register. Responses of type R1b (Auto CMD12 responses) have response data bits (R[39:8]) stored in the CMDRSP3 register. Responses with length 136 (R[135:0]) have 120-bits of the response data (R[127:8]) stored in the CMDRSP0, 1, 2, and 3 registers.

To be able to read the response status efficiently, uSDHC only stores part of the response data in the Command Response registers. This enables the host driver to efficiently read 32-bits of response data in one read cycle on a 32-bit bus system. Parts of the response, the Index field and the CRC, are checked by uSDHC (as specified by the Command Index Check Enable and the

Command CRC Check Enable bits in the Transfer Type register) and generate an error interrupt if any error is detected. The bit range for the CRC check depends on the response length. If the response length is 48, uSDHC checks R[47:1], and if the response length is 136 the uSDHC checks R[119:1].

Because uSDHC may have a multiple block data transfer executing concurrently with a CMD_wo_DAT command, uSDHC stores the Auto CMD12 response in the CMDRSP3 register. The CMD_wo_DAT response is stored in CMDRSP0. This allows uSDHC to avoid overwriting the Auto CMD12 response with the CMD_wo_DAT and vice versa. When uSDHC modifies part of the Command Response registers, as shown in the table above, it preserves the unmodified bits.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CMDRSP3															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CMDRSP3															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 CMDRSP3	Command response 3 See Command Response3 (CMD_RSP3) for the mapping of command responses from the SD bus to this field for each response type.

81.6.1.10 Data Buffer Access Port (DATA_BUFF_ACC_PORT)

Offset

Register	Offset
DATA_BUFF_ACC_PORT	20h

Function

The Buffer Data Port register is for 32-bit data access by the Arm platform or the external DMA. When the internal DMA is enabled, any write to this field is ignored, and any read from this field always yields 0s.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DATCONT															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DATCONT															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 DATCONT	Data content This field is used to access the internal buffer.

81.6.1.11 Present State (PRES_STATE)

Offset

Register	Offset
PRES_STATE	24h

Function

The host driver can get status of uSDHC from this 32-bit read only register.

The host driver can issue CMD0, CMD12, CMD13 (for memory) and CMD52 (for SDIO) when the DATA lines are busy during a data transfer. These commands can be issued when Command Inhibit (CMD) is set to zero. Other commands are issued when Command Inhibit (DATA) is set to zero. Possible changes to the SD Physical Specification may add other commands to this list in the future.

NOTE

The reset value of Present State register depends on board connectivity.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DLSL								CLSL	0			WPSPL	Reserved	0	CINST
W																
Reset	u	u	u	u	u	u	u	u	u	0	0	0	u	u	0	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0			0	BREN	BWEN	RTA	WTA	Reserved				SDSTB	DLA	CDIHB	CIHB
W																
Reset	1	0	0	0	0	0	0	0	1	0	0	0	u	0	0	0

Fields

Field	Function
31-24 DLSL	DATA[7:0] line signal level This field is used to check the DATA line level to recover from errors, and for debugging. This is especially useful in detecting the busy signal level from DATA0. The reset value is affected by the external pull-up / pull-down resistors. By default, the read value of this field after reset is 8'b11110111, when DATA3 is pulled down and the other lines are pulled up. 0000_0000b - Data 0 line signal level 0000_0001b - Data 1 line signal level 0000_0010b - Data 2 line signal level 0000_0011b - Data 3 line signal level 0000_0100b - Data 4 line signal level 0000_0101b - Data 5 line signal level 0000_0110b - Data 6 line signal level 0000_0111b - Data 7 line signal level
23 CLSL	CMD line signal level This field is used to check the CMD line level to recover from errors, and for debugging. The reset value is affected by the external pull-up / pull-down resistor, by default, the read value of this field after reset is 1'b1, when the command line is pulled up.
22-20 —	Reserved
19 WPSPL	Write protect switch pin level The Write Protect switch is supported for memory and combo cards. This field reflects the inverted value of the WP pin of the card socket. A software reset does not affect this field. The reset value is affected by the external write protect switch. If the WP pin is not used, it should be tied low, so that the reset value of this field is high and write is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Write protected (WP = 1) 1b - Write enabled (WP = 0)
18 —	Reserved
17 —	Reserved
16 CINST	Card inserted This field indicates whether a card has been inserted. The uSDHC module debounces this signal so that the host driver does not need to wait for it to stabilize. Changing from a 0 to 1 generates a Card Insertion interrupt in the Interrupt Status register. Changing from a 1 to 0 generates a Card Removal interrupt in the Interrupt Status register. A write to the Force Event Register does not affect this field. The Software Reset For All in the System Control register does not affect this field. A software reset does not affect this field. 0b - Power on reset or no card 1b - Card inserted
15-13 —	Reserved
12 —	Reserved
11 BREN	Buffer read enable This status field is used for non-DMA read transfers. The uSDHC module implements an internal buffer to transfer data efficiently. This read only flag indicates that valid data exists in the host side buffer. If this field is high, valid data greater than the watermark level exist in the buffer. A change of this field from 1 to 0 occurs when some reads from the buffer (read DATPORT (Base + 0x20)) are made and the buffer hasn't valid data greater than the watermark level. A change of this field from 0 to1 occurs when there is enough valid data ready in the buffer and the Buffer Read Ready interrupt has been generated and enabled. 0b - Read disable 1b - Read enable
10 BWEN	Buffer write enable This status field is used for non-DMA write transfers. The uSDHC module implements an internal buffer to transfer data efficiently. This read only flag indicates if space is available for write data. If this field is 1, valid data greater than the watermark level can be written to the buffer. A change of this field from 1 to 0 occurs when some writes to the buffer (write DATPORT (Base + 0x20)) are made and the buffer hasn't valid space greater than the watermark level. A change of this field from 0 to 1 occurs when the buffer can hold valid data greater than the write watermark level and the Buffer Write Ready interrupt is generated and enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Write disable 1b - Write enable
9 RTA	Read transfer active This status field is used for detecting completion of a read transfer. This field is set for either of the following conditions: • After the end field of the read command • When writing a 1 to the Continue Request field in the Protocol Control register to restart a read transfer A transfer complete interrupt is generated when this field changes to 0. This field is cleared for either of the following conditions: • When the last data block as specified by block length is transferred to the System, that is, all data are read away from uSDHC internal buffer. • When all valid data blocks have been transferred from uSDHC internal buffer to the System and no current block transfers are being sent because of the Stop At Block Gap Request being set to 1. 0b - No valid data 1b - Transferring data
8 WTA	Write transfer active This status field indicates a write transfer is active. If this field is 0, it means no valid write data exists in uSDHC. This field is set in either of the following cases: • After the end field of the write command • When writing 1 to the Continue Request field in the Protocol Control register to restart a write transfer This field is cleared in either of the following cases: • After getting the CRC status of the last data block as specified by the transfer count (Single and Multiple) • After getting the CRC status of any block where data transmission is about to be stopped by a Stop At Block Gap Request During a write transaction, a Block Gap Event interrupt is generated when this field is changed to 0, as result of the Stop At Block Gap Request being set. This status is useful for the host driver in determining when to issue commands during Write Busy state. 0b - No valid data 1b - Transferring data
7-4 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
3 SDSTB	SD clock stable This status field indicates that the internal card clock is stable. This field is for the host driver to poll clock status when changing the clock frequency. It is recommended to clear FRC_SDCLK_ON field in System Control register to remove glitches on the card clock when the frequency is changing. Before changing clock divisor value (SDCLKFS or DVS), host driver should make sure the SDSTB field is high. 0b - Clock is changing frequency and not stable. 1b - Clock is stable.
2 DLA	Data line active This status field indicates whether one of the DATA lines on the SD bus is in use. In the case of read transactions: This status indicates if a read transfer is executing on the SD bus. Changes in this value from 1 to 0, between data blocks, generates a Block Gap Event interrupt in the Interrupt Status register. This field is set in either of the following cases: • After the end field of the read command • When writing a 1 to the Continue Request field in the Protocol Control register to restart a read transfer This field is cleared in either of the following cases: • When the end field of the last data block is sent from the SD bus to uSDHC. • When the Read Wait state is stopped by a Suspend command and the DATA2 line is released. The uSDHC module waits at the next block gap by driving Read Wait at the start of the interrupt cycle. If the Read Wait signal is already driven (data buffer cannot receive data), uSDHC can wait for a current block gap by continuing to drive the Read Wait signal. It is necessary to support Read Wait to use the suspend / resume function. This field remains 1 during Read Wait. In the case of write transactions: This status indicates that a write transfer is executing on the SD bus. Changes in this value from 1 to 0 generate a Transfer Complete interrupt in the Interrupt Status register. This field is set in either of the following cases: • After the end field of the write command • When writing to 1 to the Continue Request field in the Protocol Control register to continue a write transfer This field is cleared in either of the following cases: • When the SD card releases Write Busy of the last data block, uSDHC also detects if the output is not busy. If the SD card does not drive the busy signal after the CRC status is received, uSDHC assumes the card drive "Not Busy". • When the SD card releases write busy, prior to waiting for write transfer, and because of a Stop At Block Gap Request.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	In the case of command with busy pending: This status indicates that a busy state follows the command and the data line is in use. This field is cleared when the DATA0 line is released. 0b - DATA line inactive 1b - DATA line active
1 CDIHB	Command Inhibit Data (DATA) This status field is generated if either the DAT Line Active or the Read Transfer Active is set to 1. If this field is 0, it indicates that uSDHC can issue the next SD / eMMC Command. Commands with a busy signal belong to Command Inhibit (DATA) (for example. R1b, R5b type). Changing from 1 to 0 generates a Transfer Complete interrupt in the Interrupt Status register. **NOTE** The SD host driver can save registers for a suspend transaction after this field has changed from 1 to 0. 0b - Can issue command that uses the DATA line 1b - Cannot issue command that uses the DATA line
0 CIHB	Command inhibit (CMD) If this status bit is 0, it indicates that the CMD line is not in use and uSDHC can issue a SD / eMMC command using the CMD line. This field is set also immediately after the Transfer Type register is written. This field is cleared when the command response is received. Even if the Command Inhibit (DATA) is set to 1, commands using only the CMD line can be issued if this field is 0. Changing from 1 to 0 generates a Command Complete interrupt in the Interrupt Status register. If uSDHC cannot issue the command because of a command conflict error (see Command CRC Error) or because of a Command Not Issued By Auto CMD12 Error, this field remains 1 and the Command Complete is not set. The Status of issuing an auto CMD12 does not show on this field. 0b - Can issue command using only CMD line 1b - Cannot issue command

81.6.1.12 Protocol Control (PROT_CTRL)

Offset

Register	Offset
PROT_CTRL	28h

Function

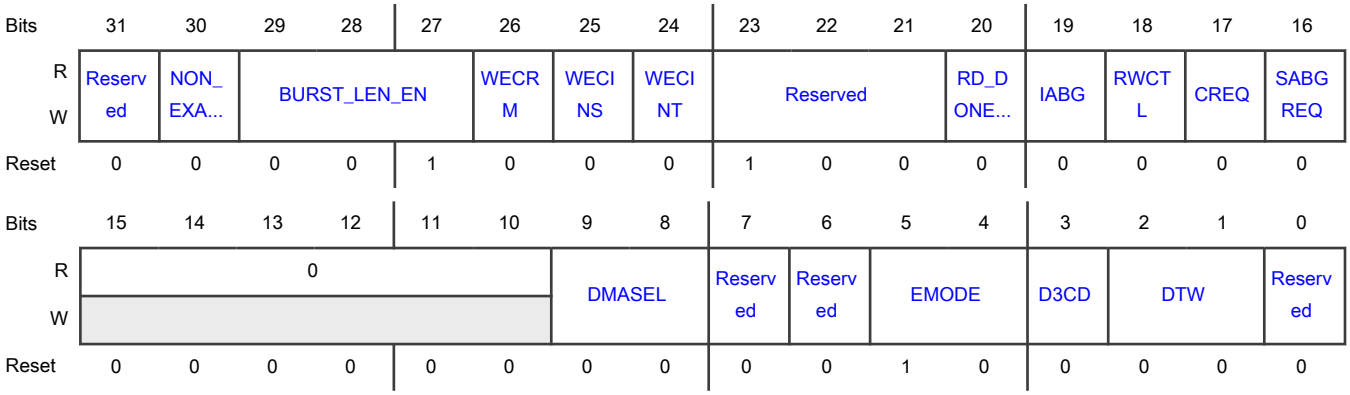
This register controls three cases to restart the transfer after stop at the block gap. Which case is appropriate depends on whether uSDHC issues a Suspend command or the SD card accepts the Suspend command.

- If the host driver does not issue a Suspend command, the Continue request is used to restart the transfer.

- If the host driver issues a Suspend command and the SD card accepts it, a Resume command is used to restart the transfer.
- If the host driver issues a Suspend command and the SD card does not accept it, the Continue request is used to restart the transfer.

Any time stop at block gap request stops the data transfer, the host driver waits for a Transfer Complete (in the Interrupt Status register), before attempting to restart the transfer. When restarting the data transfer by Continue Request, the host driver clears the Stop At Block Gap Request before or simultaneously.

Diagram



Fields

Field	Function
31 —	Reserved Always write as 0
30 NON_EXACT_BLK_RD	Non-exact block read Current block read is non-exact block read. It is only used for SDIO. 0b - The block read is exact block read. Host driver does not need to issue abort command to terminate this multi-block read. 1b - The block read is non-exact block read. Host driver needs to issue abort command to terminate this multi-block read.
29-27 BURST_LEN_EN	BURST length enable for INCR, INCR4 / INCR8 / INCR16, INCR4-WRAP / INCR8-WRAP / INCR16-WRAP This field is used to enable or disable the burst length for the external AHB2AXI bridge. It is useful especially for INCR transfer because without burst length indicator, the AHB2AXI bridge does not know the burst length in advance. Without burst length indicator, AHB INCR transfers can only be converted to SINGLEs on the AXI side. 1xxb - Burst length is enabled for INCR4-WRAP / INCR8-WRAP / INCR16-WRAP. x1xb - Burst length is enabled for INCR4 / INCR8 / INCR16. xx1b - Burst length is enabled for INCR.
26 WECRM	Wakeup event enable on SD card removal

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field enables a wakeup event, via a card removal, in the Interrupt Status register. FN_WUS (Wakeup Support) in CIS does not affect this field. When this field is set, the Card Removal Status and uSDHC interrupt can be asserted without CLK toggling. When the wakeup feature is not enabled, the CLK must be active to assert the Card Removal Status and uSDHC interrupt. 0b - Disables wakeup event enable on SD card removal 1b - Enables wakeup event enable on SD card removal
25 WECINS	Wakeup event enable on SD card insertion This field enables a wakeup event, via a card insertion, in the Interrupt Status register. FN_WUS (Wakeup Support) in CIS does not affect this field. When this field is set, the Card Insertion Status and uSDHC interrupt can be asserted without CLK toggling. When the wakeup feature is not enabled, the CLK must be active to assert the Card Insertion Status and uSDHC interrupt. 0b - Disable wakeup event enable on SD card insertion 1b - Enable wakeup event enable on SD card insertion
24 WECINT	Wakeup event enable on card interrupt This field enables a wakeup event, via a card interrupt, in the Interrupt Status register. This field can be set to 1 if FN_WUS (Wakeup Support) in CIS is set to 1. When this field is set, the Card Interrupt Status and uSDHC interrupt can be asserted without CLK toggling. When the wakeup feature is not enabled, the CLK must be active to assert the Card Interrupt Status and uSDHC interrupt. 0b - Disables wakeup event enable on card interrupt 1b - Enables wakeup event enable on card interrupt
23-21 —	Reserved Always write as 3'b100
20 RD_DONE_NO_8CLK	Read performed number 8 clock According to the SD/eMMC spec, for read data transaction, 8 clocks are needed after the end field of the last data block. So, by default(RD_DONE_NO_8CLK=0), eight clocks are active after the end field of the last read data transaction. However, these 8 clocks should not be active if user wants to use stop at block gap (include the auto stop at block gap in boot mode) feature for read and the RWCTL field (bit18) is not enabled. In this case, software should set RD_DONE_NO_8CLK to avoid these 8 clocks. Otherwise, the device might send extra data to uSDHC while uSDHC ignores these data. In a summary, this field should be set only if the use case needs to use stop at block gap feature while the device can't support the read wait feature.
19 IABG	Interrupt at block gap This field is valid only in 4-bit mode, of SDIO, and selects a sample point in the interrupt cycle. Setting to 1 enables interrupt detection at the block gap for a multiple block transfer. Setting to 0 disables interrupt detection during a multiple block transfer. If SDIO cannot signal an interrupt during a multiple block transfer, this field should be set to 0 to avoid an inadvertent interrupt. When the host driver detects an SDIO insertion, it sets this field according to the CCCR of the card.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disables interrupt at block gap 1b - Enables interrupt at block gap
18 RWCTL	Read wait control The read wait function provided by this field is optional for SDIO. If the card supports read wait, set this field to enable use of the read wait protocol to stop read data using the DATA2 line. Otherwise, uSDHC has to stop the SD clock to hold read data, which restricts commands generation. When the host driver detects an SDIO insertion, it sets this field according to the CCCR of the card. If the card does not support read wait, this field should never be set to 1; otherwise, DATA line conflicts might occur. If this field is set to 0, stop at block gap during read operation is also supported, but uSDHC stops the SD clock to pause reading operation. 0b - Disables read wait control and stop SD clock at block gap when SABGREQ field is set 1b - Enables read wait control and assert read wait without stopping SD clock at block gap when SABGREQ field is set
17 CREQ	Continue request This field is used to restart a transaction which was stopped using the stop at block gap request. When a suspend operation is not accepted by the card, it is also by setting this field to restart the paused transfer. To cancel stop at the block gap, set stop at block gap request to 0 and set this field to 1 to restart the transfer. The uSDHC module automatically clears this field, therefore it is not necessary for the host driver to set this field to 0. If both stop at block gap request and this field are 1, the continue request is ignored. 0b - No effect 1b - Restart
16 SABGREQ	Stop at block gap request This field is used to stop executing a transaction at the next block gap for both DMA and non-DMA transfers. Until the transfer complete is set to 1, indicating a transfer completion, the host driver leaves this field set to 1. Clearing both the stop at block gap request and continue request does not cause the transaction to restart. Read Wait is used to stop the read transaction at the block gap. The uSDHC module supports the stop at block gap request for write transfers, but for read transfers it requires that SDIO support read wait. Therefore, the host driver does not set this field during read transfers unless SDIO supports Read Wait and has set the read wait control to 1; otherwise, uSDHC stops the SD bus clock to pause the read operation during block gap. In the case of write transfers in which the host driver writes data to the Data Port register, the host driver sets this field after all block data is written. If this field is set to 1, the host driver does not write data to the Data Port register after a block is sent. Once this field is set, the host driver does not clear this field before the Transfer Complete field in Interrupt Status register is set, otherwise uSDHC's behavior is undefined. This field effects read transfer active, write transfer active, DATA Line Active and Command Inhibit (DATA) in the Present State register. 0b - Transfer 1b - Stop
15-10 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
9-8 DMASEL	DMA select This field is valid while DMA (SDMA or ADMA) is enabled and selects the DMA operation. 00b - No DMA or simple DMA is selected. 01b - ADMA1 is selected. 10b - ADMA2 is selected. 11b - Reserved
7 —	Reserved
6 —	Reserved
5-4 EMODE	Endian mode This field supports all three endian modes in data transfer. See Data buffer for more details. 00b - Big endian mode 01b - Half word big endian mode 10b - Little endian mode 11b - Reserved
3 D3CD	DATA3 as card detection pin If this field is set, DATA3 should be pulled down to act as a card detection pin. Be cautious when using this feature, because DATA3 is also a chip-select for the SPI mode. A pull-down on this pin and CMD0 might set the card into the SPI mode, which uSDHC does not support. 0b - DATA3 does not monitor card insertion 1b - DATA3 as card detection pin
2-1 DTW	Data transfer width This field selects the data width of the SD bus for a data transfer. The host driver sets it to match the data width of the card. Possible data transfer width is 1-bit, 4-bits or 8-bits. 00b - 1-bit mode 01b - 4-bit mode 10b - 8-bit mode 11b - Reserved
0 —	Reserved

81.6.1.13 System Control (SYS_CTRL)

Offset

Register	Offset
SYS_CTRL	2Ch

Function

This register provides control of the system. See detail in the field description.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				INITA	RSTD	RSTC	RSTA	IPP_RST...	RST_FIFO	Reserved	0	DTOCV			
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SDCLKFS								DVS				Reserved			
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1

Fields

Field	Function
31-28 —	Reserved
27 INITA	Initialization active When this field is set, 80 SD-clocks are sent to the card. After the 80 clocks are sent, this field is self cleared. This field is very useful during the card power-up period when 74 SD-clocks are needed and the clock auto gating feature is enabled. Writing 1 to this bit when this field is already 1 has no effect. Writing 0 to this field at any time has no effect. When either of the CIHB and CDIHB fields in the Present State register are set, writing 1 to this field is ignored (that is, when command line or data lines are active, write to this field is not allowed). On the other-hand, when this field is set, that is, during initialization active period, it is allowed to issue command, and the command bit stream appears on the CMD pad after all 80 clock cycles are done. So, when this command ends, the driver can make sure the 80 clock cycles are sent out. This is very useful when the driver needs to send 80 cycles to the card and does not want to wait till this field is self cleared.
26 RSTD	Software reset for data line Only part of the data circuit is reset. DMA circuit is also reset. After this field is set, the software waits for self-clear. The following registers and bits are cleared by this field:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• Data Port register — Buffer is cleared and initialized • Present State register — Buffer read enable — Buffer write enable — Read transfer active — Write transfer active — DATA line active — Command Inhibit (DATA) • Protocol Control register — Continue request • Interrupt Status register — Buffer read ready — Buffer write ready — DMA interrupt — Block gap event — Transfer complete **NOTE** When reset, the software must make sure there is no incomplete data transferring. If there is data transfer going on, the software needs to wait TC or DC INT_STATUS register is set. 0b - No reset 1b - Reset
25 RSTC	Software reset for CMD line Only part of the command circuit is reset. After this field is set, the software waits for self-clear. The following registers and bits are cleared by this field: • Present State Register — Command Inhibit (CMD) • Interrupt Status Register — Command Complete 0b - No reset 1b - Reset
24 RSTA	Software reset for all

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This reset affects the entire host controller except for the card detection circuit. RSTA resets all the registers that can be reset by RSTC/RSTD. During its initialization, the host driver is set this field to 1 to reset uSDHC. The uSDHC module resets this field to 0 when the capabilities registers are valid and the host driver can read them. Additional use of Software Reset For All does not affect the value of the capabilities registers. After this field is set, it is recommended that the host driver reset the external card and re-initialize it. After this field is set, the software should wait for self-clear. **NOTE** When reset, the software must make sure there is no incomplete data transferring. If there is data transfer going on, the software needs to wait TC or DC INT_STATUS register is set. 0b - No reset 1b - Reset
23 IPP_RST_N	Hardware reset This field's value is output to card through pad directly to hardware reset pin of the card if the card supports this feature.
22 RST_FIFO	Reset the async FIFO Reset the async FIFO between card interface and the internal logic. After this field is set, the software waits for self-clear.
21 —	Reserved
20 —	Reserved
19-16 DTOCV	Data timeout counter value This value determines the interval by which DAT line timeouts are detected. See the Data Timeout Error field in the Interrupt Status register for information on factors that dictate time-out generation. Time-out clock frequency is generated by dividing the base clock SDCLK value by this value. The host driver can clear the Data Timeout Error Status Enable (in the Interrupt Status Enable register) to prevent inadvertent time-out events. 0000b - SDCLK x 2^{14} 0001b - SDCLK x 2^{15} 0010b - SDCLK x 2^{16} 0011b - SDCLK x 2^{17} 0100b - SDCLK x 2^{18} 0101b - SDCLK x 2^{19} 0110b - SDCLK x 2^{20} 0111b - SDCLK x 2^{21}

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1000b - SDCLK x 2^{22} 1001b - SDCLK x 2^{23} 1010b - SDCLK x 2^{24} 1011b - SDCLK x 2^{25} 1100b - SDCLK x 2^{26} 1101b - SDCLK x 2^{27} 1110b - SDCLK x 2^{28} 1111b - SDCLK x 2^{29}
15-8 SDCLKFS	SDCLK frequency select This field is used to select the frequency of the SDCLK pin. The frequency is not programmed directly, rather this field holds the prescaler (of this register) and divisor (next register) of the Base Clock Frequency register. In Single Data Rate mode (DDR_EN field of MIXERCTRL is '0') Only the following settings are allowed: 80h) Base clock divided by 256 40h) Base clock divided by 128 20h) Base clock divided by 64 10h) Base clock divided by 32 08h) Base clock divided by 16 04h) Base clock divided by 8 02h) Base clock divided by 4 01h) Base clock divided by 2 00h) Base clock divided by 1 While in Dual Data Rate mode (DDR_EN field of MIXERCTRL is '1') Only the following settings are allowed: 80h) Base clock divided by 512 40h) Base clock divided by 256 20h) Base clock divided by 128 10h) Base clock divided by 64 08h) Base clock divided by 32 04h) Base clock divided by 16 02h) Base clock divided by 8 01h) Base clock divided by 4 00h) Base clock divided by 2

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When the software changes the DDR_EN field, SDCLKFS might need to be changed also. In Single Data Rate mode, setting 00h bypasses the frequency prescaler of the SD clock. Multiple bits must not be set, or the behavior of this prescaler is undefined. The two default divider values can be calculated by the frequency of ipg_perclk and the following Divisor bits. The frequency of SDCLK is set by the following formula: Clock Frequency = (Base Clock) / (prescaler x divisor) For example, in Single Data Rate mode, if the Base Clock Frequency is 96 MHz, and the target frequency is 25 MHz, then choosing the prescaler value of 01h and divisor value of 1h yields 24 MHz, which is the nearest frequency less than or equal to the target. Similarly, to approach a clock value of 400 kHz, the prescaler value of 08h and divisor value of eh yields the exact clock value of 400 kHz. The reset value of this field is 80h, so if the input Base Clock (ipg_perclk) is about 96 MHz, the default SD clock after reset is 375 kHz. Before changing clock divisor value (SDCLKFS or DVS), host driver should make sure the SDSTB field is high. If setting SDCLKFS and DVS can generate the same clock frequency,(for example, in SDR mode, SDCLKFS = 01h is same as DVS = 01h.), SDCLKFS is highly recommended.
7-4 DVS	Divisor This field is used to provide a more exact divisor to generate the desired SD clock frequency. Note the divider can even support odd divisors without deterioration of duty cycle. Before changing clock divisor value (SDCLKFS or DVS), Host driver should make sure the SDSTB field is high. The settings are as follows: 0000b - Divide-by-1 0001b - Divide-by-2 1110b - Divide-by-15 1111b - Divide-by-16
3-0 —	Reserved Always write as 1.

81.6.1.14 Interrupt Status (INT_STATUS)

Offset

Register	Offset
INT_STATUS	30h

Function

An interrupt is generated when the normal interrupt signal enable is enabled and at least one of the status fields is set to 1. For all fields, writing 1 to a bit clears it; writing to 0 keeps the bit unchanged. More than one status can be cleared with a single register write. For card interrupt, before writing 1 to clear, it is required that the card stops asserting the interrupt, meaning that when the card driver services the interrupt condition; otherwise, the CINT field is asserted again.

The table below shows the relationship between the command timeout error and the command complete.

Table 818. uSDHC status for command timeout error/command complete bit combinations

Command complete	Command timeout error	Meaning of the status
0	0	X
X	1	Response not received within 64 SDCLK cycles
1	0	Response received

The table below shows the relationship between the transfer complete and the data timeout error.

Table 819. uSDHC status for data timeout error/transfer complete bit combinations

Transfer complete	Data timeout error	Meaning of the status
0	0	X
0	1	Timeout occurred during transfer
1	X	Data transfer complete

The table below shows the relationship between the command CRC error and command timeout error.

Table 820. uSDHC status for command CRC error/command timeout error bit combinations

Command CRC error	Command timeout error	Meaning of the status
0	0	No error
0	1	Response timeout error
1	0	Response CRC error
1	1	CMD line conflict

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0		DMAE	0	0	0	AC12E	0	DEBE	DCE	DTOE	CIE	CEBE	CCE	CTOE
W				W1C				W1C		W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ERR_INT...	0	0	0	0	Reserved		CINT	CRM	CINS	BRR	BWR	DINT	BGE	TC	CC
W						W1C		W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30-29 —	Reserved
28 DMAE	DMA error Occurs when an Internal DMA transfer has failed. This field is set to 1, when some error occurs in the data transfer. This error can be caused by either simple DMA or ADMA, depending on which DMA is in use. The value in DMA System Address register is the next fetch address where the error occurs. Because any error corrupts the whole data block, the host driver restarts the transfer from the corrupted block boundary. The address of the block boundary can be calculated either from the current DS_ADDR value or from the remaining number of blocks and the block size. 0b - No error 1b - Error
27 —	Reserved
26 —	Reserved
25 —	Reserved
24 AC12E	Auto CMD12 error Occurs when detecting that one of the fields in the Auto CMD12 Error Status register has changed from 0 to 1. This field is set to 1, not only when the errors in Auto CMD12 occur, but also, when the Auto CMD12 is not executed due to the previous command error.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No error 1b - Error
23 —	Reserved
22 DEBE	Data end bit error Occurs either when detecting 0 at the end field position of read data that uses the DATA line, or at the end field position of the CRC. 0b - No error 1b - Error
21 DCE	Data CRC error Occurs when detecting a CRC error when transferring read data that uses the DATA line, or when detecting the Write CRC status having a value other than 010. 0b - No error 1b - Error
20 DTOE	Data timeout error Occurs when detecting one of following time-out conditions. • Busy time-out for R1b, R5b type • Busy time-out after Write CRC status • Read Data time-out. 0b - No error 1b - Time out
19 CIE	Command index error Occurs if a command index error occurs in the command response. 0b - No error 1b - Error
18 CEBE	Command end bit error Occurs when detecting that the end field of a command response is 0. 0b - No error 1b - End bit error generated
17 CCE	Command CRC error Command CRC Error is generated in two cases.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• If a response is returned and the Command Timeout Error is set to 0 (indicating no time-out), this field is set when detecting a CRC error in the command response. • The uSDHC module detects a CMD line conflict by monitoring the CMD line when a command is issued. If uSDHC drives the CMD line to 1, but detects 0 on the CMD line at the next SDCLK edge, then uSDHC aborts the command (Stop driving CMD line) and set this bit to 1. The Command Timeout Error should also be set to 1 to distinguish CMD line conflict. 0b - No error 1b - CRC error generated
16 CTOE	Command timeout error Occurs only if no response is returned within 64 SDCLK cycles from the end field of the command. If uSDHC detects a CMD line conflict, in which case a Command CRC Error is also be set (as shown in Interrupt Status (INT_STATUS)), this field is set without waiting for 64 SDCLK cycles. This is because the command is aborted by uSDHC. 0b - No error 1b - Time out
15 ERR_INT_STATUS	Error Interrupt Status This bit is set when any of the error status bit DMAE, AC12E, DEBE, DCE, DTOE, CIE, CEBE, CCE and CTOE is set.
14 —	Reserved
13 —	Reserved
12 —	Reserved
11 —	Reserved
10-9 —	Reserved
8 CINT	Card interrupt This status field is set when an interrupt signal is detected from the external card. In 1-bit mode, uSDHC detects the Card Interrupt without the SD clock to support wakeup. In 4-bit mode, the card interrupt signal is sampled during the interrupt cycle, so the interrupt from card can only be sampled during interrupt cycle, introducing some delay between the interrupt signal from SDIO and the interrupt to the host system. Writing this field to 1 can clear this field, but as the interrupt source from SDIO does not clear, this field is set again.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	to clear this field, it is required to reset the interrupt source from the external card followed by a writing 1 to this field. When this status has been set, and the host driver needs to service this interrupt, the Card Interrupt Signal Enable in the Interrupt Signal Enable register should be 0 to stop driving the interrupt signal to the host system. After completion of the card interrupt service (It should reset the interrupt sources in SDIO and the interrupt signal might not be asserted), write 1 to clear this field, set the Card Interrupt Signal Enable to 1, and start sampling the interrupt signal again. 0b - No card interrupt 1b - Generate card interrupt
7 CRM	Card removal This status field is set if the Card Inserted field in the Present State register changes from 1 to 0. When the host driver writes this field to 1 to clear this status, the status of the Card Inserted in the Present State register should be confirmed. Because the card state might possibly be changed when the host driver clears this field and the interrupt event might not be generated. When this field is cleared, it is set again if no card is inserted. to leave it cleared, clear the Card Removal Status Enable field in Interrupt Status Enable register. 0b - Card state unstable or inserted 1b - Card removed
6 CINS	Card insertion This status field is set if the Card Inserted field in the Present State register changes from 0 to 1. When the host driver writes this field to 1 to clear this status, the status of the Card Inserted in the Present State register should be confirmed. Because the card state might possibly be changed when the host driver clears this field and the interrupt event might not be generated. When this field is cleared, it is set again if a card is inserted. to leave it cleared, clear the Card Inserted Status Enable field in Interrupt Status Enable register. 0b - Card state unstable or removed 1b - Card inserted
5 BRR	Buffer read ready This status field is set if the Buffer Read Enable field, in the Present State register, changes from 0 to 1. See the Buffer Read Enable field in the Present State register for additional information. 0b - Not ready to read buffer 1b - Ready to read buffer
4 BWR	Buffer write ready This status field is set if the Buffer Write Enable field, in the Present State register, changes from 0 to 1. See the Buffer Write Enable field in the Present State register for additional information. 0b - Not ready to write buffer 1b - Ready to write buffer
3	DMA interrupt

Table continues on the next page...

Table continued from the previous page...

Field	Function
DINT	Occurs only when the internal DMA finishes the data transfer successfully. Whenever errors occur during data transfer, this field does not be set. Instead, the DMAE field is set. Either Simple DMA or ADMA finishes data transferring, this field is set. 0b - No DMA interrupt 1b - DMA interrupt is generated.
2 BGE	Block gap event If the Stop At Block Gap Request field in the Protocol Control register is set, this field is set when a read or write transaction is stopped at a block gap. If Stop At Block Gap Request is not set to 1, this field is not set to 1. In the case of a Read Transaction: This field is set at the falling edge of the DATA Line Active Status (When the transaction is stopped at SD bus timing). The Read Wait must be supported to use this function. In the case of Write Transaction: This field is set at the falling edge of Write Transfer Active Status (After getting CRC status at SD Bus timing). 0b - No block gap event 1b - Transaction stopped at block gap
1 TC	Transfer complete This field is set when a read or write transfer is completed. In the case of a Read Transaction: This field is set at the falling edge of the Read Transfer Active Status. There are two cases in which this interrupt is generated. The first is when a data transfer is completed as specified by the data length (after the last data has been read to the host system). The second is when data has stopped at the block gap and completed the data transfer by setting the Stop At Block Gap Request field in the Protocol Control register (after valid data has been read to the host system). In the case of a Write Transaction: This field is set at the falling edge of the DATA Line Active Status. There are two cases in which this interrupt is generated. The first is when the last data is written to the SD card as specified by the data length and the busy signal is released. The second is when data transfers are stopped at the block gap, by setting the Stop At Block Gap Request field in the Protocol Control register, and the data transfers are completed. (after valid data is written to the SD card and the busy signal released). In the case of a command with busy, this field is set when busy is deasserted. 0b - Transfer does not complete 1b - Transfer complete
0 CC	Command complete This field is set when you receive the end field of the command response (except auto CMD12). See the Command Inhibit (CMD) in the Present State register. 0b - Command not complete 1b - Command complete

81.6.1.15 Interrupt Status Enable (INT_STATUS_EN)

Offset

Register	Offset
INT_STATUS_EN	34h

Function

Setting any bit in this register to 1 enables the corresponding interrupt status be requested to the system. If any bit is set to 0, the corresponding interrupt request gets blocked.

- Depending on IABG field setting, uSDHC might be programmed to sample the card interrupt signal during the interrupt period and hold its value in the flip-flop. There are some delays on the Card Interrupt, asserted from the card, to the time the host system is informed.
- To detect a CMD line conflict, the host driver must set both Command Timeout Error Status Enable and Command CRC Error Status Enable to 1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	Reserved		DMAESEN	0	Reserved	0	AC12ESEN	0	DEBESEN	DCESEN	DTOESEN	CIESEN	CEBESEN	CCESEN	CTOESEN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	Reserved	0	Reserved		CINTSEN	CRMSEN	CINSSEN	BRRSEN	BWRSEN	DINTSEN	BGESEN	TCSEN	CCSEN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30-29 —	Reserved
28 DMAESEN	DMA error status enable 0b - Masked 1b - Enabled
27 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
26 —	Reserved
25 —	Reserved
24 AC12ESEN	Auto CMD12 error status enable 0b - Masked 1b - Enabled
23 —	Reserved
22 DEBESEN	Data end bit error status enable 0b - Masked 1b - Enabled
21 DCESEN	Data CRC error status enable 0b - Masked 1b - Enabled
20 DTOESEN	Data timeout error status enable 0b - Masked 1b - Enabled
19 CIESEN	Command index error status enable 0b - Masked 1b - Enabled
18 CEBESEN	Command end bit error status enable 0b - Masked 1b - Enabled
17 CCESEN	Command CRC error status enable 0b - Masked 1b - Enabled
16 CTOESEN	Command timeout error status enable 0b - Masked 1b - Enabled
15	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
14 —	Reserved
13 —	Reserved
12 —	Reserved
11 —	Reserved
10-9 —	Reserved
8 CINTSEN	Card interrupt status enable If this field is set to 0, uSDHC clears the interrupt request to the system. The Card Interrupt detection is stopped when this field is cleared and restarted when this field is set to 1. The host driver should clear the Card Interrupt Status Enable before servicing the Card Interrupt and should set this field again after all interrupt requests from the card are cleared to prevent inadvertent interrupts. 0b - Masked 1b - Enabled
7 CRMSEN	Card removal status enable 0b - Masked 1b - Enabled
6 CINSSEN	Card insertion status enable 0b - Masked 1b - Enabled
5 BRRSEN	Buffer read ready status enable 0b - Masked 1b - Enabled
4 BWRSEN	Buffer write ready status enable 0b - Masked 1b - Enabled
3	DMA interrupt status enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
DINTSEN	0b - Masked 1b - Enabled
2 BGESEN	Block gap event status enable 0b - Masked 1b - Enabled
1 TCSEN	Transfer complete status enable 0b - Masked 1b - Enabled
0 CCSEN	Command complete status enable 0b - Masked 1b - Enabled

81.6.1.16 Interrupt Signal Enable (INT_SIGNAL_EN)

Offset

Register	Offset
INT_SIGNAL_EN	38h

Function

This register is used to select which interrupt status is indicated to the host system as the interrupt. These status fields all share the same interrupt lines. Setting any of these fields to 1 enables interrupt generation. The corresponding Status register field generates an interrupt when the corresponding interrupt signal enable field is set.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	Reserved		DMAEIEN	0	Reserved	0	AC12EIEN	0	DEBEIEN	DCEIEN	DTOEIEN	CIEIEN	CEBEIEN	CCEIEN	CTOEIEN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	Reserved	0	Reserved		CINTIEN	CRMIEN	CINSIEN	BRRIEN	BWRIEN	DINTIEN	BGEIEN	TCIEN	CCIEN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30-29 —	Reserved
28 DMAEIEN	DMA error interrupt enable 0b - Masked 1b - Enable
27 —	Reserved
26 —	Reserved
25 —	Reserved
24 AC12EIEN	Auto CMD12 error interrupt enable 0b - Masked 1b - Enabled
23 —	Reserved
22 DEBEIEN	Data end bit error interrupt enable 0b - Masked 1b - Enabled
21 DCEIEN	Data CRC error interrupt enable 0b - Masked 1b - Enabled
20 DTOEIEN	Data timeout error interrupt enable 0b - Masked 1b - Enabled
19 CIEIEN	Command index error interrupt enable 0b - Masked 1b - Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
18 CEBEIEN	Command end bit error interrupt enable 0b - Masked 1b - Enabled
17 CCEIEN	Command CRC error interrupt enable 0b - Masked 1b - Enabled
16 CTOEIEN	Command timeout error interrupt enable 0b - Masked 1b - Enabled
15 —	Reserved
14 —	Reserved
13 —	Reserved
12 —	Reserved
11 —	Reserved
10-9 —	Reserved
8 CINTIEN	Card interrupt enable 0b - Masked 1b - Enabled
7 CRMIEN	Card removal interrupt enable 0b - Masked 1b - Enabled
6 CINSIEN	Card insertion interrupt enable 0b - Masked 1b - Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
5 BRRIEN	Buffer read ready interrupt enable 0b - Masked 1b - Enabled
4 BWRIEN	Buffer write ready interrupt enable 0b - Masked 1b - Enabled
3 DINTIEN	DMA interrupt enable 0b - Masked 1b - Enabled
2 BGEIEN	Block gap event interrupt enable 0b - Masked 1b - Enabled
1 TCIEN	Transfer complete interrupt enable 0b - Masked 1b - Enabled
0 CCIEN	Command complete interrupt enable 0b - Masked 1b - Enabled

81.6.1.17 Auto CMD12 Error Status (AUTOCMD12_ERR_STATUS)

Offset

Register	Offset
AUTOCMD12_ERR_STATUS	3Ch

Function

When the Auto CMD12 Error Status field in the Status register is set, the host driver checks this register to identify what kind of error the Auto CMD12 / CMD 23 indicated. Auto CMD23 errors are indicated in field 04-01. This register is valid only when the Auto CMD12 Error status field is set.

The table below shows the relationship between the Auto CMGD12 CRC Error and the Auto CMD12 Command Timeout Error.

Table 821. Relationship between command CRC error and command timeout error for auto CMD12

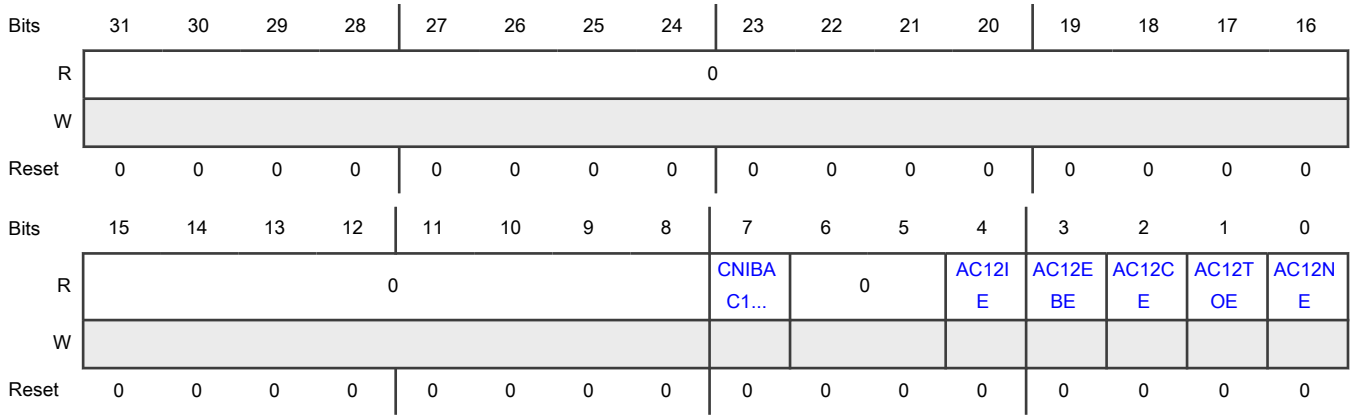
Auto CMD12 CRC error	Auto CMD12 timeout error	Type of error
0	0	No error
0	1	Response timeout error
1	0	Response CRC error
1	1	CMD line conflict

Changes in Auto CMD12 Error Status register can be classified in three scenarios:

- When uSDHC is going to issue an Auto CMD12
  - Set field 0 to 1 if the Auto CMD12 cannot be issued due to an error in the previous command
  - Set field 0 to 0 if the Auto CMD12 is issued
- At the end field of an Auto CMD12 response
  - Check errors correspond to fields 1-4
  - Set fields 1-4 corresponding to detected errors
  - Clear fields 1-4 corresponding to detected errors
- Before reading the Auto CMD12 Error Status field 7
  - Set field 7 to 1 if there is a command that can't be issued
  - Clear field 7 if there is no command to issue

The timing for generating the Auto CMD12 Error and writing to the Command register are asynchronous. After that, field 7 is sampled when the driver is not writing to the Command register. So, it is suggested to read this register only when the AC12E field in Interrupt Status register is set. An Auto CMD12 Error Interrupt is generated when one of the error fields (0-4) is set to 1. The Command Not Issued By Auto CMD12 Error does not generate an interrupt.

Diagram



Fields

Field	Function
31-8 —	Reserved
7 CNIBAC12E	Command not issued by Auto CMD12 error Setting this field to 1 means CMD_wo_DAT is not executed due to an Auto CMD12 error (D04-D01) in this register. 0b - No error 1b - Not issued
6-5 —	Reserved
4 AC12IE	Auto CMD12 / 23 index error Occurs if the command index error occurs in response to a command. 0b - No error 1b - Error, the CMD index in response is not CMD12/23
3 AC12EBE	Auto CMD12 / 23 end bit error Occurs when detecting that the end field of command response is 0 which should be 1. 0b - No error 1b - End bit error generated
2 AC12CE	Auto CMD12 / 23 CRC error Occurs when detecting a CRC error in the command response. 0b - No CRC error 1b - CRC error met in Auto CMD12/23 response
1 AC12TOE	Auto CMD12 / 23 timeout error Occurs if no response is returned within 64 SDCLK cycles from the end field of the command. If this field is set to1, the other error status fields (2-4) have no meaning. 0b - No error 1b - Time out
0 AC12NE	Auto CMD12 not executed If memory multiple block data transfer is not started, due to a command error, this field is not set because it is not necessary to issue an Auto CMD12. Setting this field to 1 means uSDHC cannot issue the Auto CMD12 to stop a memory multiple block data transfer due to some error. If this field is set to 1, other error status fields (1-4) have no meaning. 0b - Executed 1b - Not executed

81.6.1.18 Host Controller Capabilities (HOST_CTRL_CAP)

Offset

Register	Offset
HOST_CTRL_CAP	40h

Function

This register provides the host driver with information specific to uSDHC implementation. The value in this register is the power-on-reset value and does not change with a software reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0					Reserved	VS30	VS33	SRS	DMAS	HSS	ADMAS	0	MBL		
W																
Reset	0	0	0	0	0	0	1	1	1	1	1	1	0	0	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0													Reserved	Reserved	Reserved
W																
Reset	1	0	1	1	0	1	0	0	0	0	0	0	0	1	0	0

Fields

Field	Function
31-27 —	Reserved
26 —	Reserved
25 VS30	Voltage support 3.0 V This field depends on the host system ability. 0b - 3.0 V not supported 1b - 3.0 V supported
24 VS33	Voltage support 3.3 V This field depends on the host system ability. 0b - 3.3 V not supported 1b - 3.3 V supported

Table continues on the next page...

Table continued from the previous page...

Field	Function
23 SRS	Suspend / resume support This field indicates whether uSDHC supports Suspend / Resume functionality. If this field is 0, the Suspend and Resume mechanism, as well as the read wait, are not supported, and the host driver does not issue either Suspend or Resume commands. 0b - Not supported 1b - Supported
22 DMAS	DMA support This field indicates whether uSDHC can use the internal DMA to transfer data between system memory and the data buffer directly. 0b - DMA not supported 1b - DMA supported
21 HSS	High speed support This field indicates whether uSDHC supports High Speed mode and the host system can supply a SD clock frequency from 25 MHz to 50 MHz. 0b - High speed not supported 1b - High speed supported
20 ADMAS	ADMA support This field indicates whether uSDHC supports the ADMA feature. 0b - Advanced DMA not supported 1b - Advanced DMA supported
19 —	Reserved
18-16 MBL	Max block length This field indicates the maximum block size that the host driver can read and write to the buffer in uSDHC. The buffer transfers block size without wait cycles. 000b - 512 bytes 001b - 1024 bytes 010b - 2048 bytes 011b - 4096 bytes
15-3 —	Reserved
2 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 —	Reserved
0 —	Reserved

81.6.1.19 Watermark Level (WTMK_LVL)

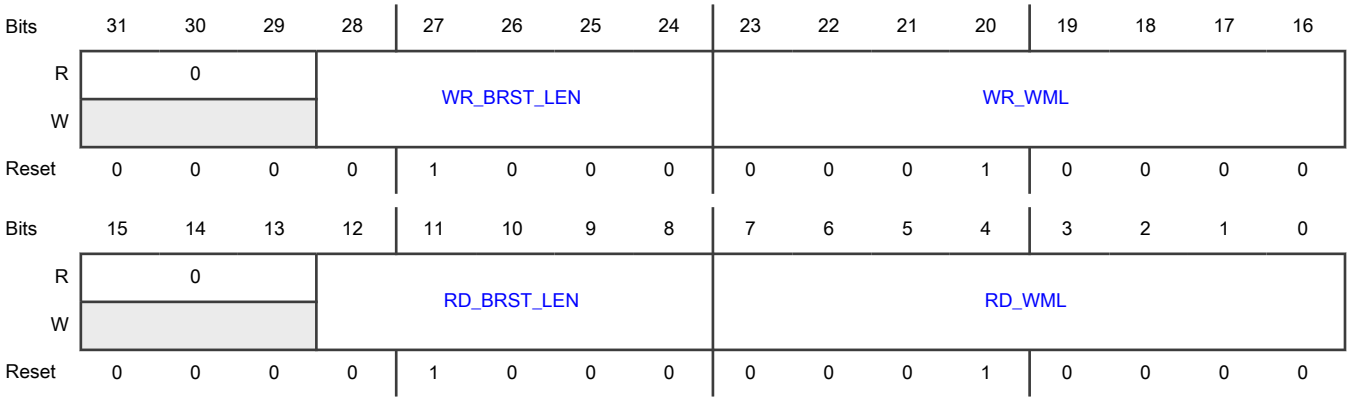
Offset

Register	Offset
WTMK_LVL	44h

Function

This register indicates configurability of write and read watermark levels (FIFO threshold). Their value can range from 1 to 128 words. Both write and read burst lengths are also configurable. Their value can range from 1 to 31 words.

Diagram



Fields

Field	Function
31-29 —	Reserved
28-24 WR_BRST_LEN	Write burst length due to system restriction, the actual burst length might not exceed 16 This field indicates the number of words uSDHC writes in a single burst. The write burst length must be less than or equal to the write watermark level, and all bursts within a watermark level transfer is in back-to-back mode. On reset, this field is 8. Writing 0 to this field results in '01000' (that is, it is not able to clear this field).

Table continues on the next page...

Table continued from the previous page...

Field	Function
23-16 WR_WML	Write watermark level This field indicates the number of words used as the watermark level (FIFO threshold) in a DMA write operation. Also, the number of words as a sequence of write bursts in back-to-back mode. The maximum legal value for the write watermark level is 128.
15-13 —	Reserved
12-8 RD_BRST_LEN	Read burst length due to system restriction, the actual burst length might not exceed 16 This field indicates the number of words uSDHC reads in a single burst. The read burst length must be less than or equal to the read watermark level, and all bursts within a watermark level transfer is in back-to-back mode. On reset, this field is 8. Writing 0 to this field results in '01000' (that is, it is not able to clear this field).
7-0 RD_WML	Read watermark level This field indicates the number of words used as the watermark level (FIFO threshold) in a DMA read operation. Also, the number of words as a sequence of read bursts in back-to-back mode. The maximum legal value for the read water mark level is 128.

81.6.1.20 Mixer Control (MIX_CTRL)

Offset

Register	Offset
MIX_CTRL	48h

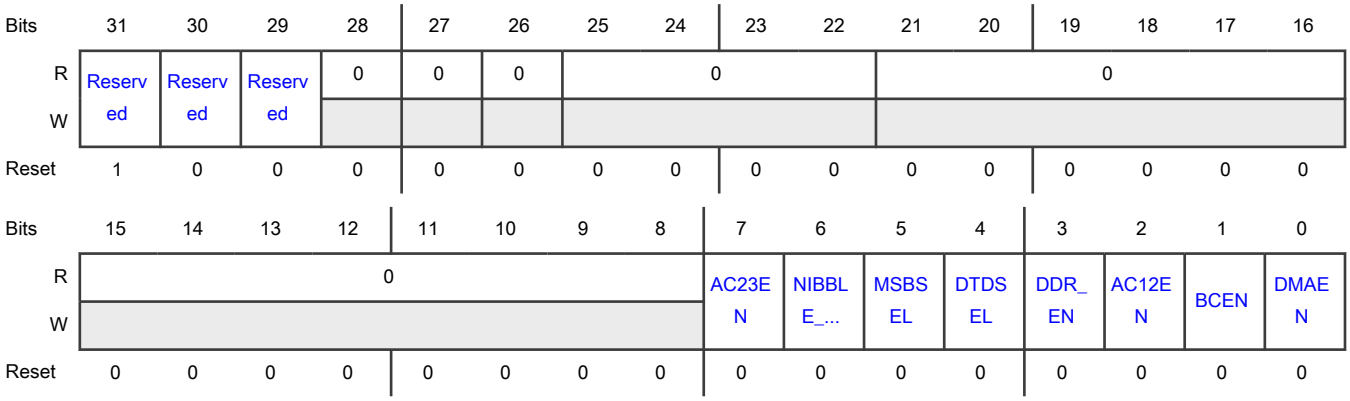
Function

This register is used to DMA and data transfer. To prevent data loss, the software should check if data transfer is active before writing this register. These fields are DPSEL, MBSEL, DTDSEL, AC12EN, BCEN, and DMAEN.

Table 822. Transfer type register setting for various transfer types

Multi/single block select	Block count enable	Block count	Function
0	Do not care	Do not care	Single transfer
1	0	Do not care	Infinite transfer
1	1	Positive number	Multiple transfer
1	1	Zero	No data transfer

Diagram



Fields

Field	Function
31 —	Reserved Always write as 1.
30 —	Reserved Always write as 0.
29 —	Reserved Always write as 0.
28 —	Reserved
27 —	Reserved
26 —	Reserved
25-22 —	Reserved
21-8 —	Reserved
7 AC23EN	Auto CMD23 enable This field is read/write when VEND_SPEC[CMD_BYTE_EN] is disabled; otherwise, this field is read-only. When this field is set to 1, the host controller issues a CMD23 automatically before issuing a command specified in the Command Register.
6	Nibble position indication

Table continues on the next page...

Table continued from the previous page...

Field	Function
NIBBLE_POS	This field indicates the nibble position in the DDR 4-bit mode. This field is read/write when VEND_SPEC[CMD_BYTE_EN] is disabled; otherwise, this field is read-only. 0- the sequence is 'odd high nibble -> even high nibble -> odd low nibble -> even low nibble'; 1- the sequence is 'odd high nibble -> odd low nibble -> even high nibble -> even low nibble'.
5 MSBSEL	Multi / Single block select This field enables multiple block DATA line data transfers. This field is read/write when VEND_SPEC[CMD_BYTE_EN] is disabled; otherwise, this field is read-only. For any other commands, this field can be set to 0. If this field is 0, it is not necessary to set the Block Count register. See Command Transfer Type (CMD_XFR_TYP). 0b - Single block 1b - Multiple blocks
4 DTDSEL	Data transfer direction select This field defines the direction of DATA line data transfers. This field is read/write when VEND_SPEC[CMD_BYTE_EN] is disabled; otherwise, this field is read-only. The field is set to 1 by the host driver to transfer data from the SD card to uSDHC and is set to 0 for all other commands. 0b - Write (Host to card) 1b - Read (Card to host)
3 DDR_EN	Dual data rate mode selection This field is read/write when VEND_SPEC[CMD_BYTE_EN] is disabled; otherwise, this field is read-only.
2 AC12EN	Auto CMD12 enable Multiple block transfers for memory require a CMD12 to stop the transaction. This field is read/write when VEND_SPEC[CMD_BYTE_EN] is disabled; otherwise, this field is read-only. When this field is set to 1, uSDHC issues a CMD12 automatically when the last block transfer has completed. The host driver is not set this field to issue commands that do not require CMD12 to stop a multiple block data transfer. In particular, secure commands defined in File Security Specification (see reference list) do not require CMD12. In single block transfer, uSDHC ignores this field no matter it is set or not. 0b - Disable 1b - Enable
1 BCEN	Block count enable This field is used to enable the Block Count register, which is only relevant for multiple block transfers. This field is read/write when VEND_SPEC[CMD_BYTE_EN] is disabled; otherwise, this field is read-only. When this field is 0, the internal counter for block is disabled, which is useful in executing an infinite transfer. 0b - Disable 1b - Enable
0 DMAEN	DMA enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field enables DMA functionality. This field is read/write when VEND_SPEC[CMD_BYTE_EN] is disabled; otherwise, this field is read-only. If this field is set to 1, a DMA operation begins when the host driver sets the DPSEL field of this register. Whether the simple DMA or the advanced DMA is active depends on the DMA Select field of the Protocol Control register. 0b - Disable 1b - Enable

81.6.1.21 Force Event (FORCE_EVENT)

Offset

Register	Offset
FORCE_EVENT	50h

Function

This register is not a physically implemented register. Rather, it is an address at which the Interrupt Status register can be written if the corresponding field of the Interrupt Status Enable Register is set. This register is a write only register and writing 0 to it has no effect. Writing 1 to this register sets the corresponding field of Interrupt Status register. A read from this register always results in 0's. to change the corresponding status fields in the Interrupt Status register, make sure to set IPGEN field in System Control Register so that peripheral clock is always active.

Forcing a card interrupt generates a short pulse on the DATA1 line, and the driver might treat this interrupt as a normal interrupt. The interrupt service routine might skip polling the card interrupt factor as the interrupt is self cleared.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0		0	0	0	0	0	0	0	0	0	0	0	0	0
W	FEVT CINT			FEVT DMAE				FEVT AC1...		FEVT DEBE	FEVT DCE	FEVT DTOE	FEVT CIE	FEVT CEBE	FEVT CCE	FEVT CTOE
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								0	0		0	0	0	0	0
W									FEVT CNI...			FEVTA C1...	FEVTA C1...	FEVTA C1...	FEVTA C1...	FEVTA C1...
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 FEVTCINT	Force event card interrupt Writing 1 to this field generates a short low-level pulse on the internal DATA1 line, as if a self-clearing interrupt was received from the external card. If enabled, the CINT field is set and the interrupt service routine might treat this interrupt as a normal interrupt from the external card.
30-29 —	Reserved
28 FEVTDMAE	Force event DMA error Forces the DMAE field of Interrupt Status register to be set
27 —	Reserved
26 —	Reserved
25 —	Reserved
24 FEVTAC12E	Force event Auto Command 12 error Forces the AC12E field of Interrupt Status register to be set
23 —	Reserved
22 FEVTDEBE	Force event data end bit error Forces the DEBE field of Interrupt Status register to be set
21 FEVTDCE	Force event data CRC error Forces the DCE field of Interrupt Status register to be set
20 FEVTDTOE	Force event data time out error Force the DTOE field of Interrupt Status register to be set
19 FEVTCIE	Force event command index error Forces the CCE field of Interrupt Status register to be set
18 FEVTCEBE	Force event command end bit error Forces the CEBE field of Interrupt Status register to be set
17 FEVTCCE	Force event command CRC error Forces the CCE field of Interrupt Status register to be set

Table continues on the next page...

Table continued from the previous page...

Field	Function
16 FEVTCTOE	Force event command time out error Forces the CTOE field of Interrupt Status register to be set
15-8 —	Reserved
7 FEVTCNIBAC12E	Force event command not executed by Auto Command 12 error Forces the CNIBAC12E field in the Auto Command12 Error Status register to be set
6-5 —	Reserved
4 FEVTAC12IE	Force event Auto Command 12 index error Forces the AC12IE field in the Auto Command12 Error Status register to be set
3 FEVTAC12EBE	Force event Auto Command 12 end bit error Forces the AC12EBE field in the Auto Command12 Error Status register to be set
2 FEVTAC12CE	Force event auto command 12 CRC error Forces the AC12CE field in the Auto Command12 Error Status register to be set
1 FEVTAC12TOE	Force event auto command 12 time out error Forces the AC12TOE field in the Auto Command12 Error Status register to be set
0 FEVTAC12NE	Force event auto command 12 not executed Forces the AC12NE field in the Auto Command12 Error Status register to be set

81.6.1.22 ADMA Error Status (ADMA_ERR_STATUS)

Offset

Register	Offset
ADMA_ERR_STATUS	54h

Function

When an ADMA Error Interrupt has occurred, the ADMA error sates field in this register holds the ADMA state and the ADMA System Address register holds the address around the error descriptor.

For recovering from this error, the host driver requires the ADMA state to identify the error descriptor address as follows:

- ST_STOP: Previous location set in the ADMA System Address register is the error descriptor address.
- ST_FDS: Current location set in the ADMA System Address register is the error descriptor address.

- ST_CADR: This state is never set because it only increments the descriptor pointer and does not generate an ADMA error.
- ST_TFR: Previous location set in the ADMA System Address register is the error descriptor address.

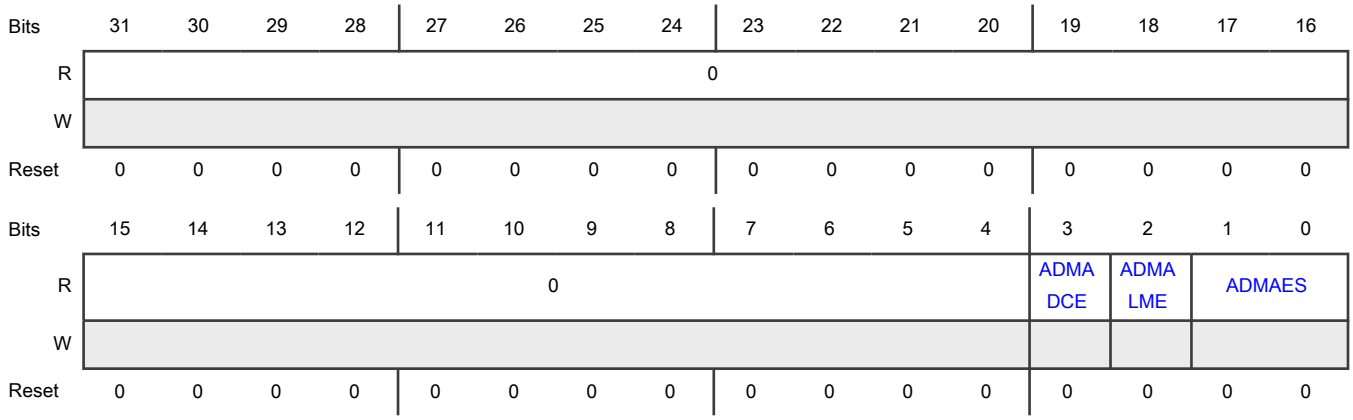
In case of a write operation, the host driver should use the ACMD22 to get the number of the written block, rather than using this information, because unwritten data might exist in the host controller.

The host controller generates the ADMA Error Interrupt when it detects invalid descriptor data (Valid=0) in the ST_FDS state. The host driver can distinguish this error by reading the Valid field of the error descriptor.

Table 823. ADMA error state coding

D01-D00	ADMA error state (when error has occurred)	Contents of ADMA System Address register
00	ST_STOP (Stop DMA)	Holds the address of the next executable descriptor command
01	ST_FDS (Fetch descriptor)	Holds the valid descriptor address
10	ST_CADR (Change address)	No ADMA error is generated
11	ST_TFR (Transfer data)	Holds the address of the next executable descriptor command

Diagram



Fields

Field	Function
31-4 —	Reserved
3 ADMADCE	ADMA descriptor error This error occurs when invalid descriptor fetched by ADMA. 0b - No error 1b - Error

Table continues on the next page...

Table continued from the previous page...

Field	Function
2 ADMALME	ADMA length mismatch error This error occurs in the following two cases: • While the block count enable is being set, the total data length specified by the descriptor table is different from that specified by the block count and block length. • Total data length cannot be divided by the block length. 0b - No error 1b - Error
1-0 ADMAES	ADMA error state (when ADMA error is occurred) This field indicates the state of the ADMA when an error has occurred during an ADMA data transfer. See ADMA Error Status (ADMA_ERR_STATUS) for more details.

81.6.1.23 ADMA System Address (ADMA_SYS_ADDR)

Offset

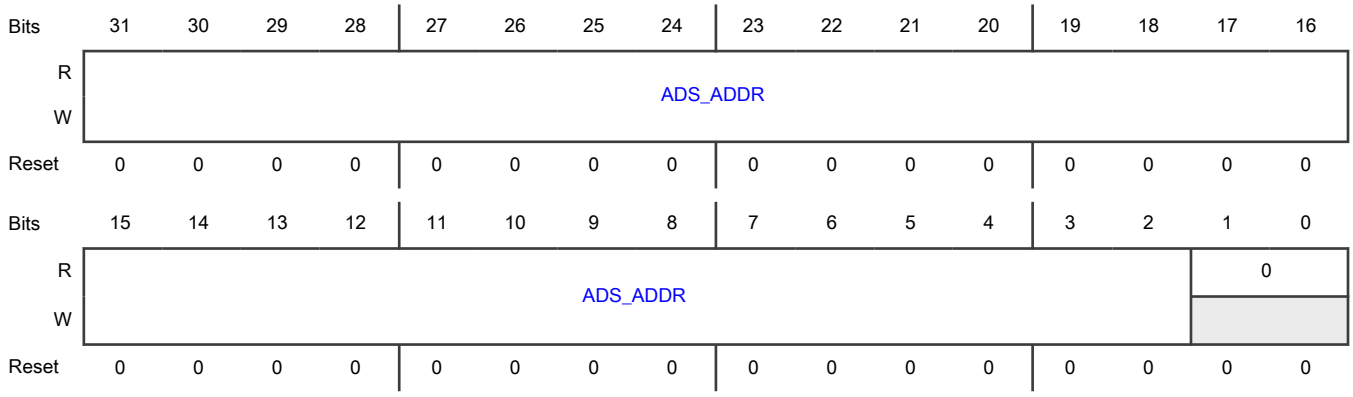
Register	Offset
ADMA_SYS_ADDR	58h

Function

This register holds the word address of the executing command in the Descriptor table. At the start of ADMA, the host driver sets the start address of the Descriptor table. The ADMA engine increments this register address whenever fetching a Descriptor command. When the ADMA is stopped at the Block Gap, this register indicates the address of the next executable Descriptor command. When the ADMA Error Interrupt is generated, this register holds the valid Descriptor address depending on the ADMA state. The lower 2 bits of this register is tied to '0' so the ADMA address is always word aligned.

Because this register supports dynamic address reflecting, when TC field is set, it automatically alters the value of internal address counter, so the software cannot change this register when TC field is set. Such restriction is also listed in Software restrictions.

Diagram



Fields

Field	Function
31-2 ADS_ADDR	ADMA system address This field contains the physical system memory address used for ADMA transfers.
1-0 —	Reserved

81.6.1.24 DLL (Delay Line) Control (DLL_CTRL)

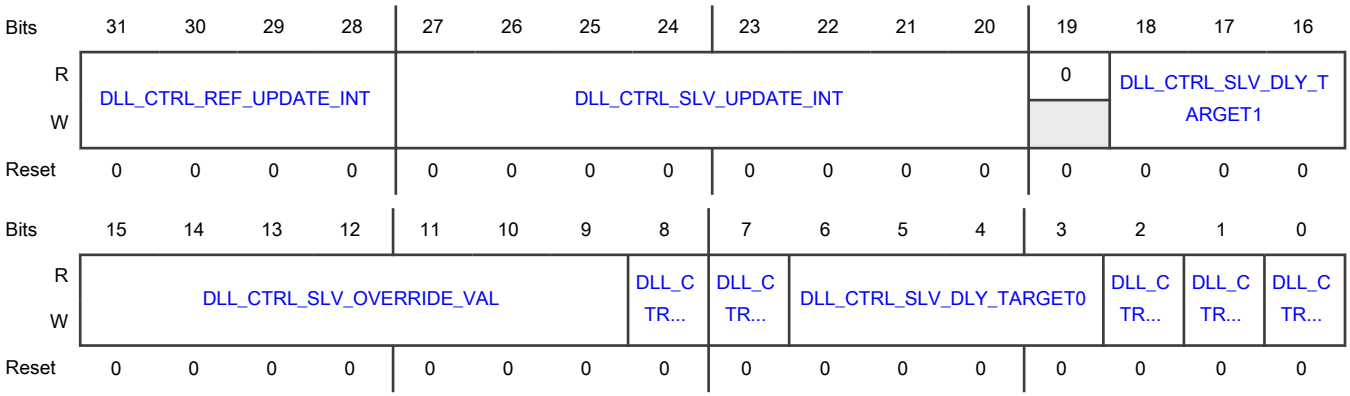
Offset

Register	Offset
DLL_CTRL	60h

Function

This register contains control fields for DLL.

Diagram



Fields

Field	Function
31-28 DLL_CTRL_REF_UPDATE_INT	DLL control loop update interval The interval cycle is (2 + REF_UPDATE_INT) * REF_CLOCK. By default, the DLL control loop updates every two REF_CLOCK cycles. It should be noted that increasing the reference delay-line update interval reduces the ability of the DLL to adjust to fast changes in conditions that might effect the delay (such as voltage and temperature)
27-20	Slave delay line update interval

Table continues on the next page...

Table continued from the previous page...

Field	Function
DLL_CTRL_SLV_UPDATE_INT	If default 0 is used, it means 256 cycles of REF_CLOCK. A value of 0x0f results in 15 cycles and so on. Note that software can always cause an update of the slave-delay line using the SLV_FORCE_UPDATE register. Note that the slave delay line also updates automatically when the reference DLL transitions to a locked state (from an un-locked state).
19 —	Reserved
18-16 DLL_CTRL_SLV_DLY_TARGET1	DLL slave delay target1 See DLL_CTRL_SLV_DLY_TARGET0 below.
15-9 DLL_CTRL_SLV_OVERRIDE_VAL	DLL slave override val When SLV_OVERRIDE = 1, this field is used to select 1 of 128 physical taps manually. A value of 0 selects tap 1, and a value of 0x7f selects tap 128.
8 DLL_CTRL_SLV_OVERRIDE	DLL slave override Set this field to 1 to Enable manual override for slave delay chain using SLV_OVERRIDE_VAL; to set 0 to disable manual override. This feature does not require the DLL to be enabled using the ENABLE field. In fact to reduce power, if SLV_OVERRIDE is used, it is recommended to disable the DLL with ENABLE = 0
7 DLL_CTRL_GATE_UPDATE	DLL gate update Set this field to 1 to prevent the DLL from updating (because when clock_in exists, glitches might appear during DLL updates). This field might be used by software if such a condition occurs. Clear the bit to 0 to allow the DLL to update automatically.
6-3 DLL_CTRL_SLV_DLY_TARGET0	DLL slave delay target0 The delay target for uSDHC loopback read clock can be programmed in 1/16th increments of an ref_clock half-period. The delay is (({DLL_CTRL_SLV_DLY_TARGET1,DLL_CTRL_SLV_DLY_TARGET0} +1)* REF_CLOCK / 2) / 16 So the input read-clock can be delayed relative input data from (REF_CLOCK / 2) / 16 to REF_CLOCK * 4. **NOTE** For the restrictions of delay cell implementation, the delay target must be set between REF_CLOCK/16 and REF_CLOCK*2 when REF_CLOCK is running at 200 MHz. When REF_CLOCK frequency is slower than 100 MHz, the maximum delay target might not reach REF_CLOCK*2.
2 DLL_CTRL_SLV_FORCE_UPD	DLL slave delay line Setting this field to 1, forces the slave delay line to update to the DLL calibrated value immediately. The slave delay line updates automatically based on the SLV_UPDATE_INT interval or when a DLL lock condition is sensed. Subsequent forcing of the slave-line update can only occur if SLV_FORCE_UP is set back to 0 and then asserted again (edge triggered). Be sure to use it when uSDHC is idle. This function might not work when uSDHC is working on data / cmd / response.

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 DLL_CTRL_RESET	DLL reset Setting this field to 1 force a reset on DLL. This causes the DLL to lose lock and re-calibrate to detect an REF_CLOCK half period phase shift. This signal is used by the DLL as edge-sensitive, so to create a subsequent reset, RESET must be taken low and then asserted again.
0 DLL_CTRL_ENABLE	DLL and delay chain Set this field to 1 to enable the DLL and delay chain; otherwise; set to 0 to bypasses DLL. Note that using the slave delay line override feature with SLV_OVERRIDE and SLV_OVERRIDE VAL, the DLL does not need to be enabled.

81.6.1.25 DLL Status (DLL_STATUS)

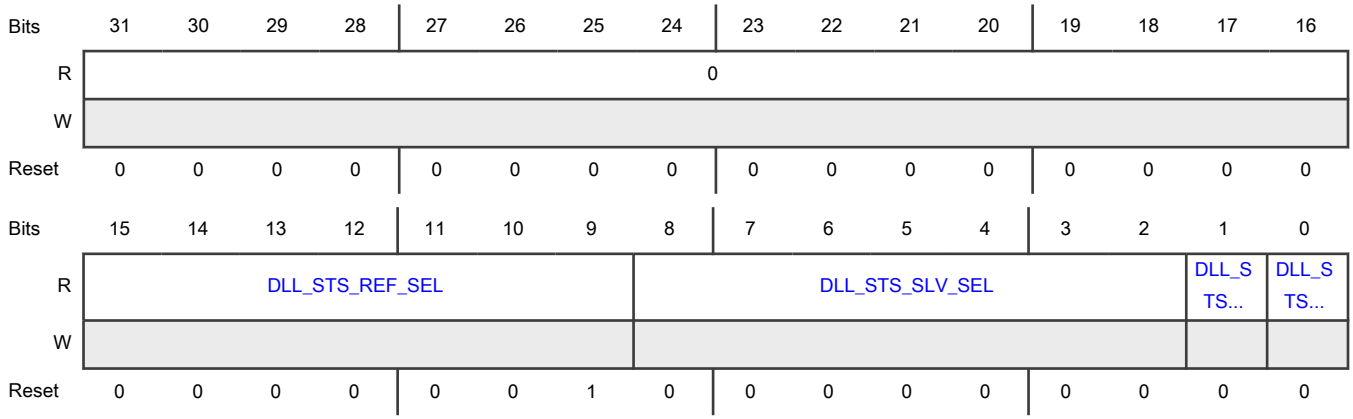
Offset

Register	Offset
DLL_STATUS	64h

Function

This register contains the DLL status information. All fields are read only and reads the same as the power-reset value.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-9	Reference delay line select taps

Table continues on the next page...

Table continued from the previous page...

Field	Function
DLL_STS_REF_SEL	This is encoded by 7 fields for 127 taps.
8-2 DLL_STS_SLV_SEL	Slave delay line select status This is the instant value generated from reference chain. Because the reference chain can only be updated when REF_CLOCK is detected, this value should be the right value to be updated when the reference is locked.
1 DLL_STS_REF_LOCK	Reference DLL lock status This signifies that the DLL has detected and locked to a half-phase ref_clock shift, allowing the slave delay-line to perform programmed clock delays
0 DLL_STS_SLV_LOCK	Slave delay-line lock status This signifies that a valid calibration has been set to the slave-delay line and that the slave-delay line is implementing the programmed delay value

81.6.1.26 Vendor Specific Register (VEND_SPEC)

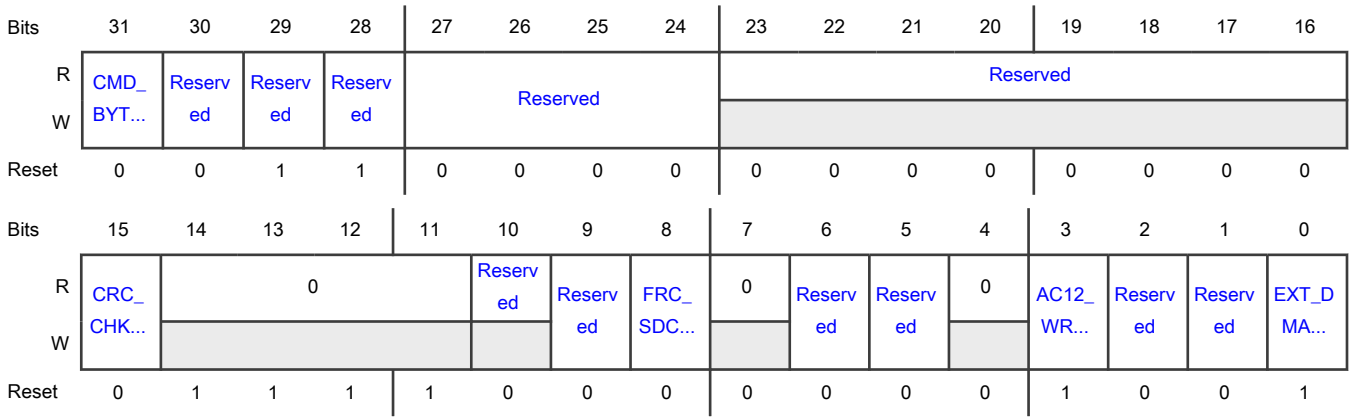
Offset

Register	Offset
VEND_SPEC	C0h

Function

This register contains the vendor specific control/status register.

Diagram



Fields

Field	Function
31 CMD_BYTE_EN	Register byte access for CMD_XFR_TYP This field controls the register byte access for Command Transfer Type (CMD_XFR_TYP). If this field is enabled, the register can be configured through byte write enable. IPS_bus can write only one byte once for one write operation. 0b - Disable. MIX_CTRL[7:0] is read/write and CMD_XFR_TYP[7:0] is read-only. 1b - Enable. MIX_CTRL[7:0] is read-only and CMD_XFR_TYP[7:0] is read/write.
30 —	Reserved Always write as 0.
29 —	Reserved Always write as 1
28 —	Reserved Always write as 0.
27-24 —	Reserved Always write as 4'b0000.
23-16 —	Reserved Always write as 8'h00.
15 CRC_CHK_DIS	CRC Check Disable 0b - Check CRC16 for every read data packet and check CRC fields for every write data packet 1b - Ignore CRC16 check for every read data packet and ignore CRC fields check for every write data packet
14-11 —	Reserved Reserved
10 —	Reserved Always write as 0.
9 —	Reserved Always write as 0.
8 FRC_SDCLK_ON	Force CLK Force CLK output active Do not set this bit to 1 unless it is necessary. Also, make sure that this bit is cleared when uSDHC's clock is about to be changed (frequency change, clock source change, or delay chain tuning). 0b - CLK active or inactive is fully controlled by the hardware. 1b - Force CLK active

Table continues on the next page...

Table continued from the previous page...

Field	Function
7 —	Reserved
6 —	Reserved
5 —	Reserved
4 —	Reserved
3 AC12_WR_CHKBUSY_EN	Check busy enable Check busy enable after auto CMD12 for write data packet 0b - Do not check busy after auto CMD12 for write data packet 1b - Check busy after auto CMD12 for write data packet
2 —	Reserved
1 —	Reserved
0 EXT_DMA_EN	External DMA request enable Enable the request to external DMA. When the internal DMA (either Simple DMA or Advanced DMA) is not in use, and this field is set, uSDHC sends out DMA request when the internal buffer is ready. This field is particularly useful when transferring data by Arm platform polling mode, and it is not allowed to send out the external DMA request. By default, this field is set. 0b - In any scenario, uSDHC does not send out external DMA request. 1b - When internal DMA is not active, the external DMA request is sent out.

81.6.1.27 eMMC Boot (MMC_BOOT)

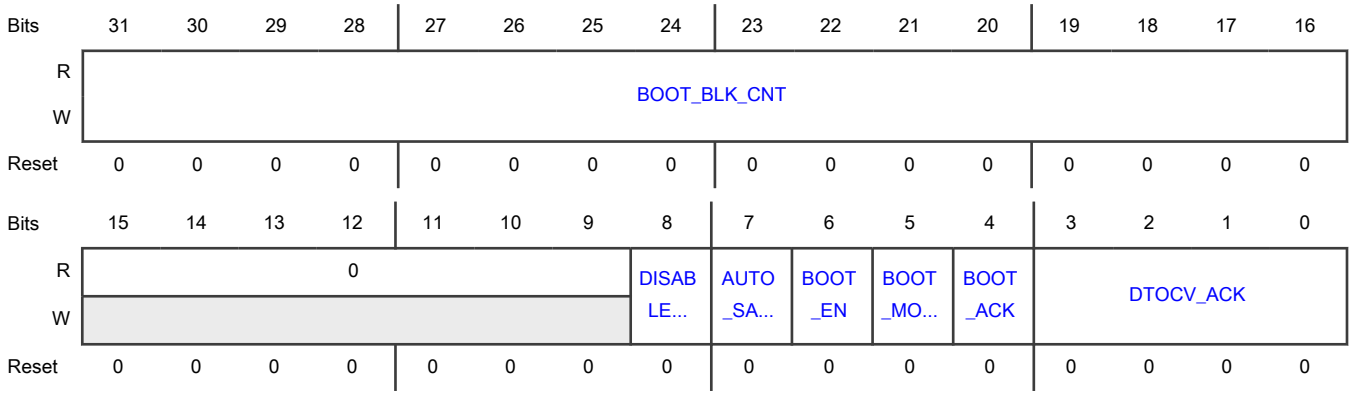
Offset

Register	Offset
MMC_BOOT	C4h

Function

This register contains the eMMC Fast Boot control register.

Diagram



Fields

Field	Function
31-16 BOOT_BLK_CNT	Stop At Block Gap value of automatic mode The value defines the Stop At Block Gap value of automatic mode. When received, card block cnt is equal to (BLK_CNT - BOOT_BLK_CNT) and AUTO_SABG_EN is 1, then Stop At Block Gap. Here, BLK_CNT is defined in the Block Attributes Register, field 31 - 16 of 0x04.
15-9 —	Reserved
8 DISABLE_TIME_OUT	Time out **NOTE** When this field is set, there is no timeout check no matter whether BOOT_EN is set or not. 0b - Enable time out 1b - Disable time out
7 AUTO_SABG_EN	Auto stop at block gap During boot, enable auto stop at block gap function. This function is triggered, and host stops at block gap when received card block cnt is equal to (BLK_CNT - BOOT_BLK_CNT).
6 BOOT_EN	Boot enable Boot mode enable 0b - Fast boot disable 1b - Fast boot enable
5 BOOT_MODE	Boot mode Boot mode select 0b - Normal boot 1b - Alternative boot

Table continues on the next page...

Table continued from the previous page...

Field	Function
4 BOOT_ACK	BOOT ACK Boot ACK mode select 0b - No ack 1b - Ack
3-0 DTOCV_ACK	Boot ACK time out Boot ACK time out counter value. 0000b - SDCLK x 2^14 0001b - SDCLK x 2^15 0010b - SDCLK x 2^16 0011b - SDCLK x 2^17 0100b - SDCLK x 2^18 0101b - SDCLK x 2^19 0110b - SDCLK x 2^20 0111b - SDCLK x 2^21 1110b - SDCLK x 2^28 1111b - SDCLK x 2^29

81.6.1.28 Vendor Specific 2 Register (VEND_SPEC2)

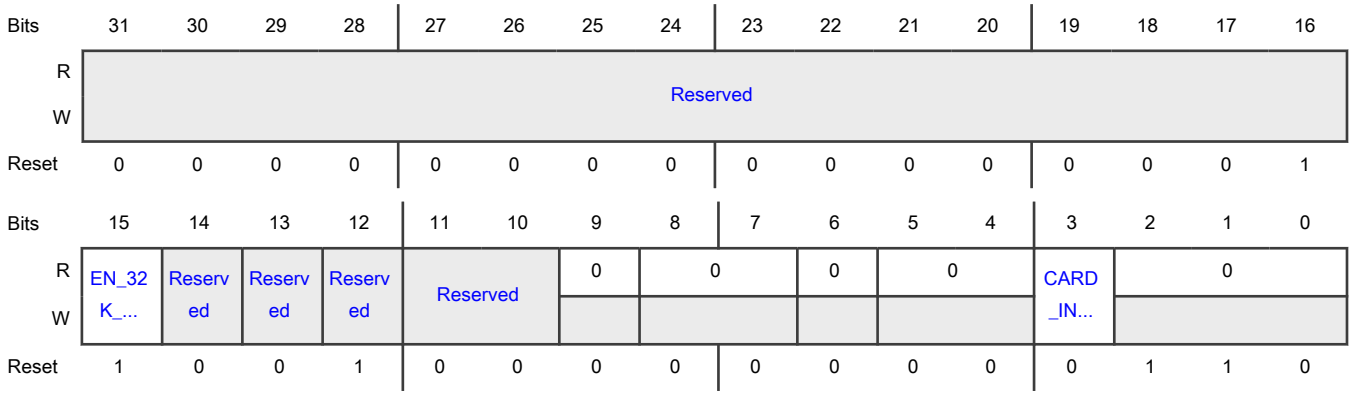
Offset

Register	Offset
VEND_SPEC2	C8h

Function

This register contains the vendor specific control 2 register.

Diagram



Fields

Field	Function
31-16 —	Reserved
15 EN_32K_CLK	Select the clock source for host card detection. It can use low power clock for card detection, by setting this bit to 1. 0b - Use the peripheral clock (ipg_clk) for card detection. 1b - Use the low power clock (ipg_clk_lp) for card detection.
14 —	Reserved
13 —	Reserved
12 —	Reserved
11-10 —	Reserved
9 —	Reserved
8-7 —	Reserved
6 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
5-4 —	Reserved
3 CARD_INT_D3_TEST	Card interrupt detection test This field is used only for debugging. 0b - Check the card interrupt only when DATA3 is high. 1b - Check the card interrupt by ignoring the status of DATA3.
2-0 —	Reserved

Chapter 82
Debug Subsystem

82.1 Introduction

This chapter discusses the debug and trace architecture of the chip that is based on the specifications provided in the *Arm® CoreSight™ SoC-400 Technical Reference Manual*. See References for a link to this document and to other related documentation available on the Arm website.

The chip architecture includes debug and trace modules. See Features for details on the debug and trace features that Cortex-M7 core clusters support.

The debug components that the Cortex-M7 core supports are accessible via the Arm DAP controller-based architecture. The DAP controller works in parallel with the system JTAGC. Both these controllers share the JTAG port and JTAG instruction set.

The system components are similar for different chips in the same family and include the primary core debug interfaces, the chip-level debug interfaces, and chip-level trace interfaces. The accelerator components include debug and trace circuits. These circuits are necessary for any application-specific accelerators required for the different application spaces that a chip supports. They vary from one chip to another and could include only trace components.

82.2 Interfaces supported in S32K3 family

Table 824. Interfaces supported in S32K3xx family

S32K344	S32K324	S32K322	S32K342 / S32K341	S32K312 / S32K311 / S32K310	S32K314	S32K338	S32K358 / S32K356	S32K348	S32K328	S32K388 / S32K389
Cortex-M7_0 (CTI, DWT, BPU, ETM, ITM)		Cortex-M7_0 (CTI, DWT, BPU, ITM)			Cortex-M7_0 (CTI, DWT, BPU, ETM, ITM)					
—	Cortex-M7_1 (CTI, DWT, BPU, ETM, ITM)	Cortex-M7_1 (CTI, DWT, BPU, ITM)	—		Cortex-M7_1 (CTI, DWT, BPU, ETM, ITM)					
—					Cortex-M7_2 (CTI, DWT, BPU, ETM, ITM)					
										Cortex-M7_3 (CTI, DWT, BPU, ETM, ITM)
Cortex-M0+										
HTM (CTI+DMA/ EMAC TRACE)		—			HTM (CTI+DMA/EMAC/GMAC TRACE)					HTM (CTI+DM

Table continues on the next page...

Table 824. Interfaces supported in S32K3xx family (continued)

S32K344	S32K324	S32K322	S32K342 / S32K341	S32K312 / S32K311 / S32K310	S32K314	S32K338	S32K358 / S32K356	S32K348	S32K328	S32K388 / S32K389
										A/ GMAC_0/ GMAC_1)
SWO										
TPIU		—			TPIU					
4x ETF		—			4x ETF					
3x Funnel		—			3x Funnel					
3x CTI	4x CTI	3x CTI	2x CTI		3x CTI	5x CTI	4x CTI	3x CTI	4x CTI	6x CTI
1x CTM						2x CTM				
Timestamp										
MDM_AP										
SDA_AP										

82.3 Block diagram

This figure illustrates the DAP architecture of the S32K3xx family.

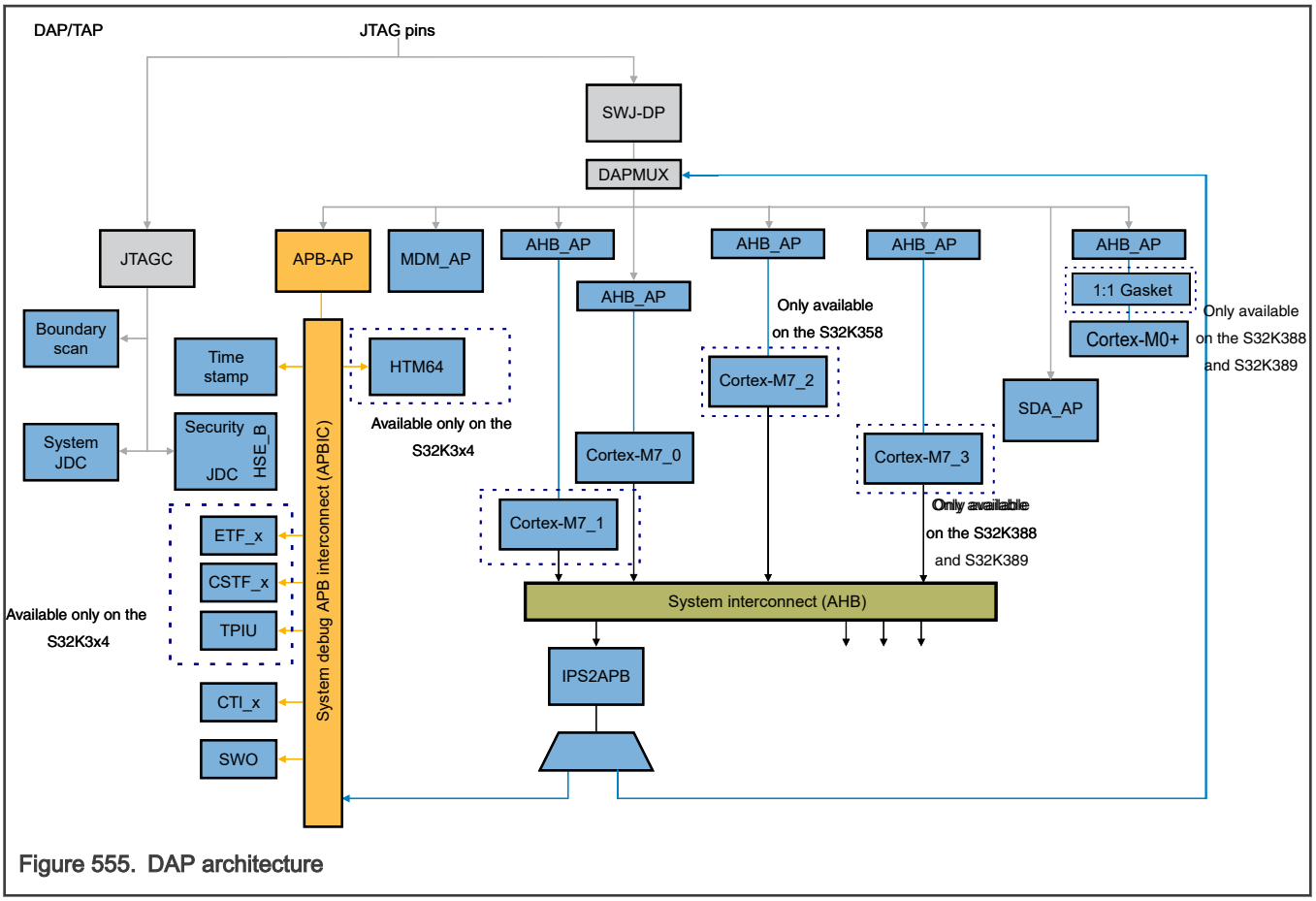


Figure 555. DAP architecture

NOTE

Debug components other than the Cortex-M0+ core are accessible through software path (CORE > AXBS > AIPS > DAPMUX > APs), that is, by directly specifying the memory mapped addresses without doing challenge response.

82.4 Features

This chip includes these features that support the functions listed under each feature type:

- Debug control
  - Is implemented via IEEE 1149.1-compliant JTAG
- Test control
  - Is implemented via IEEE 1149.1-compliant JTAG
  - Uses IEEE 1149.6 extension to IEEE 1149.1
- Trace interface
  - Includes four high-speed data pads and one clock pad up to 125 MHz (125 MB/s throughput per pad)
  - Includes 16 low-speed data pads and one clock pad @25 MHz (100 MB/s throughput per pad)
- Debug security
  - Includes a DAP/TAP interface that controls all debug features and is gated by Cortex M0+ JDC module
  - Includes security modes as described in the "Security" chapter
- Debugging and run control

- Is implemented through stopping points, starting points, breakpoints, and watchpoints
- Each Cortex-M7 core supports eight instruction comparators and a watchpoint unit having four watchpoints

- Trace source
  - Includes Cortex-M7:
    - ETM that provides instruction and data trace
    - ITM that provides DWT, time-stamping, and diagnostic information
- Cross-triggering support
  - Controls run-control options of cores based on other cores
  - Provides a matrix having connections to:
    - HTM via system CTI
    - All Cortex-M7 CPUs
- Ability to view and modify all memory-mapped areas that are not otherwise blocked—includes a system bus debug access port
- Performance monitoring of cores—implements a performance monitoring unit (PMU) on each Cortex core
- Safety
  - Supports monitoring of debug signals to avoid common mode faults
  - Allows checking of erroneous activation of debugging, especially if intrusive (for example, a CPU entering Debug state)
- Timestamps
  - Generates a timestamp bus for distribution to the trace sources
  - Includes a 48-bit binary timestamp bus
  - Clocks timestamp generator by a frequency given in Table 832

82.5 Debug

82.5.1 TAP connectivity

This figure shows a detailed view of TAP connectivity. JTAG select, SWJ-DP, and JTAG-DP are parts of DAP.

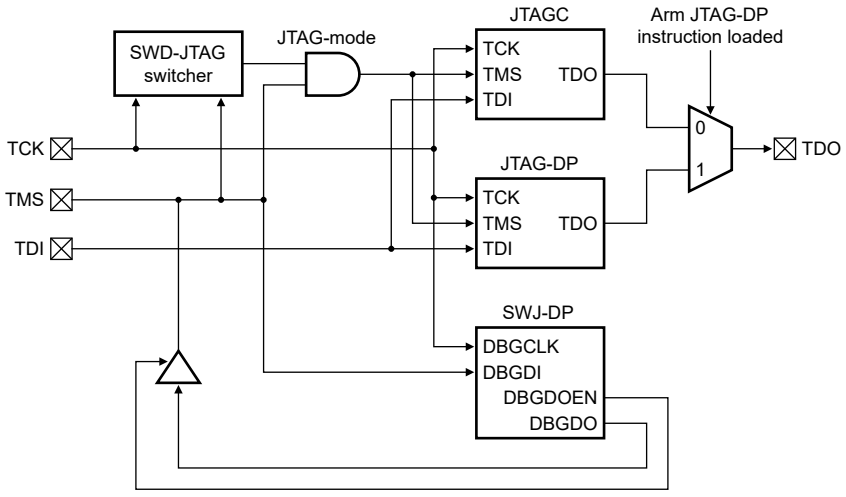


Figure 556. TAP connectivity

The debug port comes out of reset in a standard JTAG mode. It switches to another mode by using the change sequences. After the mode changes, unused debug pins can be reassigned to any of their alternative muxed functions.

JTAG-DP and JTAGC are connected in an overlay scheme and both have an Instruction Register (IR) length of 8 bits.

82.5.2 DAP TAP

DAP is an Arm component that provides multiple-master driving ports. A single external interface port accesses and controls these ports to provide system-wide debug.

The DAP Instruction Register (DAP IR) overlays with the system JTAG Instruction Register (JTAGC IR). Table 825 presents DAP instructions. In addition to the four codes listed in the table, DAP uses BYPASS, which is identical to JTAGC BYPASS and is therefore not shown in the table.

Table 825. DAP IR codes

Code	DAP IR
1111_1000b	ABORT
1111_1010b	DPACC
1111_1011b	APACC
1111_1110b	IDCODE

DAP offers an AHB master interface to access system buses. It also exports the internal DAP bus to extend the access ports. For more information on DAP TAP, see the *Arm Debug Interface Architecture Specification* document available in References.

In this chip:

- The AHB slave ports of all Cortex-M*x* cores provide debugger access to all memory units and registers in the system.
- The new pass-through approach allows the debugger to see a cache coherent view of the memory map for that core.
- The debugger must access the corresponding AHB_AP port to access the AHB-S port of that subsystem.
- XRDC controls system access.
- A core can access the debug components of another core through a DAPMUX to the bus interconnect.
- The exported DAP bus hosts AHB_AP and MDM_AP.
  - MDM_AP hosts system-level JTAG status and control registers. These registers can be used for cross triggering, synchronized debug, and other miscellaneous control and status functions.
  - APB_AP uses an APSEL value of 1h to access any APB-mapped debug modules.

Table 829 describes the access addresses of APB-mapped debug modules. The value of the DAP select signal in the APSEL field of SWJ-DP's Select register selects the access ports in the DAP. Table 826 shows APSEL decoding.

Table 826. DAP master address mapping

DAP master #	DAP component	DAPBUS base address (access from debugger) [1]	System memory map address (access from cores)		Selected port or master	Applicability
			Page number	Base address		
0h	Reserved	0000_0000h	0	4025_0000h	Reserved	-

Table continues on the next page...

Table 826. DAP master address mapping (continued)

DAP master #	DAP component	DAPBUS base address (access from debugger) [1]	System memory map address (access from cores)		Selected port or master	Applicability
			Page number	Base address		
1h	APB_AP to all APB-mapped debug modules (for example, ETFs, HTM, and trace funnels)	0100_0000h	0	4025_0100h	APB_AP to all APB-mapped debug modules (for example, ETFs, HTM, CSTF, and so on)	All (for ETF, HTM, and CSTF see Table 824).
2h	Reserved	0200_0000h	0	4025_0200h	Reserved	-
3h	AHB_AP to Cortex-M7_2 debug modules and chip system memory map	0300_0000h	0	4025_0300h	AHB_AP to Cortex-M7_2 debug modules and chip system memory map	S32K358, S32K338, S32K388, S32K389
4h	AHB_AP to Cortex-M7_0 debug modules and chip system memory map	0400_0000h	0	4025_0400h	AHB_AP to Cortex-M7_0 debug modules and chip system memory map	All
5h	AHB_AP to Cortex-M7_1 debug modules and chip system memory map	0500_0000h	0	4025_0500h	AHB_AP to Cortex-M7_1 debug modules and chip system memory map	S32K344, S32K324, S32K322, S32K358, S32K388, S32K389
6h	MDM_AP	0600_0000h	0	4025_0600h	MDM_AP	All
7h	SDA_AP	0700_0000h	0	4025_4700h	SDA_AP used for challenge-response (CR) in SWJ-DP mode	All
8h	AHB_AP to Cortex-M7_3 debug modules and chip system memory map	0800_0000h	0	4025_0700h	AHB_AP to Cortex-M7_3 debug modules and chip system memory map	S32K388, S32K389
9h—FFh	Reserved (default AP response)	—	—	—	—	—

1. For example, 4025_XXYYh address is provided when accessing access ports (APs) via the core. Here, XX shows the AP select number and YY shows the address to be accessed. If you access MDM_AP via the cores on 4h, you must provide the address 4025_0604h

NOTE

Accessing SDA_AP simultaneously from the core and the debugger is prohibited. If you try to do so, you may receive unpredictable responses to core-initiated transactions (for example, incorrect data read, a failed attempt to write to the register, or a transfer error). Debugger-initiated transactions proceed correctly.

82.5.3 System JTAGC

JTAGC connects in parallel with the TAP controller (JTAG-DP of DAP), which has an IR length of 8 bits. The JTAGC IR codes overlay the ones of the DAP controller. DAP uses four instructions and JTAGC uses the remaining ones. The TAP outputs (TPOs) are multiplexed based on the selected IR code. This chip is fully JTAG-compliant and appears as a single TAP to the JTAG chain.

This table shows the JTAGC IR codes. The instructions that are used by Arm DAP TAP are shown in Table 825.

Table 827. JTAG instructions for JTAGC

Code	JTAGC IR
0000_0000b	IDCODE
0000_0001b	Reserved
0000_0010b	SAMPLE/PRELOAD
0000_0011b	SAMPLE
0000_0100b	EXTEST
0000_0101b	HI-Z
0000_0110b	Reserved
0000_0111b	Reserved
0000_1000b—0000_1001b	Reserved
0000_1010b—0000_1011b	Reserved
0000_1100b	CLAMP
0000_1101b	ENABLE_SOC_DATA1
0000_1110b	Reserved
0000_1111b	Reserved
1000_0000b	Reserved
1000_0001b	Reserved
1000_0010b	Reserved
1000_0011b	Reserved

Table continues on the next page...

Table 827. JTAG instructions for JTAGC (continued)

Code	JTAGC IR
1000_0100b	Reserved
1000_0101b	Reserved
1000_0110b—1000_0111b	Reserved
1000_1000b	Reserved
1000_1001b	Reserved
1000_1010b—1000_1111b	Reserved
1001_0000b	Security JDC
1001_0001b	System JDC
1001_0010b—1001_0111b	Reserved for other system auxiliary clients
1001_1000b—1001_1011b	Reserved
1001_1100b—1011_1111b	Reserved for other system auxiliary clients
1100_0000b	Reserved
1100_0001b	Reserved
1100_0010b—1110_1111b	Reserved
1111_1000b	ABORT (Arm)
1111_1001b	Reserved
1111_1010b	DPACC (Arm)
1111_1011b	APACC (Arm)
1111_1100b	Reserved
1111_1101b	Reserved
1111_1110b	IDCODE (Arm)
1111_1111b	BYPASS

82.5.3.1 Chip JTAG/Target ID

Each chip in the S32K3xx family includes a unique JTAG ID, which must be changed when instantiated with a different die in an SiP. The next table shows the JTAGC ID for this chip.

Table 828. JTAG/Target ID

Chip	PRN	DC	PIN	MIC	IDCODE ID	JTAG ID	Target ID
S32K344	0	26h (38d)	160h (352d)	Eh (14d)	1	0996_001Dh	0995_C01Dh

Table continues on the next page...

Table 828. JTAG/Target ID (continued)

Chip	PRN	DC	PIN	MIC	IDCODE ID	JTAG ID	Target ID
S32K324	0	26h (38d)	160h (352d)	Eh (14d)	1	0996_001Dh	0996_001Dh
S32K314	0	26h (38d)	160h (352d)	Eh (14d)	1	0996_001Dh	0996_001Dh
S32K311/ S32K310	0	26h (38d)	16Ch (364d)	Eh (14d)	1	0996_C01Dh	0996_C01Dh
S32K312	0	26h (38d)	168h (360d)	Eh (14d)	1	0996_801Dh	0996_801Dh
S32K342 / S32K341 / S32K322	0	26h (38d)	164h (356d)	Eh (14d)	1	0996_401Dh	0996_401Dh
S32K358 / S32K348 / S32K338 / S32K328	0	26h (38d)	15Eh (350d)	Eh (14d)	1	1995_E01Dh	1995_E01Dh
S32K388	0	26h(38d)	15Ah (346d)	Eh (14d)	1	0995_A01Dh	0995_A01Dh
S32K389	0	26h(38d)	162h(354d)	Eh (14d)	1	0996_201Dh	0996_201Dh

82.5.3.2 JDC

JDC allows you to access two 32-bit data registers by using the JTAG interface and by software running on one of the CPUs in the chip. These registers exchange data between an internal CPU and an external debug tool.

82.5.4 Peripheral IPG debug implementation

Any core can control a peripheral instance individually. You define the core's control over a peripheral during application development.

For peripheral halt, all individual cores have the same capability and are gated by a dedicated MDM_AP core halt register field. For more information, see the configurations defined in MDM_AP register descriptions.

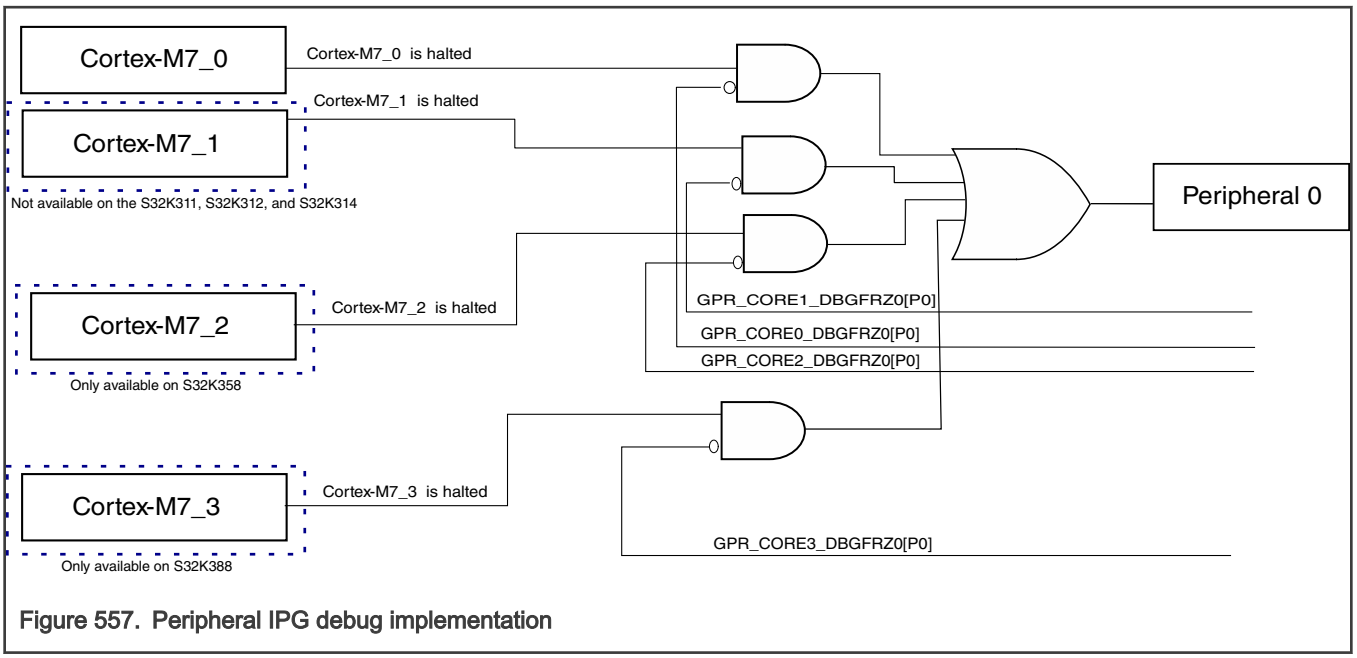


Figure 557. Peripheral IPG debug implementation

82.5.5 Application debugging

This section covers the following two cases:

- Case 1: Debugger connected—application debugging from the first instruction
- Case 2: Debugger not connected

82.5.5.1 Application debugging from first instruction

This chip supports debugging from the first instruction on system power-up, destructive reset, functional reset, self-test done, and standby exit. However, by default, debugging from the first instruction is disabled.

The next figure shows the timing diagram of application debug implementation from first instruction on system power-up or destructive reset.

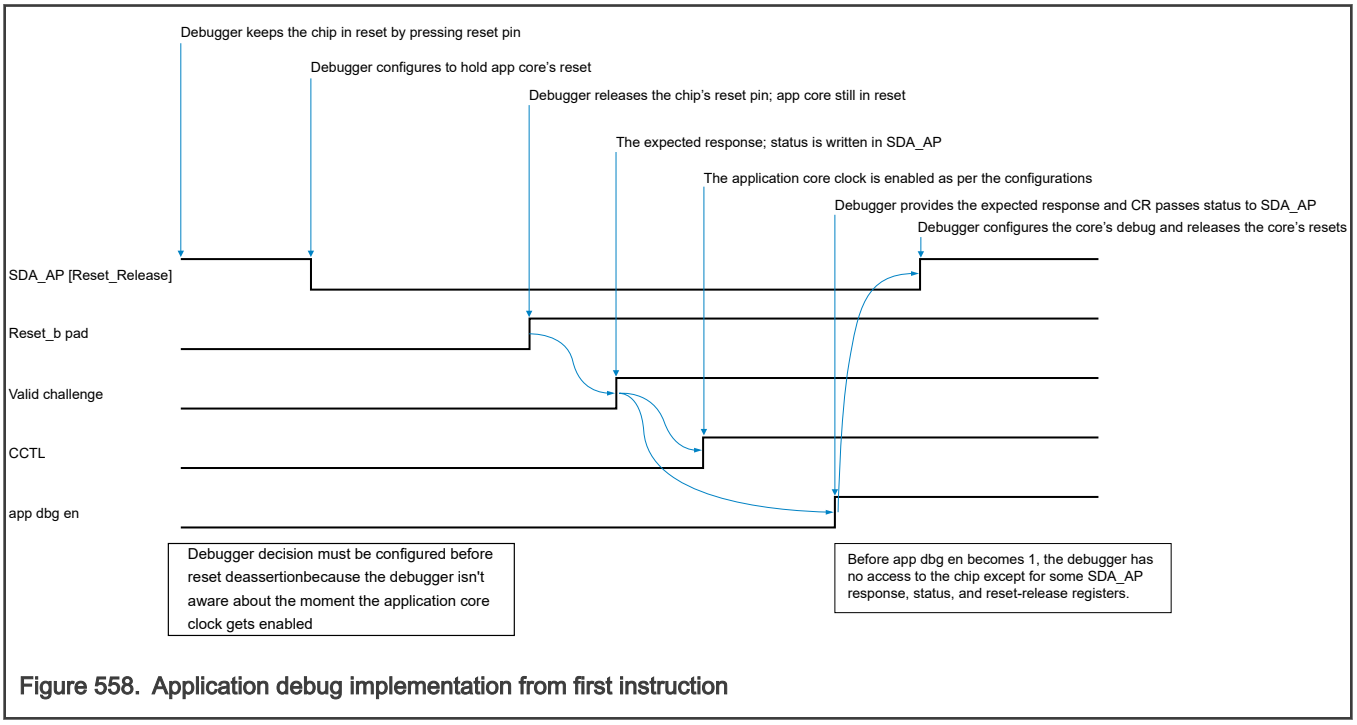


Figure 558. Application debug implementation from first instruction

These are the steps involved in enabling application debug from the first instruction on system power-up or destructive reset. Similar steps are to be performed for application debugging for core Cortex-M7_2 and core Cortex-M7_3 as applicable.

1. After the reset is applied, the debugger provides an option to debug from the first instruction. To do this:
   a. Write 0 to SDAAPRSTCTRL[RSTRELTLCM70] and SDAAPRSTCTRL[RSTRELTLCM71].
   b. Set the reset value of Reset Control (SDAAPRSTCTRL).
2. Cortex M0+ starts after the reset pin is released.
3. Debugger challenge-response (CR) starts.
4. Cortex-M7 cores remain in reset until MC_ME's PRTN0_COREx_PCONF[CCE] field becomes 1 for respective Cortex-M7 core.
5. When BAF or FW writes 1 to MC_ME's PRTN0_CORE0_PCONF[CCE] field for Cortex-M7_0 and to PRTN0_CORE1_PCONF[CCE] field for Cortex-M7_1, and the debugger CR has passed:
   - Configure the core debug registers, then write SDAAPRSTCTRL[RSTRELTLCM70] and SDAAPRSTCTRL[RSTRELTLCM71] to 1 that start the code execution.

To debug from the first instruction during the low-power debug protocol, the debugger:

1. Writes 0 to SDAAPRSTCTRL[RSTRELTLCM71] and SDAAPRSTCTRL[RSTRELTLCM70] for their respective cores.

2. Writes 1 to MDMAPWIRREL[WTRSTRGM].

82.5.5.1.1 Debug from first instruction on functional reset onwards

If you enable debug, on functional reset, the debug connection is retained. That is because the complete debug infrastructure is on destructive reset domain.

82.5.5.1.2 Debug on standby exit by wake-up or functional reset

If you enable debug before standby entry, then the Standby domain stores the final status. The configuration for reset release is reset (SDA_AP.SDAAPRSTCTRL[RSTRELTLCM70], SDA_AP.SDAAPRSTCTRL[RSTRELTLCM71] and SDA_AP.SDAAPRSTCTRL[RSTRELTLCM72]. The reset value of the three fields is 1), which defaults to debug from the first instruction disabled.

You must configure the reset release on standby exit. It uses MC_RGM's low-power debug protocol before ungating the reset deassertion.

82.5.5.2 Debugger not connected

If the debugger is not connected, the value of the DBGPWRUP_ACK signal is 0 as there is no DBGPWRUP_REQ. This results in the following scenario:

1. The booting core writes 1 to MC_ME's PRTN0_CORE0_PCONF[CCE] field for Cortex-M7_0, PRTN0_CORE1_PCONF[CCE] field for Cortex-M7_1, PRTN0_CORE4_PCONF[CCE] field for Cortex-M7_2, and PRTN0_CORE3_PCONF[CCE] field for Cortex-M7_3 (as per core's availability).
2. The application core starts running. In case of Lock-step mode, the checker core executes the same code after a delay of two cycles.

82.5.6 Debugger considerations while flash program/erase

The flash programming can be done by application cores as well as debugger. The debugger can program/erase flash using either of the following two options:

1. Loading the program/erase code to SRAM from the debugger and then executing the program/erase sequence by application code from SRAM.
   - Debugger is connected and authenticated.
   - The debugger loads the code in the form of application binary image into the on-chip SRAM.
   - The debugger then initiates the application core to execute the binary from SRAM.
2. Debugger executing the program/erase sequence.
   - Debugger is connected and authenticated.
   - The debugger then initiates the flash programming by reading/writing the registers involved and following the program/erase sequence.

First option is the recommended because of less execution time. In second option, the debugger is the master and since the debug interface is serial, the execution takes more time.

In some cases, you might experience flash program/erase failure due to flash watchdog timeout when debugger executes the flash program/erase sequence due to the serial interface. Therefore, it is recommended to use the first option. In case second option is required, it should be performed in reduced clocking options (Option E and E2 only. For details, refer to the **Clocking details** section in the **Clocking** chapter).

82.5.7 SWJ-DP sequence for debug authentication

To perform the SWJ-DP sequence for debug authentication, the debugger:

1. Polls the AUTHSTTS[CHALRDY] field until it indicates the challenge status.

2. Reads Key Challenge (KEYCHAL0 - KEYCHAL7) if the challenge is valid and creates an authenticated 256-bit response key.
3. Writes a 256-bit response key to Key Response (KEYRESP0 - KEYRESP7).
4. Indicates that the response is ready by configuring AUTHCTL[HSEAUTHREQ].
5. Checks the status of Authentication Status (AUTHSTTS) to evaluate if the operation is successful.

   If the authentication process is successful and the debugger has write access to Debug Enable Control (DBGENCTRL), the challenge or response is considered successful too.

NOTE

SDA_AP supports challenge and response based on both JTAG and SWJ-DP modes. See SDAAPGENCTRL0[JTAG_CR_EN] for details.

82.6 APB memory map

You can access the debug registers via the APB_AP bus. The next table shows all the addresses for the CoreSight APB components and the addressing used for accessing the DAP components via the memory interface. You can also access all APB registers from the processing cores.

Table 829. APB components mapping

Debug APB component	APBIC base address (access from debugger)	System memory map address (access from cores)		Memory allocation (KB)	APB-AP0 slot number	Applicability
		Page number	Base address			
APB_AP ROM table	F8h	0	4024_00F8h	4	APBIC base	All
Funnel 0	1000h	0	4024_1000h	4	0	ETM/ ITM:S32K344, S32K324, S32K314 S32K358, S32K348, S32K338, S32K328, S32K388, S32K389 ITM: S32K312, S32K310, S32K311, S32K342, S32K341, S32K322
Funnel 1	2000h	0	4024_2000h	4	1	S32K344, S32K324, S32K314 S32K358, S32K348, S32K338, S32K328,

Table continues on the next page...

Table 829. APB components mapping (continued)

Debug APB component	APBIC base address (access from debugger)	System memory map address (access from cores)		Memory allocation (KB)	APB-AP0 slot number	Applicability
		Page number	Base address			
						S32K388, S32K389
Funnel 2	3000h	0	4024_3000h	4	2	S32K344, S32K324, S32K314 S32K358, S32K348, S32K338, S32K328, S32K388, S32K389
CM7_cluster_ETF_ETMI	4000h	1	4024_4000h	4	3	S32K344, S32K324, S32K314 S32K358, S32K348, S32K338, S32K328, S32K388, S32K389
CM7_cluster_ETF_ETMD	5000h	1	4024_5000h	4	4	S32K344, S32K324, S32K314 S32K358, S32K348, S32K338, S32K328, S32K388, S32K389
HTM ETF	6000h	1	4024_6000h	4	5	S32K344, S32K324, S32K314 S32K358, S32K348, S32K338, S32K328, S32K388, S32K389
Shared_system_ETF	7000h	1	4024_7000h	4	6	S32K344, S32K324, S32K314 S32K358, S32K348, S32K338,

Table continues on the next page...

Table 829. APB components mapping (continued)

Debug APB component	APBIC base address (access from debugger)	System memory map address (access from cores)		Memory allocation (KB)	APB-AP0 slot number	Applicability
		Page number	Base address			
						S32K328, S32K388, S32K389
HTM 0	8000h	2	4024_8000h	4	7	S32K344, S32K324, S32K314 S32K358, S32K348, S32K338, S32K328, S32K388, S32K389
HTM 0 CTI	9000h	2	4024_9000h	4	8	S32K358, S32K348, S32K338, S32K328, S32K344, S32K324, S32K314, S32K388, S32K389
TPIU	A000h	2	4024_A000h	4	9	S32K344, S32K324, S32K314 S32K358, S32K348, S32K338, S32K328, S32K388, S32K389
System SWO	B000h	2	4024_B000h	4	10	All
Timestamp CTL	C000h	3	4024_C000h	4	11	All

Funnel 0 should be configure to buffer the data coming from the different sources, to avoid padding data with null packets that affect the bandwidth.

82.7 Trace

82.7.1 Trace modules and connectivity

The Trace subsystem:

- Combines trace data from all internal clients that generate trace information
- Includes a 32-bit TPIU
- Includes these components:

— ATB
— ATBR
— CSTF
— Debug APB
— ETF
— TPIU

Multiple options for trace output allow parallel tracing. Trace information can be read from the trace interface or the DAP interface, and traces can be alternatively read out from ETF at a slow speed via APB_AP. Table 830 shows EFT sizes.

Table 830. ETF sizes

FIFO	Memory interface data width (in bits)	Size (KB)Applicable for all, except S32K388/S32K389	Size (KB)Applicable for S32K388/S32K389
Cortex-M7 ETM/ITM cluster ETF	64	1	4
Cortex-M7 ETMD cluster ETF	128	2	4
HTM ETF	64	1	2
Shared system ETF	64	2	4

NOTE

- The DMA-HTM trace supports four words at 80 MHz. For HTM trace, both DBGENCTRL[GSPNIDEN] and DBGENCTRL[GSPIDEN] must be 1.
- Enabling any one of data trace causes overflow inside ETM and trace packets get dropped. Enabling instruction trace of both the core simultaneously does not cause any overflow.

Control (MDMAPCTL) provides fields to override the speed control (see Table 831 for details) from some of the trace sinks (and TPIU). The complete trace pipeline bandwidth is limited by the slowest sink component. The default settings of these fields allow maximum bandwidth for the TPIU to trace. When tracing to memory (if supported), the fields may need to be changed.

Table 831. Trace output overrides

Trace destination	SWO_override	TPIU_override
TPIU	1	0

The trace sources of the chip are the cores and their related modules. Trace funnels exist for all possible trace clients. For more information on the various components in the trace bus connectivity, see the *CoreSight Components Technical Reference Manual* (available in References). See Table 832 for details on funnel assignments.

The ATBR is integrated to send a shared funnel output across TPIU. This figure illustrates the chip's detailed trace architecture.

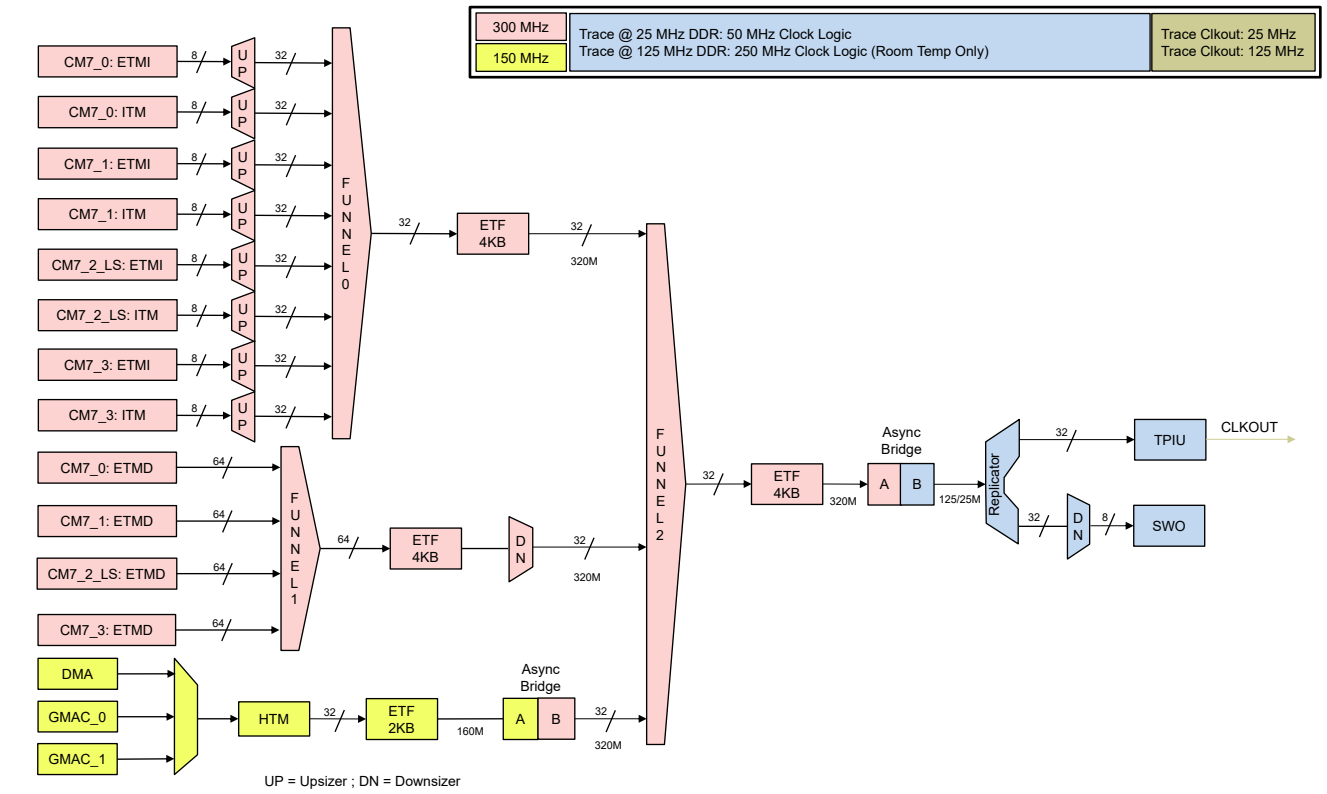


Figure 559. S32K389 detailed trace architecture

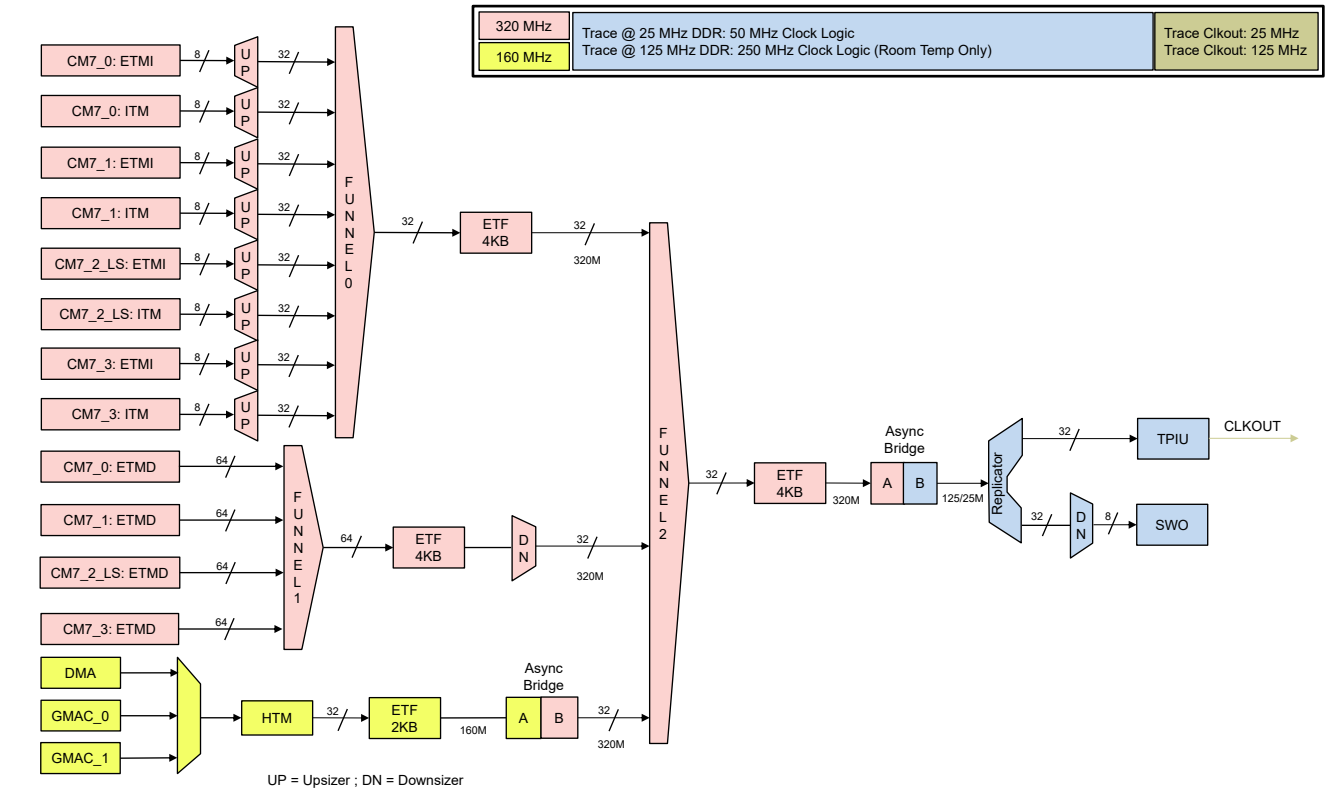


Figure 560. S32K388 detailed trace architecture

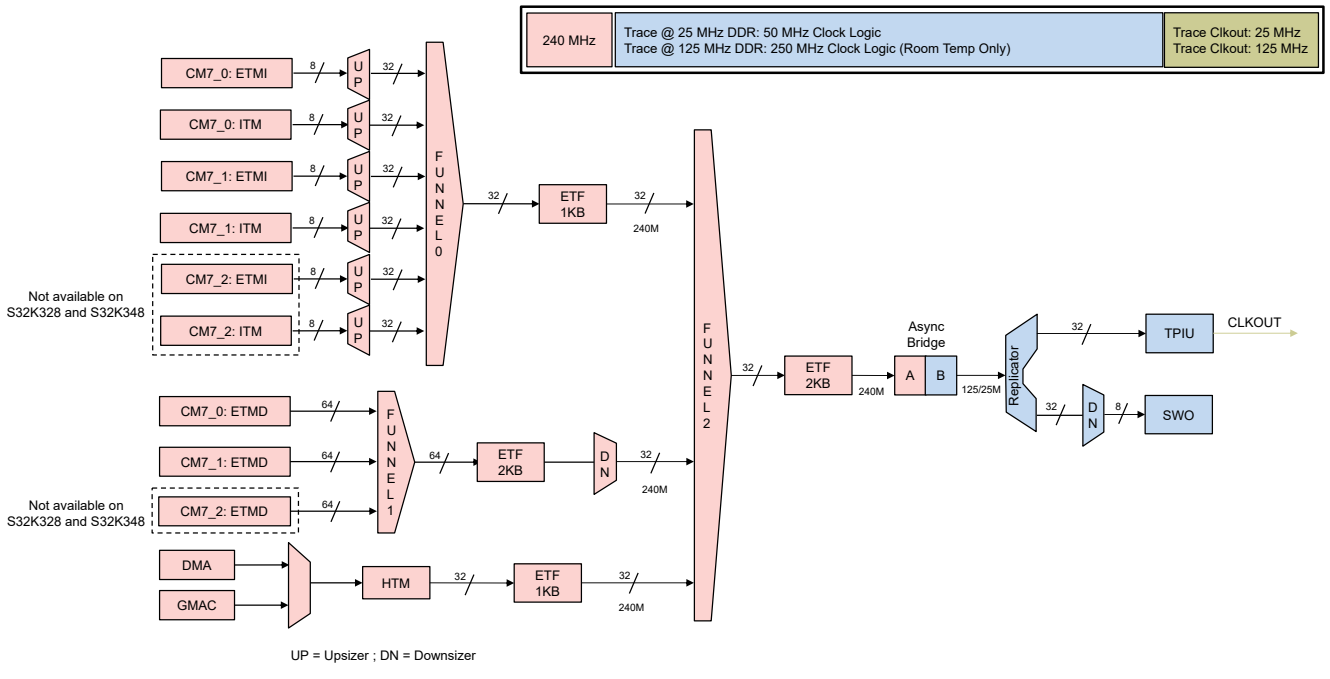


Figure 561. S32K358, S32K348, S32K338, and S32K328 detailed trace architecture

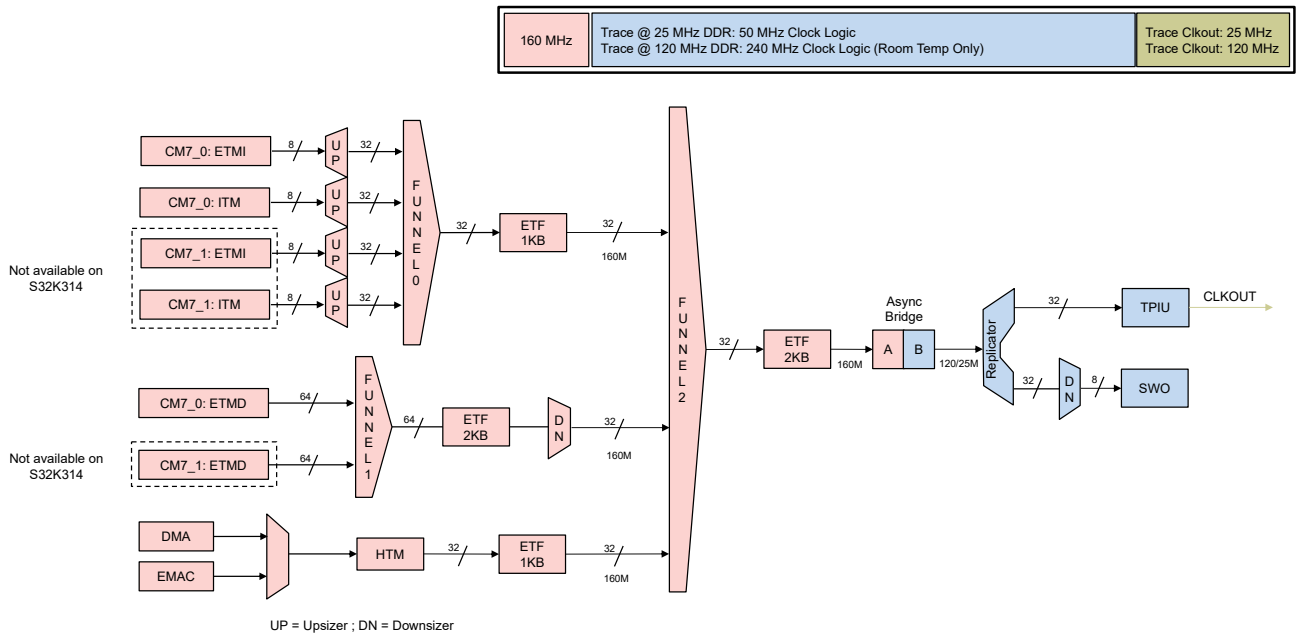


Figure 562. S32K344, S32K324, and S32K314 detailed trace architecture

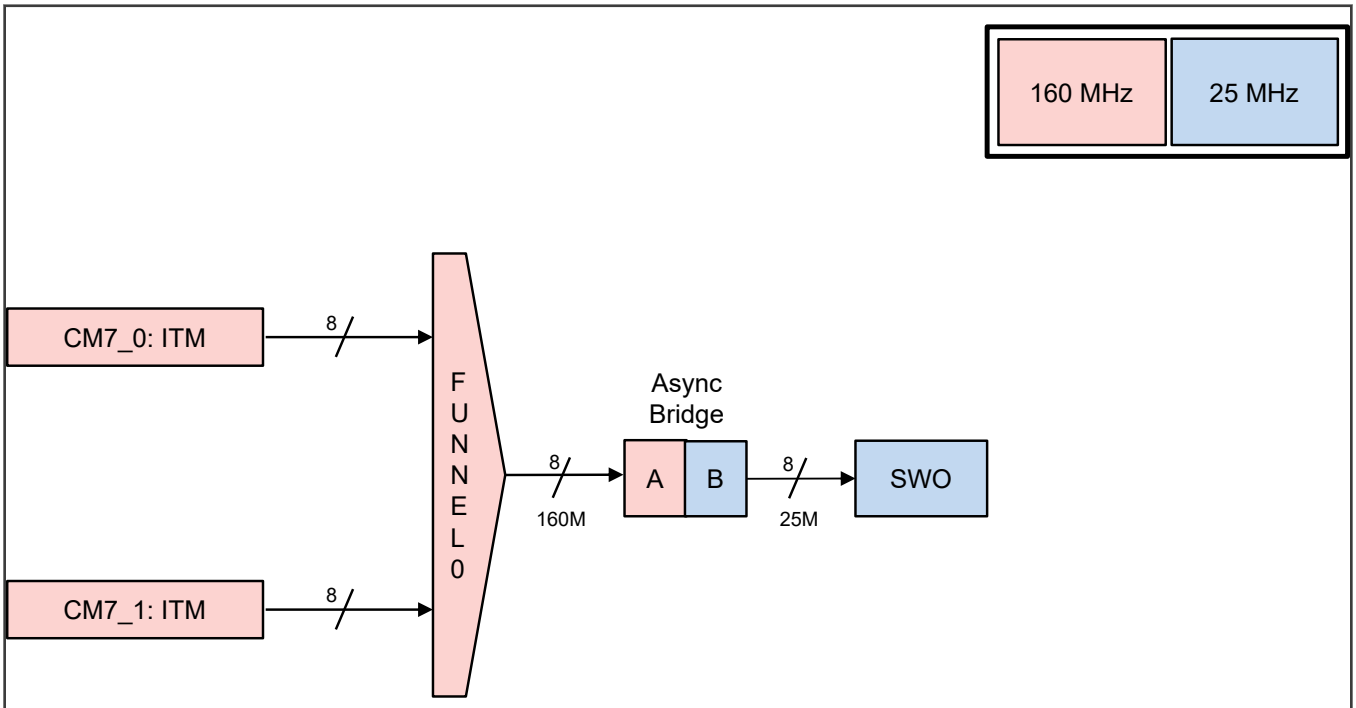


Figure 563. S32K342, S32K341, and S32K322 detailed trace architecture

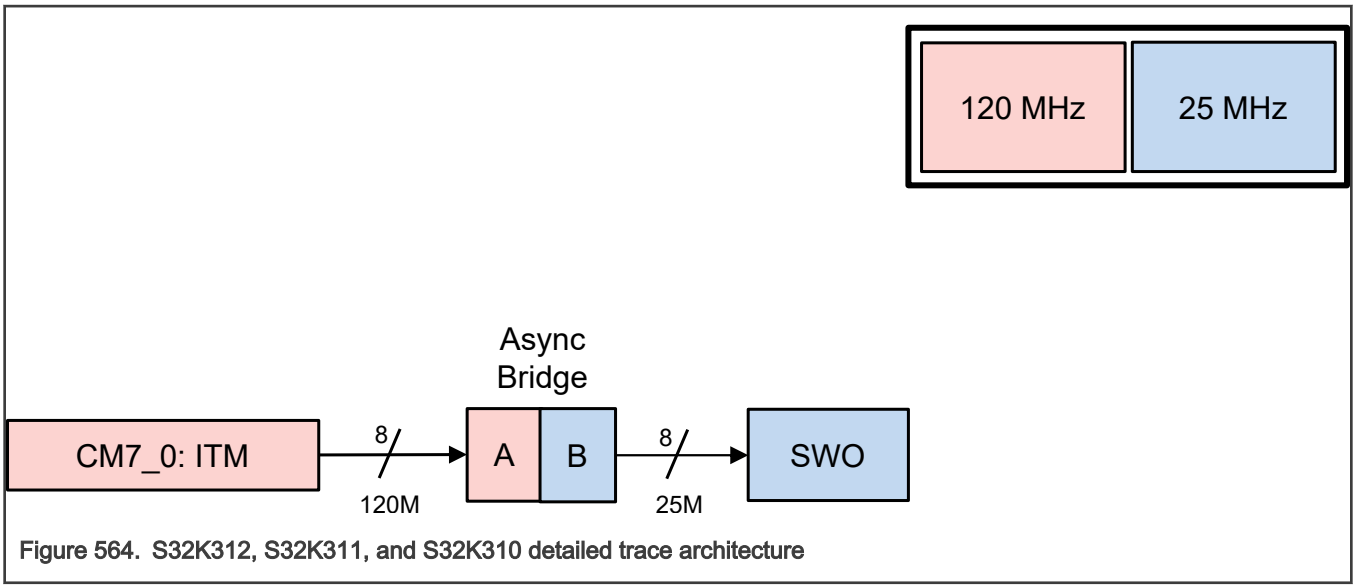


Figure 564. S32K312, S32K311, and S32K310 detailed trace architecture

Table 832. Funnel assignments

Funnel	Port	Input	Frequency (in MHz)			
			S32K342/S32K341/S32K322 S32K344/S32K324/S32K314	S32K328/S32K338/S32K348/S32K358	S32K388	S32K389
0	0	Cortex-M7_0 ETMI	160	240	320	300
0	1	Cortex-M7_0 ITM	160	240	320	300
0	2	Cortex-M7_1 ETMI	160	240	320	300
0	3	Cortex-M7_1 ITM	160	240	320	300
0	4	Cortex-M7_2 ETMI	-	240	320	300
0	5	Cortex-M7_2 ITM	-	240	320	300
0	6	Cortex-M7_3 ETMI	-	-	320	300
0	7	Cortex-M7_3 ITM	-	-	320	300
1	0	Cortex-M7_0 ETMD	160	240	320	300
1	1	Cortex-M7_1 ETMD	160	240	320	300
1	2	Cortex-M7_2 ETMD	-	240	320	300

Table continues on the next page...

Table 832. Funnel assignments (continued)

Funnel	Port	Input	Frequency (in MHz)			
			S32K342/S32K341/S32K322 S32K344/S32K324/S32K314	S32K328/S32K338/S32K348/S32K358	S32K388	S32K389
1	3	Cortex-M7_3 ETMD	-	-	320	300
1	4-7	Reserved	-	-	-	-
2	0	Cortex-M7 instruction ETF through an asynchronous bridge	160	240	320	300
2	1	Cortex-M7 data ETF through an asynchronous bridge	160	240	320	300
2	2	HTM ETF through an asynchronous bridge	160	240	320	300
2	3-7	Reserved	-	-	-	-

82.7.1.1 Chip's bus trace client

Chips in the S32K3xx family include a bus trace client.

HTM provides the address and data trace information about AXBS buses. The information from an HTM can be used with the debugger to enable easy, accurate debugging on AXBS-based embedded systems. The chip implements an HTM64 configuration to trace 64-bit AXBS masters in the system. To simplify the implementation, instead of tracing the individual ports of various AXBS masters and slaves, which may be running at different frequencies, HTM64 snoops the AXBS crossbar master ports that are all synchronous with a 160 MHz system clock. This table provides details related to the HTM input connectivity.

Table 833. HTM connections

HTM64 port	AHB crossbar port
HTMBUSSELECT0	M0 AXBS_Lite XBIC (DMA)
HTMBUSSELECT1	M3 AXBS (Ethernet)
HTMBUSSELECT2[1]	M6 AXBS (Ethernet 2)

1. Only applicable for S32K388 and S32K389.

82.7.1.2 TPIU interface

A standard 16-bit parallel TPIU is integrated into the debug subsystem. Chips in the S32K3xx family generate the trace via the trace port.

To prevent instruction trace from being dropped in a multi-core environment, a TPIU throughput of 55.2 MB/s or higher must be maintained. You could use these pin and frequency combinations:

- Four high-speed data + one clock pads @120 MHz (240 Mbit/s throughput per pad) totaling 120 MB/s

- 16 low-speed data + one clock pads @25 MHz (50 Mbit/s throughput per pad) totaling 100 MB/s

High-end chips from the S32K3xx family support at least 56 MB/s of trace throughput on the TPIU interface.

Table 834. Throughput for S32K388 and S32K389

Core	Throughput value
Cortex-M7_0	31.06 MB/s
Cortex-M7_1	31.06 MB/s
Cortex-M7_2	31.06 MB/s
Cortex-M7_3	31.06 MB/s
Total	124.24 MB/s

Table 835. Throughput for all chips except S32K388 and S32K389

Core	Throughput value	Applicability
Cortex-M7_0	27.6 MB/s	S32K344, S32K324, S32K314 S32K358, S32K348, S32K338, S32K328
Cortex-M7_1	27.6 MB/s	S32K338, S32K328, S32K324
Cortex-M7_2	27.6 MB/s	S32K358, S32K338
Total	82.8 MB/s	

82.7.1.3 TPIU flush

To allow the chip to enter Standby mode, software executes an WFI instruction indicating Standby entry. When the chip enters this mode, MC_RGM asserts a trace flush request to the TPIU and waits for a trace flush done signal from the TPIU before requesting MC_PCU to proceed. The TPIU and debug infrastructure clocks can be gated after this. Because the TPIU is sourced from system clock, the clock must not be gated until this point.

82.7.1.4 Trace through reset

All debug and trace components (HTM, ETM, ETF, and so on) are on destructive reset. When tracing through reset, the clock switches from PLL to FIRC. It is recommended to preserve the following pad configurations across reset, which are performed through MDMAPCTL[DBGRSTFASTPAD] and MDMAPCTL[DBGRSTSLOWPAD]:

- Four high-speed data + one clock pads @120 MHz (240 Mbits/s throughput per pad) totaling 120 MB/s
- 16 low-speed data + one clock pads @25 MHz (50 Mbits/s throughput per pad) totaling 100 MB/s

The implementation involves writing 1 to MDMAPCTL[DBGRSTFASTPAD] or MDMAPCTL[DBGRSTSLOWPAD], depending on which set of trace pads you need to enable. These fields also control:

- The trace pad mux.
- The "obe" of the trace pads, which, if configured, is retained over functional reset because the reset is controlled using the above-mentioned corresponding fields.

82.8 Embedded cross trigger (ECT)

ECT allows multi-core run control and trace cross-triggering, such as synchronous stop-start for all cores or trigger trace on a trigger event from another core or module. See the *CoreSight Components Technical Reference Manual* (available in References) for detailed information on ECT.

ECT architecture involves CTMs and CTIs. The CTIs provide a cross-triggering interface between the cores and other debug and trace modules. The channels of these CTIs are interconnected using CTMs, as shown in this figure.

The following figure shows ECT

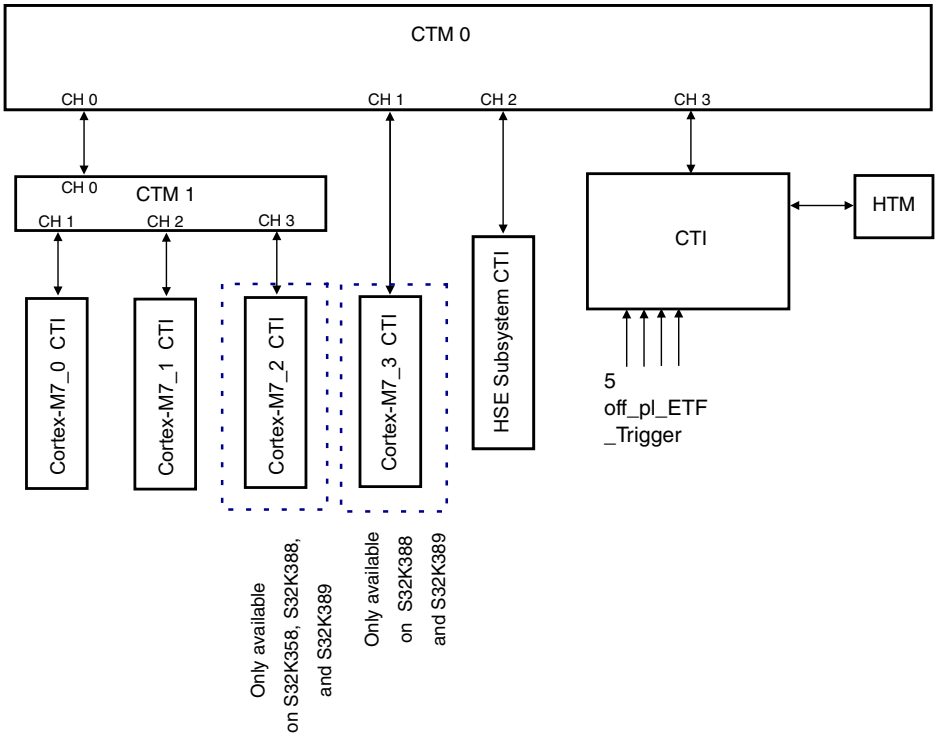


Figure 565. ECT

82.8.1 CTI assignments

Table 836. CTI assignments

CTI instance	Trigger number	Trigger in	Trigger out
CTI_Cortex-M7_0	7	ETM event output 3	Processor restart
	6	ETM event output 2	ETM event input 3
	5	ETM event output 1	ETM event input 2
	4	ETM event output 0	ETM event input 1
	3	DWT comparator output 2	ETM event input 0
	2	DWT comparator output 1	Interrupt request 1
	1	DWT comparator output 0	Interrupt request 0
	0	Processor halted	Processor debug request
CTI_Cortex-M7_1	7	ETM event output 3	Processor restart
	6	ETM event output 2	ETM event input 3

Table continues on the next page...

Table 836. CTI assignments (continued)

CTI instance	Trigger number	Trigger in	Trigger out
	5	ETM event output 1	ETM event input 2
	4	ETM event output 0	ETM event input 1
	3	DWT comparator output 2	ETM event input 0
	2	DWT comparator output 1	Interrupt request 1
	1	DWT comparator output 0	Interrupt request 0
	0	Processor halted	Processor debug request
CTI_Cortex-M7_2	7	ETM event output 3	Processor restart
	6	ETM event output 2	ETM event input 3
	5	ETM event output 1	ETM event input 2
	4	ETM event output 0	ETM event input 1
	3	DWT comparator output 2	ETM event input 0
	2	DWT comparator output 1	Interrupt request 1
	1	DWT comparator output 0	Interrupt request 0
	0	Processor halted	Processor debug request
CTI_Cortex-M7_3	7	ETM event output 3	Processor restart
	6	ETM event output 2	ETM event input 3
	5	ETM event output 1	ETM event input 2
	4	ETM event output 0	ETM event input 1
	3	DWT comparator output 2	ETM event input 0
	1	DWT comparator output 0	Interrupt request 0
	0	Processor halted	Processor debug request
	2	DWT comparator output 1	Interrupt request 1
CTI_0	7	Reserved (grounded)	Reserved (no connection)
	6	ETF_3 full	ETF_3 trigger input
	5	ETF_2 full	ETF_2 trigger input
	4	ETF_1 full	ETF_1 trigger input
	3	ETF_0 full	ETF_0 trigger input
	2	HTM trigger out 2	HTM trigger in 1
	1	HTM trigger out 1	HTM trigger in 0
	0	HTM trigger out 0	Reserved

82.9 Low-power debug handshake protocol

The debugger must perform this sequence to enter or exit Standby mode, if the debugger handshake is enabled:

1. Power on DAP.

The debugger connection is established.

2. Configure the Debug subsystem for the relevant operations.
3. Write 1 to MDMAPWIREN[LWPWREN] to gate entry into Standby mode with the debugger handshake.
4. Disable POR_WDG for monitoring the entry to or exit from Standby mode by writing 1 to DCM's DCMRWP1[8] field. This is required for low-power debug handshake because debugger configurations can be more time consuming than the POR_WDG threshold levels and can raise a false POR_WDG event.
5. Write 1 to DCMRWF1[STANDBY_IO_CONFIG] in the Device Configuration Module (DCM). This causes padkeeping to be disabled on standby entry itself without any software dependency. This is needed in case of low-power debug since otherwise the padkeeping on TDO would result in no debugger communication. In other cases, DCMRWF1[STANDBY_IO_CONFIG] should be configured as 0 before standby entry.
6. Initiate entry into Standby mode. See the "Power Management" chapter for the Standby mode entry sequence and configurations.
7. Identify whether the low-power debug is enabled on the low-power entry.
   - If low-power debug is enabled and TPIU is enabled too, trace flush starts, and the debugger acknowledges the following:
     - Low-power debug traces
     - DAP-related configuration reception and context saving

       After this, the chip enters Standby mode.
   - If TPIU is disabled:
     - The debugger acknowledges DAP-related configuration reception and context saving by writing 1 to MDMAPWIRREL[PRVNTRSTRGM].

       After this, the chip enters Standby mode.
   - If low-power debug is disabled, the chip enters Standby mode without waiting for debugger acknowledgment.

   On any wakeup event, the chip starts the Standby mode exit sequence described in the "Power Management" chapter. After the chip exits Standby mode, the debugger connection is restored. By this time, the debugger is already aware that it has enabled the low-power debug handshake. In this case, the debugger must poll MDMAPSTTS[DESTRST] to check whether:
   - The debug infrastructure is out of reset.
   - The DAP connection can be established.

   After DAP is powered on (the debugger connection is established), the debugger:
   a. Reconfigures the debug and trace attributes using the fields in DBGENCTRL
   b. Writes 1 to MDMAPWIRREL[WTRSTRGM] after the debugger trace context is restored

      The chip exits Standby mode.

      If low-power debug is not configured, the chip exits Standby mode without waiting for the debugger to write to MDMAPWIRREL[WTRSTRGM].
8. Reconfigure the debug and trace configuration in SDA_AP register descriptions after the chip exits Standby mode.

NOTE

If the chip wakes up from Standby mode through pad reset, the debugger must perform a CR again.

82.10 Debug resets

The debug subsystem follows this sequence on the source of reset:

1. POR resets the complete debug logic.

2. The destructive reset resets all types of debug logic except JTAGC.

Conversely, the debug subsystem can generate a system reset using these mechanisms:

- System destructive reset defined in Control (MDMAPCTL) that allows the debugger to provide the destructive reset to the system. The debugger loses connection to the system with this reset.
- System functional reset defined in Control (MDMAPCTL) that allows the debugger to hold the system in functional reset.

To program various debug registers, the functional clocks must be enabled. All debug and trace components must be on destructive reset.

82.11 Debug across device LifeCycles

The debug access to the system is available in early lifecycles and subsequently based on NVM configuration settings.

After the debug is set to 'Trusted', the system access is allowed after a successful authorization step between the debugger and the system.

The authorization provides provisions to allow debug for Cortex M0+ core and Cortex M7 cores.

Following table shows the debug access based on various configuration bits.

Table 837. Debug access based on LifeCycle and bit configurations

LifeCycle	Debug access
	Appl Core Debug
MCU_PROD	Open
	Open
CUST_DEL	Open
	Open
OEM_PROD	Closed
	Trusted
	Trusted
	Disabled
IN_FIELD	Closed
	Trusted
	Trusted
	Disabled
PRE_FA	Trusted
	Trusted
FA	Open

NOTE

1. **Open:** Debug is always possible
2. **Trusted:** Debug is possible after successful authentication (Challenge/Response handshake with correct credentials)
3. During functional reset, the debug-enable will retain its last status, while DCM scans NVM for lifecycle and DCF values. Debug status will be re-evaluated once dcm_done gets 1.
4. During temporary advancement of lifecycle, debug-en will immediately reflect the status based on updated lifecycle/DCF bits.

82.12 Pin interface

This table presents a summary of functional and power pins that are used for debugging purposes.

Table 838. Pin interface

Pin type	Pins	Number of pins (balls)[1]	Nominal voltage
Functional JTAG	JCOMP, TCK, TMS, TDI, TDO	Five (only TDI and TDO can be multiplexed with GPIO or other functions)	3.3 V, 5 V
Functional trace pins (parallel trace)	TRACE_CLK	One	3.3 V, 5 V
	TRACE_D[15:0]	16	3.3 V, 5 V
Ground	VSS	See the IOMUX file attached to this document for information on the number of VSS pins in various packages.	0 V

1. The terms pins and balls are used interchangeably. Some chip packages include pins and others include balls.

NOTE

Fast Trace pad TRACE_D[15:0] is available only in S32K344.

82.12.1 Debug port and pin descriptions

The pads to which the debug signals are mapped operate using the JTAG functionality out of reset but can later be reassigned to their alternate functionalities. TDI and TDO can operate as alternate GPIO functions. See this table for pin assignments in different modes.

Table 839. Debug port pins

Pin name	JTAG debug port		Internal pullup or pulldown logic
	Type	Description	
TMS	I/O	JTAG test mode selection (TMS)	Pullup
TCK/SWCLK	I	JTAG test clock (TCK)	Pulldown
TDI	I	JTAG test data input (TDI)	Pullup
TDO/TRACESWO	O	JTAG test data output (TDO)	Not connected

82.12.2 Trace port pin descriptions

This chip generates trace via TPIU that transmits the trace data out of the chip over a parallel trace port. The trace port consists of an ETM trace clock and 16 parallel trace data outputs. The Arm optional trace port control (TRACECTRL), debug request (DBGRQ), and debug acknowledge (DBGACK) signals are not implemented.

Table 840. Trace output port pins

Pin name	Description
TRACE_DATA00	ETM parallel trace data output 01
TRACE_DATA01	ETM parallel trace data output 01
TRACE_DATA02	ETM parallel trace data output 02
TRACE_DATA03	ETM parallel trace data output 03
TRACE_DATA04	ETM parallel trace data output 04
TRACE_DATA05	ETM parallel trace data output 05
TRACE_DATA06	ETM parallel trace data output 06
TRACE_DATA07	ETM parallel trace data output 07
TRACE_DATA08	ETM parallel trace data output 08
TRACE_DATA09	ETM parallel trace data output 09
TRACE_DATA10	ETM parallel trace data output 10
TRACE_DATA11	ETM parallel trace data output 11
TRACE_DATA12	ETM parallel trace data output 12
TRACE_DATA13	ETM parallel trace data output 13
TRACE_DATA14	ETM parallel trace data output 14
TRACE_DATA15	ETM parallel trace data output 15
TRACE_CLK	ETM parallel trace clock output

NOTE

ETM is supported in S32K344 only

NOTE

By default, Rx pins float and are not pulled inside. An internal active pulldown logic exists only when you enable Rx via software (IBE).

82.13 Timestamp distribution network

The timestamp distribution network uses CoreSight timestamp components to generate a 48-bit timestamp value for the trace sources, as shown in the next figure. The CoreSight timestamp generator generates a 64-bit counter value, but only the least significant 48 bits are distributed to the trace sources. A 7-bit narrow timestamp is derived from the 48-bit timestamp and distributed to the trace client, where a decoder regenerates the 48-bit timestamp. The generator must be programmed, and you could find the related programming information in the *CoreSight Components Technical Reference Manual* (available in References).

This chip supports 48-bit timestamping. The generator operates at a frequency of up to 320 MHz and gives a 64-bit timestamp value. .

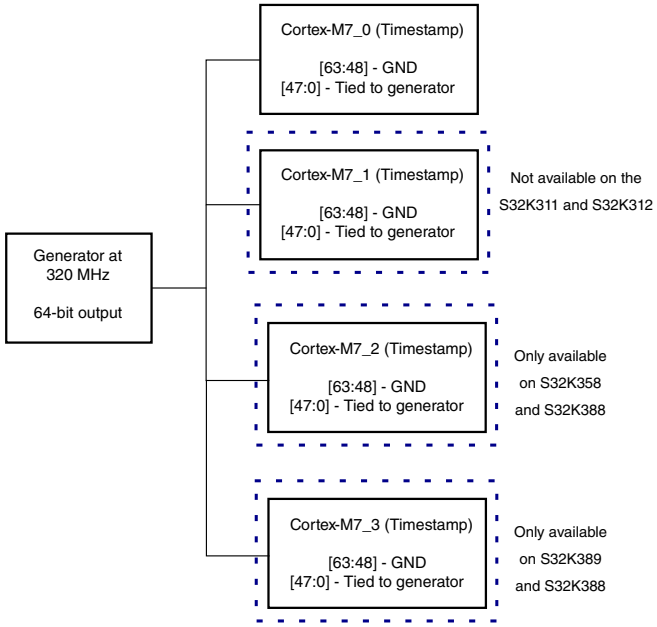


Figure 566. Timestamp distribution network

82.14 Peripheral debug freeze register descriptions

Implement the logic provided in this section for each peripheral instance supporting the debug operation.

NOTE

Debug freeze is enabled for all the peripherals so that as soon as the core halts, peripherals can be frozen to support debugging from the first instruction.

For register details, refer to the following registers in the Device Configuration Module (DCM):

- Read Write GPR On Destructive Reset Register (DCMRWD6)
- Read Write GPR On Destructive Reset Register (DCMRWD7)
- Read Write GPR On Destructive Reset Register (DCMRWD8)
- Read Write GPR On Destructive Reset Register (DCMRWD9)

82.15 MDM_AP register descriptions

The debugger has access to the status and control elements implemented as registers in MDM_AP, which is selected by APSEL (6h) on the DAP bus. These registers provide additional control and status information for typical debug, cross-triggering, and run-control scenarios. Also, the register fields provide a way for the debugger to get the updated status of the core without initiating a bus transaction across the crossbar switch, thus remaining less intrusive during a debug session.

MDM_AP is accessible as DAP (see DAP TAP for details).

82.15.1 MDM_AP memory map

MDM_AP base address: 4025_0600h

Offset	Register	Width (In bits)	Access	Reset value
0h	Status (MDMAPSTTS)	32	R	F800_0000h
4h	Control (MDMAPCTL)	32	RW	0640_0000h
30h	WIR Enable (MDMAPWIREN)	32	RW	0000_0000h
38h	MDM AP WIR Release (MDMAPWIRREL)	32	RW	0000_0000h

82.15.2 Status (MDMAPSTTS)

Offset

Register	Offset
MDMAPSTTS	0h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CM72 DBG...	Reserved	CM71 DBG...	CM70 DBG...	CM73 DBG...	CM73 SLP...	CM73 DPS...	CM73 HLT	CM0P SLP...	CM72 SLP...	CM71 SLP...	CM70 SLP...	CM0P DPS...	CM72 DPS...	CM71 DPS...	CM70 DPS...
W																
Reset	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	HSEHLT	CM72 HLT	CM71 HLT	CM70 HLT	Reserved									FUNC RST	DEST RST	Reserved
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 CM72DBGRSTRD	Cortex-M7_2 Debug Restarted Indicates if Cortex-M7_2 has returned to Normal mode from Debug mode. 0b - In Debug mode 1b - In Normal mode
30 —	Reserved
29	Cortex-M7_1 Debug Restarted

Table continues on the next page...

Table continued from the previous page...

Field	Function
CM71DBGRSTRD	Indicates if Cortex-M7_1 has returned to Normal mode from Debug mode. 0b - In Debug mode 1b - In Normal mode
28 CM70DBGRSTRD	Cortex-M7_0 Debug Restarted Indicates if Cortex-M7_0 has returned to Normal mode from Debug mode. 0b - In Debug mode 1b - In Normal mode
27 CM73DBGRSTRD	Cortex-M7_3 Debug Restarted Indicates if Cortex-M7_3 has returned to Normal mode from Debug mode. 0b - In Debug mode 1b - In Normal mode
26 CM73SLPNG	CM7_3 Sleeping Indicates if Cortex-M7_3 is in Sleep mode. 0b - Not in Sleep mode 1b - In Sleep mode
25 CM73DPSLP	Cortex-M7_3 Deep Sleep Indicates if Cortex-M7_3 is in Deep Sleep mode. 0b - Not in Deep Sleep mode 1b - In Deep Sleep mode
24 CM73HLT	CM7_3 Debug Halted Indicates if Cortex-M7_3 is halted because of entry into Debug mode. 0b - Core is not halted 1b - Core is halted
23 CM0PSLPNG	Cortex-M0+ Sleeping Indicates if Cortex-M0+ is in Sleep mode. 0b - Not in Sleep mode 1b - In Sleep mode
22 CM72SLPNG	CM7_2 Sleeping Indicates if Cortex-M7_2 is in Sleep mode. 0b - Not in Sleep mode 1b - In Sleep mode

Table continues on the next page...

Table continued from the previous page...

Field	Function
21 CM71SLPNG	CM7_1 Sleeping Indicates if Cortex-M7_1 is in Sleep mode. 0b - Not in Sleep mode 1b - In Sleep mode
20 CM70SLPNG	Cortex-M7_0 Sleeping Indicates if Cortex-M7_0 is in Sleep mode. 0b - Not in Sleep mode 1b - In Sleep mode
19 CM0PDPSLP	Cortex-M0+ Deep Sleep Indicates if Cortex-M0+ is in Deep Sleep mode. 0b - Not in Deep Sleep mode 1b - In Deep Sleep mode
18 CM72DPSLP	Cortex-M7_2 Deep Sleep Indicates if Cortex-M7_2 is in Deep Sleep mode. 0b - Not in Deep Sleep mode 1b - In Deep Sleep mode
17 CM71DPSLP	Cortex-M7_1 Deep Sleep Indicates if Cortex-M7_1 is in Deep Sleep mode. 0b - Not in Deep Sleep mode 1b - In Deep Sleep mode
16 CM70DPSLP	Cortex-M7_0 Deep Sleep Indicates if Cortex-M7_0 is in Deep Sleep mode. 0b - Not in Deep Sleep mode 1b - In Deep Sleep mode
15 HSEHLT	Cortex-M0+ Halted Indicates if Cortex-M0+ is halted because of entry into Debug mode. 0b - Core is not halted 1b - Core is halted
14 CM72HLT	CM7_2 Debug Halted Indicates if Cortex-M7_2 is halted because of entry into Debug mode. 0b - Core is not halted

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Core is halted
13 CM71HLT	CM7_1 Debug Halted Indicates if Cortex-M7_1 is halted because of entry into Debug mode. 0b - Core is not halted 1b - Core is halted
12 CM70HLT	Cortex-M7_0 Halted Indicates if Cortex-M7_0 is halted because of entry into Debug mode. 0b - Core is not halted 1b - Core is halted
11-3 —	Reserved
2 FUNCRST	Functional Reset Indicates the system reset state. 0b - Not in functional reset 1b - In functional reset
1 DESTRST	Destructive Reset Indicates the system reset state. 0b - Not in destructive reset 1b - In destructive reset
0 —	Reserved

82.15.3 Control (MDMAPCTL)

Offset

Register	Offset
MDMAPCTL	4h

Function

Allows the debugger:

- To give the destructive reset to the system. A system destructive reset enables this. The debugger also loses connection to the system with this reset.
- To hold the system in functional reset. A system functional reset enables this.

NOTE

- The trace functionality on trace pins is selected with this register and is retained across the functional reset.
- For S32K388, you must always write:
  - Bitfields 18 and 19 in conjunction.
  - Write bitfields 16 and 17 in conjunction if lockstep is enabled for Cortex-M7_0 and Cortex-M7_1.
- For S32K358, S32K348, S32K338, S32K328, S32K342, S32K341, S32K322, S32K344, S32K324, and S32K314, write bitfields 16 and 17 in conjunction if lockstep is enabled for Cortex-M7_0 and Cortex-M7_1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CM72 DBG...	Reserved	CM71 DBG...	CM70 DBG...	CM73 DBG...	Reserved				SWOO VRD	CM7_3 _C...	TRIUO VRD	CM7_2 _C...	CM7_2 _C...	CM7_1 _C...	CM7_0 _C...
W																
Reset	0	0	0	0	0	1	1	0	0	1	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	POR_ WDG...	CM73 DBG...	DBGR STF...	DBGR STS...	Reserved	CM72 DBG...	CM71 DBG...	CM70 DBG...	Reserved		SYSF UNC...	SYSR ESE...	Reserved			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 CM72DBGRSRT	Cortex-M7_2 Debug Restart Indicates if a request to Cortex-M7_2 to leave the debug halt state is asserted. 0b - Normal operation 1b - Request asserted
30 —	Reserved
29 CM71DBGRSRT	Cortex-M7_1 Debug Restart Indicates if a request to Cortex-M7_1 to leave the debug halt state is asserted. 0b - Normal operation 1b - Request asserted
28 CM70DBGRSRT	Cortex-M7_0 Debug Restart Indicates if a request to Cortex-M7_0 to leave the debug halt state is asserted. 0b - Normal operation 1b - Request asserted
27	Cortex-M7_3 Debug Restart

Table continues on the next page...

Table continued from the previous page...

Field	Function
CM73DBGRSRT	Indicates if a request to Cortex-M7_3 to leave the debug halt state is asserted. 0b - Normal operation 1b - Request asserted
26-23 —	Reserved
22 SWOOVRD	SWO Override Indicates if the SWO trace response is overriden. When SWO is not the selected trace sink target, you must override the trace response. 0b - Not overridden, and SWO generates the trace response 1b - Is overridden
21 CM7_3_CORE_ACCESS	Debugger Access To Application Cortex-M7_3 Indicates if debugger access to Cortex-M7_3 across the functional reset phase is supported. After programming the Cortex-M7_3 core debug logic, the debugger can write 1 to this field. Because of this, the clock gating control for CLK and FCLK of Cortex-M7_3 shifts to CCTL. 0b - Supported 1b - Not supported
20 TRIUOVRD	TPIU Override Indicates if TPIU trace response is overridden. When TPIU is not the selected trace sink target, you must override the trace response. 0b - Not overridden, and TPIU generates the trace response 1b - Is overridden and asserted
19 CM7_2_CHK	Cortex-M7_2 Check Indicates if debugger access to Cortex-M7_2 CHK across the functional reset phase is supported. After programming the Cortex-M7_2 CHK core debug logic, the debugger can write 1 to this field. Because of this, the clock gating control for CLK and FCLK of Cortex-M7_2 CHK shifts to CCTL. 0b - Supported 1b - Not supported
18 CM7_2_CORE_ACCESS	Debugger Access To Application Cortex-M7_2 Indicates if debugger access to Cortex-M7_2 across the functional reset phase is supported. After programming the Cortex-M7_2 core debug logic, the debugger can write 1 to this field. Because of this, the clock gating control for CLK and FCLK of Cortex-M7_2 shifts to CCTL. 0b - Supported 1b - Not supported

Table continues on the next page...

Table continued from the previous page...

Field	Function
17 CM7_1_CORE_ACCESS	Debugger Access To Application Cortex-M7_1 Indicates if debugger access to Cortex-M7_1 across the functional reset phase is supported. After programming the Cortex-M7_1 core debug logic, the debugger can write 1 to this field. Because of this, the clock gating control for CLK and FCLK of Cortex-M7_1 shifts to CCTL. 0b - Supported 1b - Not supported
16 CM7_0_CORE_ACCESS	Debugger Access To Application Cortex-M7_0 Indicates if debugger access to Cortex-M7_0 across the functional reset phase is supported. After programming the Cortex-M7_0 core debug logic, the debugger can write 1 to this field. Because of this scenario, the clock gating control for CLK and FCLK of Cortex-M7_0 shifts to CCTL. 0b - Supported 1b - Not supported
15 POR_WDG_DIS_FUNC_RST	Power Watchdog Status 0b - Power watchdog is disabled 1b - Power watchdog is enabled
14 CM73DBGREQ	Cortex-M7_3 Debug Request Drives the EDBGREQ input for Cortex-M7_3 and indicates if the debug request is generated. When the core goes into debug state, the field acknowledges that with a halted output signal (see MDMAPSTTS[CM73HLT]). If the core is in Stop mode, this field is used to wake up the core and transition it to the debug halt state. 0b - Debug request is not generated 1b - Debug request is generated
13 DBGRSTFASTPAD	Debug Over Reset Via Fast Pads Enables or diasables trace via fast pads. If enabled, the trace pads have trace over functional reset feature enabled. This field does not take care of the clock configuration. 0b - Disabled 1b - Enabled
12 DBGRSTSLOWPAD	Debug Over Reset Via Slow Pads Enables or diasables trace via slow pads. If enabled, the trace pads have trace over functional reset feature enabled. This field does not take care of the clock configuration. 0b - Disabled 1b - Enabled
11 —	Reserved

Table continues on the next page...

Table continued from the previous page...

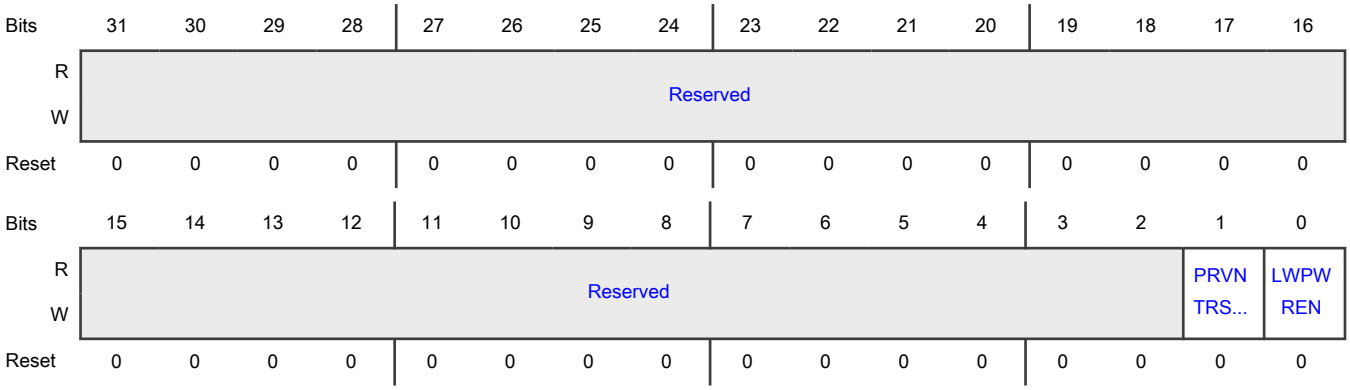
Field	Function
10 CM72DBGREQ	Cortex-M7_2 Debug Request Drives the EDBGREQ input for Cortex-M7_2 and indicates if the debug request is generated. When the core goes into debug state, the field acknowledges that with a halted output signal (see MDMAPSTTS[CM72HLT]). If the core is in Stop mode, this field is used to wake up the core and transition it to the debug halt state. 0b - Debug request is not generated 1b - Debug request is generated
9 CM71DBGREQ	Cortex-M7_1 Debug Request Drives the EDBGREQ input for Cortex-M7_1 and indicates if the debug request is generated. When the core goes into debug state, the field acknowledges that with a halted output signal (see MDMAPSTTS[CM71HLT]). If the core is in Stop mode, this field is used to wake up the core and transition it to the debug halt state. 0b - Debug request is not generated 1b - Debug request is generated
8 CM70DBGREQ	Cortex-M7_0 Debug Request Drives the EDBGREQ input for Cortex-M7_0 and indicates if the debug request is generated. When the core goes into debug state, the field acknowledges that with a halted output signal (see MDMAPSTTS[CM70HLT]). If the core is in Stop mode, this field is used to wake up the core and transition it to the debug halt state. 0b - Debug request is not generated 1b - Debug request is generated
7-6 —	Reserved
5 SYSFUNCRST	System Functional Reset Asserts or deasserts functional reset to the chip. The debugger maintains connection with the chip. 0b - Deasserted 1b - Asserted
4 SYSRESETREQ	System Destructive Reset Asserts or deasserts destructive reset to the chip. The debugger also loses connection with this reset. When this field is reset, the entire system comes out of reset. 0b - Deasserted 1b - Asserted
3-0 —	Reserved

82.15.4 WIR Enable (MDMAPWIREN)

Offset

Register	Offset
MDMAPWIREN	30h

Diagram



Fields

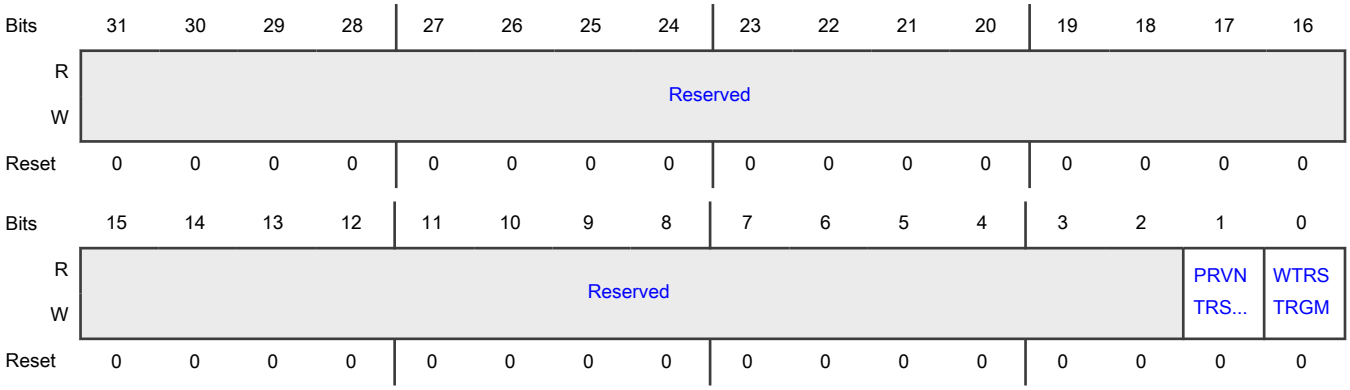
Field	Function
31-2 —	Reserved
1 PRVNTRSTEN	Prevent Reset Enable Indicates if bit field PRVNTRSTRGM in register MDMAPWIRREL is capable of preventing MC_RGM from generating reset. **NOTE** Reserved for S32K31x. 0b - Automatic low power entry enabled 1b - Low power entry enabled controlled by bit field PRVNTRSTRGM of register MDMAPWIRREL.
0 LWPWREN	Low Power Debug Enable Enables or disables low-power debug. 0b - Disabled 1b - Enabled

82.15.5 MDM AP WIR Release (MDMAPWIRREL)

Offset

Register	Offset
MDMAPWIRREL	38h

Diagram



Fields

Field	Function
31-2 —	Reserved
1 PRVNTRSTRGM	Prevent Reset Indicates if MC_RGM is prevented from generating reset. After TPIU flush, this field prevents MC_RGM from generating reset even if MC_RGM receives an acknowledge response from TPIU. This is valid for low-power entry. 0b - Normal operation 1b - MC_RGM prevented
0 WTRSTRGM	Wait In Reset B Indicates if waiting of MC_RGM from generating reset is supported. On exiting Standby mode, MC_RGM waits until the debugger writes to another field in the MDM_AP register to allow it to exit reset. 0b - Normal operation 1b - Wait supported

82.16 SDA_AP register descriptions

The debugger and system have access to secure authentication control and status, implemented as registers in SDA_AP on the DAP bus. These registers provide authentication control, authentication status, key exchange information, and debug enable controls.

82.16.1 SDA_AP memory map

SDA_AP base address: 4025_4700h

Offset	Register	Width (In bits)	Access	Reset value
0h	Authentication Status (AUTHSTTS)	32	R	6000_0004h
4h	Authentication Control (AUTHCTL)	32	W	0000_0000h
10h - 2Ch	Key Challenge (KEYCHAL0 - KEYCHAL7)	32	R	See section
40h - 5Ch	Key Response (KEYRESP0 - KEYRESP7)	32	RW	0000_0000h
70h	Unique Device Identifier 0 (UID0)	32	R	See section
74h	Unique Device Identifier 1 (UID1)	32	R	See section
80h	Debug Enable Control (DBGENCTRL)	32	RW	See section
90h	Reset Control (SDAAPRSTCTRL)	32	RW	1E00_0000h
A0h	SDA_AP Generic Status (SDAAPGENSTATUS0)	32	R	0000_0000h
A4h	Generic Control 0 (SDAAPGENCTRL0)	32	RW	0000_0000h
B0h	SDA_AP Generic Status (SDAAPGENSTATUS1)	32	R	0000_0000h
C0h	SDA_AP Generic Status (SDAAPGENSTATUS2)	32	R	0000_0000h
D0h	SDA_AP Generic Status (SDAAPGENSTATUS3)	32	R	0000_0000h
E0h	SDA_AP Generic Status (SDAAPGENSTATUS4)	32	R	0000_0000h
FCh	Identity (ID)	32	R	001C_0040h

82.16.2 Authentication Status (AUTHSTTS)

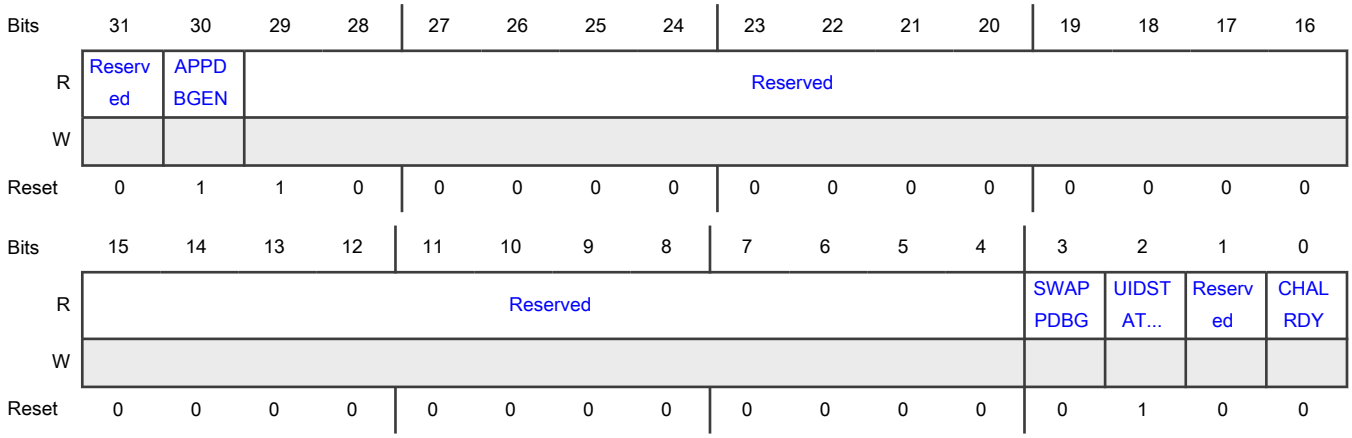
Offset

Register	Offset
AUTHSTTS	0h

Function

Indicates the status of the authentication process.

Diagram



Fields

Field	Function
31 —	Reserved
30 APPDBGEN	Application Debug Enabled or Disabled Indicates: • The status of application debug • Whether CR is satisfied • Whether access to other APs is allowed 0b - Application debug disabled 1b - Application debug enabled
29-4 —	Reserved
3 SWAPPDBG	Software Application Debug Indicates: • The status of software application debug • Whether access to debug controls is allowed 0b - Software application debug disabled 1b - Software application debug enabled
2 UIDSTATUS	User Identification Status Indicates: • The status of UID • Whether DCM has finished reading the flash memory user section

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - UID is not ready and is invalid 1b - UID is ready and is valid
1 —	Reserved
0 CHALRDY	Challenge Ready Indicates: • The status of challenge ready when the value of export control is 0 • The status of the DCM_DONE signal if the value of export control is 1 0b - Challenge is not ready 1b - Challenge is ready

82.16.3 Authentication Control (AUTHCTL)

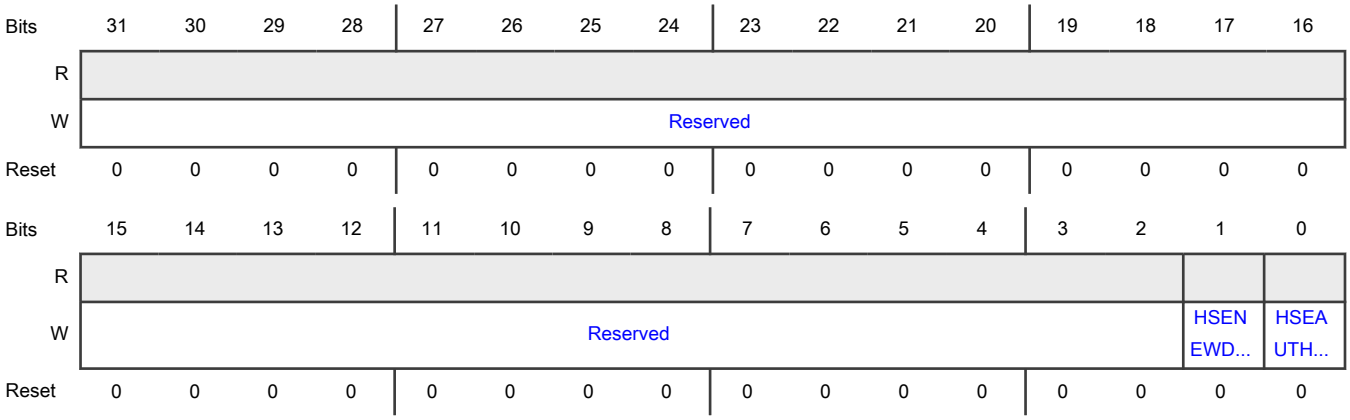
Offset

Register	Offset
AUTHCTL	4h

Function

Controls the authentication process.

Diagram



Fields

Field	Function
31-2 —	Reserved
1 HSENEWDATACTL	New Data Control Indicates that the debugger has consumed the data registers. It is alright for the core to provide new data. 0b - Does not indicate that the debugger has consumed the data registers 1b - Indicates that the debugger has consumed the data registers
0 HSEAUTHREQ	Debug Enablement Authentication Request Indicates that all key values are written and the chip can start the authentication request. 0b - Does not start the authentication request 1b - Starts the authentication request

82.16.4 Key Challenge (KEYCHAL0 - KEYCHAL7)

Offset

For a = 0 to 7:

Register	Offset
KEYCHALa	10h + (a × 4h)

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	KEYCHAL															
W																
Reset [1]	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	KEYCHAL															
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

1. The reset value of this register depends on NXP factory settings.

Fields

Field	Function
31-0 KEYCHAL	Debug Enablement Key Challenge

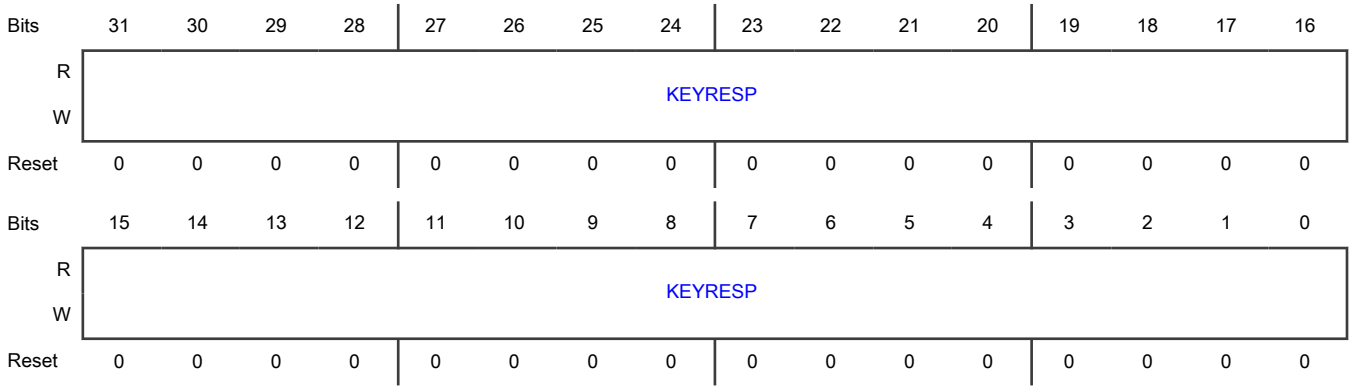
82.16.5 Key Response (KEYRESP0 - KEYRESP7)

Offset

For a = 0 to 7:

Register	Offset
KEYRESPa	40h + (a × 4h)

Diagram



Fields

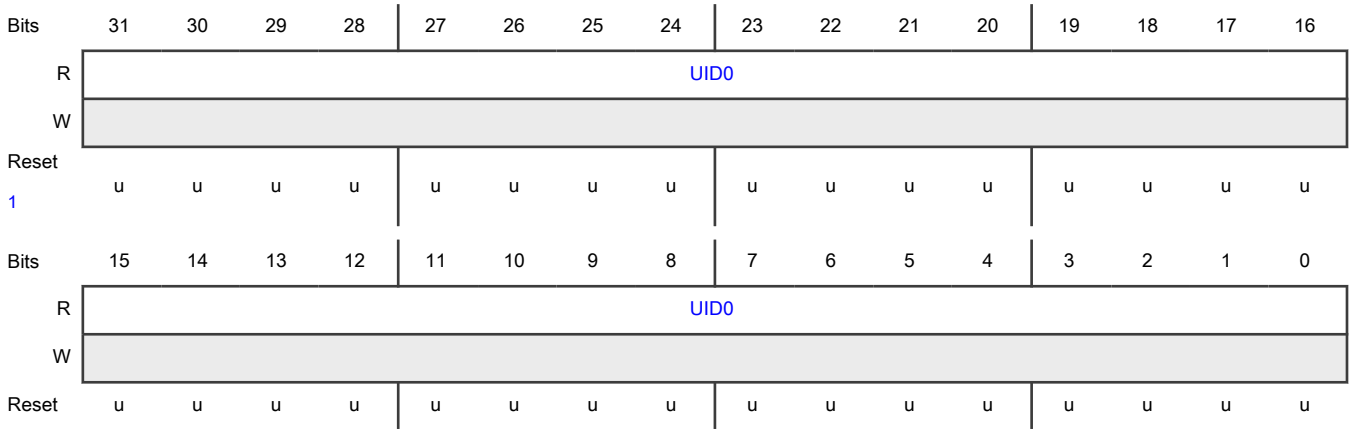
Field	Function
31-0 KEYRESP	Debug Enablement Key Response

82.16.6 Unique Device Identifier 0 (UID0)

Offset

Register	Offset
UID0	70h

Diagram



1. The reset value of this register depends on NXP factory settings.

Fields

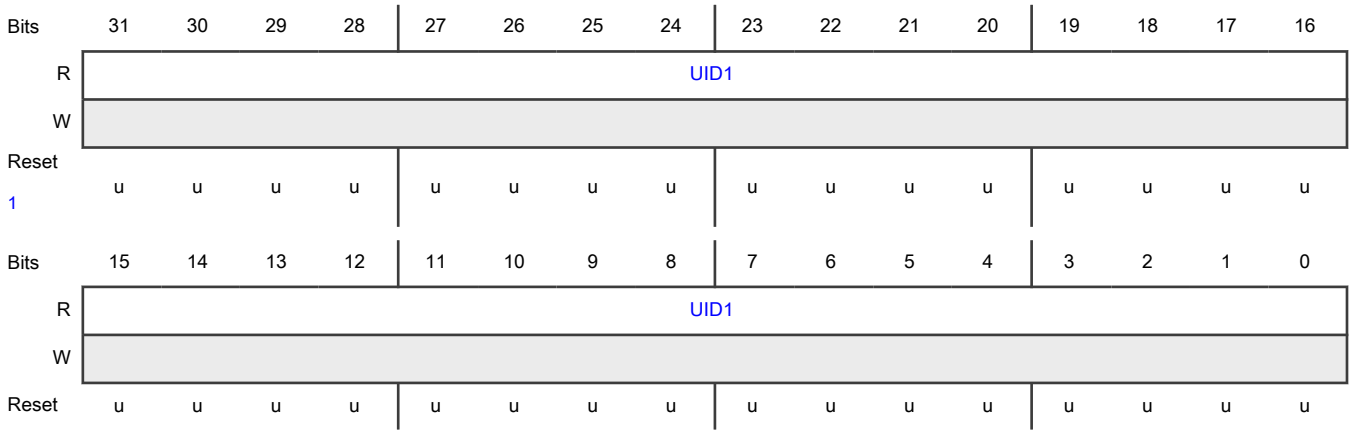
Field	Function
31-0 UID0	User ID 0 Indicates the JTAG user ID bits of the lower word.

82.16.7 Unique Device Identifier 1 (UID1)

Offset

Register	Offset
UID1	74h

Diagram



1. The reset value of this register depends on NXP factory settings.

Fields

Field	Function
31-0 UID1	User ID 1 Indicates the JTAG user ID bits of the upper word.

82.16.8 Debug Enable Control (DBGENCTRL)

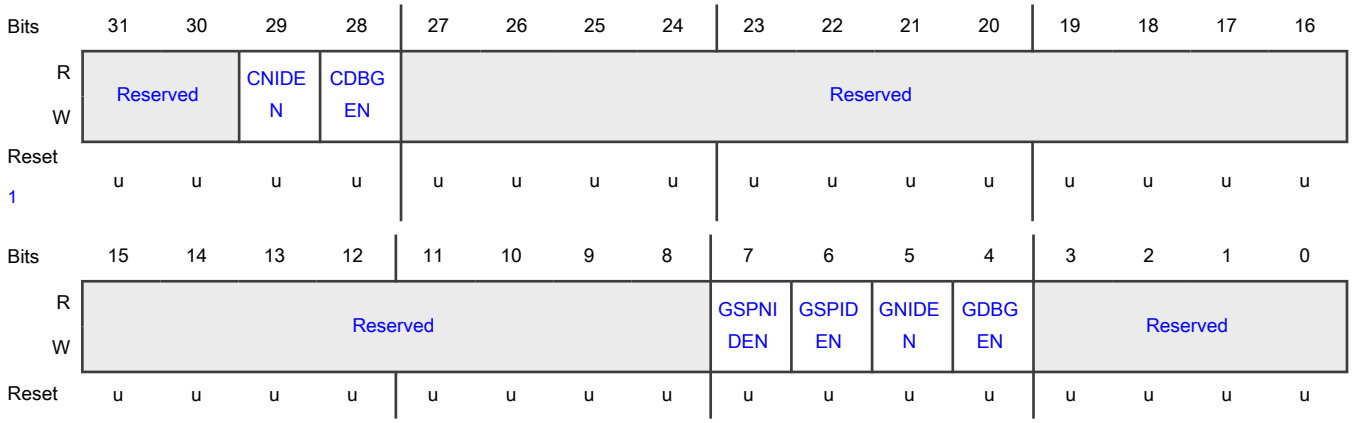
Offset

Register	Offset
DBGENCTRL	80h

Function

Includes a special protection that allows access from Cortex-M0+ only if bit 30 of Authentication Status (AUTHSTTS) is 1.

Diagram



1. The reset value is controlled by an export control enable input. If export control is enabled, the reset value of this register is FFFF_FFF0h. Otherwise, it is 0000_0000h.

Fields

Field	Function
31-30 —	Reserved
29 CNIDEN	Core Non-Invasive Debug Enable Controls CNIDEN of debug blocks coupled with the Cortex-M7 core. 0b - Disabled 1b - Enabled

Table continues on the next page...

Table continued from the previous page...

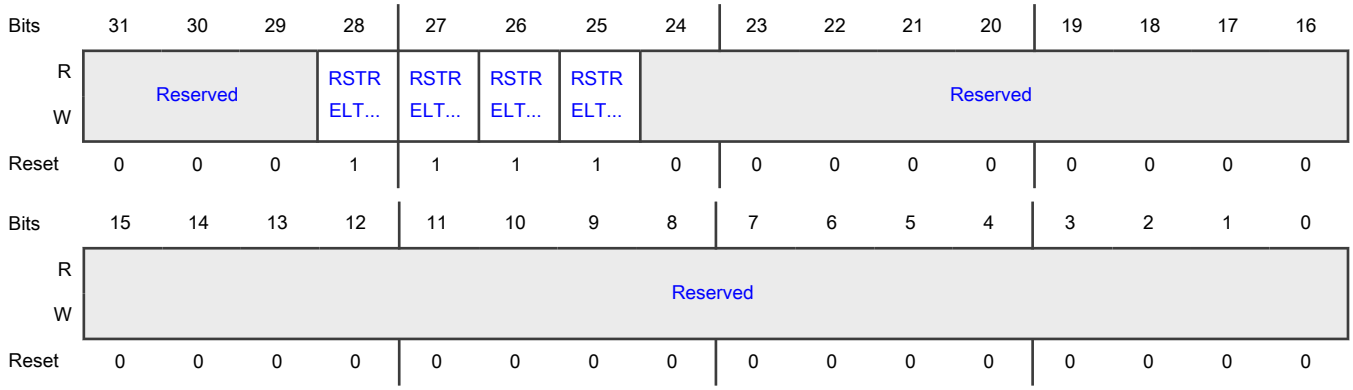
Field	Function
28 CDBGEN	Core Debug Enable Controls CDBGEN of debug blocks coupled with the Cortex-M7 core. 0b - Disabled 1b - Enabled
27-8 —	Reserved
7 GSPNIDEN	Global Secure Privileged Non-Invasive Debug Enable Controls GSPNIDEN of debug blocks coupled with Cortex-M7's subsystems, ETM, ITM, and CTI. 0b - Disabled 1b - Enabled
6 GSPIDEN	Global Secure Privileged Debug Enable Controls GSPIDEN of debug blocks coupled with Cortex-M7's subsystems, ETM, ITM, and CTI. 0b - Disabled 1b - Enabled
5 GNIDEN	Global Non-Invasive Debug Enable Controls GNIDEN of debug blocks coupled with Cortex-M7's subsystems, ETM, ITM, and CTI. 0b - Disabled 1b - Enabled
4 GDBGEN	Global Debug Enable Controls GDBGEN of debug blocks coupled with Cortex-M7's subsystems, ETM, ITM, and CTI. 0b - Disabled 1b - Enabled
3-0 —	Reserved

82.16.9 Reset Control (SDAAPRSTCTRL)

Offset

Register	Offset
SDAAPRSTCTRL	90h

Diagram



Fields

Field	Function
31-29 —	Reserved
28 RSTRELTLCM73	Reset Release Cortex-M7_3 Indicates if the control signal released the reset for Cortex-M7_3. The reset is released to debug the core from first instruction. The default value of this field is 1. 0b - Core is in reset 1b - Reset is released
27 RSTRELTLCM72	Reset Release Cortex-M7_2 Indicates if the control signal released the reset for Cortex-M7_2. The reset is released to debug the core from first instruction. The default value of this field is 1. 0b - Core is in reset 1b - Reset is released
26 RSTRELTLCM71	Reset Release Cortex-M7_1 Indicates if the control signal released the reset for Cortex-M7_1. The reset is released to debug the core from first instruction. The default value of this field is 1. 0b - Core is in reset 1b - Reset is released
25 RSTRELTLCM70	Reset Release Cortex-M7_0 Indicates if the control signal released the reset for Cortex-M7_0. The reset is released to debug the core from the first instruction.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	The default value of this field is 1. 0b - Core is in reset 1b - Reset is released
24-0 —	Reserved

82.16.10 SDA_AP Generic Status (SDAAPGENSTATUS0 - SDAAPGENSTATUS4)

Offset

Register	Offset
SDAAPGENSTATUS0	A0h
SDAAPGENSTATUS1	B0h
SDAAPGENSTATUS2	C0h
SDAAPGENSTATUS3	D0h
SDAAPGENSTATUS4	E0h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SDAAPGENSTATUS															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SDAAPGENSTATUS															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

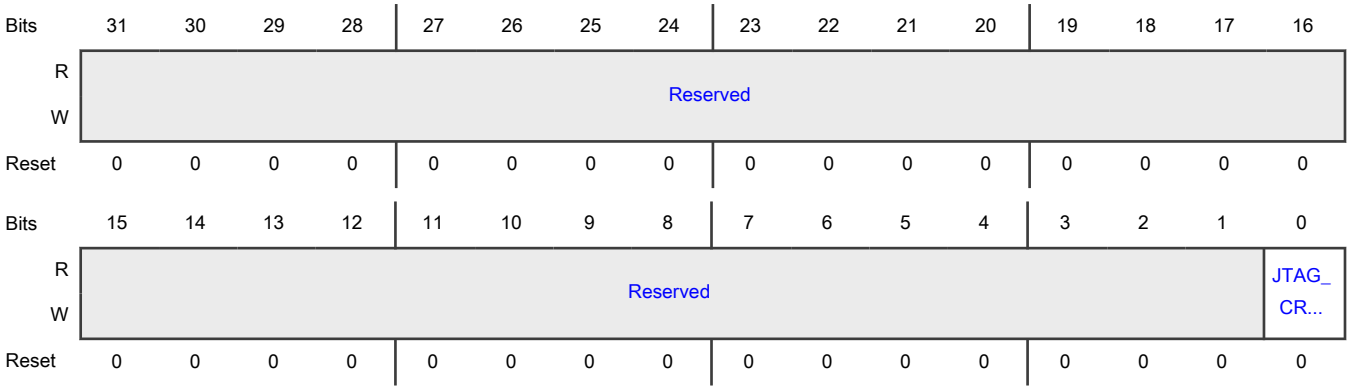
Field	Function
31-0 SDAAPGENSTATUS	DAP Generic Status Is a generic status field for future use. 0b - Does not show generic status 1b - Shows generic status

82.16.11 Generic Control 0 (SDAAPGENCTRL0)

Offset

Register	Offset
SDAAPGENCTRL0	A4h

Diagram



Fields

Field	Function
31-1 —	Reserved
0 JTAG_CR_EN	JTAG CR Enable Performs CR or password comparison based on SWJ-DP mode or JTAG mode. If you write 1 to this field, this function is performed using JTAG irrespective of SWJ-DP mode of the debugger. 0b - Function performed on the basis of SWJ-DP mode 1b - Function performed on the basis of JTAG mode

82.16.12 Identity (ID)

Offset

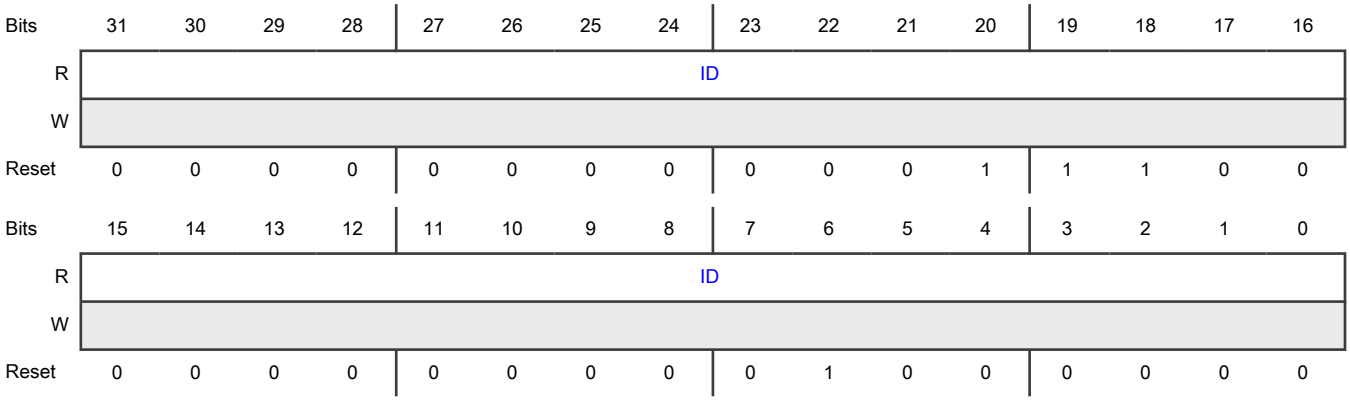
Register	Offset
ID	FCh

Function

NOTE

This register does not generate the bus error on performing the write operation.

Diagram



Fields

Field	Function
31-0 ID	Identity

82.17 Glossary

AHB	Advanced high-performance bus
AHB_AP	Advanced high-performance bus access port
APB	Advanced peripheral bus
APB_AP	Advanced peripheral bus access port
ATB	Advanced trace bus
ATBR	ATB replicator
CSTF	CoreSight trace funnel
DAP	Debug access port
DC	Design center
ETF	Embedded CoreSight funnels
JTAG-DP	JTAG debug port
MDM_AP	Miscellaneous debug module access port
MIC	Manufacturer identity code
PIN	Part identification number
PRN	Part revision number
SDA_AP	Serial data access port
SiP	System-in-package
SWJ-DP	Serial wire/JTAG debug port
TAP	Test and debug access port
TPIU	Trace port interface unit

82.18 References

- Arm CoreSight SoC-400 Technical Reference Manual

  https://developer.arm.com/documentation/100536/0302/?lang=en
- Arm CoreSight Architecture Specification

  http://infocenter.arm.com/help/topic/com.arm.doc.ihi0029d/IHI0029D_coresight_architecture_spec_v2_0.pdf
- CoreSight Components Technical Reference Manual

  http://infocenter.arm.com/help/topic/com.arm.doc.ddi0314h/DDI0314H_coresight_components_trm.pdf
- Arm Debug Interface Architecture Specification

  https://developer.arm.com/documentation/ihi0031/g/?lang=en

Chapter 83
JTAG Controller (JTAGC)

83.1 Chip-specific JTAGC information

The following figure shows the detailed TAP connectivity. SWD/JTAG SELECT, SWDP and JTAG-DP are part of ARM DAP

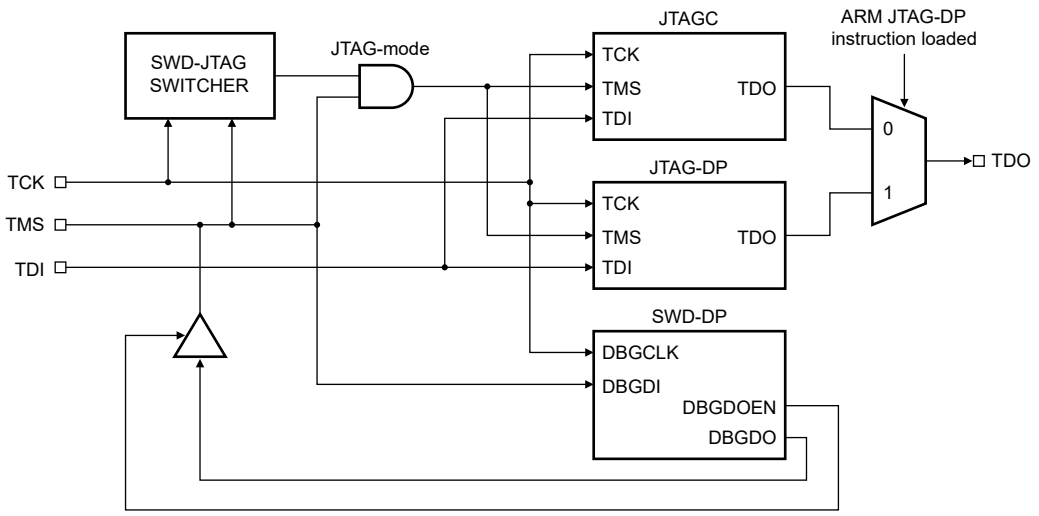


Figure 567. Test and Debug Access Port (TAP) connectivity

The debug port comes out of reset in a standard JTAG mode. It is switched to SWD mode by the change sequences described in JTAG to SWD change sequence . Once the mode has been changed, unused debug pins can be reassigned to any of their alternative muxed functions

JTAG-to-SWD change sequence:

1. Send more than 50 TCK cycles with TMS (SWDIO) =1
2. Send the 16-bit sequence on TMS (SWDIO) = 0111_1001_1110_0111 (MSB transmitted first)
3. Send more than 50 TCK cycles with TMS (SWDIO) =1

NOTE

In case of any reset event, JTAGC is moved to Test Logic Reset (TLR) state first and then JTAGC should be used.

NOTE

If JTAG pins are used for alternative functionality and reset event arrives, then these pins need to be reconfigured to required functionality before using it, in order to avoid unintentional entry in JTAG mode.

83.2 Overview

JTAGC allows you to test chip functionality and connectivity while remaining transparent to system logic when not in test mode. Testing is performed via a boundary scan technique, as defined in the IEEE 1149.1-2001 standard. All data input to and output from JTAGC is communicated in serial format.

83.2.1 Block diagram

The following is a simplified block diagram of the JTAG Controller (JTAGC) block. See the chip-specific configuration information within this chapter as well as Register description for more information about the JTAGC registers.

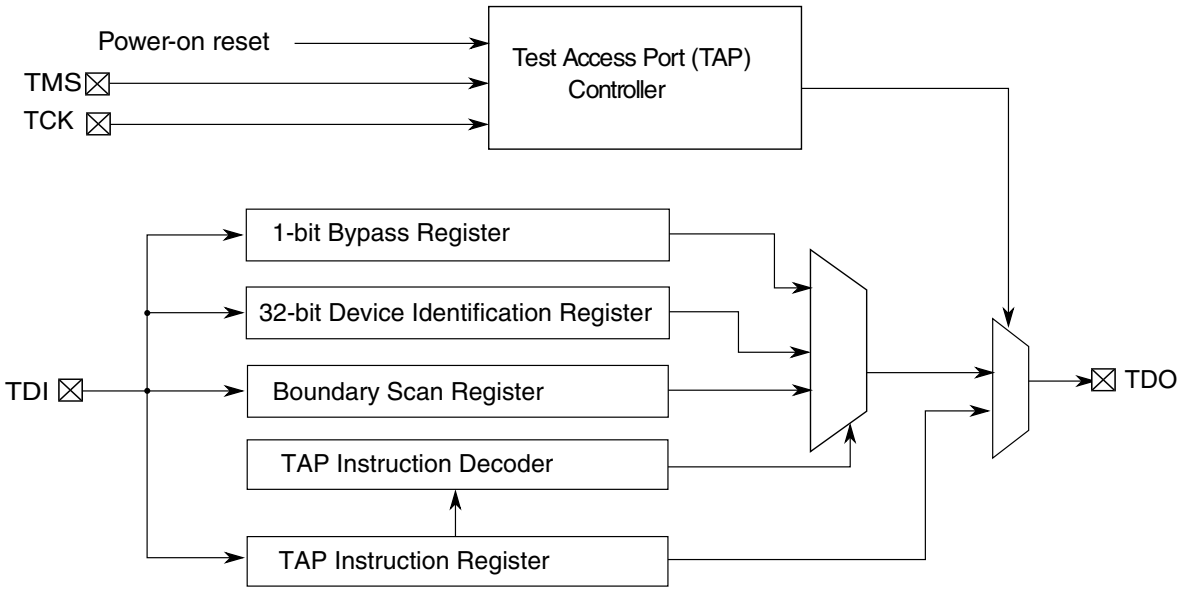


Figure 568. JTAG (IEEE 1149.1) block diagram

83.2.2 Features

The JTAGC block is compliant with the IEEE 1149.1-2001 standard, and supports the following features:

- IEEE 1149.1-2001 TAP interface
  - Four pins (TDI, TMS, TCK, and TDO)
- Instruction register that supports several IEEE 1149.1–2001 defined instructions as well as several public and private device-specific instructions (see JTAGC block instructions for a list of supported instructions).
- Sharing of the TAP with other TAP controllers via ACCESS_AUX_x instructions
- Bypass register, boundary scan register, and device identification register
- TAP controller state machine that controls the operation of the data registers, instruction register, and associated circuitry

83.3 Functional description

83.3.1 Modes of operation

The JTAGC block uses a power-on reset indication as its primary reset signals. Several IEEE 1149.1–2001 defined test modes are supported, as well as a bypass mode.

83.3.1.1 IEEE 1149.1–2001 defined test modes

The JTAGC block supports several IEEE 1149.1–2001 defined test modes. A test mode is selected by loading the appropriate instruction into the instruction register when the JTAGC is enabled. Supported test instructions include EXTEST, HIGHZ, CLAMP, PRELOAD.

Each instruction defines the set of data register(s) that may operate and interact with the on-chip system logic when the instruction is current. Only one test data register path is enabled to shift data between TDI and TDO for each instruction.

The boundary scan register is enabled for serial access between TDI and TDO when the EXTEST, PRELOAD instructions are active. The single-bit bypass register shift stage is enabled for serial access between TDI and TDO when the following instructions are active:

- BYPASS

- HIGHZ
- CLAMP

The functionality of each test mode is explained in more detail in JTAGC block instructions.

83.3.1.2 Bypass mode

When no test operation is required, the BYPASS instruction can be loaded to place the JTAGC block into bypass mode. When in bypass mode, the single-bit bypass shift register is used to provide a minimum-length serial path to shift data between TDI and TDO.

83.3.2 IEEE 1149.1-2001 (JTAG) TAP

The JTAGC block uses the IEEE 1149.1-2001 TAP for accessing registers. This port can be shared with other TAP controllers on the MCU. Ownership of the port is determined by the value of the currently loaded instruction. For more detail on TAP sharing via JTAGC instructions, see ACCESS_AUX_x instructions .

Data is shifted between TDI and TDO through the selected register (instruction, test data, or the bypass register) starting with the least significant bit, as illustrated in the following figure.

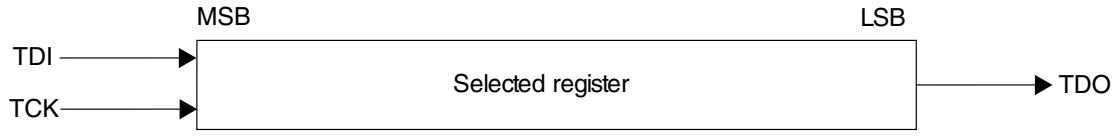
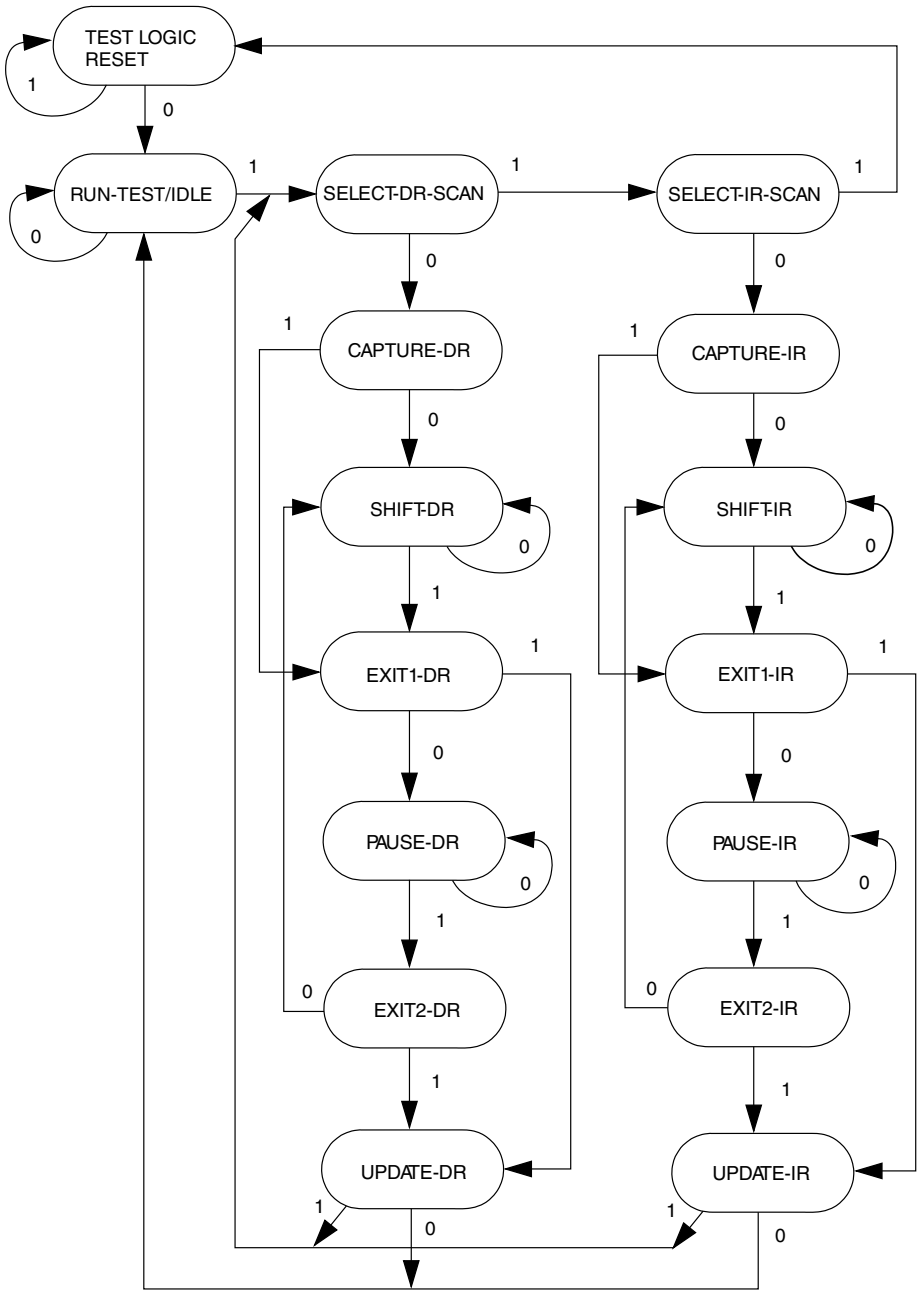


Figure 569. Shifting data through a register

83.3.3 TAP controller state machine

The TAP controller is a synchronous state machine that interprets the sequence of logical values on the TMS pin.

The following figure shows the machine's states. The value shown next to each state is the value of the TMS signal sampled on the rising edge of the TCK signal. As the figure shows, holding TMS at logic 1 when clocking TCK through a sufficient number of rising edges also causes the state machine to enter the Test-Logic-Reset state.



The value shown adjacent to each state transition in this figure represents the value of TMS at the time of a rising edge of TCK.

Figure 570. IEEE 1149.1-2001 TAP controller finite state machine

83.3.3.1 Enabling the TAP controller

The JTAGC TAP controller is enabled by setting the JTAGC enable to a logic 1 value.

83.3.3.2 Selecting an IEEE 1149.1-2001 register

Access to the JTAGC data registers is achieved by loading the instruction register with any of the JTAGC block instructions when the JTAGC is enabled. Instructions are shifted in via the Select-IR-Scan path and loaded in the Update-IR state. At this point, all data register access is performed via the Select-DR-Scan path.

The Select-DR-Scan path is used to read or write the register data by shifting in the data (LSB first) during the Shift-DR state. When reading a register, the register value is loaded into the IEEE 1149.1-2001 shifter during the Capture-DR state. When writing a register, the value is loaded from the IEEE 1149.1-2001 shifter to the register during the Update-DR state. When reading a register, there is no requirement to shift out the entire register contents. Shifting may be terminated after the required number of bits have been acquired.

83.3.4 JTAGC block instructions

The JTAGC implements instructions defined in IEEE 1149.1-2001. See the chip-specific JTAGC information for the JTAGC instructions implemented on this chip. See the IEEE 1149.1-2001 standard for additional information. All undefined opcodes are reserved.

83.3.4.1 IDCODE instruction

IDCODE selects the 32-bit device identification register as the shift path between TDI and TDO. This instruction allows interrogation of the MCU to determine its version number and other part identification data. IDCODE is the instruction placed into the instruction register when the JTAGC block is reset.

83.3.4.2 PRELOAD instruction

The PRELOAD instruction has two functions:

- The PRELOAD portion of the instruction initializes the boundary scan register cells before selecting the EXTEST or CLAMP instructions to perform boundary scan tests. This is achieved by shifting in initialization data to the boundary scan register during the Shift-DR state. The initialization data is transferred to the parallel outputs of the boundary scan register cells on the falling edge of TCK in the Update-DR state. The data is applied to the external output pins by the EXTEST or CLAMP instruction. System operation is not affected.

83.3.4.3 EXTEST external test instruction

EXTEST selects the boundary scan register as the shift path between TDI and TDO. It allows testing of off-chip circuitry and board-level interconnections by driving preloaded data contained in the boundary scan register onto the system output pins. Typically, the data is loaded into the boundary scan register using the PRELOAD instruction before the selection of EXTEST. EXTEST asserts the internal system reset for the MCU to force a predictable internal state when performing external boundary scan operations.

83.3.4.4 TEST_LEAKAGE instruction

The TEST_LEAKAGE instruction forces the jtag_leakage output signal to high. It is intended to tristate all output pad buffers and disable all of the part's pad input buffers except JCOMP and TEST. The jtag_leakage signal is asserted at the falling edge of TCK following the TAP controller state machine transition from the Update-IR state to the Run-Test-Idle state. When asserted, the part disables TCK, TMS, and TDI inputs and forces them to a logic 1. The TAP controller state machine remains in the Run-Test-Idle state until the JCOMP input is set to a value other than the JTAGC enable encoding. TEST_LEAKAGE also asserts the internal system reset for the MCU to force a predictable internal state.

83.3.4.5 HIGHZ instruction

HIGHZ selects the bypass register as the shift path between TDI and TDO. When HIGHZ is active all output drivers are placed in an inactive drive state (for example, high impedance). HIGHZ also asserts the internal system reset for the MCU to force a predictable internal state.

83.3.4.6 CLAMP instruction

CLAMP allows the state of signals driven from MCU pins to be determined from the boundary scan register when the bypass register is selected as the serial path between TDI and TDO. CLAMP enhances test efficiency by reducing the overall shift path to a single bit (the bypass register) when conducting an EXTEST type of instruction through the boundary scan register. CLAMP also asserts the internal system reset for the MCU to force a predictable internal state.

83.3.4.7 ACCESS_AUX_x instructions

The JTAGC is configurable to allow other TAP controllers on the device to share the port with it. This is done by providing ACCESS_AUX_x instructions for each of these TAP controllers.

When this instruction is loaded, control of the JTAG pins is transferred to the selected TAP controller. Any data input via TDI and TMS is passed to the selected TAP controller, and any TDO output from the selected TAP controller is sent back to the JTAGC to be output on the pins.

The JTAGC gets back control of the JTAG port during the UPDATE-DR state if the PAUSE-DR state was entered. Auxiliary TAP controllers are held in RUN-TEST/IDLE when they are inactive. Instructions not used to access an auxiliary TAP controller on a device are treated like the BYPASS instruction.

83.3.4.8 BYPASS instruction

BYPASS selects the bypass register, creating a single-bit shift register path between TDI and TDO. BYPASS enhances test efficiency by reducing the overall shift path when no test operation of the MCU is required. This allows more rapid movement of test data to and from other components on a board that are required to perform test functions. When the BYPASS instruction is active the MCU operates normally.

83.3.5 Boundary scan

The boundary scan technique allows signals at component boundaries to be controlled and observed through the shift-register stage associated with each pad. Each stage is part of a larger boundary scan register cell, and cells for each pad are interconnected serially to form a shift-register chain around the border of the design. The boundary scan register consists of this shift-register chain, and is connected between TDI and TDO when the EXTEST, PRELOAD instructions are loaded. The shift-register chain contains a serial input and serial output, as well as clock and control signals.

83.3.6 Clocking

There is one clock for this module. For more information See External signals.

83.3.7 Interrupts

This module has no interrupts.

83.3.8 Reset

The JTAGC block is placed in reset when:

- Power-on reset is asserted
- TMS input is held high for enough consecutive rising edges of TCK to sequence the TAP controller state machine into the Test-Logic-Reset state

Holding TMS high for five consecutive rising edges of TCK guarantees entry into the Test-Logic-Reset state regardless of the current TAP controller state. Asserting power-on reset results in asynchronous entry into the reset state.

When in reset, the following actions occur:

- The TAP controller is forced into the Test-Logic-Reset state, thereby disabling the test logic and allowing normal operation of the MCU.
- The instruction register is loaded with the IDCODE instruction.

83.4 External signals

JTAGC includes a set of signals that connect to off-chip development tools and allow access to test support functions. The JTAGC signals are outlined in the following table and described in the following sections.

Table 841. JTAG signal properties

Name	I/O	Function	Reset state
TCK	Input	Test clock	Weak pulldown
TDI	Input	Test data in	Weak pullup
TDO	Output	Test data out	High Z[1]
TMS	Input	Test mode select	Weak pullup

1. TDO output buffer is disabled when the JTAGC is not in the Shift-IR or Shift-DR states. A weak pull may be implemented at the TDO pad for use when JTAGC is inactive.

83.4.1 Test clock input (TCK)

Test Clock Input (TCK) is an input pin used to synchronize the test logic and control register access through the TAP.

83.4.2 Test data input (TDI)

Test Data Input (TDI) is an input pin that receives serial test instructions and data. TDI is sampled on the rising edge of TCK.

83.4.3 Test data output (TDO)

Test Data Output (TDO) is an output pin that transmits serial output for test instructions and data. TDO is tristateable and is actively driven only in the Shift-IR and Shift-DR states of the TAP controller state machine, which is described in TAP controller state machine.

83.4.4 Test mode select (TMS)

Test Mode Select (TMS) is an input pin used to sequence the IEEE 1149.1-2001 test control state machine. TMS is sampled on the rising edge of TCK.

83.5 Initialization

To initialize the JTAGC block and enable access to registers, the following sequence is required:

1. Place the JTAGC in reset through TAP controller state machine transitions controlled by TMS.
2. Load the appropriate instruction for the test or action to be performed.

83.6 Application information

The test logic is a static logic design, and TCK can be stopped in either a high or low state without loss of data. However, the system clock is not synchronized to TCK internally. Any mixed operation using both the test logic and the system functional logic requires external synchronization.

83.7 Register description

This section provides a detailed description of the JTAGC block registers accessible through the TAP interface, including data registers and the instruction register. Individual bit-level descriptions and reset states of each register are included. These registers are not memory-mapped and can only be accessed through the TAP.

83.7.1 Instruction register

The JTAGC block uses a 8-bit instruction register as shown in the following figure.

The instruction register allows instructions to be loaded into the block to select the test to be performed, or the test data register to be accessed, or both. Instructions are shifted in through TDI when the TAP controller is in the Shift-IR state, and latched on the falling edge of TCK in the Update-IR state. The latched instruction value can only be changed in the Update-IR and Test-Logic-Reset TAP controller states.

Synchronous entry into the Test-Logic-Reset state results in the IDCODE instruction being loaded on the falling edge of TCK. Asynchronous entry into the Test-Logic-Reset state results in asynchronous loading of the IDCODE instruction. During the Capture-IR TAP controller state, the instruction shift register is loaded with the value 0b1, making this value the register's read value when the TAP controller is sequenced into the Shift-IR state.

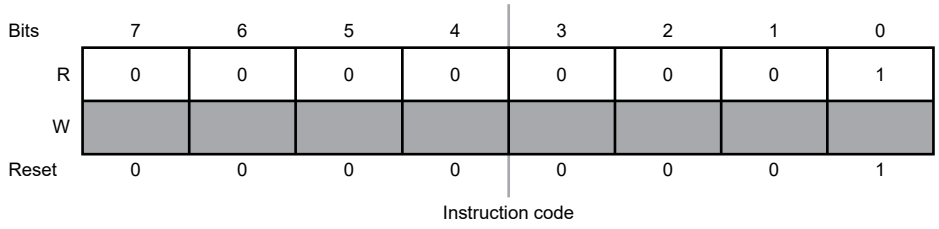


Figure 571. Instruction register

83.7.2 Bypass register

The bypass register is a single-bit shift register path selected for serial data transfer between TDI and TDO when the following instructions are active:

- BYPASS
- HIGHZ
- CLAMP

After entry into the Capture-DR state, the single-bit shift register is set to a logic 0. Therefore, the first bit shifted out after selecting the bypass register is always a logic 0.

83.7.3 Device identification register

The device identification (JTAG ID) register, shown in the following figure, allows the revision number, part number, manufacturer, and design center responsible for the design of the part to be determined through the TAP.

The device identification register is selected for serial data transfer between TDI and TDO when the IDCODE instruction is active. Entry into the Capture-DR state when the device identification register is selected loads the IDCODE into the shift register to be shifted out on TDO in the Shift-DR state. No action occurs in the Update-DR state.

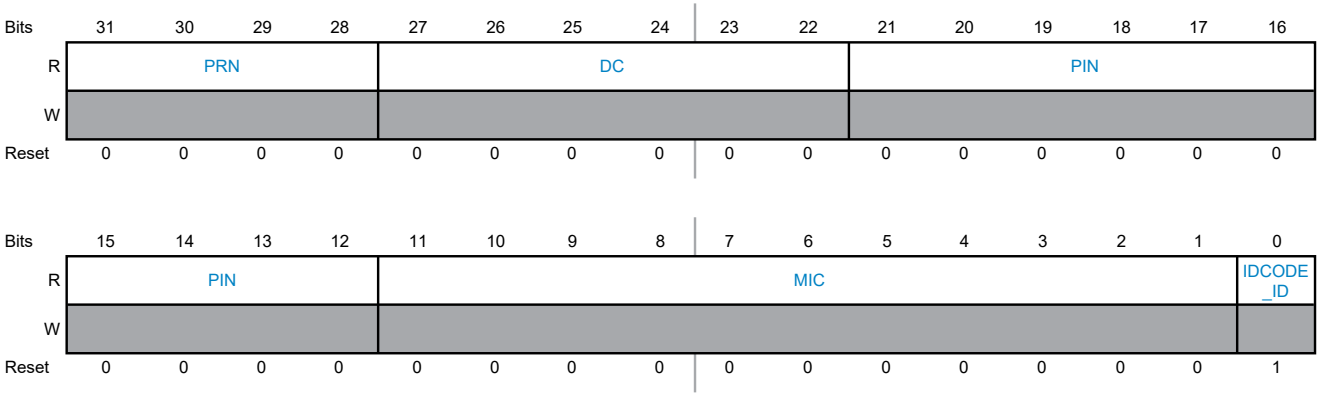


Figure 572. Device identification register

The following table describes the device identification register functions. The device identification register values are described in the chip-specific JTAGC information.

Table 842. Device identification register field descriptions

Field	Function
PRN	Part revision number Contains the revision number of the part.
DC	Design center Indicates the design center.
PIN	Part identification number Contains the part number of the device.
MIC	Manufacturer identity code Contains the reduced Joint Electron Device Engineering Council (JEDEC) ID .
IDCODE ID	IDCODE register ID Identifies this register as the device identification register and not the bypass register. Always set to 1.

83.7.4 Boundary scan register

The boundary scan register is connected between TDI and TDO when the EXTEST, PRELOAD instructions are active. It is used to:

- Capture input pin data
- Force fixed values on output pins
- Select a logic value and direction for bidirectional pins

Each bit of the boundary scan register represents a separate boundary scan register cell, as described in the IEEE 1149.1-2001 standard and discussed in Boundary scan. The size of the boundary scan register and bit ordering is device-dependent and can be found in the device BSDL file.

83.8 Glossary

TAP	Test access port
TCK	Test clock
TDI	Test data in
TDO	Test data out
TMS	Test mode select
JCOMP	JTAG compliance

Chapter 84
JTAG Data Communication (JDC)

84.1 Overview

JDC module provides the capability to move register data between the IPS and JTAG domains. This facilitates communication between internal resources that access memory-mapped register space and an external tool that accesses the JTAG port.

The JDC module consists of:

- IPS-accessible registers
- JTAG-accessible registers
- Associated logic to coordinate movement of data from one register domain to another

The JDC implements the following IPS data registers, occupying separate memory space:

- 1 32-bit memory mapped register that can be read or written via IPS (JTAG Output Data Register (JOUT_IPS)), and whose contents are ported out for capture into a JTAG register (JOUT) to be read via the JTAG port.
- 1 32-bit memory mapped register that can only be read via IPS (JTAG Input Data Register (JIN_IPS)), and whose contents are loaded from a JTAG register (JIN).

JDC indicates when:

- New data has been shifted in via the JTAG port and is ready to be read from the JTAG Input Data Register (JIN_IPS) register
- New data has been written to the JTAG Output Data Register (JOUT_IPS) register and is ready to be read via the JTAG port

A Module Status Register (MSR) captures the state of these flags and also provides the flags to a JTAG register (JOUT) for tool visibility.

84.1.1 Block diagram

There is a single bus interface to port register data out to the JTAGC, and a single bus interface to port data in from the JTAGC. The following figure is the JDC module block diagram.

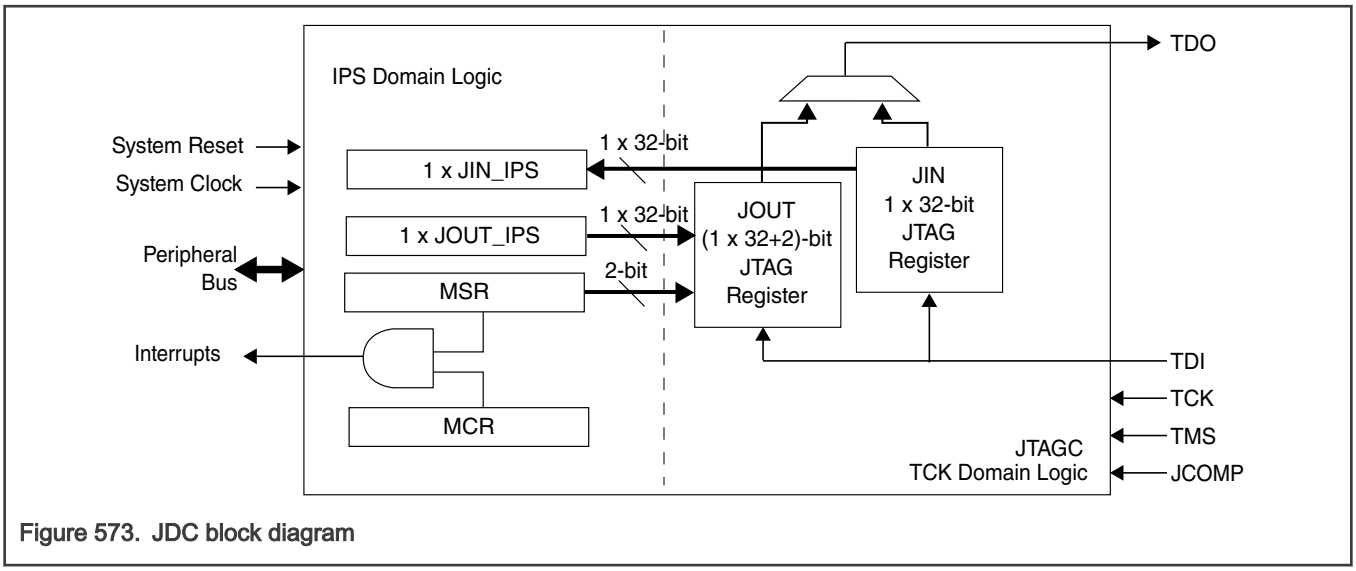


Figure 573. JDC block diagram

84.2 Functional description

The JDC module provides the ability to shift in data via the JTAG port and capture that data in memory-mapped register that can be accessed via IPS. It also provides the ability to capture data written to memory-mapped register into a JTAG shift register for output via the JTAG port.

84.2.1 JDC functionality

An overview of the module functionality is described here:

- Software write to the JOUT_IPS register:
  1. The JDC sets the JOUT_RDY flag bit, indicating new data is available to be read from the JOUT register via the JTAG port.
  2. The MSR reflects the state of the JOUT_RDY bit.
  3. The JDC also captures the state of the JOUT_RDY bit in the JOUT register.
  4. A JTAG read of the JOUT register via execution of the JOUT_READ instruction with a JOUT_RDY bit whose value is logic 1 indicates that the register contains new data.
  5. The JDC clears the JOUT_RDY flag bit upon exit of the Capture-DR JTAG state during execution of the JOUT_READ instruction.
  6. Clearing the JOUT_RDY bit indicates to software that a new data value can be written to the JOUT_IPS register.
- JTAG write to the JIN register via execution of the JIN_WRITE JTAG instruction:
  1. The JDC updates the contents of the JIN_IPS register upon exit of the Update-DR state.
  2. An update of the JIN_IPS register sets the JIN_RDY flag bit, indicating new data is available to be read via IPS.
  3. The MSR register reflects the state of the JIN_RDY bit.
  4. The JDC also captures the state of the JIN_RDY bit in the JOUT register.
  5. The JDC clears the JIN_RDY flag bit upon software read of the JIN_IPS register.
  6. A JTAG read of the JOUT register with a JIN_RDY value of logic 0 indicates that new data can be written to the JIN register.

84.2.2 JTAG register access

See the JTAGC documentation for information on how to access the JTAG registers.

84.2.2.1 JDC block instructions

The JDC block implements the instructions listed in Table 843. This section gives an overview of each instruction. All undefined opcodes are reserved.

Table 843. JTAG instructions

Instruction	Code[4:0]	Instruction summary
Reserved	00001	Factory debug reserved.
JOUT_READ	00010	Selects JOUT data register. The JDC captures data from JOUT_IPS into JOUT data register upon entry to Capture-DR state when JOUT_READ is active.
JIN_WRITE	01110	Selects JIN data register. The JDC captures data from JIN into JIN_IPS upon exit of Update-DR state when JIN_WRITE is active.

Table continues on the next page...

Table 843. JTAG instructions (continued)

Instruction	Code[4:0]	Instruction summary
BYPASS[1]	11111	Selects bypass register for data operations.
Reserved	All other opcodes	Decoded to select bypass register.

1. This is an IEEE 1149.1-2001 defined instruction. See the IEEE 1149.1-2001 standard for more details.

84.2.3 Clocking

JDC module uses TCK and inverted TCK as clock source.

84.2.4 Interrupts

This module has no interrupts.

84.3 External signals

JDC signals are listed in the following table.

Table 844. Signal properties

Name	Function	I/O	Reset
TCK	Test clock input	I	—
TMS	Test mode select	I	0
TDI	Test data in	I	0
JCOMP	JTAG compliance pin	I	0
TDO	Test data out	O	HiZ

84.4 Initialization

This module does not require initialization.

84.5 Secure challenge/response connection procedure

A summary of how the JDC can be used to interact with a security module to perform challenge/response password authorization is as follows:

1. After exit of reset, all registers are at their reset values. JIN_RDY bit value of 0 indicates that the tool may write data to the JIN register. JOUT_RDY bit value of 0 indicates that software may write data to the JOUT_IPS register.
2. The MCR register is programmed to enable JOUT and JIN interrupts, or not.
3. Once the tool is ready to start the authorization process, it writes a value to the JIN register via execution of the JTAGC JIN_WRITE instruction.
4. Software monitors the state of the JIN_RDY bit or enables the JIN interrupt and uses the interrupt assertion as a trigger to start the authorization process.
5. Once the authorization process is started, software writes the first JOUT value to the JOUT_IPS register. This sets the JOUT_RDY bit value to 1.

6. Following the initial tool write to the JIN register, the tool polls the JOUT register to determine when software has provided a challenge word. Upon reading the JOUT register with JOUT_RDY bit set, the tool records the data value to be used to generate the appropriate response. The read of the JOUT register clears the JOUT_RDY bit. The clearing of the JOUT_RDY bit also sets the JOUT_INT bit and asserts the JOUT interrupt if the MCR[JOUT_IEN] is set.
7. With the JOUT_RDY bit cleared, software may now provide the next challenge word.
8. Once the tool has generated a response word, it uses the value of the JIN_RDY bit from the JOUT register read to determine when it can perform a write to the JIN register to provide the response.
9. The security module can provide additional challenge words and read back the corresponding response words as needed, repeating the process.

Simply providing a password without using the challenge/response protocol can be done in a similar way. The tool provides 32-bits of the password at a time with each write to the JIN register, and monitors the JIN_RDY bit value of the JOUT register to determine when the next JIN register value can be written. There is no need for software to write the JOUT_IPS register in this case.

For details related to the lifecycle stage at which this process is performed, see the "Hardware Security Engine" chapter.

84.6 JDC register descriptions

The following sections describe the implemented 32-bit registers. Only 32-bit accesses are valid. The effects of access that are not 32 bits are not defined.

84.6.1 JDC memory map

JDC base address: 4039_4000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Module Configuration Register (MCR)	32	RW	0000_0000h
4h	Module Status Register (MSR)	32	RW	0000_0000h
8h	JTAG Output Data Register (JOUT_IPS)	32	RW	0000_0000h
Ch	JTAG Input Data Register (JIN_IPS)	32	R	0000_0000h

84.6.2 Module Configuration Register (MCR)

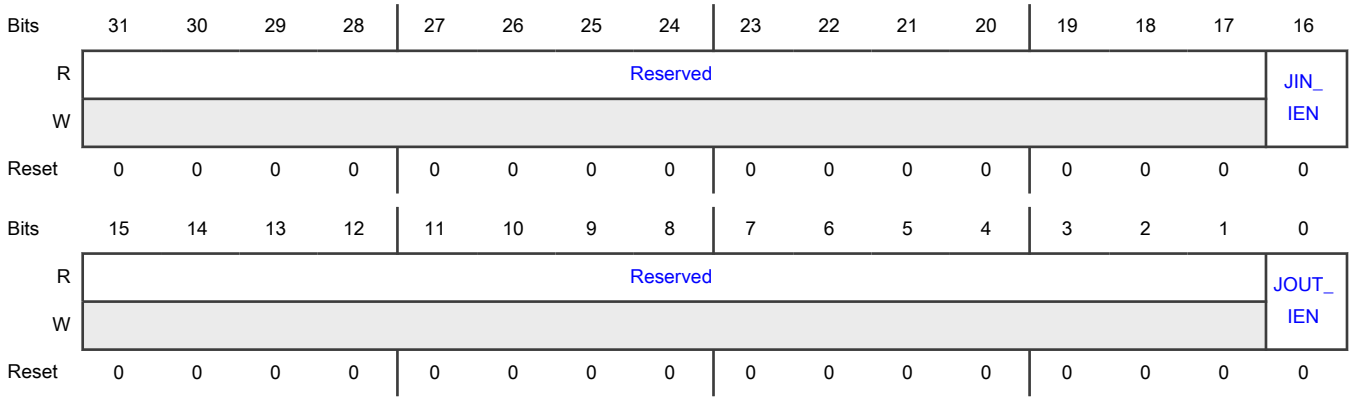
Offset

Register	Offset
MCR	0h

Function

The MCR enables the interrupt outputs of the JDC. This register is reset by system destructive reset.

Diagram



Fields

Field	Function
31-17 —	Reserved
16 JIN_IEN	JIN Interrupt Enable 0b - Setting MSR[JIN_INT] bit does not assert the JIN interrupt 1b - Setting MSR[JIN_INT] bit asserts the JIN interrupt
15-1 —	Reserved
0 JOUT_IEN	JOUT Interrupt Enable 0b - Setting MSR[JOUT_INT] bit does not assert the JOUT interrupt 1b - Setting MSR[JOUT_INT] bit asserts the JOUT interrupt

84.6.3 Module Status Register (MSR)

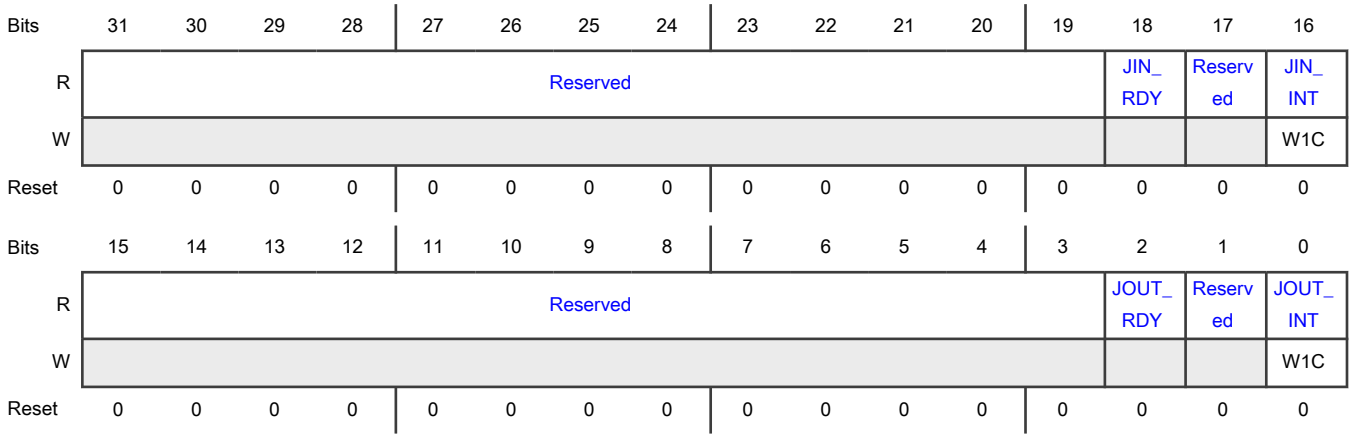
Offset

Register	Offset
MSR	4h

Function

The MSR holds the JTAG register status and interrupt bits. This register is reset by system destructive reset.

Diagram



Fields

Field	Function
31-19 —	Reserved
18 JIN_RDY	JIN Ready (read only) 0b - Cleared upon software read of JIN_IPS contents via IPS 1b - Set when new data is written to the JIN_IPS register
17 —	Reserved
16 JIN_INT	JIN Interrupt 0b - Cleared by writing logic 1 1b - Set when new data is written to the JIN_IPS register
15-3 —	Reserved
2 JOUT_RDY	JOUT Ready (read only) 0b - Cleared upon tool read of JOUT register via JTAG port 1b - Set when new data is written to the JOUT_IPS register
1 —	Reserved
0 JOUT_INT	JOUT Interrupt 0b - Cleared by writing logic 1 1b - Set when JOUT_RDY bit is cleared by tool reading JOUT register

84.6.4 JTAG Output Data Register (JOUT_IPS)

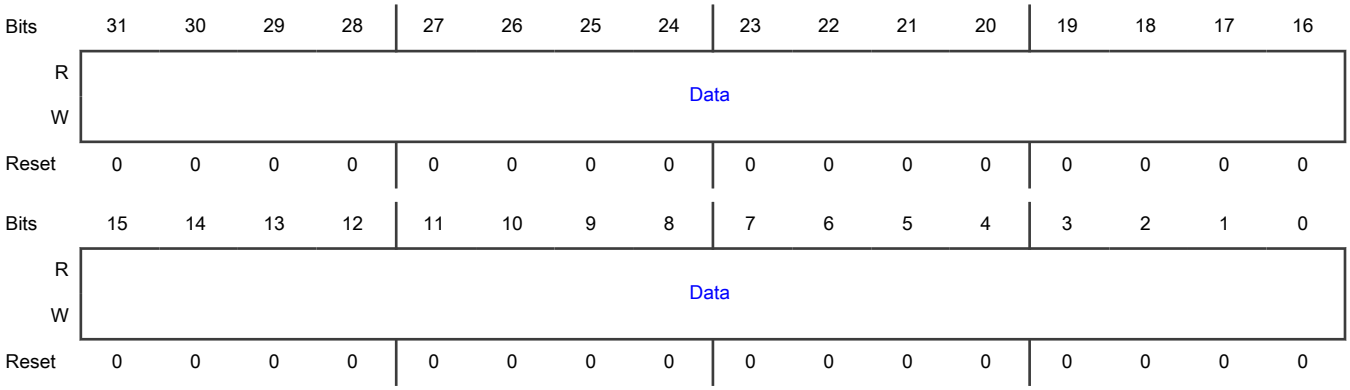
Offset

Register	Offset
JOUT_IPS	8h

Function

The JOUT_IPS register holds data written via IPS. The JDC captures the JOUT_IPS contents into the JOUT register to be read via the JTAG port. This register is reset by system destructive reset.

Diagram



Fields

Field	Function
31-0 Data	JOUT_IPS Data

84.6.5 JTAG Input Data Register (JIN_IPS)

Offset

Register	Offset
JIN_IPS	Ch

Function

The JIN_IPS register holds data written to the JTAG input data register (JIN) via the JTAG port, where the data can be read via IPS. Any IPS write to the JIN_IPS register returns a transfer error. This register is reset by system destructive reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Data															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Data															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 Data	JIN_IPS data

84.7 Non-memory-mapped register definition

The JDC also implements two JTAG accessible registers that are not memory-mapped. One JTAG register shifts in data to be placed in the JIN_IPS register. The other JTAG register captures data from the JOUT_IPS register plus ready status from the MSR, to be shifted out via the JTAG port.

84.7.1 JTAG output data register (JOUT)

The JOUT register captures data from the JOUT_IPS register upon execution of the JOUT_READ JTAG instruction. It also holds the JIN_RDY and JOUT_RDY status bits. This register is reset by system destructive reset and JTAG reset. The reset value of the JOUT register is 0. The following figure shows the format of the JOUT register.

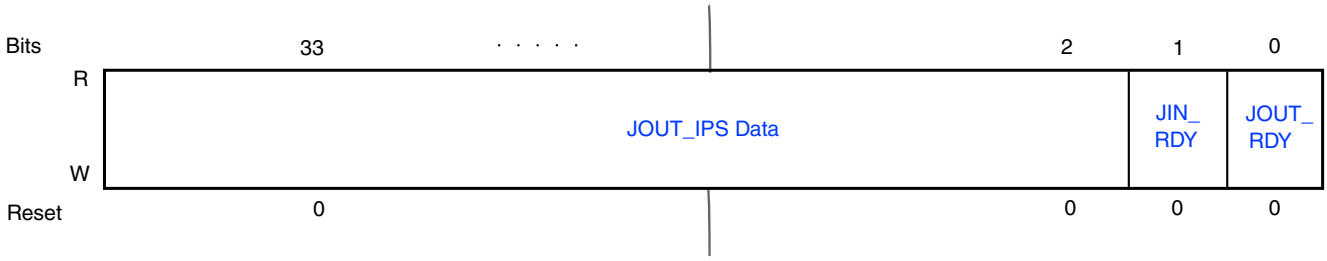


Figure 574. JOUT

The JOUT register is described in the following table.

Table 845. JOUT JTAG register field descriptions

Field	Function
JOUT_IPS Data	JOUT_IPS Data Data value from JOUT_IPS register

Table continues on the next page...

Table 845. JOUT JTAG register field descriptions (continued)

Field	Function
JIN_RDY	JIN_RDY State of JIN_RDY bit from MSR
JOUT_RDY	JOUT_RDY State of JOUT_RDY bit from MSR

84.7.2 JTAG input data register (JIN)

The external tool writes data to the JIN register via JTAG during execution of the JIN_WRITE JTAG instruction. The JDC later captures JIN data in the JIN_IPS register to be read via IPS. This register is reset by system destructive reset and JTAG reset. The reset value of the JIN register is 0. The following figure shows the format of the JIN register.

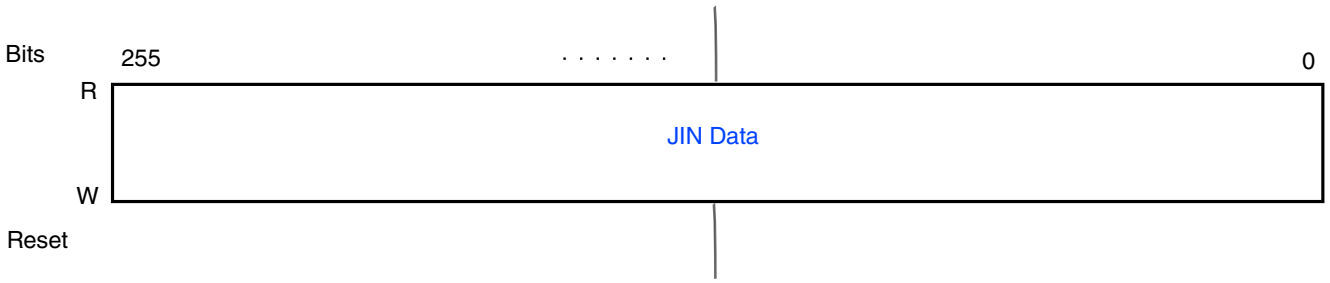


Figure 575. JIN

The JIN register is described in the following table.

Table 846. JIN JTAG register field descriptions

Field	Description
JIN Data	Contains data to be captured in JIN_IPS register upon exit of Update-DR state when executing WRITE_JIN JTAG instruction.

84.8 Glossary

IPS	Internal peripheral system
XBAR	Crossbar bus interface

Chapter 85
Temperature Sensor (TempSense)

85.1 Overview

TEMPSENSE is used to measure the temperature on the chip through an ADC and to flag if an overtemperature condition occurs.

The TEMPSENSE module contains control registers for use with the Engineering temperature sensor (ETS) hard block and the flag generation hard block. These registers use a peripheral bus interface to communicate with the chip.

85.1.1 Block diagram

The functional structure of the TEMPSENSE can be seen in the block diagram in Figure 576.

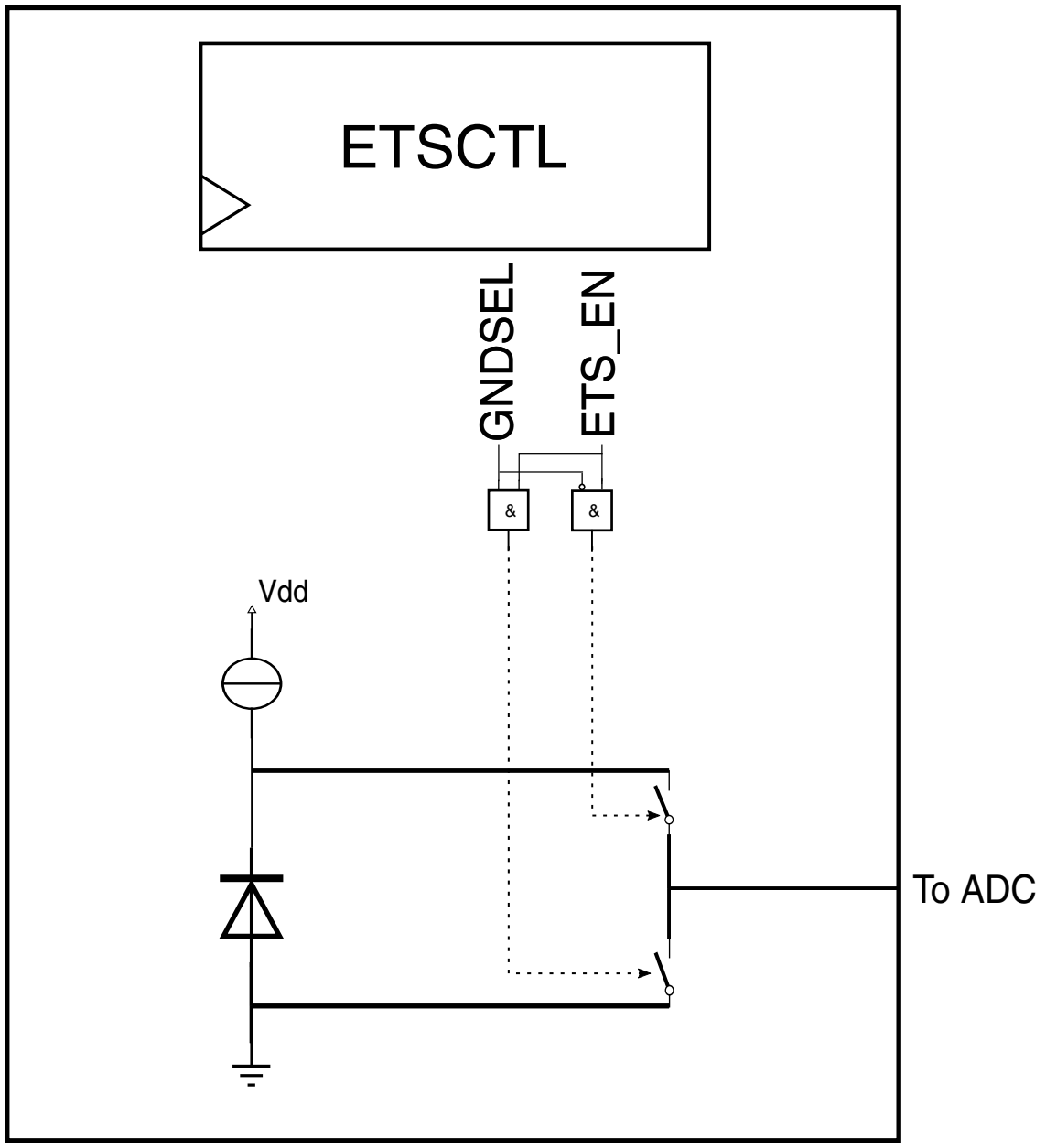


Figure 576. TEMPSENSE block diagram

85.1.2 Features

The TEMPSENSE includes the following features:

- Provide a voltage proportional to the temperature which will be read out by ADC
- Ground selection for improved precision in the ETS. The ETS ground could be different from the ADC ground. By exposing it to the ADC, the temperature calculation could be improved as the voltage value will be more precise.

85.2 Functional description

85.2.1 Clocking

This module has no clocking considerations.

85.2.2 Interrupts

This module has no interrupts.

85.2.3 General

The TEMPSENSE module is designed to measure the temperature on the chip.

The TEMPSENSE output can be read by an ADC on demand so that the software can determine the current die temperature. The software should enable the TEMPSENSE by setting ETSCTL[ETS_EN].

The TEMPSENSE output can be read by an ADC on demand so that the software can determine the current die temperature..

Coefficients are used to allow a simple and accurate calculation of linearized temperature directly from the ADC. The increase of the sampling time of the ADC from the minimum sampling value is increasing the temperature accuracy.

85.2.4 Initialization

85.2.4.1 Conversion from voltage to temperature

Solve the equation for the conversion from voltage to temperature, where V_{ETS} is the difference of V_{be} and V_{GND}. V_{GND} is expose on the ADC output when ETSCTL[GNDSEL] is set.

$$T_{ETS} = TCA_0 + TCA_1 \cdot V_{ETS} + TCA_2 \cdot V_{ETS}^2$$

The coefficients TCA0, TCA1 and TCA2 are read from the corresponding registers. They are stored in a signed fixed-point format as the following TCAx(11,4) (1 bit for the sign, 11 bits for the integer part and 4 bits for the decimal part).

The calculation of the temperature should be done with the actual coefficient values provided in the TCAx fields. See the example below for an ambient temperature of 25C, and an ADC reference voltage (V_{REFH}) of 5V. See the ADC section of the device data sheet for more details on V_{REFH} and N-bit level resolution.

$$TCA0 = (000110100100.1110)_2 = (000110100100)_2 + (1110)_2 \times \frac{1}{2^4} = +((420)_{10} + (14)_{10} \times \frac{1}{2^4}) = +420.8750\ C$$

$$TCA1 = (100010101111.1110)_2 = (100010101111)_2 + (1110)_2 \times \frac{1}{2^4} = -((175)_{10} + (14)_{10} \times \frac{1}{2^4}) = -175.8750\ C/V$$

$$TCA2 = (100000011101.1110)_2 = (100000011101)_2 + (1110)_2 \times \frac{1}{2^4} = -((29)_{10} + (14)_{10} \times \frac{1}{2^4}) = -29.8750\ C\ /V^2$$

$$V_{ETS} = (010110001011)_2 = (1419)_{10} \times \frac{V_{REFH}}{2^N}$$

By using the formula, the junction temperature calculated is 26.5838 C.

The maximal calculation error is 0.0313 C.

85.3 External signals

There are no external signals for this module.

85.4 Memory map and register definition

85.4.1 Transfer error description

The following actions will cause a transfer error and will not change the register content:

- Any access to an unused register address
- Write access to a read-only register (TCAx)

85.4.2 TEMPSENSE register descriptions

85.4.2.1 TEMPSENSE memory map

TEMPSENSE base address: 4037_C000h

Offset	Register	Width (In bits)	Access	Reset value
0h	ETS Control (ETSCTL)	32	RW	0000_0000h
8h	Temperature Coefficient (TCA0)	32	R	See section
Ch	Temperature Coefficient (TCA1)	32	R	See section
10h	Temperature Coefficient (TCA2)	32	R	See section

85.4.2.2 ETS Control (ETSCTL)

Offset

Register	Offset
ETSCTL	0h

Function

This register contains control bits that control ETS.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0														GNDSEL	ETS_EN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-2 —	Reserved
1 GNDSEL	Ground selection 0b - No exposure of the ground 1b - Expose ground on the ADC output if ETSCTL[ETS_EN]=1
0 ETS_EN	Temperature Sensor enable Power up the ETS. 0b - Power down 1b - Functional mode

85.4.2.3 Temperature Coefficient (TCA0)

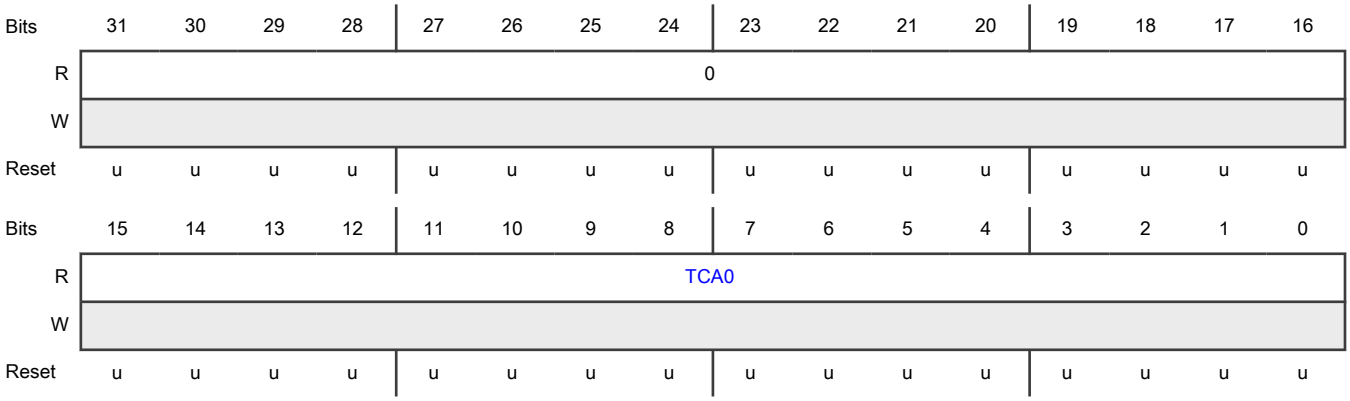
Offset

Register	Offset
TCA0	8h

Function

This register contains the coefficient TCA0 needed to calculate the temperature. the reset value is specific for each chip.

Diagram



Fields

Field	Function
31-16	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
15-0 TCA0	Temperature coefficient A0 see Conversion from voltage to temperature for the usage of this coefficient

85.4.2.4 Temperature Coefficient (TCA1)

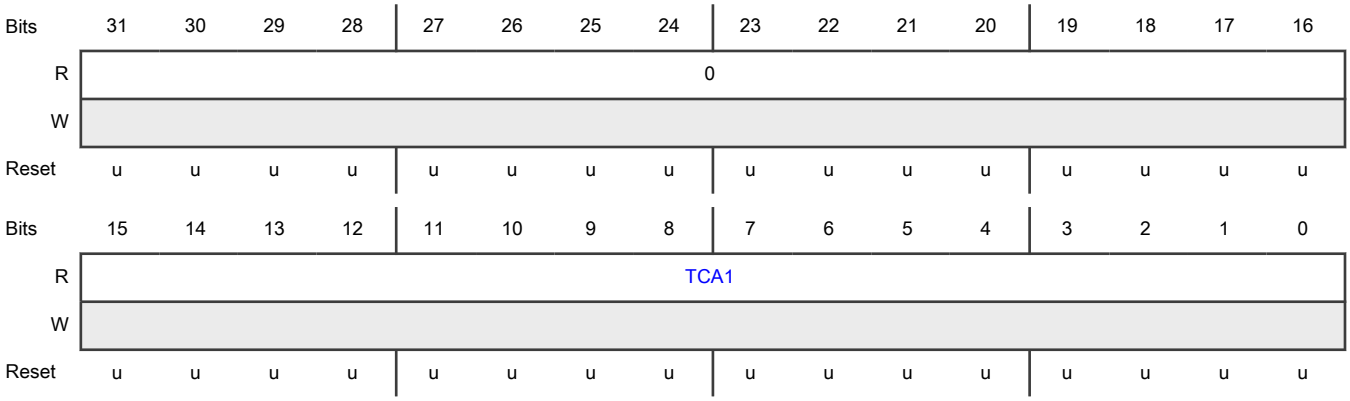
Offset

Register	Offset
TCA1	Ch

Function

This register contains the coefficient TCA1 needed to calculate the temperature. the reset value is specific for each chip.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 TCA1	Temperature coefficient A1 see Conversion from voltage to temperature for the usage of this coefficient

85.4.2.5 Temperature Coefficient (TCA2)

Offset

Register	Offset
TCA2	10h

Function

This register contains the coefficient TCA2 needed to calculate the temperature. the reset value is specific for each chip.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TCA2															
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Fields

Field	Function
31-16 —	Reserved
15-0 TCA2	Temperature coefficient A2 see Conversion from voltage to temperature for the usage of this coefficient

85.5 Glossary

ADC	Analog to digital converter
ETS	Engineering temperature sensor
CTS	Customer temperature sensor
V_{be}	Base emitter voltage
V_{ETS}	Engineering temperature sensor voltage
V_{GND}	Ground voltage

Appendix A
General Changes

A.1 General changes

• Editorial changes.

Appendix B
Release notes

B.1 About This Manual changes

No substantial content changes.

B.2 Introduction changes

• In Table 5, added 100-LQFP package information (available on S32K312, S32K314, S32K324 and S32K344). Also, removed footnote from S32K389 "This feature set is under evaluation and subject to change".
• In Overview, removed the caution "S32K389 specific information is preliminary until the device is qualified and may change without notice". • In S32K3xx product series, replaced Autosar 4.4 with AUTOSAR. • In Feature summary, added 100 LQFP in I/O and package.

B.3 Memory map module changes

No substantial content changes.

B.4 Signal Multiplexing module changes

No substantial content changes.

B.5 Cortex-M7 changes

No substantial content changes.

B.6 Miscellaneous Control Module (MCM)

B.6.1 Chip-specific MCM information changes

• In MCM instances and configuration, added S32K388 and S32K389 in the note for reset value of CM7_2.

B.6.2 MCM module changes

No substantial content changes.

B.7 Miscellaneous System Control Module (MSCM)

B.7.1 Chip-specific MSCM information changes

No substantial content changes.

B.7.2 MSCM module changes

- In Core configuration registers, added details for CP1, CP2, CP3.

B.8 Virtualization Wrapper (VIRT_WRAPPER) for S32K388 and S32K389

B.8.1 Chip-specific VIRT WRAPPER information changes

No substantial content changes.

B.8.2 VIRT_WRAPPER changes

No substantial content changes.

B.9 Virtualization Wrapper (VIRT_WRAPPER) for all chips except S32K388 and S32K389

B.9.1 Chip-specific VIRT WRAPPER information changes

No substantial content changes.

B.9.2 Virt_Wrapper changes

No substantial content changes.

B.10 System Integration Unit Lite2 (SIUL2)

B.10.1 Chip-specific SIUL2 information changes

No substantial content changes.

B.10.2 SIUL2 module changes

No substantial content changes.

B.11 Touch Sensing Pin Coupling (TSPC)

B.11.1 Chip-specific TSPC information changes

No substantial content changes.

B.11.2 TSPC module changes

No substantial content changes.

B.12 Crossbar Switch (AXBS)

B.12.1 Chip-specific AXBS information changes

No substantial content changes.

B.12.2 AXBS module changes

No substantial content changes.

B.13 Peripheral Bridge (AIPS_Lite)

B.13.1 Chip-specific AIPS_Lite information changes

No substantial content changes.

B.13.2 AIPS_Lite module changes

No substantial content changes.

B.14 Direct Memory Access Multiplexer (DMAMUX)

B.14.1 Chip-specific DMAMUX information changes

No substantial content changes.

B.14.2 DMAMUX module changes

No substantial content changes.

B.15 Enhanced Direct Memory Access (eDMA)

B.15.1 Chip-specific eDMA information changes

No substantial content changes.

B.15.2 eDMA3 module changes

No substantial content changes.

B.16 INTM module changes

No substantial content changes.

B.17 Semaphores2 (SEMA42)

B.17.1 Chip-specific SEMA42 information changes

No substantial content changes.

B.17.2 SEMA42 module changes

No substantial content changes.

B.18 Crossbar Integrity Checker (XBIC)

B.18.1 Chip-specific XBIC information changes

No substantial content changes.

B.18.2 XBIC module changes

No substantial content changes.

B.19 Extended Resource Domain Controller (XRDC)

B.19.1 Chip-specific XRDC information changes

- In Table 74, updated XRDC_MRC1 , removed _0 from PRAM0 and PRAM1 and in XRDC_MRC3, renamed PRAM3(2) to PRAM3(0).

B.19.2 XRDC module changes

- Editorial updates

B.20 Memories and Memory Interfaces changes

- In Table 101, added missing S32K358 varaints in Applicability column.

B.21 Embedded Flash Memory (c40asf)

B.21.1 Chip-specific C40asf information changes

No substantial content changes.

B.21.2 Embedded Flash Memory (c40asf)

No substantial content changes.

B.22 Flash Memory Controller (PFLASH)

B.22.1 Chip-specific PFLASH information changes

No substantial content changes.

B.22.2 PFLASH controller module changes

No substantial content changes.

B.23 RAM Controller (PRAMC)

B.23.1 Chip-specific PRAMC information changes

No substantial content changes.

B.23.2 PRAMC module changes

• "Less than 64-bit writes" section is removed. • Removed the redundant note from PRCR1[P0_BO_DIS] bit-field description. • Removed the redundant note from Read and write operations. • Name of the PRCR1 register is updated to " Platform RAM Configuration 1". The word "register" is removed from the name. • Updated the text in table: Clocking.
• Updated the statement about RAM array interface in Overview. • Updated the second note note in PRAMC register descriptions. • Updated bit field value description of 0 in P0_B0_DIS. • Updated Optional read wait state. • Removed the section "Reset". • Updated the image in Block diagram.

B.24 Clocking module changes

• In Chip clock sources, updated the PLL_AUX_PHI0_CLK range to 500 MHz. • In Table 151, updated the AIPS_SLOW_CLOCK for S32K311/S32K312/S32K310 to 40 MHz. • In Option A++ - Very High Performance mode (CM7_CORE_CLK @ 320 MHz) (For S32K388/S32K389) , updated the PLL_PHI1_CLK frequency to 160 MHz for both S32K388 and S32K389.
• In Figure 60, removed the statement that STM1_CLK is not available in S32K314. • In Figure 67, removed the statement that STM1_CLK is not available in S32K314. • In Clock monitoring , added a note and update figure "Frequency checking (FC) instances" for CMU_FC_0.

B.25 Clock Generation Module (MC_CGM)

B.25.1 Chip-specific MC_CGM information changes

No substantial content changes.

B.25.2 MC_CGM module changes

No substantial content changes.

B.26 FIRC module changes

No substantial content changes.

B.27 SIRC module changes

No substantial content changes.

B.28 Fast Crystal Oscillator Digital Controller (FXOSC)

B.28.1 Chip-specific FXOSC information changes

No substantial content changes.

B.28.2 FXOSC module changes

No substantial content changes.

B.29 SXOSC module changes

No substantial content changes.

B.30 PLL Digital Interface (PLLDIG)

B.30.1 Chip-specific PLLDIG information changes

No substantial content changes.

B.30.2 PLLDIG module changes

No substantial content changes.

B.31 Reset module changes

No substantial content changes.

B.32 Boot Overview changes

No substantial content changes.

B.33 Reset Generation Module (MC_RGM)

B.33.1 Chip-specific MC_RGM information changes

No substantial content changes.

B.33.2 MC_RGM module changes

- Added following note in the Reset During Standby Status Register (RDSS) register: Set flags in the RDSS register must be cleared before a new standby entry.

B.34 POR_WDG module changes

No substantial content changes.

B.35 Security overview changes

No substantial content changes.

B.36 HSE Subsystem Overview changes

No substantial content changes.

B.37 DCF module changes

No substantial content changes.

B.38 DCM_GPR module changes

- Updated the note in bit field DCMRWD3[LC_ERR_EN].
- Updated the description of DCMRWF2 reserved bit 6.
- Updated the reset value of Read Write GPR On Destructive Reset 17 (DCMRWD17) to 0x07FFFFFF.

B.39 DCM module changes

No substantial content changes.

B.40 Messaging Unit (MU)

B.40.1 Chip-specific MU information changes

No substantial content changes.

B.40.2 MU module changes

No substantial content changes.

B.41 Power management changes

No substantial content changes.

B.42 PMC changes

No substantial content changes.

B.43 PMC changes

No substantial content changes.

B.44 PMC changes

No substantial content changes.

B.45 PMC changes

No substantial content changes.

B.46 Mode Entry Module (MC_ME)

B.46.1 Chip-specific MC_ME information changes

No substantial content changes.

B.46.2 MC_ME module changes

No substantial content changes.

B.47 MC_PCU changes

No substantial content changes.

B.48 Wakeup Unit (WKPU)

B.48.1 Chip-specific WKPU information changes

No substantial content changes.

B.48.2 WKPU module changes

No substantial content changes.

B.49 Safety overview changes

No substantial content changes.

B.50 Error Injection Module (EIM)

B.50.1 Chip-specific EIM information changes

• In Table 279 channel number 29, removed the typo "[59:0] – ace_gskt_slave_err_inj59:0]". Also, added PRAM_3 in channel number 29 (11:10) and channel number 30 (5:4).
• In Table 279 channel number 30, added Word1[7:6] – Inject error on QSPI write data checker.

B.50.2 EIM module changes

• Updated field description of EICHD*n*_WORD*n* register except then EICHD*n*_WORD0
• Removed "The EIM supports 17 error injection channels" in Functional description section. • Updated text "Supports 17 error injection channel." to "Supports upto 32 error injection channel" in Features section.

B.51 Error Reporting Module (ERM)

B.51.1 Chip-specific ERM information changes

• In ERM channel mapping, added note "A hard fault is generated when non-correctable errors are detected".

B.51.2 ERM module changes

No substantial content changes.

B.52 Fault Collection and Control Unit (FCCU)

B.52.1 Chip-specific FCCU information changes

No substantial content changes.

B.52.2 FCCU module changes

No substantial content changes.

B.53 SELFTEST_GPR module changes

No substantial content changes.

B.54 Self-Test Control Unit (STCU2)

B.54.1 Chip-specific STCU2 information changes

- In STCU2 LBIST/MBIST mapping, added note "For module instance availability. See the respective chip-specific section for details".
- In Table 302:
  - Removed notes addressing a specific variant
  - Removed MDAC2 and MDAC3.
  - Updated EIM, ERM
  - Added REG_PROT
  - Updated iAHB Gaskets
  - Added AIPS_Lite_0

B.54.2 STCU2 module changes

No substantial content changes.

B.55 Register Protection (REG_PROT)

B.55.1 Chip-specific REGPROT information changes

No substantial content changes.

B.55.2 REG_PROT module changes

No substantial content changes.

B.56 Clock Monitoring Unit – Frequency Check (CMU_FC)

B.56.1 Chip-specific CMU_FC information changes

No substantial content changes.

B.56.2 CMU_FC module changes

- Overview: Added text to refer chip-specific section

B.57 Clock Monitoring Unit – Frequency Meter (CMU_FM)

B.57.1 Chip-specific CMU_FM information changes

No substantial content changes.

B.57.2 CMU_FM module changes

- Block diagram: Added SR[MET_CNT]

B.58 Cyclic Redundancy Check (CRC)

B.58.1 Chip-specific CRC information changes

No substantial content changes.

B.58.2 CRC module changes

No substantial content changes.

B.59 PCMC changes

- In ADC input noise, added note "When possible, use staggered ADC conversions instead of simultaneous conversions".
- Updated the title of "PWM and trigger generation" and "OPWMCB PWM and trigger generation" to OPWMCB PWM and trigger generation and OPWMB PWM and trigger generation.

B.60 Analog-to-Digital Converter (ADC)

B.60.1 Chip-specific ADC information changes

- In ADC Configuration, added two notes "No ADC fault is routed to FCCU" and "All supplies are monitored internally---informational purposes only".
- In Table 344, added information for channel 0, 7, 8, 11, 12, and 28.

B.60.2 ADC module changes

No substantial content changes.

B.61 Low Power Comparator (LPCMP)

B.61.1 Chip-specific LPCMP information changes

No substantial content changes.

B.61.2 LPCMP module changes

- Corrected the titles the images "Sampled, non-filtered (#3A).." and "Sampled, Filter (#4A).."

B.62 Logic Control Unit (LCU)

B.62.1 Chip-specific LCU information changes

No substantial content changes.

B.62.2 LCU module

No substantial content changes.

B.63 Enhanced Modular IO Subsystem (eMIOS)

B.63.1 Chip-specific eMIOS information changes

No substantial content changes.

B.63.2 eMIOS module changes

No substantial content changes.

B.64 Body Cross-triggering Unit (BCTU)

B.64.1 Chip-specific BCTU information changes

No substantial content changes.

B.64.2 BCTU module changes

No substantial content changes.

B.65 Trigger MUX (TRGMUX)

B.65.1 Chip-specific TRGMUX information changes

No substantial content changes.

B.65.2 TRGMUX module changes

No substantial content changes.

B.66 Software Watchdog Timer (SWT)

B.66.1 Chip-specific SWT information changes

No substantial content changes.

B.66.2 SWT module changes

No substantial content changes.

B.67 System Timer Module (STM)

B.67.1 Chip-specific STM information changes

No substantial content changes.

B.67.2 STM module changes

No substantial content changes.

B.68 Periodic Interrupt Timer (PIT)

B.68.1 Chip-specific PIT information changes

No substantial content changes.

B.68.2 PIT module changes

• Editorial changes.
• Editorial changes.

B.69 Real Time Clock (RTC)

B.69.1 Chip-specific RTC information changes

No substantial content changes.

B.69.2 RTC module changes

No substantial content changes.

B.70 Low Power Serial Peripheral Interface (LPSPI)

B.70.1 Chip-specific LPSPI information changes

No substantial content changes.

B.70.2 LPSPI module changes

No substantial content changes.

B.71 Low Power Inter-Integrated Circuit (LPI2C)

B.71.1 Chip-specific LPI2C information changes

No substantial content changes.

B.71.2 LPI2C module changes

No substantial content changes.

B.72 Flexible I/O (FlexIO)

B.72.1 Chip-specific FlexIO information changes

No substantial content changes.

B.72.2 FLEXIO module changes

No substantial content changes.

B.73 CAN (FlexCAN)

B.73.1 Chip-specific FlexCAN information changes

No substantial content changes.

B.73.2 FlexCAN module changes

- In the "Maximum CAN bit rate in data phase on CAN FD frames with Enhanced RX FIFO disabled" table, changed the column heading "Payload size" to "Minimum payload size".

- Changed all instances of ISO 11898-1:2015 to 11898-1_2024.

B.74 Synchronous Audio Interface (SAI)

B.74.1 Chip-specific SAI information changes

No substantial content changes.

B.74.2 SAI module changes

- Updated description in PARAM[FRAME]..
- Updated description in TCR1[TFW].

- Updated description in PARAM[FRAME]..

B.75 Ethernet Media Access Controller (EMAC)

B.75.1 Chip-specific EMAC information changes

- In EMAC instance and configuration, added note 'It is essential that all---the PHY can be reset without affecting the MAC'.

B.75.2 ENET_Controller module changes

No substantial content changes.

B.76 Gigabit Ethernet Media Access Controller (GMAC)

B.76.1 Chip-specific GMAC information changes

- In GMAC instances and configuration, added note 'It is essential that all---the PHY can be reset without affecting the MAC'.

B.76.2 GMAC module changes

No substantial content changes.

B.77 Low Power Universal Asynchronous Receiver/Transmitter (LPUART)

B.77.1 Chip-specific LPUART information changes

No substantial content changes.

B.77.2 LPUART module changes

No substantial content changes.

B.78 Quad Serial Peripheral Interface (QuadSPI) for S32K322, S32K342, S32K341, S32K314, S32K324, and S32K344

B.78.1 Chip-specific QuadSPI information changes

No substantial content changes.

B.78.2 QuadSPI module changes

No substantial content changes.

B.79 Quad Serial Peripheral Interface (QuadSPI) for S32K358, S32K356, S32K348, S32K338, and S32K328

B.79.1 Chip-specific QuadSPI information changes

No substantial content changes.

B.79.2 QuadSPI module changes

No substantial content changes.

B.80 Quad Serial Peripheral Interface (QuadSPI) for S32K388 and S32K389

B.80.1 Chip-specific QuadSPI information changes

No substantial content changes.

B.80.2 QuadSPI module changes

No substantial content changes.

B.81 Ultra Secured Digital Host Controller (uSDHC)

B.81.1 Chip-specific uSDHC information changes

No substantial content changes.

B.81.2 uSDHC module changes

No substantial content changes.

B.82 Debug changes

- In Application debugging from first instruction, added self-test done in the statement "This chip supports debugging from the first instruction on system power-up, destructive reset, functional reset, self-test done, and standby exit".

B.83 JTAG Controller (JTAGC)

B.83.1 Chip-specific JTAGC information changes

No substantial content changes.

B.83.2 JTAGC module changes

No substantial content changes.

B.84 JDC module changes

No substantial content changes.

B.85 TEMPSENSE module changes

No substantial content changes.

Legal information

Definitions

Draft — A draft status on a document indicates that the content is still under internal review and subject to formal approval, which may result in modifications or additions. NXP Semiconductors does not give any representations or warranties as to the accuracy or completeness of information included in a draft version of a document and shall have no liability for the consequences of use of such information.

Disclaimers

Limited warranty and liability — Information in this document is believed to be accurate and reliable. However, NXP Semiconductors does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information. NXP Semiconductors takes no responsibility for the content in this document if provided by an information source outside of NXP Semiconductors.

In no event shall NXP Semiconductors be liable for any indirect, incidental, punitive, special or consequential damages (including - without limitation - lost profits, lost savings, business interruption, costs related to the removal or replacement of any products or rework charges) whether or not such damages are based on tort (including negligence), warranty, breach of contract or any other legal theory.

Notwithstanding any damages that customer might incur for any reason whatsoever, NXP Semiconductors' aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the Terms and conditions of commercial sale of NXP Semiconductors.

Right to make changes — NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

Applications — Applications that are described herein for any of these products are for illustrative purposes only. NXP Semiconductors makes no representation or warranty that such applications will be suitable for the specified use without further testing or modification.

Customers are responsible for the design and operation of their applications and products using NXP Semiconductors products, and NXP Semiconductors accepts no liability for any assistance with applications or customer product design. It is customer's sole responsibility to determine whether the NXP Semiconductors product is suitable and fit for the customer's applications and products planned, as well as for the planned application and use of customer's third party customer(s). Customers should provide appropriate design and operating safeguards to minimize the risks associated with their applications and products.

NXP Semiconductors does not accept any liability related to any default, damage, costs or problem which is based on any weakness or default in the customer's applications or products, or the application or use by customer's third party customer(s). Customer is responsible for doing all necessary testing for the customer's applications and products using NXP Semiconductors products in order to avoid a default of the applications and the products or of the application or use by customer's third party customer(s). NXP does not accept any liability in this respect.

Terms and conditions of commercial sale — NXP Semiconductors products are sold subject to the general terms and conditions of commercial sale, as published at https://www.nxp.com/profile/terms, unless otherwise agreed in a valid written individual agreement. In case an individual agreement is concluded only the terms and conditions of the respective agreement shall apply. NXP Semiconductors hereby expressly objects to applying the customer's general terms and conditions with regard to the purchase of NXP Semiconductors products by customer.

Export control — This document as well as the item(s) described herein may be subject to export control regulations. Export might require a prior authorization from competent authorities.

HTML publications — An HTML version, if available, of this document is provided as a courtesy. Definitive information is contained in the applicable document in PDF format. If there is a discrepancy between the HTML document and the PDF document, the PDF document has priority.

Translations — A non-English (translated) version of a document, including the legal information in that document, is for reference only. The English version shall prevail in case of any discrepancy between the translated and English versions.

Security — Customer understands that all NXP products may be subject to unidentified vulnerabilities or may support established security standards or specifications with known limitations. Customer is responsible for the design and operation of its applications and products throughout their lifecycles to reduce the effect of these vulnerabilities on customer's applications and products. Customer's responsibility also extends to other open and/or proprietary technologies supported by NXP products for use in customer's applications. NXP accepts no liability for any vulnerability. Customer should regularly check security updates from NXP and follow up appropriately.

Customer shall select products with security features that best meet rules, regulations, and standards of the intended application and make the ultimate design decisions regarding its products and is solely responsible for compliance with all legal, regulatory, and security related requirements concerning its products, regardless of any information or support that may be provided by NXP.

NXP has a Product Security Incident Response Team (PSIRT) (reachable at PSIRT@nxp.com) that manages the investigation, reporting, and solution release to security vulnerabilities of NXP products.

Suitability for use in automotive applications (functional safety) — This NXP product has been qualified for use in automotive applications. It has been developed in accordance with ISO 26262, and has been ASIL classified accordingly. If this product is used by customer in the development of, or for incorporation into, products or services (a) used in safety critical applications or (b) in which failure could lead to death, personal injury, or severe physical or environmental damage (such products and services hereinafter referred to as "Critical Applications"), then customer makes the ultimate design decisions regarding its products and is solely responsible for compliance with all legal, regulatory, safety, and security related requirements concerning its products, regardless of any information or support that may be provided by NXP. As such, customer assumes all risk related to use of any products in Critical Applications and NXP and its suppliers shall not be liable for any such use by customer. Accordingly, customer will indemnify and hold NXP harmless from any claims, liabilities, damages and associated costs and expenses (including attorneys' fees) that NXP may incur related to customer's incorporation of any product in a Critical Application.

NXP B.V. — NXP B.V. is not an operating company and it does not distribute or sell products.

Trademarks

Notice: All referenced brands, product names, service names, and trademarks are the property of their respective owners.

NXP — wordmark and logo are trademarks of NXP B.V.

AMBA, Arm, Arm7, Arm7TDMI, Arm9, Arm11, Artisan, big.LITTLE, Cordio, CoreLink, CoreSight, Cortex, DesignStart, DynamIQ, Jazelle, Keil, Mali, Mbed, Mbed Enabled, NEON, POP, RealView, SecurCore, Socrates, Thumb, TrustZone, ULINK, ULINK2, ULINK-ME, ULINK-PLUS, ULINKpro, µVision, Versatile — are trademarks and/or registered trademarks of Arm Limited (or its subsidiaries or affiliates) in the US and/or elsewhere. The related technology may be protected by any or all of patents, copyrights, designs and trade secrets. All rights reserved.

Bluetooth — the Bluetooth wordmark and logos are registered trademarks owned by Bluetooth SIG, Inc. and any use of such marks by NXP Semiconductors is under license.

EdgeLock — is a trademark of NXP B.V.

eIQ — is a trademark of NXP B.V.

I2C-bus — logo is a trademark of NXP B.V.

SafeAssure — is a trademark of NXP B.V.

SafeAssure — logo is a trademark of NXP B.V.

SuperFlash — This product uses SuperFlash® technology. SuperFlash® is a registered trademark of Silicon Storage Technology, Inc.

Synopsys & Designware — are registered trademarks of Synopsys, Inc.

Synopsys — Portions Copyright © 2018-2022 Synopsys, Inc. Used with permission. All rights reserved.



Please be aware that important notices concerning this document and the product(s) described herein, have been included in section 'Legal information'.

© 2025 NXP B.V. All rights reserved.

For more information, please visit: https://www.nxp.com

Date of release: 11 November 2025
Document identifier: S32K3XXRM